再帰的な処理によるグラフ分割を用いた部分グラフ問い合わせ処理の効率化

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
E-mail: †kenjiro.seki@gmail.com, ††katayama@eei.metro-u.ac.jp

あらまし　近年，グラフは複雑な構造を表現し，理解するためにますます重要な構造となっている．それに伴い，グラフデータベースからのグラフの検索が必要とされている．しかし，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，それに係るグラフの検索も高コストである．ゆえに，グラフ検索手法の効率化に対して様々な研究が成されてきた．Messmer らはグラフを再帰的に分解して構築される構造を用いて，グラフデータベースから問い合わせグラフの誘導部分グラフ同型を効率的に検出する手法を提案した．本稿では，Messmer らの問い合わせ処理を再帰的にすることで，処理の効率化，単純化を行う．実験では，化合物データベースおよび人工データを用いて，提案手法と Messmer らの手法，一対一の誘導部分グラフ同型検出による逐次処理を比較し，提案手法の有効性を示す．

キーワード　部分グラフ同型判定，グラフデータベース，データ構造

1. はじめに

近年，グラフは複雑な構造を表現すつための重要なデータモデルとなっている．情報学の分野においては，パターン認識やコンピュータービジョン等に利用されている．化学，生物学の分野では，分子構造や DNA 等がグラフとして表される．その他，Web や XML 文書，ソーシャルネットワーク等，その応用は多岐にわたる．ゆえに，グラフデータベースから条件に合ったグラフを効率的に検索することが必要とされている．一方，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いる [1]．したがって，部分グラフ同型に係るグラフの検索も，膨大なコストを必要とする．これらの問題に対し，数多くの研究が成されてきた．

Messmer ら [2] は，グラフデータベースの各グラフから問い合わせグラフへの誘導部分グラフ同型の検出（誘導部分グラフ同型問い合わせ）を効率的に行う手法を提案した（以下，検索・検出の対象となるグラフデータベースをモデルグラフ集合とする）．誘導部分グラフ同型問い合わせは，モデルグラフ集合からの問い合わせグラフの誘導部分グラフの検索（誘導部分グラフ問い合わせ）を成す．図 1 に誘導部分グラフ同型問い合わせを示す．図 1 では，各グラフの各頂点には左上に ID が付されている．ID により，同じラベルが付された頂点も区別される．簡単のため，枝ラベルについては考慮しない．例えば，問い合わせグラフから g_1 への誘導部分グラフ同型は複数存在するが，誘導部分グラフの検索であれば，1 つの誘導部分グラフ同型を確認すれば足りる．比較すれば，誘導部分グラフ同型問い合わせがすべての誘導部分グラフ同型を求める問題であるのに対し，誘導部分グラフ問い合わせは誘導部分グラフ同型が少なくとも 1 つは存在するか否かを求める問題である．

Messmer らの手法では，モデルグラフ集合内の各グラフを再帰的に分解してできるデータ構造を予め構築する．構築の際，各グラフ間の共通部分がデータ構造内で共有される．このデータ構造を利用して誘導部分グラフ同型問い合わせを処理する．

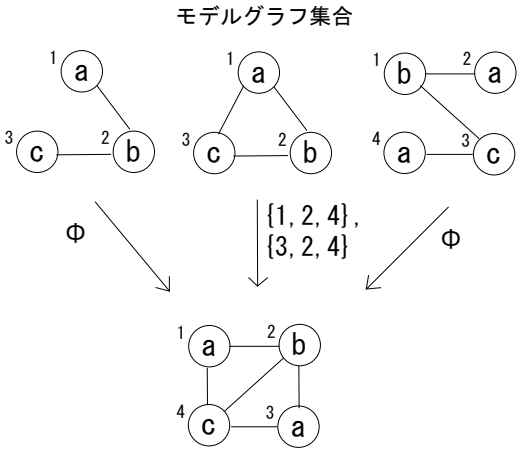

図 1　誘導部分グラフ同型問い合わせ

Messmer らの手法は，下位のグラフと問い合わせグラフとの誘導部分グラフ同型から，上位のグラフと問い合わせグラフとの誘導部分グラフ同型を検出し，データ構造内の全てのグラフをボトムアップに処理する．しかし，問い合わせによっては，データ構造内の一部のグラフについての検出で解が得られる．本稿では，Messmer らの問い合わせ処理を再帰的な処理とすることで，この処理の冗長を排除する．また，再帰化により，局所的な問い合わせへの対応，手法の拡張性の向上の効果を伴う．

布施ら [3] は，Messmer らの手法に対し，分解後のグラフを連結グラフに制限することによるコストの抑制，問い合わせグラフの部分グラフとなるグラフの検索への拡張を提案したが，

本手法による改善はこれらの改善とは独立している．

2. 関連研究

部分グラフ同型判定に係るグラフの検索は，2 つのカテゴリーに分類される．

1 つ目は，問い合わせグラフを部分グラフとして含むグラフの検索であり，グラフの索引付けとして，多くの研究が成されている．Shasha ら [4] は，モデルグラフ集合内の各グラフのパスを特徴として，索引付けを行う GraphGrep を提案した．GraphGrep は索引と問い合わせグラフの関係から，答えとなる候補のグラフを絞ることで，問い合わせ処理を効率化する．Yan ら [5] の提案する gIndex は，パスの代わりに頻出部分グラフ集合から冗長なグラフを除いた集合を特徴とした索引付けをすることで，索引により得られる候補グラフ集合の精度を高めた．Chen ら [6] による FG-index も頻出部分グラフ集合を特徴とした索引付けを行う．FG-index は，問い合わせグラフが頻出部分グラフ集合に含まれる場合は，部分グラフ同型判定を不要とするように設計されており，問い合わせ処理を高速化している．

2 つ目は，問い合わせグラフに部分グラフとして含まれるグラフの検索である．Chen ら [7] による cIndex は，対照部分グラフ集合から冗長なグラフを除いた集合を特徴とした索引付けを行うことで，今までのグラフの索引付け手法を，グラフの包含関係が逆の問題に対応させた．Chen らが解決する問題は，本稿が解決する問題に類似しているが，そのアプローチが異なる．すなわち，cInde はグラフの索引付け手法であり，正解グラフ集合の割合が少ない問い合わせを想定している．

3. 準　　備

本稿においては，ラベル付の無向グラフを議論の対象とし，グラフと呼ぶ．

［定義 1］ グラフ G は，4 つ組 $G = (V, E, \mu, \nu)$ で定義される．ここで，$V = \{v_1, v_2, ...\}$ は頂点集合，$E = \{(v, u)|v, u \in V, v \neq u\}$ は枝集合である．μ,ν はそれぞれ頂点集合，枝集合にラベルを割り当てる関数である．

慣例に基づき，グラフ G の頂点集合を $V(G)$, 枝集合を $E(G)$ とも表現する．また，グラフ G のサイズを $|V(G)|$ で定義し，$|G|$ で表す．

［定義 2］ 以下の条件を満たすとき，全単射 $\phi : V_1 \rightarrow V_2$ をグラフ $G_1 = (V_1, E_1, \mu_1, \nu_1)$ からグラフ $G_2 = (V_2, E_2, \mu_2, \nu_2)$ へのグラフ同型と呼ぶ．

- すべての $v \in V_1$ について, $\phi(v) \in V_2$ かつ $\mu_1(v) = \mu_2(\phi(v))$
- すべての $(v, u) \in E_1$ について, $(\phi(v), \phi(u)) \in E_2$ かつ $\nu_1((u, v)) = \nu_2((\phi(v), \phi(u)))$

［定義 3］ 定義 2 の条件を満たす，単射 $\phi : V_1 \rightarrow V_2$ をグラフ $G_1 = (V_1, E_1, \mu_1, \nu_1)$ からグラフ $G_2 = (V_2, E_2, \mu_2, \nu_2)$ への部分グラフ同型と呼ぶ．

グラフ G_1 からグラフ G_2 への部分グラフ同型が存在するとき，G_1 を G_2 の部分グラフと呼び，$G_1 \subseteqq G_2$ で表す．

［定義 4］ 2 つのグラフ $G_1 = (V_1, E_1, \mu_1, \nu_1)$ から $G_2 = (V_2, E_2, \mu_2, \nu_2)$ への部分グラフ同型 $\phi : V_1 \rightarrow V_2$ が以下の条件を満たすとき，ϕ を誘導部分グラフ同型と呼ぶ．

- すべての $u, v \in V_1$ について，$(\phi(u), \phi(v)) \in E_2$ が存在するならば，$(u, v) \in E_1$ が存在する．

グラフ G_1 からグラフ G_2 への誘導部分グラフ同型が存在するとき，G_1 を G_2 の誘導部分グラフと呼び，$G_1 \sqsubseteq G_2$ で表す．

［定義 5］ グラフ $G_1 = (V_1, E_1, \mu_1, \nu_1)$ とその誘導部分グラフ $G_2 = (V_2, E_2, \mu_2, \nu_2)$ が与えられたとき，G_1 と G_2 の差分を頂点集合 $V_1 - V_2$ からなる，G_1 の誘導部分グラフ G_{rest} とし，$G_{rest} = G_1 - G_2$ と表記する．

ここで，G_{rest} と G_2 のいずれにも含まれない G_1 の枝の集合 $E \subset E_1$ をカットと呼ぶ．

以下に，本稿が扱う誘導部分グラフ同型問い合わせ，本稿における部分グラフ問い合わせ等を定義する．

［定義 6］ グラフデータベース $D = \{g_1, ..., g_n\}$, 問い合わせグラフ q が与えられたとき，$g \in D$ から q への全ての部分グラフ同型 ϕ を検出する問題を部分グラフ同型問い合わせと定義する．同様に，$g \in D$ から q への全ての誘導部分グラフ同型 ϕ を検出する問題を誘導部分グラフ同型問い合わせと定義する．

問い合わせの対象となるグラフデータベースをモデルグラフ集合とよぶ．

4. 提案手法の概要

4.1 Messmer らの手法

Messmer らは，誘導部分グラフ同型問い合わせを効率的に計算する手法を提案した．Messmer らの手法は，予め行うデータ構造の構築と問い合わせに対するそのデータ構造を用いた処理の 2 つの段階から成る．

データ構造の構築では，モデルグラフ集合内の各グラフを再帰的に分解し，それらを非循環有向グラフに保存する．非循環有向グラフの各ノードが 1 つのグラフに対応しており，分解元のグラフ（親）から分解で生成されたグラフ（子）に向けて，有向辺が与えられる．グラフ分解では，1 つのグラフは必ず 2 つの誘導部分グラフに分けられ，各グラフは頂点 1 つから成るグラフになるまで再帰的に分解される．グラフ分解時には，非循環有向グラフ内のすべてのノードを走査し，誘導部分グラフを検索する．もし，誘導部分グラフが存在すれば，その中で頂点数が最大のグラフとその差分とに分解される．これにより，モデルグラフ集合内の各グラフ間の共通部分の共有が起きる．また，誘導部分グラフの検索は，後述する問い合わせに対する処理を利用し，非循環有向グラフ内のすべてのノードについて，誘導部分グラフ同型を検出することで行う．

問い合わせに対する処理では，前段階で構築した非循環有向グラフを用いる．処理は，すべての子孫の分解が確定したノードの順でボトムアップに成される．最初は，頂点 1 つから成るグラフから問い合わせグラフへの誘導部分グラフ同型の検出から始まり，子の誘導部分グラフ同型を組み合わせて，親の誘導部分グラフ同型が計算される分割統治法によるアルゴリズムである．ここで，あるグラフの誘導部分グラフが問い合わせグラ

フに対して誘導部分グラフ同型を持たないとき，そのグラフも誘導部分グラフ同型を持たない．したがって，非循環有向グラフ内の任意のノードにおいて，どちらかの子が誘導部分グラフ同型を持たないと判定されれば，そのノードも誘導部分グラフ同型を持たないと判定でき，その判定は先祖まで伝搬できる．この原理により，非常に大きなグラフに関する誘導部分グラフ同型の判定を小さなグラフに関する誘導部分グラフ同型の判定からできる場合がある．また，グラフの分解時に共通部分の共有を促進しているため，共通部分に関する誘導部分グラフ同型は共有されるといった利点がある．

4.2 提案手法

非循環有向グラフ内のノードおいて，どちらかの子が誘導部分グラフ同型を持たなければ，そのノードも誘導部分グラフ同型を持たない．したがって，その親の誘導部分グラフ同型を知るためには，もう一方の子とその子孫についての誘導部分グラフ同型の計算を必要としないといった利点がある．しかし，他の親の誘導部分グラフ同型の計算のためには，その計算が必要となる場合がある．例えば，図 2 において，S_2 から Q への誘導部分グラフ同型が存在しなければ，G_2 から Q への誘導部分グラフ同型も存在せず，G_2 から Q への誘導部分グラフ同型を調べるために，S_7,S_8,S_9 から Q への誘導部分グラフ同型を調べる必要はない．しかし，ボトムアップによる処理では，S_7,S_8,S_9 に G_2 以外の親がいる可能性を考慮し，それらについても誘導部分グラフ同型を調べなければならない．

このように，ボトムアップによる処理では，前述の利点を満足に享受できない．我々は，この非循環有向グラフのノードの冗長な計算を排除するため，問い合わせに対する処理を再帰化する．つまり，あるグラフから問い合わせグラフへの誘導部分グラフ同型を返すアルゴリズムは，その子に対する同じアルゴリズムによる問い合わせの結果を用いて答えを返す．このとき，子の一方が誘導部分グラフ同型を一つも返さなければ，もう一方に対する問い合わせは行わない．この再帰的な処理により，本当は必要のないノードへの問い合わせが無くなり，冗長を排除することができる．実際的には，モデルグラフ集合内に問い合わせグラフへの誘導部分グラフ同型を 1 つも持たないグラフが多いほど，多くの冗長ノードが発生し，検出が省略できる．

問い合わせに対する処理の再帰化によって，さらに局所的な問い合わせ（モデルグラフ集合の任意の部分集合内のグラフから問い合わせグラフへの誘導部分グラフ同型を検出する問題）

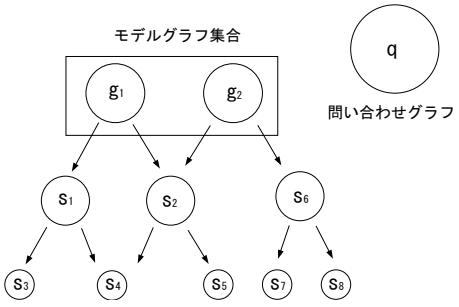

図 2　非循環有向グラフ

にも効率的に対応することが可能になる．Messmer らの手法では，たとえ局所的な解しか必要としていなくても，通常の問い合わせと同じように，非循環有向グラフ内の全てのノードについて検出をし，モデルグラフ集合内のすべてのグラフについて解を得ることでしか，局所的な問い合わせに対応できない．しかし，再帰的なアルゴリズムであれば，必要なノードのみ問い合わせることができ，より効率的な処理が可能になる．例えば図 2 において，グラフ集合 G_2 についてのみ問い合わせたい場合，G_1,S_1,S_2.S_3 に対する問い合わせは冗長である．

実際的には，モデルグラフ集合内の各グラフと問い合わせグラフの特徴を比較することで，効率的に誘導部分グラフ同型を持ち得ないグラフを枝刈りできる場合がある．簡単な例として，ラベルの数を特徴とした枝刈りをあげる（ラベルチェック）．例えば，モデルグラフ集合内の各グラフについて，各ラベルの数を予め数えておけば，問い合わせグラフの各ラベルの数を数え，それらを比較することで，問い合わせグラフの誘導部分グラフには成り得ないグラフを枝刈りすることができる．ラベルチェックは単純だが，場合により強力な手法である．Messmer らの手法を局所的な問い合わせに対応させることは，すなわち，グラフの枝刈りが効率的であれば，枝刈り後の候補者集合に対して局所的な問い合わせを行うことにより，全体として効率的な誘導部分グラフ同型問い合わせを構成できることになる．

グラフの枝刈りはグラフの索引付け [4]〜[9] において，候補者集合を得るために用いられ，さまざまな手法が提案されている．

5. 提案手法のアルゴリズム

本章では，提案手法の詳細なアルゴリズムを述べる．なお，アルゴリズムは非循環有向グラフ内のグラフを連結グラフに制限する手法 [3] を適用するが，非連結グラフにも対応させるため，「非循環有向グラフ内においては，連結グラフは必ず連結グラフに分解される」とした．非連結グラフについては，連結成分を分配することで，分解を行う．また，提案手法は Messmer らの手法を部分グラフ同型問い合わせに拡張した場合 [3] においても有効に働くが，本稿では誘導部分グラフ同型問い合わせのアルゴリズムについてのみ述べる．部分グラフ同型問い合わせへの拡張方法は，文献を参照されたい．

以下では，非循環有向グラフ DAG をグローバル変数として扱う．非循環有向グラフ内の各ノードからは 2 つの子に向けて有向辺がある．各ノードには，グラフ，親への誘導部分グラフ同型集合，問い合わせグラフへの誘導部分グラフ同型集合，グラフ分解時のカットが保存される．

5.1 非循環有向グラフの構築アルゴリズム

非循環有向グラフにグラフを挿入するアルゴリズム Insert を Algorithm1 に示す．モデルグラフ集合 D の非循環有効グラフの構築するには，D 内のすべてのグラフを挿入すればよい．

初めに，入力グラフ g_i の誘導部分グラフ s を，DAG 内のすべてのグラフから検索し，優先待ち行列 PQ に挿入する．この検索には，誘導部分グラフ同型問い合わせの処理が利用できる．つぎに PQ において，優先される誘導部分グラフから，以下の

処理がなされる．

g_i と同じサイズの誘導部分グラフ s が見つかった場合，DAG 内に g_i と同型なグラフ s がすでに存在していたことを意味する．したがって，s を g_i の親の子とし，g_i を破棄することで，s と g_i を入れ替える．

多くの場合は，g_i よりサイズの小さい誘導部分グラフ s が見つかることが想定される．このとき，g_i と s の差分 s_{rest} が計算される．s_{rest} が連結であった場合，g_i は s と s_{rest} に分解され，カットが保存される．s_{rest} は Insert により再帰的に DAG に挿入される．もし，s_{rest} が非連結であった場合は，連結グラフを連結グラフに分解する制限を守ることができない．このとき，s_{rest} は破棄され，次に優先される誘導部分グラフ同型における差分が計算される．差分の計算は，s_{rest} が連結グラフとなるまで，PQ 内で優先される順番で，すべての誘導部分グラフ同型が試される．提案手法においては，できるだけ大きなサイズのグラフが共有され，その計算結果も共有されることが望ましい．ゆえに，PQ における優先順位は，グラフのサイズが大きいほど優先されることとする．

もし，DAG 内に入力グラフ g_i の誘導部分グラフが存在しない場合，もしくは g_i との差分が連結グラフになるような，誘導部分グラフ同型が存在しない場合，RandomPartition(Algorithm2) が g_i を 2 つの誘導部分グラフに分解する．分解時にはカットが g_i に保存され，2 つの誘導部分グラフはそれぞれ DAG に再帰的に挿入される．

g_i が 1 つの頂点からなるグラフの場合は，グラフ分解の終端あり，その時点で再帰的な挿入を終了する．また，非連結グラフが入力とされ場合には，連結成分を分配して分解することで，できるだけ連結グラフが生成されるようにした．

RandomPartition は，連結グラフ g を 2 つの連結な誘導部分グラフ $s_1, s_2 \sqsubseteq g$ に分解するためのアルゴリズムである．初めに，g の頂点を s_1, s_2 に半分ずつ分配し，それらの頂点から誘導される誘導部分グラフが連結になるまで，連結成分をもう一方に移動し続ける．

5.2 誘導部分グラフ同型問い合わせのアルゴリズム

誘導部分グラフ同型問い合わせにおいて，DAG 内のノードは 3 種類のタグ *alive*,*dead*,*unsolved* により区別され，それぞれ誘導部分グラフ同型が存在する，誘導部分グラフ同型が存在しない，未解決であることを表す．

Search(Algorithm3) は，誘導部分グラフ同型問い合わせを行うアルゴリズムである．問い合わせ処理では，まず DAG 内のすべてのノードに初期値として *undolved* をマークする．SubgraphQ(Algorithm4) は，DAG 内の任意のノードから問い合わせグラフへの誘導部分グラフ同型を返すアルゴリズムである．したがって，誘導部分グラフ同型問い合わせは，SubgraphQ により，DAG 内でモデルグラフ集合に含まれていたグラフから問い合わせグラフへの誘導部分グラフ同型をすべて求めることで成される．

SubgraphQ は，DAG 内の任意のノードから問い合わせグラフへの誘導部分グラフ同型集合を，ノードの子から再帰的に計算する．もし，入力グラフ g にすでに *alive* もしくは *dead* のタグが与えられている場合には，すでに誘導部分グラフ同型集合が計算されているため，以前の結果を返す．

もし，g が頂点 1 つからなるグラフである場合は，子が存在しない．このとき，SubgraphQ は AssignVertex を呼び出す．AssignVertex は，問い合わせグラフ内におけるその頂点の写像の集合を検索し，返す関数である．

SubgraphQ は誘導部分グラフ同型を子の誘導部分グラフ同型から，再帰的に計算する．したがって，g が *unsolved* である場合には，g の子 c_1,c_2 を入力として，再帰的に SubgraphQ を呼び出す．

ここで，子から問い合わせグラフへの誘導部分グラフ同型が存在しなければ，親から問い合わせグラフへの誘導部分グラフ同型も存在しないという事実が利用できる．まず，冗長な計算を排除するため，子を入力とした SubgraphQ の呼び出しの前に，子のタグを調べる．いずれかの子に *dead* のタグが付いていれば，g のタグを *dead* とすることで，SubgraphQ の呼び出

Algorithm 1 Insert

```
Input: グラフ g_i
1: s := ∅, s_rest := ∅ とする
2: if g_i が連結 then
3:    for 各 DAG 内の g do
4:       if g ⊑ g_i then
5:          PQ.insert(g)
6:       end if
7:    end for
8: end if
9: repeat
10:   s = PQ.dequeue()
11:   if |s| = |g_i| then
12:      s を g_i の親の子とし，g_i を破棄する
13:      return
14:   else
15:      for 各 s から g_i への誘導部分グラフ同型 φ do
16:         s_rest = g_i − s(φ により定まる)
17:         if s_rest が連結 then
18:            s_rest を g_i の子とする
19:            Insert(s_rest)
20:            return
21:         end if
22:      end for
23:   end if
24: until PQ が空
25: if g_i が頂点 1 つから成るグラフでない then
26:   if g_i が連結 then
27:      (s, s_rest) = RandomPartition(g_i)
28:   else
29:      g_i の連結成分を s と s_rest に半分づつ分配する
30:   end if
31:   s と s_rest を g_i の子とする
32:   Insert(s)
33:   Insert(s_rest)
34: end if
```

Algorithm 2 RandomPartition

Input: グラフ g
Output: グラフの組 (s_1, s_2)

$s_1 := \emptyset$, $s_2 := \emptyset$ とする
g の頂点を s_1 と s_2 へ半分づつランダムに分配する
s_1 と s_2 に頂点集合により誘導される枝集合を与える
repeat
  if s_1 と s_2 のどちらも非連結 **then**
    より大きいグラフの最小連結成分を他方に移動する
  else
    非連結グラフの最小連結成分を他方に移動する
  end if
  s_1 と s_2 の頂点集合により誘導され枝集合を与え，誘導部分グラフを生成し直す
until s_1 と s_2 のどちらも連結
return (s_1, s_2)

Algorithm 3 Search

Input: 問い合わせグラフ q, モデルグラフ集合 D
Output: 誘導部分グラフ同型集合の集合 Z

$Z := \emptyset$
DAG 内のすべてのノードに *unsolved* を付す
for 各 $g \in D$ **do**
  $F :=$ SubgraphQ(g, q)
  if $F \neq \emptyset$ **then**
    $Z = Z \cup \{F\}$
  end if
end for
return Z

Algorithm 4 SubgraphQ

Input: グラフ g, 問い合わせグラフ q
Output: 誘導部分グラフ同型集合 F

if g が *unsolved* **then**
  $F := \emptyset$
  if g が頂点 1 つから成るグラフ **then**
    $F =$AssignVertex(g, q)
  else
    c_1 と c_2 を g の子とする
    if c_1 か c_2 のいずれかが *dead* **then**
      g に *dead* を付す
    else if SubgraphQ$(c_1, q) = \emptyset$ もしくは SubgraphQ$(c_2, q) = \emptyset$ **then**
      g に *dead* を付す
    else
      $F =$ Combine(g, q)
    end if
  end if
  if $F \neq \emptyset$ **then**
    g に *alive* を付す
  else
    g に *dead* を付す
  end if
else
  F を既に計算されている g から q への誘導部分グラフ同型集合とする
end if
return F

しを行わずにアルゴリズムを終了できる．また，子を入力としたSubgraphQ のいずれかが空集合を返した場合も，子に *dead* のタグが付いたことを意味するため，g のタグを *dead* とする．

どちらの子も *dead* のタグが付いておらずかつ空でない誘導部分グラフ同型集合が返ってきた場合のみ，結合アルゴリズム Combine(Algorithm5) を呼び出す．Combine の結果，g にも誘導部分グラフ同型が存在した場合は，g に *alive* をマークする．Combine が空集合を返す場合には，*dead* のタグが付される．

Combine はグラフ $g = (V, E, \mu, \nu)$ から問い合わせグラフ $q = (V_q, E_q, \mu_q, \nu_q)$ への誘導部分グラフ同型集合を g の子グラフから q への誘導部分グラフ同型集合を利用して計算するアルゴリズムである．Combine が呼び出されたとき，g の 2 つの子 c_1,c_2 から q への誘導部分グラフ同型集合 F_1,F_2 は既に計算されていることになっている．2 子グラフから q への任意の誘導部分グラフ同型のペア $f_1 \in F_1$，$f_2 \in F_2$ に対して，V から V_q への写像 $f : V \to V_q$ を次式により定める．

$$f(v) = \begin{cases} f_1(p_1(v)) & (p_1(v) \in V(c_1)) \\ f_2(p_2(v)) & (p_2(v) \in V(c_2)) \end{cases} \quad (1)$$

次の 2 つの条件を満たすとき，f は g から q への誘導部分グラフ同型である．

（1） q にカットに対応する枝集合が存在し，かつ q 内での f_1，f_2 の像の間にある枝に対応する枝がカット内に存在する

（2） f_1 と f_2 が交わりを持たない

1 つ目の条件は，カットが誘導部分グラフの条件に対応しているかどうかを調べる条件であり，2 つ目の条件は f が単射であるための条件である．Combine は $f_1 \in F_1$，$f_2 \in F_2$ のすべてのペアについて，2 つの条件を調べ，f を計算する．

6. 実験と評価

提案手法を Messmer らによる手法および一対一の誘導部分グラフ同型検出による逐次処理（SCAN）と比較し評価する．SCAN では，VF2 [10] を利用する．アルゴリズムはすべて Visual Studio 2005 上で C++により実装し，Intel PC 2.5GHz, 8GBRAM, Windows Vista 上で動作させる．

6.1 データセット

本稿では，誘導部分グラフ同型判定の応用の一つとして，化合物の検索により実験を行う．実験データには，NCI(注1)により提供されている AIDS Antiviral Screen のデータセット（以下，AIDS データ）を利用する．AIDS データは 42687 個の化合物で構成され，平均頂点数 45，平均枝数 47，最大頂点数 438，最大枝数 441，頂点ラベル数 63 種，枝ラベル数 3 種類といった特徴を持つ．実験ではこの中から指定した条件に合うグラフをランダムに抽出し，グラフデータベースを構成する．

(注1)：National Cancer Institute http://dtp.nci.nih.gov/

Algorithm 5 Combine

Input: グラフ $g = (V, E, \mu, \nu)$，問い合わせグラフ $q = (V_q, E_q, \mu_q, \nu_q)$

Output: 誘導部分グラフ同型集合 F

1: g の子を c_1 と c_2 とする
2: g から c_1，c_2 への誘導部分グラフ同型を p_1，p_2 とする
3: c_1，c_2 から q への誘導部分グラフ同型集合を F_1，F_2 とする
4: $F := \emptyset$
5: g の分解時におけるカットを $E_c \subset E$ とする
6: **for** 各 $f_1, f_2 \in F_1, F_2$ **do**
7: **if** 各 $e = (v_1, v_2) \in E_c$ について，$\nu(e) = \nu_q(e_q)$ となる枝 $e_q = (f_1(v_1), f_2(v_2)) \in E_q$ が存在し，かつ，各 $e_q = (u_1, u_2) \in E_q$ s.t. $u_1 \in f_1(V(c_1)), u_2 \in f_2(V(c_2))$ について，$\nu_q(e_q) = \nu(e)$ となる枝 $e = (f_1^{-1}(u_1), f_2^{-1}(u_2)) \in E_c$ が存在する **then**
8: **if** $f_1(V(c_1)) \cap f_2(V(c_2)) = \emptyset$ **then**
9: 誘導部分グラフ同型 $f : V \to V_q$ を式 (1) で定義する
10: $F = F \cup \{f\}$
11: **end if**
12: **end if**
13: **end for**
14: **return** F

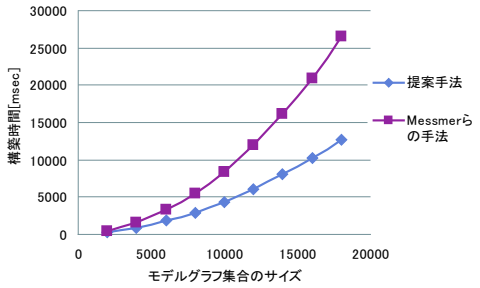

図 3　モデルグラフ集合のサイズと非循環有向グラフの構築時間の関係

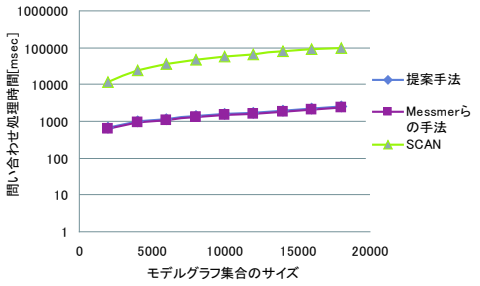

図 4　モデルグラフ集合のサイズと問い合わせの処理時間の関係

また，提案手法が顕著に有効なケースを示すため，人工データを用いて実験を行う．データセットは，ジェネレータによってランダムに生成されたグラフで構成する．ジェネレータが生成するグラフは連結グラフに限定される．また，平均枝確率を 50 %，頂点ラベル 10 種，枝ラベル 10 種とした．ラベルは一様に分布される．

6.2　記述子の検索による実験

化合物の化学的，物理的特徴を示す部分構造を記述子と呼ぶ．化合物に含まれる記述子の検索を効率化することは，化合物の解析等の助けとなる．本実験では，記述子の集合をモデルグラフ集合とし，化合物から成る問い合わせ集合による記述子の検索を行う．AIDS データの中から，頂点数 40 未満のグラフをランダムに 10000 個の抽出し，グラフデータベース W とする．グラフ数，頂点数の制限は頻出部分グラフマイニングを容易にするためである．W の頻出部分グラフ集合を求め記述子の集合とみなし，モデルグラフ集合とする．頻出部分グラフ集合は gSpan [11] により求める．出現頻度を 5 %以上とし，18930 個のグラフが得られた．

6.2.1　モデルグラフ集合のサイズに関する実験

モデルグラフ集合のサイズを変化させた時の非循環有向グラフの構築時間を提案手法と Messmer らの手法で比較し，評価する．モデルグラフ集合のサイズを 2000 から 18000 まで変化させた．図 3 にモデルグラフ集合のサイズと構築時間の関係を示す．図 3 より，提案手法は Messmer らの手法より短い時間で非循環有向グラフを構築できる．したがって，非循環有向グラフの構築時間においては，提案手法が Messmer らの手法よりも優れている．

モデルグラフ集合のサイズを変化させた時の問い合わせの処理時間を提案手法と Messmer らの手法および SCAN で比較し，評価する．モデルグラフ集合のサイズを 200 から 18000 まで変化させた．問い合わせグラフ集合を，W から 100 個のグラフをランダムに抽出し構成する．図 4 にモデルグラフ集合のサイズと問い合わせの処理時間の関係を示す．問い合わせの処理時間は問い合わせグラフ集合内のすべてのグラフの問い合わせを処理するのにかかる合計時間である．また，提案手法，Messmer らの手法において，前処理である非循環有効グラフの構築時間は含まれない．SCAN では前処理を必要としない．図 4 より，提案手法は Messmer らの手法はほぼ同じ時間で，提案手法，Messmer らの手法は SCAN より 10 倍以上高速に問い合わせを処理する．

以下，結果を考察する．提案手法の Messmer らの手法からの改善点は 2 つ，(1) 分解時に非連結グラフが生成されると，問い合わせ時に誘導部分グラフ同型の組み合わせ爆発が起こるため，分解後のグラフを連結グラフに限定したこと，(2) 再帰的な処理により，非循環有向グラフ内のノードに対する冗長な問い合わせを排除したことである．しかし，本実験には 2 つの改善点はそれぞれ下記の理由により，有効でない．(1) について，本実験では頻出部分グラフ集合をモデルグラフ集合としており，アルゴリズムにより発生する頻出部分グラフ間の差分では，分解により非連結グラフが生成されることが稀であったこと．(2) について，頻出部分グラフマイニング時の出現頻度の設定から，モデルグラフが問い合わせグラフに対して誘導部分グラフ同型を持つ確率が少なくとも 5 %以上あり，問い合わせることが冗長なノードが極僅かであったこと．したがって，提案手法と Messmer らの手法で差が見られなかったと考えられる．

6.2.2　問い合わせグラフのサイズに関する実験

問い合わせのサイズを変化させたときの問い合わせの処理時間を提案手法と Messmer らの手法および SCAN で比較し，評

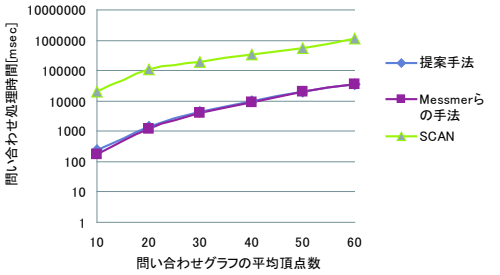

図 5　問い合わせグラフのサイズと問い合わせの処理時間の関係

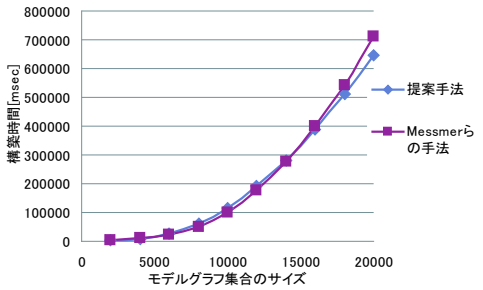

図 6　モデルグラフ集合のサイズと非循環有向グラフの構築時間の関係

価する．平均頂点数を指定して 10 から 60 まで変化させ，それぞれグラフをランダムに 100 個，AIDS データから抽出し，問い合わせグラフ集合とする．図 5 に問い合わせグラフのサイズと問い合わせの処理時間の関係を示す．図 5 より，提案手法は Messmer らの手法はほぼ同じ時間で，提案手法，Messmer らの手法は SCAN より 10 倍以上高速に問い合わせを処理する．この結果は，モデルグラフ集合のサイズに関する実験と同様である．本実験は人工データによる実験との対比を目的とする．

6.3　人工データによる実験

記述子の検索による実験では，提案手法の有効性が確認できなかった．提案手法が顕著に有効なケースを示すため，人工データを用いて実験を行う．

6.3.1　モデルグラフ集合のサイズに関する実験

モデルグラフ集合のサイズを変化させた時の非循環有向グラフの構築時間を提案手法と Messmer らの手法で比較し，評価する．モデルグラフ集合は，平均頂点数 10 のグラフを 20000 個生成し，サイズを 2000 から 20000 まで変化させた．図 6 にモデルグラフ集合のサイズと構築時間の関係を示す．図 6 より，提案手法は Messmer らの手法と同等の時間で非循環有向グラフを構築する．

モデルグラフ集合のサイズを変化させた時の問い合わせの処理時間を提案手法と Messmer らの手法および SCAN で比較し，評価する．問い合わせグラフ集合は，平均頂点数 60 のグラフを 100 個生成し構成する．図 7 にモデルグラフ集合のサイズと問い合わせの処理時間の関係を示す．図 7 より，提案手法は Messmer らの手法より 5 倍以上，SCAN より 10 倍以上高速に問い合わせを処理する．したがって，問い合わせの処理時間に関しては，提案手法が Messmer らの手法よりも優れている．

この結果には 2 つの原因が考えられる．1 つ目は，Messmer らの手法では分解において，非連結グラフが多く発生し，誘導部分グラフ同型の組み合わせ爆発が起こることである．

2 つ目は，本実験におけるモデルグラフ集合と問い合わせ集合の関係では，モデルグラフから問い合わせグラフへの誘導部分グラフ同型が存在するケースは少なく，非循環有向グラフ内に問い合わせることが冗長なノードが数多く含まれており，再帰化による恩恵が大きかったためである．図 8 は本実験における，モデルグラフ集合のサイズと非循環有向グラフ内のノードの数および問い合わせ処理後に *unsolved* のタグが付されたノード数の平均の関係を示す．提案手法は，Messmer らの手

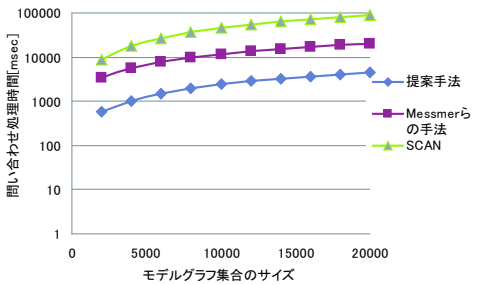

図 7　モデルグラフ集合のサイズと問い合わせの処理時間の関係

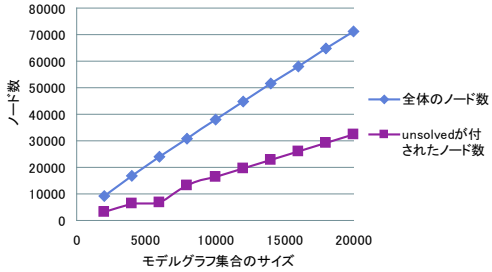

図 8　モデルグラフ集合のサイズと非循環有向グラフ内のノード数および問い合わせ処理後に *unsolved* のタグが付されたノード数の関係

法が問い合わせ処理の過程で非循環有向グラフ内のすべてのノードについて検出を行うのに対し，不要な検出を排除することで，処理時間の改善する．つまり，全体のノード数に対して *unsolved* のタグが付されたノード数が多いほど提案手法による改善が有効であったと言える．図 8 から，本実験では平均して約 40 %のノードに対する冗長な問い合わせが排除されたことがわかり，提案手法の有効性が確認される．

6.3.2　問い合わせグラフの頂点数に関する実験

問い合わせグラフの頂点数を変化させたときの問い合わせの処理時間を，提案手法と Messmer らの手法および SCAN で比較し，評価する．平均頂点数を 20 から 100 まで変化させ，それぞれグラフを 100 個生成し，問い合わせグラフ集合とする．モデルグラフ集合は，平均頂点数 10 のグラフを 20000 個で構成する．図 9 に問い合わせグラフのサイズと問い合わせの処理時間の関係を示す．図 9 より，提案手法は Messmer らの手法，SCAN より短い時間で問い合わせを処理する．また，Messmer らの手法との差は問い合わせグラフのサイズが大きくなるにつ

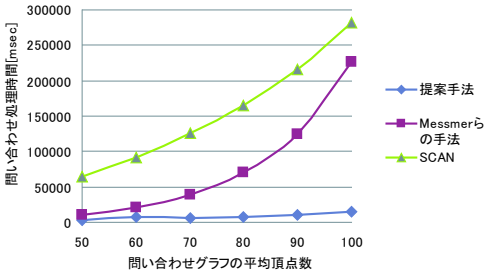

図 9　問い合わせグラフのサイズと問い合わせの処理時間の関係

れて大きくなっている．これは，問い合わせグラフのサイズが大きくなるほど，問い合わせ時の誘導部分グラフ同型の組み合わせが爆発的に増えるためである．本実験により，提案手法は Messmer らの手法では扱えない大規模なグラフに対しても有効であることがわかる．

7. まとめと今後の課題

本稿では，モデルグラフ集合に含まれるグラフから問い合わせグラフへの誘導部分グラフ同型をすべて検出する問題に対する Messmer らの手法を，再帰的な処理により効率化することを提案した．実験により，Messmer らの手法と比較し，非循環有向グラフの構築時間，問い合わせの処理時間について顕著に改善されるケースを示した．特に，提案手法はより大規模な問い合わせグラフに対して有効であることが判明した．

今後の課題としては，XML や画像などの化合物以外の実データによる応用実験があげられる．また，類似グラフの検索手法へと応用することを考えている．

謝辞　gSpan を提供してくださった Dr. Xifeng Yan と Prof. Jiawei Han に深く感謝いたします．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（C）（課題番号:19500089）による．

文　献

[1] S. A. Cook: “The complexity of theorem-proving procedures”, Proc. of STOC ’71: Proceedings of the third annual ACM symposium on Theory of computing, pp. 151–158 (1971).

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

[3] 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”, DMIN, pp. 397–403 (2007).

[4] D. Shasha, J. T. L. Wang and R. Giugno: “Algorithmics and applications of tree and graph searching”, Proc. of PODS, pp. 39–52 (2002).

[5] X. Yan, P. S. Yu and J. Han: “Graph indexing: a frequent structure-based approach”, SIGMOD ’04: Proceedings of the 2004 ACM SIGMOD international conference on Management of data, New York, NY, USA, ACM, pp. 335–346 (2004).

[6] J. Cheng, Y. Ke, W. Ng and A. Lu: “Fg-index: Towards verification-free query processing on graph databases”, SIGMOD’07: Proceedings of the 2007 ACM SIGMOD international conference on Managemant of data, ACM, pp. 857–872 (2007).

[7] C. Chen, X. Yan, P. S. Yu, J. Han, D.-Q. Zhang and X. Gu: “Towards graph containment search and indexing”, VLDB, pp. 926–937 (2007).

[8] H. He and A. K. Singh: “Closure-tree: An index structure for graph queries”, ICDE, p. 38 (2007).

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

[10] 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 Recoginition, pp. 149–159 (2001).

[11] X. Yan and J. Han: “gspan: Graph-based substructure pattern mining”, Proceedings of the 2002 IEEE International Conference on Data Mining(ICDM 2002), pp. 721–724 (2002).