有向グラフに対する圧縮法および圧縮グラフに対する頂点選択問合せ評価法の提案

武田　健志† 　橋本　健二† 　関　浩之†

† 名古屋大学大学院情報学研究科　〒 464–8601 愛知県名古屋市千種区不老町
E-mail: †takeda.takeshi@a.mbox.nagoya-u.ac.jp, ††{k-hasimt,seki}@i.nagoya-u.ac.jp

あらまし　実用データを表現したグラフは非常に大規模になりやすいため，様々なグラフ圧縮の研究が行われているが，圧縮された有向グラフに対する直接問合せ評価法については報告されていない．本研究では有向グラフに対して合流頂点分解という手法を導入することで辺の向きも保存するような，木文法に基づく圧縮法と，圧縮グラフに対して解凍することなく問合せを行う手法について提案し，圧縮法については提案手法に基づく実験結果についても述べる．
キーワード　有向グラフ，木文法，圧縮，問合せ，TreeRePair

1. まえがき

グラフは頂点と辺からなるデータ構造で，その汎用性の高さから SNS における対人関係，通信ネットワークトポロジーやセマンティックウェブにおける RDF(Resource Description Framework) など，様々な関係を表現することに用いられている．一方で，実用的なグラフは非常に大規模になりやすいため，メモリ量を節減した記憶方式の開発が必要となっている．

このような問題を解決するため，データ工学分野では，オブジェクト指向データベースや科学データベースを含むグラフデータベースにおいて，グラフ圧縮の研究 [7] [3] [5] [4] [1] が盛んに行われている一方，圧縮グラフに対する直接問合せ評価の有用性を示したという研究はまだ報告されていない．

本研究では，グラフを SLCFTG と呼ばれる木文法に圧縮する手法と，SLCFTG に圧縮されたグラフに対して，それを解凍することなく頂点選択を行う問合せ評価法を提案し，実データを用いて提案手法を評価することを目的としている．圧縮法の基本的な方針としては，与えられたグラフに対して，合流頂点分解とよばれる手法で複数の木からなる森を構成したのち，各木に対して TreeRePair アルゴリズム [6] を適用することでグラフから SLCFTG $\mathcal{G}$ を得る．圧縮結果は $\mathcal{G}$ と逆辺 (向きを反転させた辺) の集合 R の対 $(\mathcal{G}, R)$ とする．次に，グラフに対する問合せモデルとして決定性選択グラフオートマトン (Deterministic Selecting Graph Automaton: DSGA) を導入した．これは頂点に状態を割り当てていき，頂点のラベルと状態の対に応じて頂点選択を行うモデルである．そして，圧縮されたグラフ $(\mathcal{G}, R)$ と DSGAA が与えられたとき，$\mathcal{G}$ に対して A を直接適用することで頂点選択を行う手法を提案した．本研究では以上の圧縮法および直接問合せ評価法を提案し，圧縮法については実装・評価実験を行なった．

本稿の以降の構成は以下の通りである．まず 2. 節で諸定義を与える．続いて，3. 節でグラフから SLCFTG を得るための具体的な圧縮法を提案し，4. 節では提案手法によって圧縮されたデータに対する問合せの直接評価法，すなわち，圧縮データである SLCFTG $\mathcal{G}$，逆辺情報 R および問合せを表す DSGAA が与えられたとき，$(\mathcal{G}, R)$ に対して A を直接適用して問合せ評価する方法を述べる．5. 節では実際のグラフに対する実験の結果について述べる．6. 節ではまとめと今後の課題を述べる．

2. 準　　備

2.1 根付き連結グラフと木

ランクと呼ばれる自然数を持つ記号の有限集合 Σ をランク付きアルファベットとよぶ．記号 $a \in \Sigma$ のランクを $rank(a)$ と表記し，$\Sigma_n = \{a \in \Sigma \mid rank(a) = n\}$ とする．また，空系列を ϵ で表す．有限集合 S の要素数を $|S|$ と表す．

定義 2.1　Σ 上の根付き連結 (有向) グラフとは，5 項組 $G = (V, E, v_0, \Sigma, lab_G)$ である．ここで，

(1) V は頂点の有限集合．
(2) E は有向辺 (辺と略す) の有限集合．辺は頂点の順序対 (u, v) である．$(u, v \in V)$．
(3) $v_0 \in V$ は根頂点 (または開始頂点)．
(4) $lab_G : V \to \Sigma$ は頂点へのラベル付け関数．

ただし，G において自己閉路辺 (u, u) は存在しないとする．また，全ての頂点 $v \in V$ は，v_0 から到達可能でなければならない．$(u, v) \in E$ であるとき，単に $u \to_E v$ とかき，$\to_E$ の推移閉包を $\to_E^+$, 反射推移閉包を $\to_E^*$ とかく．ある頂点 u から頂点 v へ到達可能であるとは，$u \to_E^* v$ であることをいう．$u \to_E v$ であるとき，u は v の親頂点といい，v は u の子頂点という．$lab_G(v) = a$ $(v \in V)$ であるとき，a を v のラベルという．頂点 v について，$dg_{in_G}(v) = |\{(u, v) \mid (u, v) \in E\}|$ を v の入次数といい，同様に $dg_{out_G}(v) = |\{(v, u) \mid (v, u) \in E\}|$ を v の出次数という．入次数が 2 以上の頂点 v を合流頂点とよぶ．根付き連結グラフ $G = (V, E, v_0, \Sigma, lab_G)$ において，$\forall v \in V. rank(lab_G(v)) = dg_{out_G}(v)$ であるなら，G をランク付きであるという．　□

定義 2.2　根付き連結グラフ $G = (V, E, v_0, \Sigma, lab_G)$ が $|E| =$

$|V|-1$ を満たすとき，G を Σ 上の木という．さらに，G がランク付きであるとき，以下の条件を満たす全単射 $pos : V \to dom(G) \subseteqq \mathbb{N}^*$ に対して，G を (pos で決まる) ランク付き順序木という．

(1) $pos(v_0) = \epsilon$.

(2) $pp' \in dom(G)$ $(p, p' \in \mathbb{N}^*)$ ならば，$p \in dom(G)$.

(3) $v \in V$ かつ $lab_G(v) \in \Sigma_n$ であるとき，$1 \leqq i \leqq n$ であるような全ての i について，ある $v_i \in V$ が存在して，$(v, v_i) \in E$，$pos(v_i) = pos(v)i$．v を v_i の親頂点，v_i を v の i 番目の子頂点という．

G がラベル付き順序木であるとき，$v \in V$ と $pos(v)$ を同一視して，$pos(v)$ を頂点とよぶこともある．ランク付き順序木に対し，各頂点の先行順番号 $pre : V \to \mathbb{N} \cup \{0\}$ を以下のように定義する．ただし，$<_{lex}$ は辞書式順序を表す．

- $pre(v_0) = 0$
- $pre(v) = max\{pre(u) \mid pos(u) <_{lex} pos(v)\} + 1$

先行順番号による順序を単に先行順とよぶ． □

定義 2.3 根付き連結グラフ $G = (V, E, v_0, \Sigma, lab_G)$ が次の条件を満たすとき，G を非巡回グラフ (DAG) とよぶ．

- $v \to_E^+ v$ であるような頂点 $v \in V$ が存在しない．

G が DAG であるとき，以下の条件を満たすような順序 $<_{DAG}$ をトポロジカル順序とよぶ．

- $\forall u, v \in V. u <_{DAG} v$ であるなら $v \to_E^+ u$ でない．

□

定義 2.4 ラベル付き順序木 $t = (V, E, v_0, \Sigma, lab_G)$ について，以下のような条件を満たすラベル付き順序木 $t_v = (V_t, E_t, v, \Sigma, lab_G)$ のことを，頂点 $v \in V$ を根頂点とする t の部分木という．

(1) $V_t \subseteqq V$ は v から到達可能なすべての頂点からなる集合．

(2) $E_t \subseteqq E$ は v から到達可能なすべての辺からなる集合．

ここで，頂点 u から頂点 w に到達可能であるとき，$(w, x) \in E$ を頂点 u から到達可能な辺であるという． □

ラベル付き順序木 t において，$lab_G(v) = f \in \Sigma_n$ であるとき，$t = f(t_1, \ldots, t_n)$ と表し，t の項表現とよぶ．ここで，$t_i (1 \leqq i \leqq n)$ は再帰的に t の根頂点の i 番目の子頂点を根とする部分木の項表現である．

各ラベルのランクが固定されていないラベル付き順序木のことをランクなしラベル付き順序木 (ランクなし木と略) とよぶ．ランクなし木は次に述べる fcns 符号化を用いることで，全てのラベルについて，常にランクが 2 であるような 2 分木に変換できることが知られている [2].

定義 2.5 first-child next-sibling(fcns) 符号化

$t_1, \ldots, t_n$ を任意のランクなし木とし，$t_1 = f(s_1, \ldots, s_m)$ とする．$fcns(t_1, \ldots, t_n)$ を以下のように定義する．

(1) $fcns() = \#$.

(2) $fcns(t_1, \ldots, t_n) = f(fcns(s_1, \ldots, s_m), fcns(t_2, \ldots, t_n))$.

□

本稿では以降，ランクなし木を fcns 符号化した 2 分木を単に木とよぶ．

2.2 木 圧 縮

2.2.1 直線的文脈自由木文法

$\Sigma \cap \mathcal{X} = \emptyset$ であるような変数の可算集合を $\mathcal{X} = \{x_1, x_2, \ldots\}$ とする．また，$\mathcal{X}_n = \{x_1, \ldots, x_n\}$ とする．$\mathcal{X}$ に属する変数をランク 0 の記号として含む木の全体集合を $T(\Sigma, \mathcal{X})$ とする．木に出現するどの変数についても高々1 回しか出現しないとき，その木は線形であるという．$t \in T(\Sigma, \mathcal{X}_n)$，$t_1, \ldots, t_n \in T(\Sigma, \mathcal{X})$ としたとき，$t[x_1/t_1, \ldots, x_n/t_n]$ はラベルが $x_i (1 \leqq i \leqq n)$ であるような頂点を木 t_i で置き換えた木を表す．また，木 t に対し，$t = x \in \mathcal{X}$ のとき $|t| = 0, t = f(t_1, \ldots, t_n)$ のとき $|t| = 1 + \sum_{i=1}^n |t_i|$ と定義する．

定義 2.6 文脈自由木文法 (Context-Free Tree Grammar, CFTG) は 4 項組 $\mathcal{G} = (N, \Sigma, R, S)$ である．ここで，

(1) N はランク付きアルファベット (非終端記号の有限集合).

(2) Σ は $N \cap \Sigma = \emptyset$ を満たすランク付きアルファベット (終端記号の有限集合).

(3) R は生成規則の有限集合．生成規則とは $A(x_1, \ldots, x_n) \to t$ という形をしたもののことをいう．ただし，$A \in N$, $rank(A) = n$, $t \in T(\Sigma \cup N, \mathcal{X}_n)$ で，t には $x_1, \ldots, x_n$ が出現する．

(4) $S \in N$ は開始非終端記号で $rank(S) = 0$.

$A(x_1, \ldots, x_n) \to t \in R$ について，$x_1, \ldots, x_n$ を A の仮引数とよび，また t に出現する非終端記号 B の子頂点を根とした部分木のことを B の実引数，または単に実引数とよぶ． □

$s \Rightarrow_{\mathcal{G}} s'$ とは，$A(x_1, \ldots, x_n) \to t \in R$, $C \in T(\Sigma \cup N, \mathcal{X} \cup \{z\})$ が存在して，$s = C[z/A(t_1, \ldots, t_n)]$，$s' = C[z/t[t_1, \ldots, t_n]]$ となることを表す．また，$\Rightarrow_{\mathcal{G}}$ の反射推移閉包，推移閉包を $\Rightarrow_{\mathcal{G}}^*$, $\Rightarrow_{\mathcal{G}}^+$ とかく．$L(\mathcal{G}) = \{t \mid S \Rightarrow_{\mathcal{G}}^* t$ かつ $t \in T(\Sigma)\}$ を $\mathcal{G}$ から生成される木言語とよぶ．

定義 2.7 直線的文脈自由木文法 (Straight-Line CFTG, SLCFTG) は以下の条件を満たすような CFTG $\mathcal{G} = (N, \Sigma, R, S)$ である．

(1) 全ての $A \in N$ について，ただひとつの生成規則 $A(x_1, \ldots, x_n) \to t \in R$ が存在し，t は線形である．このとき t を t_A とかき，A の右辺とよぶ．

(2) 全ての $A \in N$ について，$A(x_1, \ldots, x_n) \Rightarrow_{\mathcal{G}}^+ s$ のとき，s に A が出現しない．

また，SLCFTG $\mathcal{G}$ のサイズ $|\mathcal{G}|$ は以下のように定義する．

$$|\mathcal{G}| = \sum_{A \to t \in R} (|t| + rank(A))$$

□

SLCFTG $\mathcal{G}$ について，(1) より各非終端記号について決定的に生成規則を適用することができ，また (2) より生成規則が再帰的に適用されることがないので，$L(\mathcal{G})$ に属する木はただ一つである．

定義 2.8 SLCFTG $\mathcal{G} = (N, \Sigma, R, S)$, $t \in T(\Sigma \cup N, \mathcal{X})$ とする．$t \Rightarrow_{\mathcal{G}}^{*} t' \in T(\Sigma, \mathcal{X})$ となるただ一つの t' に対して，$||t|| = |t'|$ と定義し，t の展開サイズと呼ぶ．また，$||A(x_1, \ldots, x_n)||$ を $||A||$ と略記する． □

SLCFTG $\mathcal{G}$ における非終端記号 $A \in N$ について，$A \to t_A \in R$ としたとき，t_A の任意の部分木 $t' = f(t_1, \ldots, t_n)$ について，$||t_1||, \ldots, ||t_n||$ が計算できているなら $||t'||$ は $O(n)$ で計算することができる．したがって，ボトムアップに計算を行うことで，規則の右辺の全ての部分木について，その展開サイズを合わせて $O(|\mathcal{G}|)$ で計算することが可能である．

定義 2.9 木 $t \in T(\Sigma, \mathcal{X})$ について，t に出現する変数を先行順に $x_1, \ldots, x_n$ とする．各 $i (1 \leqq i < n)$ について，$size_{tree}(t, i)$ を先行順番号が x_i よりも大きくかつ x_{i+1} より小さい終端記号の出現数と定義する．また，$size_{tree}(t, 0)$ は先行順番号が x_1 より小さい終端記号の出現数，$size_{tree}(t, n)$ は先行順番号が x_n よりも大きい終端記号の出現数とする． □

定義 2.10 SLCFTG $\mathcal{G}$ における非終端記号 $A \in N$ について，$0 \leqq i \leqq rank(A)$, $A(x_1, \ldots, x_n) \to t_A \in R$, $A(x_1, \ldots, x_n) \Rightarrow_{\mathcal{G}}^{*} t_A^{*} \in T(\Sigma, \mathcal{X}_n)$ としたとき，$size_{direct}(A, i) = size_{tree}(t_A, i)$, $size(A, i) = size_{tree}(t_A^{*}, i)$ とする． □

非終端記号 A と $1 \leqq i \leqq rank(A)$ について，$size_{direct}(A, i)$ と $size(A, i)$ は全ての非終端記号に対して，合わせて $O(|\mathcal{G}|)$ で計算できる．

2.2.2 TreeRePair

SLCFTG を用いた木の圧縮法の一つに TreeRePair が知られている [6]．TreeRePair はダイグラム (digram) と呼ばれる頂点間の関係を用いて木を SLCFTG に変換する．ダイグラムとは，葉ではない頂点のラベルを f，その頂点の i 番目の子頂点のラベルを g としたとき，(f, i, g) のことをいう．木に対して 2 回以上出現するダイグラムのうち最頻出するものを非終端記号に置き換える操作を，そのようなダイグラムが出現しなくなるまで繰り返す．このようにして，木を SLCFTG に変換することを，木圧縮，または単に圧縮とよぶ．逆に，圧縮された SLCFTG $\mathcal{G}$ から元の木を得ること，すなわち，$\mathcal{G}$ から $L(\mathcal{G}) = \{t\}$ である t を求めることを解凍という．

2.3 決定性選択グラフオートマトン

定義 2.11 Σ 上の決定性選択グラフオートマトン (Deterministic Selecting Graph Automaton, DSGA) は 4 項組 $A = (Q, q_0, \delta, S)$ である．ここで，

(1) Q は状態の有限集合．
(2) $q_0 \in Q$ は初期状態．
(3) $\delta : \Sigma \times Q \to Q$ は状態遷移関数．
(4) $S \subseteqq \Sigma \times Q$ は頂点選択の指定．

□

定義 2.12 グラフ $G = (V, E, v_0, \Sigma, lab_G)$，DSGA $A = (Q, q_0, \delta, S)$ について，以下の条件を満たすグラフ r のことを，G に対する A の実行という．

(1) r は G と同型であり，$r = (V, E, v_0, 2^Q, lab_r)$．
(2) $lab_r : V \to 2^Q$ は，以下の条件を満たす最小の状態集合族である．

- $lab_r(v) = \cup_{(u,v)\in E}\{q' \mid \exists q \in lab_r(u).\delta(lab_G(u), q) = q'\}$.
- $q_0 \in lab_r(v_0)$.

G に対する A の実行を r，$q \in lab_r(v)$，$lab_G(v) = f$ としたとき，$(f, q) \in S$ であるならば，頂点 $v \in V$ は A によって選択される，または v は頂点選択であるという．頂点選択全部からなる集合を $A(G)$ とかく． □

3. 圧 縮 法

本節では，グラフ G が与えられたとき，SLCFTG を得るための手法を提案する．基本的な方針として，グラフ G に対して後述する合流頂点分解を行なうことでグラフを複数の木からなる森に変換し，TreeRePair アルゴリズムにより SLCFTG に変換する．

3.1 TreeRePair を用いたグラフ圧縮法の概要

TreeRePair は木に対して SLCFTG を得るアルゴリズムであるため，与えられたグラフが木でない場合は直接圧縮することができない．また，TreeRePair は最頻出するダイグラムを非終端記号に置き換えて SLCFTG に変換するアルゴリズムであるが，頂点に同じラベルがほとんど出現しないときは非終端記号の置き換えができないため，効率的な圧縮ができない場合もある．以上の問題を解決するため，次のようなアルゴリズムでグラフから SLCFTG を得る．

(1) グラフを DAG に変換する．
(2) グラフの頂点のラベルをすべて同じラベルに書き換え，辺を頂点に置き換える．
(3) グラフを合流頂点分解することで森を構成し，各木に対して TreeRePair アルゴリズムを適用する．

根付き連結グラフ $G = (V, E, v_0, \Sigma, lab_G)$ から，深さ優先探索における上昇辺の向きを逆にすることで DAG $G_{DAG} = (V, E_{DAG}, v_0, \Sigma, lab_G)$ を得る．このとき，逆向きにした辺については解凍時や直接問合せをする際に必要となってくるので，逆辺集合 R として保存しておく．なお，逆辺は根頂点からの先行順番号の組で保存する．

(1) の変換で得られた DAG に対して，(2) の変換で得られるグラフの例を図 1 に載せる．置き換える前のラベルの情報は解凍時に復元できるようにするため，根頂点からの先行順番号との組で保存する．また，辺にラベルがついているグラフの場合，TreeRePair にそのまま入力として与えると辺のラベルの情報が失われてしまうため，辺を頂点に置き換える処理を行い，辺

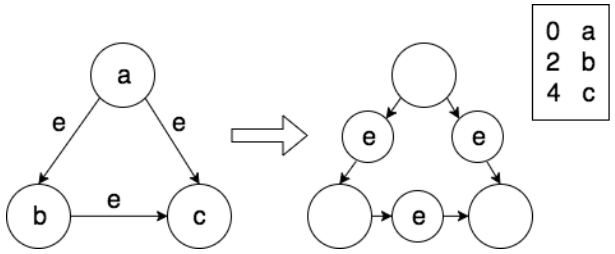

図 1　(2) の具体例

のラベルを保存するようにする．

最後に，合流頂点分解によってグラフを森に分解し，各木に対して TreeRePair アルゴリズムを適用することで SLCFTG $\mathcal{G}$ を得ることができる．

3.2　合流頂点分解

合流頂点分解とは，DAG $G = (V, E, v_0, \Sigma, lab_G)$ をラベル付き順序木の集合に変換するアルゴリズムである．基本的な方針は，G 上の合流頂点 v において，v から到達可能であるような部分グラフを G から切り分けるということを，全ての頂点の入次数が 1 以下になるまで繰り返す．

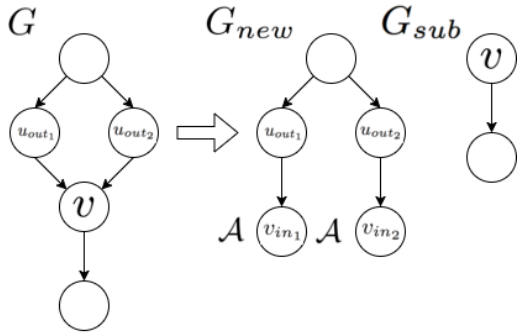

図 2　合流頂点分解の例

図 2 に示すように，ある合流頂点 v において合流頂点分解を行うとは，$n = dg_{in}(v)$ としたとき，v から到達可能であるような部分グラフ $G_{sub} = (V_{sub}, E_{sub}, v, \Sigma, lab_G)$ を複製し，$G_{new} = ((V \setminus V_{sub}) \cup \{v_{in_1}, \ldots, v_{in_n}\}, (E \setminus (E_{sub} \cup \{(u_{out_1}, v), \ldots, (u_{out_n}, v)\})) \cup \{(u_{out_1}, v_{in_1}), \ldots, (u_{out_n}, v_{in_n})\}, \Sigma, lab_{G_{new}})$ とすることである．ここで，$u_{out_1}, \ldots, u_{out_n}$ とは v の親頂点を表しており，$lab_{G_{new}}$ は以下のようなラベル付け関数である．

- $\forall u \in V. lab_{G_{new}}(u) = lab_G(u)$
- $\forall u \in \{v_{in_1}, \ldots, v_{in_n}\}. lab_{G_{new}}(u) = \mathcal{A}$

$\mathcal{A}$ は合流非終端記号とよぶ．

ある合流頂点 v について合流頂点分解を行うとき，v から到達可能であるような頂点のうち，v 以外の頂点で入次数が 2 以上の頂点 $v_{dg_{out_2}}$ がある場合，$(u, v_{dg_{out_2}}) \in E$ でありかつ v から到達不能な頂点 u が存在する可能性がでてきてしまう．このような場合に任意の合流頂点から合流頂点分解を行ってしまうと不整合が生じる可能性があるため，トポロジカル順序の逆順で合流頂点分解を行う必要がある．

4.　問合せ評価法

本節では，圧縮データとして SLCFTG $\mathcal{G} = (N, \Sigma, R, S)$，逆辺集合 R，問合せ DSGA $A = (Q, q_0, \delta, S_A)$ が与えられたとき，$\mathcal{G}$ を解凍することなく，A によって選択される頂点集合を求める手法を提案する．

4.1　評価法の概要

評価法の基本方針は，$\mathcal{G}$ の根頂点からトップダウンに走査しながら状態を割り振っていき，頂点選択集合を求めるというものである．$\mathcal{G}$ 上の走査中の頂点 v が，解凍された木 $t_{\mathcal{G}}$ のどの頂点と対応しているかを管理するため，頂点 v に状態を割り当てながら，$t_{\mathcal{G}}$ 上での先行順番号 π_v を更新する．頂点 v のラベルが非終端記号 A の場合は，v に状態を割り振ったあと，その非終端記号の右辺の木 t_A に移り同様に状態を割り振っていく．t_A に対する状態割り当てが終了し，t_A に出現する変数 $x_1, \ldots, x_n$ に割り当てられた状態が $q_1, \ldots, q_n$ であったとき，v に戻り，v の実引数 $t_1, \ldots, t_n$ の根に対し状態 $q_1, \ldots, q_n$ を割り当てる．その後，部分木 t_i に対してトップダウンに状態を割り当てていく．頂点 v のラベルが合流非終端記号 $\mathcal{A}$ の場合は，非終端記号の場合と同様に，$\mathcal{A}$ に対応する部分グラフを表す木文法の根頂点から同様に状態を割り振っていく．

ここで，状態割り当てを行なっている頂点を v とし，$(\pi_v, \pi_{v'}) \in R$ であるような辺 (v, v') について考える．(v, v') は元のグラフでは (v', v) という辺であるため，v' のラベル $lab_G(v')$ と v' に割り当てられた状態集合 $Q_{v'}$ から，次に割り当てる状態集合 Q_{next} を計算し，頂点 v に割り当てる．さらに頂点 v の元のグラフ上での v の親頂点 v_p に対して同様の状態割り当てを行い，v_p から再びトップダウンに状態を割り当てることを行う．ここで，TreeRePair は入力された木を fcns 符号化した二分木に対して圧縮を行うため，頂点 v の元のグラフ上での親頂点と文法上の木における親頂点が異なることがある (図 3)．TreeRePair に入力として与えるグラフは辺を頂点に置

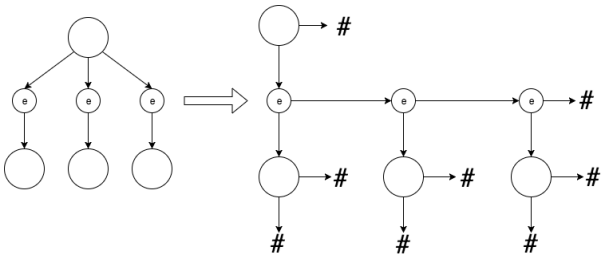

図 3　fcns 符号化により親頂点が変化する例

き換えたものであるので，fcns 符号化の定義より，頂点 v の元のグラフ上での親頂点は，元のグラフ上で頂点であったもののうち，木文法上での直近の先祖にあたる．したがって，元のグラフ上での頂点であった頂点を記憶しながらトップダウンで木文法の各頂点に状態を割り振ることにより，頂点 v の元のグラフ上での親頂点を定数時間で特定することができる．

4.2　メモ化による重複評価の回避

ラベルが非終端記号 $A \in N$ である頂点 v_A に対し状態割り当

てを行う際，A の右辺の木 t_A に対する状態割り当ては，v_A に割り当てられる状態が同じときは一致する．したがって，各非終端記号 $A \in N$ について，(A, A に割り当てられた状態 q) をキーとして，(A の右辺に出現する選択頂点集合，$A \to t_A \in R$ の右辺 t_A に出現する変数へ割り当てられた状態) をマップ $map_{\mathcal{G}}$ によりメモ化することで，再び A にその状態が割り当てられたときには t_A の評価は行わず，メモ化された情報から A の実引数に割り当てられる状態を取得することで，t_A に対する重複評価を回避する．

ただし，逆辺を (u, v) としたとき，t_A 中に頂点 u が出現する場合はメモ化による重複回避を行なった後，4.1 節で述べた逆辺処理を行う．

5. 評価実験

本節では，Wikipedia から情報を構造化データ (RDF グラフ) として抽出するプロジェクト DBPedia が公開しているデータセット [8] と，スタンフォード大学が公開している SNS の交友関係やネットワークを表現したグラフデータセット [9] を用いて，提案手法でグラフを SLCFTG に変換したときの圧縮率について示すとともに，圧縮後のファイルサイズに関して先行研究である gRePair [7] と比較を行う．

5.1 実験環境とデータセット

実験環境を以下に示す．

- OS:Ubuntu 14.04.5 LTS 64bit
- CPU:Intel®Core™ i5-3470 3.20GHz
- メモリ:8GB

提案手法は c++で実装し，コンパイラは g++4.8.4 を使用した．実験に使用したグラフに関する情報を表 1 に示す．なお，#1 から#4 までのグラフは頂点と辺にラベルがついているような RDF グラフで，#5 から#7 までのグラフは頂点と辺にラベルがついていないようなグラフである．なお，実装したプログラムの都合上，#5 から#7 までのグラフは辺に空文字列のラベルがついているものとして圧縮を行なった．

表 1　実験に用いたグラフ

#	ファイル名	ファイルサイズ (kb)	頂点数	辺数	辺ラベルの種類
1	Types ru	148,386	642,378	642,364	1
2	Types es	121,161	818,833	819,780	1
3	Geographic pt	2,608	15,743	18,316	4
4	Bookmashup	1,269	8,794	8,903	2
5	email-EuAll	5,705	265,214	420,045	1
6	wiki-Vote	1,272	7,115	103,689	1
7	p2p-Gnutella06	318	8,114	26,013	1

5.2 圧縮率

今回用いたデータセットでは，#1 から#4 についてはグラフの頂点にラベルがついており，それぞれのラベルが各頂点に固有であったため，そのまま圧縮を行うと 2 回以上現れるダイグラムが存在せず，圧縮率が大幅に下がってしまうため，3. 節で触れたように，各頂点のラベルについては別途保存したのち，頂点のラベルを全て同一視して圧縮を行なった．

提案手法で圧縮した結果を表 2 に示す．ここでファイルサイズとは，圧縮で得られた SLCFTG を保存したファイル，逆辺情報を保存したファイル，頂点のラベル情報を保存したファイルの合計サイズとなっている．

表 2　圧縮した結果

#	SLCFTG + 逆辺 (kb)	頂点ラベル (kb)	ファイルサイズ (kb)	圧縮率
1	275	94,352	94,627	0.634
2	481	50,912	51,393	0.424
3	43	838	881	0.338
4	4	536	540	0.425
5	1,366	0	1,366	0.239
6	425	0	425	0.334
7	215	0	215	0.677

グラフ圧縮の先行研究である gRePair [7] では#1,#2,#5 のファイルについて，頂点のラベルを削除したものを圧縮に用いており，圧縮後のファイルサイズはそれぞれ 1kB，3kB，329kB であった．同条件で比較をすると，圧縮後のファイルサイズはそれぞれ表 2 から 275kB,481kB,1366kB となり，圧縮率に関しては gRePair と比べて劣っていることがわかった．提案手法では合流頂点分解を行った木に関して TreeRePair による圧縮を行うが，本来ならダイグラムとして検出されるものが合流頂点分解で分断されてしまい，効率よく圧縮できなかったことが原因だと考えられる．

#5 から#7 のデータは，圧縮後のデータサイズ (SLCFTG と逆辺の合計ファイルサイズ) と元のファイルサイズの比が#1 から#4 のデータと比べて大きくなっているため，さらに詳しく圧縮後のデータについて調査した．図 4 は，合流頂点分解して得られた部分グラフのファイルサイズ毎の圧縮率を示したものである．この図から，木文法の開始非終端記号のラベルなど，元のグラフに出現しないようなものも圧縮ファイルに含めるため，圧縮率が 1 より大きくなってしまうことがわかった．また，ファイルサイズが小さいものはあまり圧縮率がよくなく，大きなものは比較的圧縮率がよいことが確認できる．ファイルサイズが小さいものに関して，どのようなグラフの形になっているかを確認したところ，深さが浅く，部分グラフ中の頂点のラベルがあまり重複していないようなものが大半であった．また，部分グラフの頂点のラベルは多くの場合，元のグラフの辺のラ

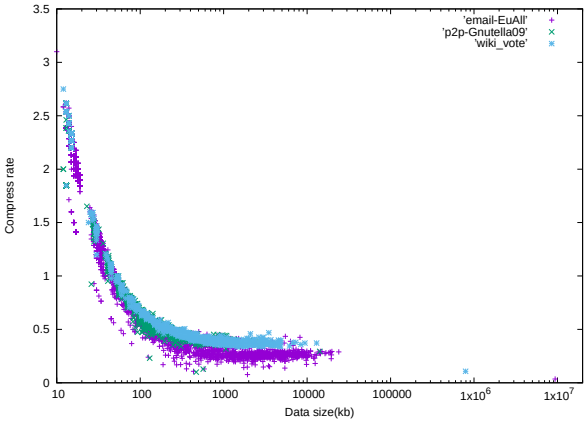

図 4　ファイルサイズ毎の圧縮率

ベルではなく，合流頂点分解をした際に生じる合流非終端記号であった．TreeRePair は共通するダイグラムが少ないとあまり効率よく圧縮できないので，ファイルサイズが小さい部分グラフに対しての効率良い圧縮法を考える必要があると思われる．

6. あとがき

6.1 本研究の成果

本研究では，グラフ圧縮法として，与えられたグラフ G を合流頂点分解し，得られた森に対して木構造圧縮法 TreeRePair を用いて SLCFTG $\mathcal{G}$ に圧縮し，逆辺集合 R を構成して $(\mathcal{G}, R)$ を圧縮データとする手法を提案した．次に，圧縮データ $(\mathcal{G}, R)$ と DSGA で記述された頂点選択問合せ A が与えられたとき，$\mathcal{G}$ を解凍することなく，直接問合せ処理を行うアルゴリズムを提案した．また，提案した圧縮法を実装し，圧縮率について評価を行なった．

評価の結果，先行研究である gRePair と比べて圧縮率が劣っていることがわかった．また，合流頂点分解をした際に多くの小さな部分グラフが生成され，各部分グラフ内で共通するダイグラムの数が少ないことに起因して圧縮率が低下することがわかった．

6.2 今後の課題

本研究で提案した圧縮法は 5.2 節で述べたように，部分グラフ内で共通するダイグラムが少ないことに起因して圧縮率が低下することがわかったが，一方で頂点のラベルが合流非終端記号が多いこともわかった．合流非終端記号はグラフ全体では必ず 2 つ以上存在するため，それらでダイグラムを形成するようにすれば，より効率よく圧縮できることが見込める．

また，本稿では圧縮データに対する直接問合せ評価法を提案したが，実装に基づく評価実験は行なっていないため，実行時間・メモリ量の観点から実際に実装を行い評価する必要がある．一般に，実データを表現したグラフは頂点のラベルを書き換えたり，グラフの構造を変えるといった更新操作も行われる．したがって，圧縮データに対する直接更新についても考える必要がある．

文　献

[1] P. Boldi and S. Vigna. The webgraph framework I: compression techniques, *World Wide Web*(WWW), pp. 595–602, 2004.

[2] H. Common, M. Dauchet, R. Gileron, F. Jacquemard, D. Lugiez, C. Löding, S. Tison and M. Tommasi. Tree automata techniques and applications, 2008, http://tata.gforge.inria.fr/

[3] W. Fan. Graph pattern matching revised for social network analysis, *International Conference on Database Theory* (ICDT), pp. 8–21, 2012.

[4] W. Fan, J. Li, X. Wang and Y. Wu. Query preserving graph compression, *Special Interest Group on Management of Data*(SIGMOD), pp. 157–168, 2012.

[5] U. Kang, C. E. Tsourakakis and C. Faloutsos. PEGASUS: mining peta-scale graphs, *Knowledge and Information Systems*(KAIS), pp. 303–325, 2011.

[6] M. Lohrey, S. Maneth and R. Mennicke. XML tree structure compression using RePair, *Information Systems*(IS), Vol. 38, pp. 1150–1167, 2013, https://code.google.com/p/treerepair/.

[7] S. Maneth and F. Peternek. Compressing graphs by grammars, *International Conference on Data Engineering*(ICDE), pp.109–120, 2016.

[8] DBpedia. http://wiki/dbpedia.org/Downloads2015-04

[9] Stanford Large Network Dataset Collection. http://snap.stanford.edu/data/index.html