大量グラフ集合からのある部分グラフを含むグラフの効率的な検索手法の提案

布施　貴義† 　片山　薫†† 　石川　博††

† 東京都立大学工学部　〒 192-0397　東京都八王子市南大沢 1–1
†† 首都大学東京システムデザイン学部　〒 192-0397　東京都八王子市南大沢 1–1
E-mail: †huse-takayosi@c.metro-u.ac.jp, ††{katayama,ishikawa}@eei.metro-u.ac.jp

あらまし　二つのグラフが与えられたとき，一方が他方に含まれるかどうかを判定する問題を部分グラフ同型判定問題という．部分グラフ同型判定は多様な応用を持つが，NP 完全であるため多量のグラフを扱うには膨大な計算コストがかかる．Messmer らは，入力グラフとそれよりも小さなグラフの集合との部分グラフ同型判定を，予めグラフ集合を分解しておくことで効率よく行う手法を提案している．Messmer らの手法では，あるグラフが入力グラフに含まれなければ，その親グラフも入力グラフには含まれないことを利用して無駄な部分グラフ同型判定を行わないようにしている．しかし，入力グラフがグラフ集合内の各グラフよりも小さい場合，入力グラフがあるグラフに含まれないからといってその親グラフにも含まれないとは限らないため，Messmer らの手法を直接利用することはできない．そこで本論文では，Messmer らの手法において入力グラフとグラフ集合内のグラフの大小関係を逆転した，入力グラフとそれよりも大きなグラフの集合との部分グラフ同型判定問題を想定し，それらをグラフ分解を用いて効率よく行う手法を提案する．さらに，代表的な部分グラフ同型判定アルゴリズムである VF2 と比較実験し，入力グラフの大きさがグラフ集合の各グラフの大きさに近い場合やグラフ集合内に入力グラフを含むグラフが多くある場合に提案手法が有効であることを確認した．

キーワード　グラフ分解，部分グラフ同型，情報検索

Retrieving Graphs Including an Input Graph from a Large Graph Set

Takayoshi FUSE†, Kaoru KATAYAMA††, and Hiroshi ISHIKAWA††

† Tokyo Metroporitan University, Engineering, Minamioosawa1–1, Hachioji-shi, Tokyo, 192–0397, Japan
†† Tokyo Metroporitan University, System Design, Minamioosawa1–1, Hachioji-shi, Tokyo, 192–0397, Japan
E-mail: †huse-takayosi@c.metro-u.ac.jp, ††{katayama,ishikawa}@eei.metro-u.ac.jp

Abstract　Deciding if a graph is a subgraph of another graph is called subgraph isomorphism problem. Subgraph isomorphism is applied in various area such as pattern recognition. It is computationally very expensive to detect subgraph isomorphism because this plobrem is NP-complete. Messmer et al. proposed the algorithm based on graph decomposition to detect each subgraph isomorphism from a large graph set to an input graph efficiently, but their algorithm can't solve a problem to detect each subgraph isomorphism from an input graph to a large graph set. We propose a new efficient algorithm for that problem. We compared our algorithm with typical subgraph isomorphism algorithm VF2, and show that our algorithm is more effective than VF2 when an input graph is included in many graphs in the graph set or an input graph size close to average graph size of the graph set.

Key words　graph decomposition, subgraph isomorphism, infomation retrieval

1. はじめに

二つのグラフが与えられたとき，一方が他方に含まれるかどうかを判定する問題を部分グラフ同型判定問題という．部分グラフ同型判定問題はグラフマイニング [4] やグラフの索引法 [10]，パターン認識，コンピュータビジョンなど，多量のグラフデータの検索が必要な分野で広く応用されるが，NP 完全であるため多量のグラフを扱うには膨大な計算コストがかかる．

よく知られた部分グラフ同型判定アルゴリズムとして，Ullmann [9] のアルゴリズムがある．Ullmann のアルゴリズムは

探索範囲を減少するための枝刈りを含む深さ優先探索によるアルゴリズムであり，今日広く利用されている．Ullmann のアルゴリズムでは頂点の次数や隣接する頂点との関係が枝刈りに利用されている．VF [1] も Ullmann のアルゴリズムと同じく深さ優先探索により部分グラフ同型判定を行う．VF では探索空間における各状態での部分的なマッチングとまだマッチングが見つかっていない頂点との間の関係を利用して枝刈りが行われる．VF は Ullmann のアルゴリズムと比べてより効率的に探索範囲を削減し，高速に動作する．また，VF のメモリ消費量を押さえ，複雑で大きなグラフを処理できるように改良された VF2 [2] が提案されている．Messmer [6] らは，入力グラフとそれよりも小さなグラフの集合との部分グラフ同型判定を，予めグラフ集合を分解しておくことで効率よく行う手法を提案している．西村ら [12] は，Messmer らの手法におけるグラフの分解時に非連結なグラフが生成されると計算コストが非常に増加するという問題を，常に連結なグラフを生成するようにすることで解決した．ただし，VF，VF2，Messmer らの手法，西村らの手法は全て，誘導部分グラフのみを対象としたアルゴリズムとなっている．

Messmer らの手法には，グラフ集合の各グラフよりも大きい入力グラフにしか対応していないという問題がある．そこで本論文では，Messmer らの手法において入力グラフとグラフ集合内のグラフの大小関係を逆転した，入力グラフとそれよりも大きなグラフの集合との部分グラフ同型判定問題を想定し，それらをグラフ分解を用いて効率よく行う手法を提案する．提案手法では，グラフを分解する際に，枝のカットによってグラフを分割するのではなく，2 つの連結な枝集合に分解することで，グラフ集合内の共有可能な部分をより多くとれるように改良した．また，Messmer らの手法は誘導部分グラフのみを対象としているが，提案手法は一般的な部分グラフを対象とする．

2. 関 連 研 究

グラフ集合から入力グラフを含むグラフを検索する問題は，グラフデータベースに対する問い合わせの一種と考えられる．グラフデータベースから問い合わせのグラフを含むグラフを検索する問題は，画像検索 [7] や知識発見 [3]，科学物質解析 [5] など様々な分野で応用されている．グラフデータベースに関する主な研究には，データベースの索引手法，問い合わせの最適化，グラフマイニングなどがある [8]．グラフデータベースの索引手法である gIndex [10] は，頻出部分グラフに索引を付けることで，グラフデータベースから問い合わせのグラフを含むグラフを高速に検索する．また，厳密な部分グラフではなく，問い合わせと類似した部分グラフを含むグラフの検索 [11] も研究されている．

3. 提案手法の概要

3.1 Messmer らの部分グラフ同型判定手法

グラフの集合の各グラフが入力として与えられたグラフに含まれているかどうかを全て調べる問題を考える．Messmer らは，グラフの分解を利用してグラフ集合内の共通部分を共有するデータ構造を作成しておくことで，この問題を効率良く解く方法を考案した．

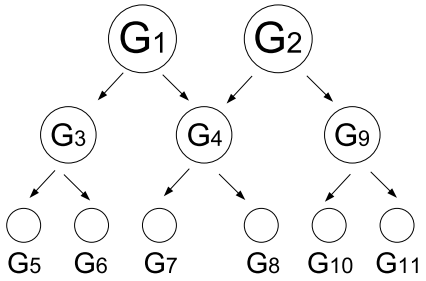

図 1　グラフ集合の分解

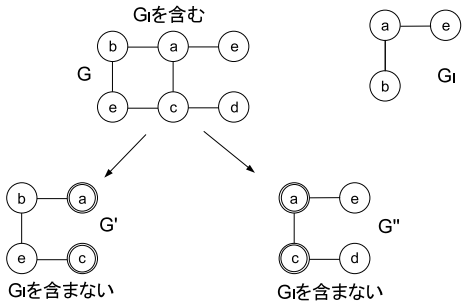

図 2　部分グラフに入力グラフは含まれないが，
親グラフには入力グラフが含まれる分解データ

図 1 にグラフ集合 G_1, G_2 の分解から得られるデータ構造の例を示す．図 1 のデータ構造において，あるグラフとそのグラフを分解して得られる二つの部分グラフからなるデータ構造を分解データと呼ぶことにする（より詳しい定義を 4.1 で与える）．例えば，G_1，G_3，G_4 は一つの分解データである．G_1 と G_2 は，共通の部分グラフである G_4 を共有している．もし入力グラフに G_4 が含まれないならば，部分グラフ同型判定を行うことなく G_1 と G_2 も入力グラフに含まれないと判断できる．

このように，あらかじめグラフ集合を分解しておくことで，無駄な部分グラフ同型判定の回数を減らすことができる．

3.2 想定する問題と我々のアプローチ

グラフの集合 $G_1, ..., G_n$ があるとき，グラフ G_I が $G_1, ..., G_n$ のそれぞれのグラフに含まれるかどうかを全て判定する問題を考える．$G_1, ..., G_n$ をそれぞれモデルグラフ，G_I を入力グラフとする．

単純な方法として，なんらかの部分グラフ同型判定アルゴリズムを用いて各モデルグラフに入力グラフが含まれるかどうかを順次調べていくというものがある．しかし，この方法には処理時間がモデルグラフ集合の大きさに比例して増大する欠点がある．さらに，各モデルグラフに共通部分がある場合，それらの共通部分と入力グラフのマッチングが繰り返し行われることになり，無駄が多い．

Messmer らの手法より，あらかじめモデルグラフ集合を分解し，共通部分を共有するデータ構造を作成しておくことで，共通部分が繰り返し比較されることを防ぐことができる．しかし，この問題に対して Messmer らの手法を直接適用することはできないことを以下に述べる．

Messmer らの手法では，3.1 で示したように，

- あるグラフが入力グラフに含まれていなければ，その親グラフもまた入力グラフには含まれない

ということを無駄な部分グラフ同型判定の回避に利用する．しかし，想定する問題では入力グラフの方がモデルグラフより小さいため，Messmer らの手法における無駄な部分グラフ同型判定の回避方法では入力グラフがあるグラフに含まれるかどうかを判定することはできない．さらに，Messmer らの手法における入力グラフと分解データ構造内のあるグラフとの関係を単純に逆転した，あるグラフが入力グラフに含まれていなければ，その親グラフもまた入力グラフには含まれないという判定方法では，正しい解が得られない．図 2 に示される分解データの例では，G_I は G' にも G'' にも含まれていないが，G'，G'' の親である G には含まれている．

そこで，我々は部分グラフ同型判定の回避方法として

- あるグラフに入力グラフが含まれているならば，その親グラフにもまた入力グラフは含まれる

という判定方法を提案する．ある分解データの部分グラフのどちらか一方にでも入力グラフが含まれている場合には，親グラフとの部分グラフ同型判定を回避することが出来る．図 2 のように分解データの部分グラフの両方とも入力グラフを含まない場合は，入力グラフと親グラフとの部分グラフ同型判定を行う．ただし，このような場合であっても分解データの情報を用いていくつかの枝刈りを行い，無駄な部分グラフ同型判定を回避することが出来る．分解データを利用した部分グラフ同型判定アルゴリズムの詳細は 4.3 で述べる．

3.3 提案手法と Messmer らの手法の違い

ここで，対象とする問題と部分グラフ同型判定の回避方法以外での提案手法と Messmer らの手法との違いをまとめておく．

モデルグラフ集合の分解において，Messmer らはグラフの連結性を考慮せず，単純に枝をカットして頂点の集合とそれに付随する枝という形で新たな部分グラフを作成する．図 3 に Messmer らの手法におけるグラフの分解を示す．破線の通っている枝をカットし，二つの部分グラフを生成する．枝がカットされた結果，片方の部分グラフが非連結になっている．これに対して，我々はグラフを分解する際，枝をカットしてグラフを分割するのではなく，グラフを二つの連結な枝集合に分けることで分解する．図 4 に，提案手法における図 3 と同じグラフの分解を示す．提案手法では，元のグラフから枝を振り分けて二つのグラフを生成するため，いくつかの頂点は生成された二つのグラフ間で共有される．共有頂点は部分グラフ同型判定アルゴリズムの中で枝刈りに利用される．また，グラフ G からの部分グラフ S_{max} の減算において，減算結果 S_{rest} が連結でない場合，提案手法では S_{rest} が連結になるように S_{max} と S_{rest} で枝を共有する．これらの技法により，分解データ集合内での共通部分の共有数を増やし，無駄な部分グラフ同型判定の回数を減らす．

部分グラフ同型判定において，Messmer らの手法では入力グラフがモデルグラフより大きいため，分解データ集合内の各グラフを初期化する際，すべてのグラフを処理の必要ありとして

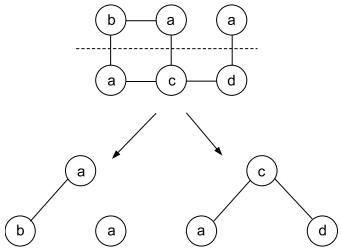

図 3　Messmer らの手法におけるグラフの分解

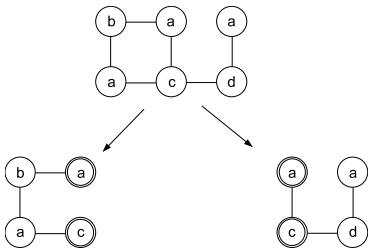

図 4　提案手法におけるグラフの分解

初期化する．提案手法ではモデルグラフよりも入力グラフのほうが小さいので，分解データ集合内のグラフが入力グラフより小さい場合にはあらかじめ処理をする必要がないことがわかり，調べる分解データの数を減らすことができる．また，分解データの二つの部分グラフがどちらも alive でなかったとき，部分グラフ同型判定を行う前に二つの部分グラフの共有頂点を利用した簡単な枝刈りを行うことで，無駄な判定を避ける．このように，提案手法ではモデル集合の分解，部分グラフ同型判定の二つのアルゴリズムの各所に無駄な部分グラフ同型判定を避けるための改良を加えている．

4. アルゴリズムの詳細

提案手法はモデルグラフ集合の分解と，分解によって得られたデータ構造を利用しての入力グラフとモデルグラフ集合との部分グラフ同型判定という，二つのアルゴリズムからなる．まず基礎的な定義と表記を与え，つづいて提案手法における二つのアルゴリズムそれぞれについて詳細を述べる．

4.1 定　　義

提案手法はラベル付きグラフを対象とする．L_V，L_E をそれぞれ頂点，枝のラベルの集合とする．

定義 1　ラベル付きグラフ G は，4 つの要素 $G = (V, E, \mu, \nu)$ からなる．V は頂点の集合，$E \subseteqq V \times V$ は枝の集合である．$\mu : V \to L_V$，$\nu : E \to L_E$ はそれぞれ頂点，枝にラベルを割り当てる関数である．

以降，ラベル付きグラフを単にグラフと呼ぶ．

定義 2　以下の条件を満たすとき，かつそのときに限り $G' = (V', E', \mu', \nu')$ は $G = (V, E, \mu, \nu)$ の部分グラフである．また，G' が G の部分グラフであるとき，G は G' の親グラフであるという．

- $V' \subseteqq V$
- $\forall v \in V', (\mu(v) = \mu'(v))$
- $E' \subseteqq E \wedge (v_1, v_2) \subseteqq E' \to v_1, v_2 \subseteqq V'$

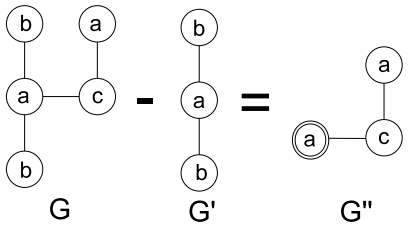

図 5　グラフの差分

Decomposition(B)
1. $B = \{G_1, ..., G_n\}$，$D = \phi$ とする．
2. for $i = 1$ to n
3.　　Decompose(G_i, D)

図 6　モデルグラフ集合分解アルゴリズム

- $\forall e \in E', (\nu(e) = \nu'(e))$

G' が G の部分グラフであることを，$G' \subseteqq G$ と表記する．特に，$E' \subset E$ であるとき，$G' \subset G$ と表記する．

定義 3　グラフ $G = (V, E, \mu, \nu)$ とその部分グラフ $G' = (V', E', \mu', \nu')$ が与えられたとき，G と G' の差分を $E - E'$ の枝の集合からなる G の部分グラフとし，$G - G'$ と表記する．

図 5 にグラフの差分の例を示す．各頂点内のアルファベットはラベルである．G から G' を減算した結果，二重線の頂点は G' と G'' で共有されている．

定義 4　以下の条件を満たすとき，全単射 $f : V \to V'$ はグラフ $G = (V, E, \mu, \nu)$ から $G' = (V', E', \mu', \nu')$ へのグラフ同型である．

- $\forall v \in V, (\mu(v) = \mu'(f(v)))$
- $\forall u, v \in V, ((u, v) \in E \Leftrightarrow (f(u), f(v)) \in E')$
- $\forall (u, v) \in E, (\nu(u, v) = \nu'(f(u), f(v)))$

定義 5　$f : V \to V'$ がグラフ G からグラフ G' の部分グラフ G'' へのグラフ同型であるとき，G から G' への単射 f を G から G' への部分グラフ同型とする．

定義 6　分解データ $T = (G, G', G'', N)$ は，グラフの分解によって得られるデータ構造であり，

- $G' \subset G \wedge G'' \subset G$
- $N = V' \cap V''$
- $V = V' \cup V'' - N$
- $E = E' \cup E'' - E' \cap E''$

定義 7　$B = \{G_1, ..., G_n\}$ をモデルグラフ集合とする．B の分解データの集合を D とする．

4.2　モデルグラフ集合の分解

図 6 に，モデルグラフ集合の分解のアルゴリズム Decomposition を示す．

Decomposition は，モデルグラフ集合内の各グラフに対して，図 7 のグラフ分解の手続き Decompose を呼び出す．

Decompose は，与えられたグラフ G を枝一つからなるグラフになるまで再帰的に分解する．まず，G が枝一つからなるか調べ，そうであれば G を返して処理を終える．次に，それまでに生成した分解データの集合 D から，G の部分グラフのうち

Decompose(G,D)
1. $S_{max} = \phi$，S_{rest} とする．
2. G が唯一つの枝からなるならば，G を返す．
3. for each $T = (G_i, G_i', G_i'', N_i) \in D$
4.　　$G_i \subseteqq G$ かつ S_{max} が G_i より小さいならば，S_{max}=G_i
5. S_{max} が G と同型ならば，S_{max} を返す．
6. $S_{max} = \phi$ ならば
7.　　(S_{max}, S_{rest})=Partition(G)
8.　　S_{max} =Decompose(S_{max},D)
9. そうでなければ
10.　　$S_{rest} = G - S_{max}$
11.　　S_{rest} が連結で無いならば
12.　　S_{rest} が連結になるように
　　S_{max} といくつかの枝を共有する．
13. S_{rest} =Decompose(S_{rest},D)
14. N=S_{max} と S_{rest} が共有する頂点
15. D に $(G, S_{max}, S_{rest}, N)$ を加える．
16. G を返す．

図 7　グラフ分解手続き

最大のものを調べ，S_{max} とする．このとき，引数として与えられたグラフと D 内のグラフとの部分グラフ同型判定を行う必要がある．提案手法では，VF2 を誘導部分グラフだけでなく一般の部分グラフ同型判定を行えるように拡張したものを使用した．S_{max} が G と同型の場合，G の分解データはすでに D に存在するので，S_{max} を返して処理を終える．S_{max} が発見できた場合，G と S_{max} の差分を S_{rest} とし，必要であれば連結になるように S_{max} と枝を共有する．こうすることで，D におけるグラフの共有数を増やすことができる．S_{max} が発見できなかったなら，G に対して図 8 に示す手続き Partition を呼び出して G を二つの連結な枝集合に分解する．分解結果を S_{max}，S_{rest} に代入し，S_{max} に対して Decompose を呼び出し，分解する．続いて，残った S_{rest} に対して Decompose を呼び出し，分解する．最後に，以上の処理から得られた新たな分解データを D に加え，G 自身を返して処理を終える．Decompose が返すグラフは，引数として与えられたグラフ G または D 内の G と同型なグラフである．

Partition は，与えられたグラフ G を連結な二つの枝集合に分解する．はじめに，G の枝を空のグラフ S_1，S_2 に半分ずつ割り振る．その結果，S_1，S_2 はいくつかの連結成分を持つ G

Partition(G)
1. $S_1 = \phi$，$S_1 = \phi$ とする．
2. G の枝を半分ずつ，ランダムに S_1，S_2 に振り分ける．
3. while S_1 が連結でないまたは S_2 が連結でない
4.　　S_1，S_2 がともに連結でないならば
5.　　枝を多く持つ方の最小の連結成分を他方に移動する．
6.　　そうでなければ
7.　　連結でない方の最小の連結成分を他方に移動する．
8. (S_1, S_2) を返す．

図 8　グラフを二つの連結な枝集合に分解する手続き

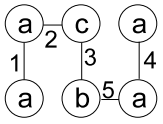

図 9　分解元のグラフ

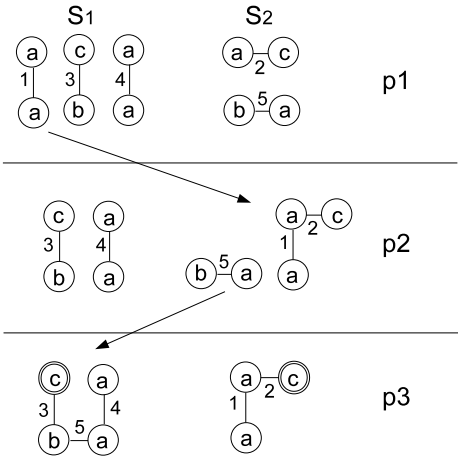

図 10　枝の振り分けによるグラフの分解

SearchSubgraph(G_I,D)

1. G は入力グラフ，D は分解データ集合である．
2. for each $T = (G_i, G'_i, G''_i, N_i) \in D$
3. 　G_i の頂点数もしくは枝数が G_I よりも少なければ
4. 　　G_i に dead をマークする．
5. 　そうでなければ
6. 　　G_i に unsolved をマークする．
7. for each $T = (G_i, G'_i, G''_i, N_i) \in D$
8. 　G_i が unsolved ならば
9. 　　G'_i が alive または G''_i が alive ならば
10. 　　G_i に alive をマークする．
11. 　　そうでなければ
12. 　　G_I に，N_i 内の頂点のどれかと同じラベルで次数が等しいか少ない頂点が一つでもあるならば
13. 　　G_I と G_i の部分グラフ同型判定をする．
14. 　　$G_I \subseteqq G_i$ ならば
15. 　　G_i に alive をマークする．
16. 　　そうでなければ
17. 　　G_i に dead をマークする．

図 11　部分グラフ同型判定アルゴリズム

の部分グラフとなる．続いて，S_1，$S2$ がともに連結なグラフとなるまで最小の連結成分を移動していく．最小の連結成分を移動するのは，S_1 と S_2 の枝数をなるべく平衡にするためである．

例として，図 9 のグラフを分解する様子を図 10 に示す．図 9，10 において，枝の近くにある数字は枝の ID である．図 10 の p1 が，二つのグラフ S_1，S_2 に図 9 のグラフの枝がランダムに割り振られた状態である．S_1，S_2 はともに非連結であり，S_1 の方が枝を多くもつので，G_1 の連結成分のうち最小のものとして枝 1 を選び，S_2 に渡す．p2 において，S_1，S_2 はともに非連結であり，S_2 の方が枝を多く持つので，S_2 の連結成分のうち最小のものである枝 5 を S_1 に渡す．p3 において，S_1，S_2 はともに連結なので，Partition は S_1，S_2 を返して処理を終える．二重線の頂点は S_1 と S_2 で共有される頂点である．

4.3　部分グラフ同型判定のアルゴリズム

分解データ $T = (G, G', G'', N)$ とグラフ G_I があるとき，もし $G_I \subseteqq G'$ または $G_I \subseteqq G''$ ならば，G_I と G の部分グラフ同型判定をせずとも $G_I \subseteqq G$ であることがわかる．部分グラフ同型判定のアルゴリズムは，このことを利用して部分グラフ同型判定の回数を減らすことで，効率よく分解データ集合の中から部分グラフを発見する．

図 11 に入力グラフからモデルグラフ集合への同型判定のアルゴリズムを示す．分解データ集合 D 内の各グラフは，入力グラフを部分グラフとして持つことを示す alive，そうでないことを示す dead，まだ入力グラフとの部分グラフ同型判定が調べられていないことを示す unsolved のいずれかのマークを持つ．

まず，分解データ集合 D を初期化する．初期化では，D 内のすべてのグラフに対して，入力グラフ G_I より小さければ dead，そうでなければ unsolved をマークする．初期化が終わったら，D 内の各 $T = (G_i, G'_i, G''_i, N_i)$ に対して，入力グラフとの部分グラフ同型判定の処理を行う．G_I が G'_i にも G''_i にも含まれない場合，G_I も G_i に含まれていないとは限らないので，G_I と G_iの 部分グラフ同型判定を行う必要がある．このとき，G_I が G'_i と G''_i の共有頂点を一つも含まないならば G_i に G_I が含まれることはないということを利用して，無駄な部分グラフ同型判定を行わないようにする．同様に，部分グラフの各頂点の次数は対応する親グラフの頂点の次数以下でなければならないということも判定の回数を減らすために利用する．枝刈りの例として，共有頂点を利用したものを図 12 に示す．二重線の頂点は G' と G'' の共有頂点である．入力グラフ G_I は G'，G'' に含まれないことはわかっている．G_I と G との部分グラフ同型判定をする前に，G_I が G' と G'' の共有頂点を含むかどうかを調べる．G_I はラベルが a である頂点も c である頂点も持たないので，G_I は G に含まれないことがわかる．

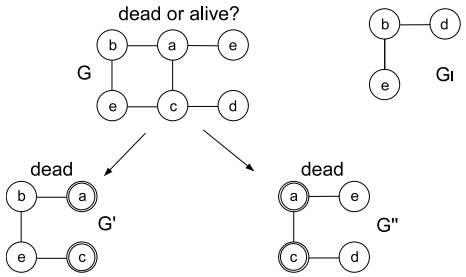

図 12　共有頂点を利用した枝刈り

4.4　モデルグラフ集合の分解と部分グラフ同型判定の例

4.4.1　モデルグラフ集合の分解の例

モデルグラフ集合 $B = \{G_1, G_2\}$ の分解を図 13 に示す．簡単のため，枝のラベルは省略してある．各グラフの近くにある G_i はグラフの ID，T_j は分解データの ID である．グラフのインデクス i はグラフの生成される順番，分解データ T_j のイン

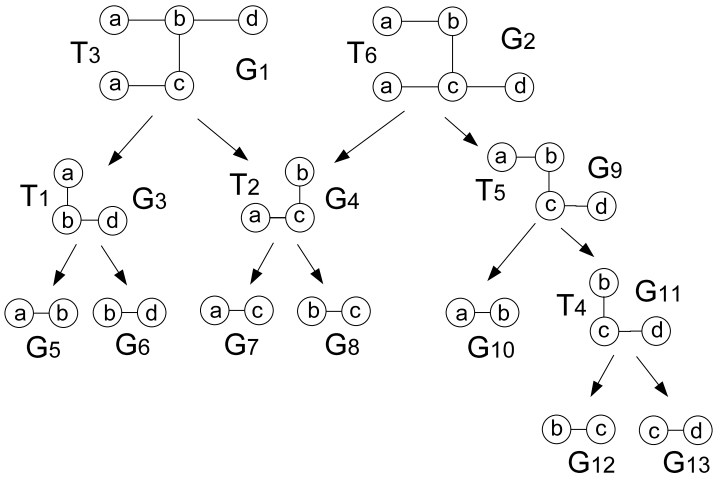

図 13　モデルグラフ集合の分解

デクス j は分解データ集合に加えられる順番に付けられていることに注意する必要がある．図 13 におけるモデルグラフ集合の分解の手順を以下に述べる．

B に対して Decomposition が呼び出されると，まず G_1 に対して Decompose が呼ばれる．まだ D には分解データはないので，G_1 は二つの連結な部分グラフ G_3 と G_4 に分解され，G_3 対して Decompose が呼ばれる．同様に分解を繰り返され，分解した二つのグラフが両方とも枝一つのグラフ（G_5，G_6）になると分解は止まり，最初の分解データである T_1 が D に加えられる．続いて T_2，T_3 が順に D に加えられていき，G_3 のすべての分解データが加えられた後に G_4 に対して Decompose が呼ばれる．このように，あるグラフ G_i の分解データはその部分グラフの分解データが D に加えられた後に，D に加えられる．次に，G_2 に対して Decompose が呼び出される．ここまでの $D = \{T_1, T_2, T_3\}$ において，G_2 の最大の部分グラフは G_4 であるため，G_2 から G_4 が減算される．しかし，減算結果 $G_2 - G_4$ は非連結なグラフであるので，ラベル b の頂点とラベル c の頂点の間の枝を G_4 と共有し，連結なグラフである G_9 が生成される．後は，それまでの分解と同様に枝一つからなるグラフになるまで再帰的に分解され，小さいグラフの分解データから順に D に加えられていく．

4.4.2　部分グラフ同型判定の例

図 14 に示す入力グラフ G_I と 4.4.1 で得られた分解データ集合 $D = \{T_1, T_2, T_3, T_4, T_5, T_6\}$ との部分グラフ同型判定を例に，図 11 の部分グラフ同型判定アルゴリズムの手順を説明する．

まず，D 内の各グラフのマークを初期化する．G_I より小さい G_5〜G_8，G_{10}，G_{12}，G_{13} に dead をマークし，残りには

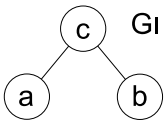

図 14　入力グラフ G_I

表 1　実験に使用したグラフデータのパラメータ

実験	モデルグラフ集合				入力グラフ集合	
図表番号	D	T	i	p	D	T
図 15	100	100	50%	-	100	*
図 16	100	100	-	*	100	30
図 17	100	100	70%	-	100	*
図 18	100	*	50%	-	-	-
図 19	*	50	50%	-	-	-

unsolved をマークする．

次に，G_I が unsolved なグラフに含まれるかどうかを，分解データを利用して調べていく．G_I は G_3 の部分グラフではないので，G_3 に dead をマークする．同様に G_I と G_4 の部分グラフ同型判定を行い，$G_I \subseteqq G_4$ なので G_4 に alive をマークする．G_4 が alive なので，部分グラフ同型判定を行うことなく $G_I \subseteqq G_1$ であることがわかり，G_1 に alive をマークする．続いて G_{11}，G_9 との部分グラフ同型判定を調べるが，G_2 については $G_I \subseteqq G_4$ であることから $G_I \subseteqq G_2$ であることが直ちにわかり，G_2 に alive をマークする．モデルグラフ集合のうち，alive をマークされたグラフが G_I を部分グラフとして持つグラフであり，この場合は G_1，G_2 である．

5.　評 価 実 験

5.1　実 験 方 法

想定する問題に対して，VF2 を用いて入力グラフからモデルグラフ集合の各グラフへの部分グラフ同型判定を順次行った場合と，我々のアルゴリズムを用いた場合の処理速度を測定した．実験に使用したモデルグラフ集合と入力グラフは，倉持らの開発したグラフデータ生成ソフトウェアで作成した．実験に使用したグラフデータのパラメータを表 1 に示す．各パラメータの意味は，

- D　グラフ集合の大きさ
- T　グラフ集合内の一つのグラフの平均枝数
- i　モデルグラフ集合に含まれる共通の部分グラフの枝数

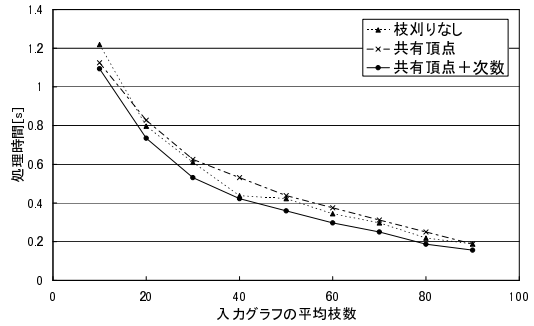

図 15　枝刈りの使用と
部分グラフ同型判定の処理時間

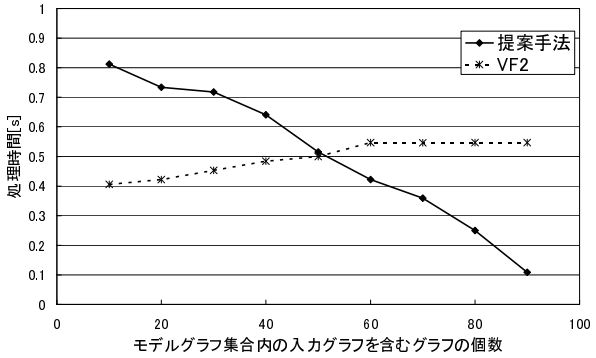

図 16　モデルグラフ集合内の入力グラフを含むグラフの個数と
部分グラフ同型判定の処理時間

の割合

● p　モデルグラフ集合内の入力グラフを含むグラフの数

である．表の中で，“ * ”になっているマスが該当する実験におけるパラメータである．“ - ”になっているマスは該当する実験では使用しなかったパラメータである．実験に使用した全てのグラフデータについて，頂点のラベルは 10 個，枝はラベル無しとして作成した．枝のラベル無しとは，全ての枝に同じラベルが付けられていることと等しい．グラフ集合内のグラフをランダムに生成した場合，ラベルの数が増えるほど提案手法ではグラフの共有がされにくくなるために処理速度が落ちる．しかし，実験に使用したグラフ集合は常に共通部分を持つように生成されているため，ラベルの数が実験に与える影響はほぼ無い．

5.2 実験環境

Pentium4 2.2GHz，1024MBRAM，Windows XP proffesional を搭載したワークステーション上で各実験を行った．提案手法の測定には，C++で実装し cl7.1 でコンパイルしたプログラムを使用した．VF2 については，公開されているライブラリを誘導部分グラフだけでなく一般の部分グラフ同型判定を行えるように拡張し，cl7.1 でコンパイルしたものを使用した．

5.3 実験結果

図 15 は，4.3 で述べた枝刈りを行った場合と行わない場合での提案手法の性能の違いを，入力グラフ集合の平均枝数をパラメータとして測定した結果である．“ 共有頂点 + 次数 ”の測定結果は，共有頂点と入力グラフの頂点との整合性を調べる際に，ラベルだけでなく次数の整合性も調べた場合の処理時間で

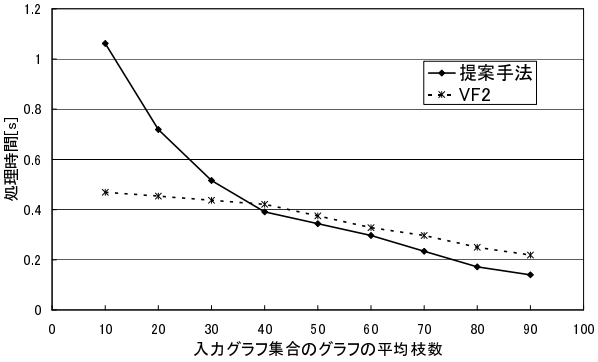

図 17　モデルグラフ集合内のグラフの平均枝数と
部分グラフ同型判定の処理時間

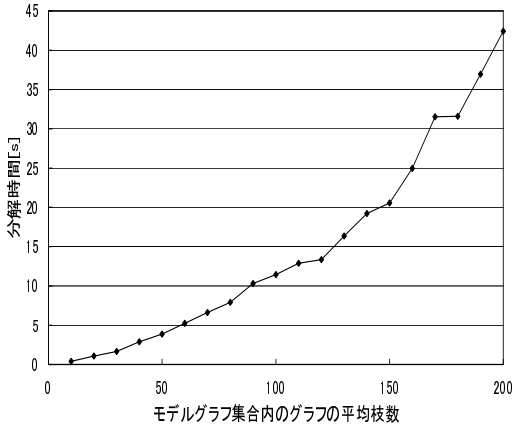

図 18　モデルグラフ集合内のグラフの平均枝数と
分解にかかった時間

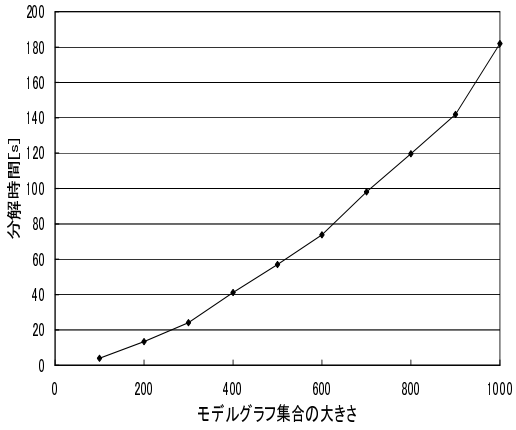

図 19　モデルグラフ集合の大きさと分解にかかった時間

ある．共有頂点による枝刈りだけでは枝刈りを行わない場合よりもかえって処理に時間がかかったが，共有頂点による枝刈りに次数での枝刈りを組み合わせた場合には枝刈りを行わない場合よりも平均で 12%，最大で 17% ほど高速に動作した．

以降の実験に使用した提案手法には共有頂点と頂点の次数による枝刈りを実装してある．

図 16 は，モデルグラフ集合内の入力グラフを含むグラフの個数を変化させた場合に，部分グラフ同型判定にかかる時間が

どのように変化するかを測定した結果である．モデルグラフ集合の平均枝数を 100 本，入力グラフ集合の平均枝数を 30 本に固定して測定した．

モデルグラフ集合内に入力グラフを含むグラフが多くなるにつれ，提案手法の部分グラフ同型判定にかかる時間は急速に減少していく．一方，VF2 での判定は単純に入力グラフと各モデルグラフとの部分グラフ同型判定を VF2 を用いて順次行うだけなので，入力グラフを含むモデルグラフの数の変化によらずほぼ一定となっている．モデルグラフ集合内に入力グラフを含むグラフが少ない場合，分解データ集合内での入力グラフを含むグラフの共有が少なくなる．そのため，分解データ集合内の多くのグラフと部分グラフ同型判定を行う必要が生じ，線形探索よりも処理速度が落ちたと考えられる．逆に，モデルグラフ集合内に入力グラフを含むグラフが多いと，分解データ集合内で入力グラフを含むグラフが多数共有されることになり，提案手法が有効になる．

図 17 は，入力グラフの大きさを変化させた場合における部分グラフ同型判定にかかった時間を測定したものである．入力グラフが大きくなるに従って提案手法，VF2 ともに高速になっていくが，提案手法はより高速化の度合いが強い．入力グラフ集合の平均枝数 40 本の段階で VF2 よりも高速になり，その後は差が広がっていく．入力グラフは分解データ集合内の小さなグラフから順に部分グラフ同型判定されるため，入力グラフがモデルグラフに対して小さい場合は判定回数が増加し，処理に時間がかかったと考えられる．

図 18 にモデルグラフ集合内のグラフの平均枝数とモデルグラフ集合の分解にかかる時間との関係を示す．図 19 にモデルグラフ集合の大きさとモデルグラフ集合の分解にかかる時間との関係を示す．どちらも，パラメータの増加に従って分解にかかる時間は曲線を描いて増加している．これは，グラフの分解によって蓄積された分解データの走査にかかる時間が分解にかかる時間に上乗せされていくためだと考えられる．

5.4 考　　察

実験結果から，提案手法はモデルグラフ集合内の多くのグラフが入力グラフを含む場合，分解データ集合の特性を生かすことができるため特に有効である．逆に，モデルグラフ集合内の多くのグラフが入力グラフを含まない場合，提案手法では分解データ集合における部分グラフ同型判定がオーバーヘッドとなり，単純な順次比較の方が効率的となる．提案手法が有効となる応用として，似た画像の集合からの問い合わせ画像を含む画像の検索などが考えられる．

6. おわりに

入力グラフとそれよりも大きいグラフの集合との部分グラフ同型判定を行う問題について，あらかじめグラフ集合を分解し，共通部分を共有するデータ構造を作成しておくことで，無駄な部分グラフ同型判定をせずに部分グラフを発見するアルゴリズムを提案した．グラフを分解する際に，分解した二つのグラフの間で頂点や枝を共有することで，共通部分の共有数を増やし，より効率的に部分グラフ同型判定を行えるようにした．代表的な部分グラフ同型判定アルゴリズムである VF2 との比較実験を行い，入力グラフの大きさがグラフ集合の各グラフの大きさに近い場合，グラフ集合内に入力グラフを含むグラフが多くある場合に提案手法が有効であるという結果が得られた．

今後の課題としては，提案手法のなかでボトルネックとなっている部分の発見と高速化，アルゴリズムの改良，計算量による評価がある．特に，入力グラフが分解データの部分グラフのいずれにも含まれていない場合の処理が課題である．また，モデルグラフ集合の分解によって得られた分解データ集合を頻出部分グラフマイニングに利用することを考えている．

謝辞　実験用グラフデータ生成ソフトウェアを提供頂いたミネソタ大学倉持道広氏に深く感謝致します．多くの貴重なコメントを頂いた査読者の方々に深く感謝致します．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 (B)(2)(課題番号:16300030) による．

文　　献

[1] L.P. Cordella, P. Foggia, C. Sansone, and M. Vento, “Performance evaluation for the vf graph matching algorithm,” Proc. of the 10th International Conference on Image Analysis and Processing, pp.1172–1177, Los Alamitos, USA, Sept. 1999.

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

[3] S. Djoko, D.J. Cook, and L.B. Holder, “An empirical study of domain knowledge and its benefits to substructure discovery,” IEEE Trans. on Knowledge and Data Engineering, vol.9, no.4, 1997.

[4] J. Huan, W. Wang, J. Prins, and J. Yang, “Spin: Mining maximal frequent subgraphs from graph databases,” Proc. of the 10th ACM SIGKDD international conference on Knowledge discovery and data mining, pp.581–586, Seattle, WA, USA, Aug. 2004.

[5] C.A. James, D. Weininger, and J. Delany, Daylight theory manual daylight version 4.9. Daylight Chemical Information Systems, Inc., 2004.

[6] B.T. Messmer and H. Bunke, “Efficient subgraph isomorphism detection: A decomposition approach,” IEEE Trans. Knouledge And Data Engineering, vol.12, pp.307–323, 2000.

[7] E.G.M. Petrakis and C. Faloutsos, “Similarity searching in medical image database,” Knowlege and Data Engineering, vol.9, pp.435–447, 2002.

[8] D. Shasha, J.T.L. Wang, and R. Giugno, “Algorithmics and applications of tree and graph searching,” Proc. 21th ACM Symp. Principles of Database Systems, pp.39–52, June 2002.

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

[10] X. Yan, P.S. Yu, and J. Han, “Graph indexing: A frequent structure-based approach,” Proc. of 2004 Int. Conf. on Management of Data (SIGMOD’04), pp.335–346, Paris, France, June 2004.

[11] X. Yan, P.S. Yu, and J. Han, “Substracture similarity search in graph database,” Proc. 2005 ACM-SIGMOD Int. Conf. on Management of Data (SIGMOD’05), Baltimore, Maryland, 2005.

[12] 西村将太郎, 片山薫, 石川博, “Messmer らのグラフ分解を利用した部分グラフ同型発見手法,” 電子情報通信学会 DE 研究会第 16 回データ工学ワークショップ（DEWS2005）, March 2005.