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

2008－DBS－146 (38)
2008／9／22

グラフ分解を用いた構造による部分グラフ問い合わせ処理の改良とその実験的評価

関 建二郎† 片山 薫†

† 首都大学東京大学院システムデザイン研究科 〒191–0065 東京都日野市旭丘 6-6
E-mail: †Kenjiro.Seki@gmail, ††kaoru@comp.metro-u.ac.jp

あらまし グラフは複雑な構造を表現する重要な構造として，ますます重要性が高まっている．それに伴い，グラフデータベースに対する効率的な検索が求められている．しかし，部分グラフ同型判定問題は NP 完全であることが知られており，そのような検索は複雑かつ困難である．そのため，グラフデータベースを効率的に検索するための索引付けの研究が数多く成されている．しかし，索引付けの多くはクエリを含むグラフの数が多いほど，候補者集合の検証に時間がかかり，非効率的になってしまう．一方，布施らはクエリを含むグラフの数が非常に多い場合により有効である検索手法を提案した．我々は，既存のグラフの枝刈り手法を利用することで，彼らの手法をより一般的なデータも効率的に処理できるように改良した．実験では，改良した手法を彼らの提案した手法，VF2 による逐次処理，He らによる Closure Tree を修正した手法と比較し性能を評価した．
キーワード インデクス，グラフデータベース，部分グラフ同型

An Improvement of Graph Query Processing Using Messmer's Decomposition Approach and its Experimental Evaluation

Kenjiro SEKI† and Kaoru KATAYAMA†

† Guraduate School of System Design, Tokyo Metoropolitan University
Asahigaoka 6–6, Hino-shi, Tokyo, 191–0065 Japan
E-mail: †Kenjiro.Seki@gmail, ††kaoru@comp.metro-u.ac.jp

Abstract Graphs have become increasingly important structures for representing and understanding complex structures. So an efficient retrieval from graph database is required. But it is complicated and expensive because the subgraph isomorphism problem is NP-complete. Recently, there have been several studies for efficient retrieval from graph database, but it become inefficient in the case that the size of answer set is large, because it needs candidate verifidation. Meanwhile, Fuse et al. proposed a graph retrieval method that is efficient especially in the case the size of answer set is very large. We improved their method for applying more typical graph database by making use of an index based graph pruning. We evaluate in our experiment that the performance of our method by comparing with their method, sequential process using VF2 and modification of Closure Tree proposed by He et al.
Key words Graph Indexing, Graph Database, Subgraph Isomorphsim

1. まえがき

グラフは複雑な構造を表現する重要な構造として，ますます重要性が高まっている．生物学や化学の領域では，化合物の分子構造を表現するためにグラフが使われる．また，近年急激に増大している Web や XML 文書は非常に大きなグラフとして表現される．その他，パターン認識やコンピュータビジョン，ソーシャルネットワークなど多くの分野でグラフが適用されている．それに伴い，グラフデータベースに対する効率的な検索が求められている．

我々は，グラフデータベース $D = \{g_1, g_2, \ldots, g_n\}$ と問い合わせグラフ q が与えられた時，D の中から q を部分グラフとして含むグラフをすべて見つける問題と部分グラフ問い合わせと定義する．同様に，D の中から q を上位グラフとするグラフをすべて見つける問題を上位グラフ問い合わせと定義する．

部分グラフ同型判定問題は NP 完全であることが知られてお

り [2]，これらの問い合わせを行うために，D 内の全てのグラフに対して順次部分グラフ同型判定を行う手法は非効率的である．

そこで，グラフを効率的に検索するためのグラフデータベースに対する索引付けが必要になってくる．近年，部分グラフ問い合わせを効率的に行うため，多くのグラフデータベースに対する索引付けの研究がされている [9] [6] [12]．これらの手法は次の 2 つステップから成る．初めに索引を利用して，正解ではないグラフの枝刈りを行い，正解の候補であるグラフの集合を生成する．次に，候補者集合に対して，部分グラフ同型判定を順次適応し，本当に正解のグラフであるかを確かめる．候補者集合の大きさはグラフデータベースの大きさより小さいため，これらの索引付けを利用した部分グラフ問い合わせは効率的である．しかし，候補者集合の大きさは正解のグラフ集合より小さくはならない．したがって，正解のグラフ集合が大きい場合，枝刈りと候補者集合の検証からなる索引付けは非効率的になってしまう．

Messmer ら [7] は，上位グラフ問い合わせに対する研究として、決定木を用いた特殊な構造を提案した．彼らの構造は，分割統治法でグラフを扱えるようにするため，グラフを再帰的に分解していくことで構成される．一方，布施ら [4] は，Messmer らの構造を利用し，部分グラフ問い合わせを分割統治法で扱う手法を提案した．彼らの手法は一般的な部分グラフ問い合わせに対しては非効率的であったが，グラフデータベースの中で正解集合に含まれるグラフの割合が非常に大きい場合に有効なケースがあった．

本稿では，まず既存の索引付け手法によるグラフの枝刈りを利用し，その後に布施らの手法を適用することで，一般的な部分グラフ問い合わせも効率的に処理できるように，彼らの手法を改良した．既存の索引付け手法は，それ単体でグラフデーベース内のグラフの枝刈りの効果があると共に，布施らの構造内のグラフも枝刈りすることができ，彼らの手法を促進できる．また，既存の索引付け手法による枝刈りを行った後，その候補者集合と問い合わせグラフによる部分グラフ問い合わせを考えたとき，正解のグラフの割合が大きいグラフ問い合わせになっており，布施らの手法にとって有利である．

本稿では，枝刈りを行う索引付けとして，He ら [6] による Closure Tree を利用した．我々は実験で，改良した手法を彼らの提案した手法と比較し性能が大きく改善されていることを示す．同時に，本手法が部分グラフ同型判定手法 VF2 [3] の逐次処理による部分グラフ問い合わせよりも効率的に処理されることを示す．しかし，我々は Closure Tree の問題点を発見し，修正したところ，その性能は提案手法より優れていることが分かった．

2. 関 連 研 究

部分グラフ同型判定問題は NP 完全であることが知られている [2]．この問題に関する研究として Ullman [10]，VF2 [3] 等がある．

近年，グラフデータベースに対するグラフの検索について，多くの研究が成されている．これらの研究は 2 種類に分類することができる．1 つはグラフデータベースから問い合わせグラフを部分グラフとして含むグラフをすべて答える問題である．この問題に関する研究として，Graph Grep [9], Closure-Tree [6], gIndex [12], GDIndex [11], FG-Index [1] 等がある．もう 1 つはグラフデータベースから問い合わせグラフを上位グラフとするグラフをすべて答える上位グラフ問い合わせに関する研究である．Messmer [7] は，上位グラフ問い合わせに対する研究として、決定木を用いた特殊な構造を提案した．彼らの構造は，分割統治法でグラフを扱えるようにするため，グラフを再帰的に分解していくことで構成される．

3. 準　　備

本章では，グラフとグラフ同型に関するいくつかの定義を確認する．本稿においては，議論の対象をラベル付の無向グラフに限定する．しかし，提案手法の他種類のグラフへ拡張は容易である．以降，ラベル付の無向グラフをグラフと呼ぶ．

［定義 1］ グラフ $g = (V, E, \mu, \nu)$ とは，頂点集合 V，枝集合 E，頂点集合を頂点ラベルに写像する関数 μ，および枝集合を枝ラベルに写像する関数 ν の 4 つ組である．グラフ g の頂点集合，枝集合をそれぞれ $V(g)$,$E(g)$ と表す．また，枝集合の大きさ $|E(g)|$ をグラフのサイズと呼ぶ．

［定義 2］ 単射写像 $\phi(\phi : V(g_1) \to V(g_2))$ が以下の条件を満たすとき，ϕ を g_1 から g_2 への部分グラフ同型と呼び，$g_1 \subseteqq g_2$ と表記する．

（1） すべての $v \in V(g_1)$ について，$\phi(v) \in V(g_2)$ かつ $\mu(v) = \nu(\phi(v))$,

（2） すべての $e = (v, u) \in E(g_1)$ について，$\phi(e) = (\phi(v), \phi(u)) \in E(g_2)$ かつ $\nu(e) = \nu(\phi(e))$

$g_1 \subseteqq g_2$ であるとき，g_1 は g_2 の部分グラフであるという．また，g_2 は g_1 の上位グラフであるという．

［定義 3］ グラフデータベース $D = \{g_1, g_2, \ldots, g_n\}$ と問い合わせグラフ q が与えられたとき，D 内から q を部分グラフとして含むすべてのグラフの集合 $D_q = \{g_i \mid q \subseteqq g_i, g_i \in D\}$ を得る問題を部分グラフ問い合わせと呼ぶ．同様に，D 内から q を上位グラフとするすべてのグラフの集合 $D_q = \{g_i \mid g_i \subseteqq q, g_i \in D\}$ を得る問題を上位グラフ問い合わせと呼ぶ．

グラフ問い合わせにより得られる集合 D_q を正解集合と呼ぶ．また，枝刈りと検証からなるグラフ問い合わせにおいて，枝刈り後に得られるグラフ集合を候補者集合と呼び C_q で表す．あるグラフ問い合わせにおいて，グラフデータベースの中で正解集合に含まれるグラフの割合を $\gamma = |D_q|/|D|$ で表す．また，枝刈りの正確さを $\alpha = |D|/|C_q|$ で表す．

4. Messmer らの構造を用いた部分グラフ問い合わせの改良

本研究の目的は部分グラフ問い合わせを効率的に処理することである．本稿では，まず既存の索引付けによるグラフの枝刈りを利用し，その後に布施らの手法を適用することで，一般的な部分グラフ問い合わせも効率的に処理できるように，彼らの手法を改良する．本節では，まず布施らにより提案され

た Messmer らの構造を用いたグラフ問い合わせを説明する．次に，既存の索引付け手法の例として，今回利用する Closure Tree について紹介し，いくつかの見解を述べる．最後に索引付け手法を利用して，布施らの手法を改良する方法を提案する．

4.1 Messmer らの構造を用いた部分グラフ問い合わせ

Messmer ら [7] は，上位グラフ問い合わせのための決定木を用いた特殊な構造を提案した．彼らの構造は，グラフを分割統治法で扱えるようにするため，グラフを再帰的に分解していくことで構成される．以降，彼らの構造を M-Tree と表記する．布施ら [5] は，M-Tree の分解において，連結グラフは連結グラフに分解されるという制約を付け，組み合わせの急増を減らした．

M-Tree の上位グラフ問い合わせはボトムアップで処理される．M-Tree 中のあるノードの問い合わせグラフへの部分グラフ同型はノードの子の部分グラフ同型から計算される．M-Tree の手法の利点を以下に示す．

（1） 部分グラフ同型を持たない子を持っているノードは部分グラフ同型を持たないという事実を利用して枝刈りを行える．

（2） 共有されているノードの部分グラフ同型の計算は 1 回で済む

（3） 分割統治法により，大きいノードの部分グラフ同型を小さいノードの部分グラフ同型から計算できる

布施ら [4] は，さらに部分グラフ問い合わせに M-Tree を使用することを提案した．この場合，あるノードの子が 1 つでも質問グラフに対して部分グラフ同型をもつとき，そのノードも部分グラフ同型をもつという事実が利用でき，処理時間の短縮が期待できる．また，M-Tree をそのまま使用しているので，この構造があれば上位グラフ問い合わせも処理することもできる．

我々が実験した結果，布施らの部分グラフ問い合わせが有効なケースは非常に少かった．しかし，γ が大きく，ラベル数が少ない場合にこの手法が有効なケースが出てくる．このようなケースは実際のグラフデータにおいて非常に稀である．しかし，既存の索引付けによるグラフの枝刈りを組み合わせることで，γ が大きいグラフ問い合わせを作ることができる．

4.2 Closure Tree

He ら [6] は部分グラフ問い合わせおよび類似グラフ問い合わせの両方の問い合わせを効率的に処理するための索引付け手法として Closure Tree(以下，C-Tree) を提案した．C-Tree はグラフ閉包という複数のグラフをまとめて表現する概念を利用し，グラフを階層的にまとめて索引付けを行う．C-Tree の階層的な索引付け手法は R-Tree の形に則っているため，パラメータとして，ノードが持つ最小の子の数 m およびノードが持てる最大の子の数 M を持っている．

C-Tree の部分グラフ問い合わせでは HistogramTest による枝刈りのあと，PseudoSubgraphIsomorphism による枝刈りを行い，最後に候補者集合を部分グラフ同型判定による逐次処理で検証する．HistogramTest は問い合わせグラフとグラフ閉包のラベル数をラベルの種類ごとに比較する枝刈りである．PseudoSubgraphIsomorphism は問い合わせグラフとグラフ閉包の各ノード間について隣接部分木のマッチングを調べ枝刈りを行う．枝刈りの後は，候補者集合に対して逐次部分グラフ同型判定を行う必要がある．

PseudoSubgraphIsomorphism は候補者集合と正解集合がほぼ一致する優れた正確さを持っている．しかし，我々の実験では枝ラベルが付くと大きく処理時間が増えてしまうことや，木の深さの処理時間への影響が大きいことなど，欠点が多いことも分かった．また，C-Tree の枝刈りの性能に大きく貢献しているのは，HistogramTest であることも分かった．HistogramTest による枝刈りはあまり正確ではないが，処理時間が少なく効率的である．本稿では，PseudoSubgraphIsomorphism の処理は行わず，HistogramTest のみを適用する．実験で使用している C-Tree の機能は HistogramTest だけであることに注意されたい．

部分グラフ問い合わせにおいて γ が小さい場合，候補者集合を小さくできるため，C-Tree は効率的に作用する．しかし，γ が大きい場合，候補者集合は正解集合より小さくならないため，候補者集合の検証による処理時間が増えてしまう．これは全ての索引を用いた枝刈りと候補者集合の検証からなる部分グラフ問い合わせ手法に対して言えることである．

4.3 提 案 手 法

我々は，既存の索引付けによるグラフの枝刈りを利用し，その後に布施らの手法を適用することで，一般的な部分グラフ問い合わせも効率的に処理できるように，彼らの手法を改良した．この改良の利点は以下にあげられる．

（1） 索引付けによる枝刈りは M-Tree のノードの枝刈りにも適用できる．

（2） 布施らの手法は γ が大きい程，有効になる．このようなデータベースは稀であるが索引付けによる枝刈りを行った後の候補者集合をグラフデータベースとして見ると，枝刈り前より γ が大きくなっている．したがって，枝刈りの後に布施らの手法を適用することで，総合的に処理が早くなることを期待できる．また，枝刈りの正確さ α が優れているほど，枝刈り後の γ は大きくなる．

以降では，枝刈りのための索引付けとして C-Tree を用いて，部分グラフ問い合わせのアルゴリズムと実験を示す．木の構築アリゴリズムは元の C-Tree と M-Tree と同じである．両方の構造で逐次的なグラフの挿入が可能であることに注目されたい．

4.3.1 部分グラフ問い合わせ

Algorithm1 は部分グラフ問い合わせのアルゴリズムを示している．2 つの木の全てのノードはマーク *dead*,*alive*,*unsolved* のいずれかをもっており，それぞれ「部分グラフである」,「部分グラフでない」,「検証が必要である」ことを示す．初めに，C-Tree の HistogramTest による枝刈りを行う（Algorithm2）．HistogramTest はあらかじめ数えてあるグラフ内のラベル数をラベルの種類ごとに比較する．1 つでも，問い合わせグラフのラベル数がグラフ閉包のラベル数を超えていれば，false が返される．この枝刈りは M-Tree のノードも含めたすべての子孫に *dead* を与えることで，後の M-Tree での処理を削減できる．次に，M-Tree のノードの内 *unsolved* なまま残っているものに対して，ボトムアップで部分グラフ同型判定をかけていく

Algorithm 1 SubgraphQuery

Input: Query graph q, C-tree $ctree$, M-tree $mtree$
Output: Answer set D_q

```
1: mark all nodes of mtree and ctree unsolved
2: TraverseUpperIndex(q, ctree.root)
3: for all nodes of database graph do
4:     TraverseUnderIndex(q, node)
5: end for
6: return D_q {all database graph marked alive}
```

Algorithm 2 TraverseUpperIndex

Input: Query graph q, C-tree's node $node$

```
1: if node is unsolved then
2:     let g be the graph or graph closure at node
3:     if HistogramTest(g,q) is true then
4:         for child c of node node do
5:             TraverseUpperIndex(q,c)
6:         end for
7:     else
8:         mark node and node's all descendants dead
9:     end if
10: end if
```

(Alogithm3)．最後に，グラフデータベースに属するノードのうち $alive$ がマークされているグラフの集合を出力する．

Algorithm 3 TraverseUnderIndex

Input: Query graph q, M-tree's node $node$
Output: Mark

```
1: if node is unsolved then
2:     for child c of node node do
3:         if TraverseUnderIndex(q, c) is alive then
4:             mark node alive
5:             return alive
6:         end if
7:     end for
8:     let g be the graph at node
9:     if q ⊆ g then
10:        mark node alive
11:    else
12:        mark node dead
13:    end if
14: end if
15: return mark of node
```

4.3.2 提案手法とそのグラフ問い合わせの例

図 1 は提案手法とその部分グラフ問い合わせの例を示している．グラフデータベースは 4 つのグラフから成る．簡単のため枝ラベルなしのグラフを考える．グラフ閉包におけるラベル e は，頂点が存在しない状態を意味している．

最初に，グラフデータベースの上に位置する C-Tree を捜査する．グラフ閉包 $c1$ において，$c1$ は問い合わせグラフ q が持つラベルを持たないため HistogramTest が false を返す．例に示されているように，このときグラフデータベースだけでなく，M-Tree のノードを含む，$c1$ のすべての子孫に $dead$ のマークを与えることができ，後の処理を削減できる．

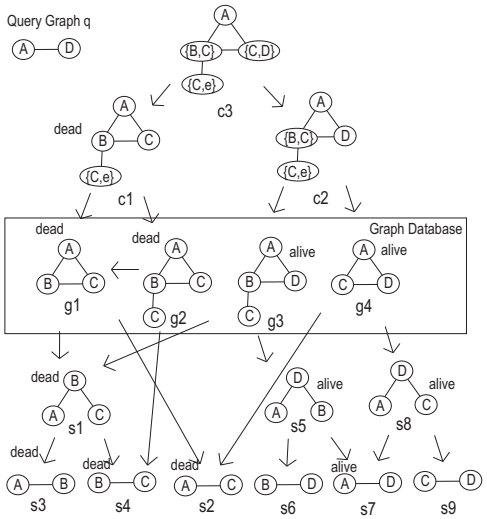

図 1　提案手法とそのグラフ問い合わせの例

次に，グラフデータベースの下に位置する M-Tree をボトムアップに捜査する．ノード $s7$ は問い合わせグラフからの部分グラフ同型を持っている．したがって，$s7$ と問い合わせグラフの一回の部分グラフ同型判定でその先祖すべてに対して $alive$ のマークを付与できる．この時点ですべてのグラフデータベースを調べ終わっているので，グラフデータベース内の $alive$ が付いたすべてのグラフの集合を返す．C-Tree による枝刈りのおかげで，グラフデータベース内の $alive$ が付くノードの子孫にも $dead$ が付いていることを確認されたい．

5. 性能評価

本章では，実データ上の実験を行うことで，本手法の性能を評価し，布施らの手法を改善していることを示す．枝刈りを行う索引付けとして，HistogramTest のみによる C-Tree を用いた．また，VF2 [3] の逐次処理による部分グラフ問い合わせと比較し，本手法の有効性を確認する．実験環境は，Intel PC, 2.5GHZ, 8GBM 上で行う．OS は Windows Vista を用い，アルゴリズムは全て C++で実装した．

5.1 実データ上の実験

実際のグラフデータベースとて，AIDS Antiviral Screen のデータセットを利用する．このデータセットは約 43,000 個の化合物データから成り，NCI [8] により提供されている．実験に使用するグラフデータベースはこのデータセットからランダムにグラフを抽出して生成する.. C-Tree のパラメータは [6] と同様に，ノードにおける最小の子の数 $m = 20$，最大の子の数

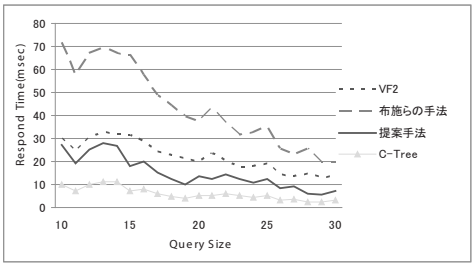

図 2　問い合わせのサイズと部分グラフ問い合わせの応答時間

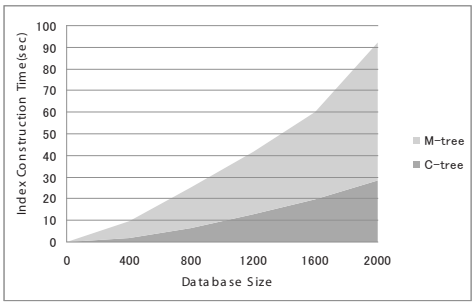

図 3　データベースサイズと 2 つの木の構築時間

を $M = 2m - 1$ とした．

図 2 は，問い合わせグラフのサイズに対する部分グラフ問い合わせの応答時間について，提案手法，布施らの手法，VF2 による逐次処理の比較を示したグラフである．データベースのサイズは 2000 で固定した．問い合わせグラフは AIDS からランダムに抽出したグラフの内，サイズが 10 から 30 であるグラフを使用し，その応答時間の平均をとった．すべての場合において，提案手法の性能が布施らの手法，VF2 による逐次処理の性能を上回っていることがわかる．しかし，この結果は提案手法における C-Tree を用いたグラフの枝刈りによるものであった．この実験では，提案手法における布施らの処理の部分は、枝刈りにより処理を減少しているにも関わらず、提案手法を遅くしている部分でしかなかった．

図 3 は，データベースサイズに対する C-Tree と M-Tree の構築時間を示している．この実験では逐次的にグラフの挿入処理を行うことで，2 つの木を構築している．しかし，C-Tree では，纏まったデータセットから木を構築する場合，階層的クラスタリングを用いることを推奨しており，それに比べて本実験の構築時間は長くなっている．枝刈りの手法として C-Tree を用いる場合，提案手法の構造の計算量は M-Tree に依存する．最悪の場合，M-Tree の構築時間，メモリ消費量はデータベースサイズに対して指数的に増える．

6. ま と め

本研究の目的は，グラフデータベース内で問い合わせグラフを部分グラフとしてもつグラフの検索を効率的に処理することである．我々は，既存の索引付けによるグラフの枝刈りを利用し，その後に布施らの手法を適用することで，一般的な部分グラフ問い合わせも効率的に処理できるように，彼らの手法を改良した．また，実データによる実験で，提案手法の性能が布施らの手法および VF2 による逐次処理の性能を上回っていることを確認した．しかし，提案手法の性能に貢献しているのは主に索引付けのの処理であり，布施らの処理の部分が有効なケースは少ないこともわかった．また，我々は Closure Tree の問題点を発見し，修正したところ，その性能は提案手法より優れていることが分かった．

本稿では，既存の索引付け手法として，Closure Tree を利用したが，提案手法は他の索引付けを利用することもできる．その場合，索引付けがより優れた正確な候補者集合を出力できるほど，提案手法が効果的と期待できる．

文　献

[1] J. Cheng, Y. Ke, W. Ng, and A. Lu. Fg-index: Towards verification-free query processing on graph databases. In *SIGMOD'07: Proceedings of the 2007 ACM SIGMOD international conference on Managemant of data*, pages 857–872. ACM, 2007.

[2] S. A. Cook. The complexity of theorem-proving procedures. In *Proc. of STOC '71: Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158, 1971.

[3] 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 Recoginition*, pages 149–159, May 2001.

[4] T. Fuse, K. Katayama, and H. Ishikawa. Retrieving graphs including an input graph from a large graph set. In *Proc. of Data Engeneering Workshop'06 (DEWS2006)*, March 2006.

[5] T. Fuse, S. Nishimura, and K. Kayayama. Enabling the messmer's graph decomposition approach for subgraph isomorphism detection to apply to a larger set of graphs. In *DMIN*, pages 397–403, 2007.

[6] H. He and A. K.Singh. Closure-tree: An index structure for graph queries. *ICDE'06, Atlanta, Georgia*, 2006.

[7] B. T. Messmer and H. Bunke. Efficient subgraph isomorphism detection: A decomposition approach. *IEEE Trans. Knowledge And Data Engineering*, 12:307–323, 2000.

[8] National Cancer Institute http://dtp.nci.nih.gov/.

[9] D. Shasha, J. T. L. Wang, and R. Giugno. Algorithmics and applications of tree and graph searching. In *Proc. of PODS*, pages 39–52, 2002.

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

[11] D. W. Williams, J. Huan, and W. Wang. Graph database indexing using structured graph decomposition. In *Proc of the 23rd IEEE International Conference on Data Engineering (ICDE)*, April 2007.

[12] X. Yan, P. S. Yu, and J. Han. Graph indexing: a frequent structure-based approach. In *SIGMOD '04: Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 335–346, New York, NY, USA, 2004. ACM.