辺縮約に基づく木構造パターンの列挙とそのデータマイニングへの応用

村井 光[1]　吉村 友太[2]　岡本 康宏[2]　正代 隆義[3,a)]　宮原 哲浩[4]

概要：本論文では，木構造データからの効率的かつ効果的な木構造パターンマイニングを目的として導入された辺縮約に基づく無順序木構造パターンである木縮約パターン (TC-パターン) の照合アルゴリズムおよび列挙アルゴリズムを提案する．TC-パターンは，無順序木のいくつかの頂点を縮約可能頂点と指定した木構造パターンである．無順序木が TC-パターンに照合するとは，辺縮約により無順序木の連結部分木が縮約可能頂点に縮約でき，その結果 TC-パターンと同型にすることが可能なときをいう．本論文では，木縮約パターンの多項式時間照合アルゴリズムと最悪多項式時間遅延列挙アルゴリズムを提案する．また，両アルゴリズムの実験的評価を報告する．

キーワード：木構造パターン，辺縮約，パターン照合，列挙アルゴリズム，グラフマイニング，

Enumeration of Unordered Tree Contraction Patterns and Its Application to Data Mining

Hikaru Murai[1]　Yuta Yoshimura[2]　Yasuhiro Okamoto[2]　Takayoshi Shoudai[3,a)]
Tetsuhiro Miyahara[4]

Abstract: In this paper, we present a concept of edge contraction-based tree-structured patterns as a graph pattern suited to represent tree-structured data. A *tree contraction pattern* (*TC-pattern*) is an unordered tree-structured pattern common to a given tree-structured data, which is obtained by merging every uncommon connected substructure into one vertex by edge contraction. In this paper, in order to establish an algorithmic foundation for the discovery of knowledge from tree-structured data, we propose a polynomial-time TC-pattern matching algorithm and an worst-case polynomial-time-delay enumeration algorithm. Moreover, we report experimental results evaluating the performance of our algorithms.

Keywords: Tree structured pattern, edge contraction, pattern matching, enumeration, graph mining

1. はじめに

大規模グラフデータからの半構造データマイニングが関心を集めている理由として，インターネットのリンク構造や，テキストマイニング，有機化合物，遺伝子ネットワークなど，非定型データが多くなっていることが挙げられる．これらのデータから効率よくデータマイニングするためには，半構造データを表現するパターンの列挙アルゴリズムの開発が必要である．本論文では，浅井ら [1] にならって，半構造データを，頂点とその関係を表す辺を使った木構造として考える．これらのラベル付き木から特徴的な木構造パターンを発見することが本論文の目標である．

木構造データ中から有意義かつ隠れた知識を抽出するためには，それらを表現することのできる木構造パターンが必要である．無順序木構造パターンとしては，木パター

[1] 九州大学理学部物理学科情報理学コース
[2] 九州大学大学院システム情報科学府
[3] 九州大学大学院システム情報科学研究院
〒 819-0395 福岡市西区元岡 744 番地
[4] 広島市立大学大学院情報科学研究科
〒 731-3194 広島市安佐南区大塚東 3-4-1
[a)] shoudai@inf.kyushu-u.ac.jp

ンや無順序項木 [3], [7] があるが，表現力や学習時間の観点から一長一短がある．本論文では，吉村ら [10] が提案した辺縮約に基づく木構造パターンである d-TC-パターンの列挙によるデータマイニングを考察する．このグラフパターンは既存の木構造パターンとは異なる木表現が可能で，正例から多項式時間帰納推論可能であることが示されている [6]．d を自然数とする．d-TC-パターンは 3 つ組 $t = (V_t, E_t, U_t)$ である．ここで，(V_t, E_t) は特定の根 r_t をもつ無順序木であり，U_t は V_t の部分集合で，U_t に属す頂点は d 個以下の子頂点しか持たない．U_t に属す頂点を縮約可能頂点，$V_t \setminus U_t$ に属す頂点を縮約不能頂点とよぶ．与えられた d-TC-パターン t と無順序木 T に対して，T が t に照合するかどうか判定する問題を d-TC-パターン照合問題とよぶ．我々は，[6], [10] で d-TC-パターン照合問題が $O(nN^{\max\{d-1,1.5\}})$ 時間で計算可能であることを示した．ここで，n と N はそれぞれ，t と T の頂点数を表す．本論文では，同問題が $O(nN(n+\sqrt{N}))$ 時間で計算可能であることを示す．従って，d-TC-パターン照合問題はパラメータ d に関して FPT アルゴリズム (Fixed Parameter Tractable Algorithm) を持つことになる．

現在までにいくつかの列挙アルゴリズムの技法が知られている [5], [8]．無順序木を効率良く列挙するアルゴリズムとして，中野と宇野 [4] のアルゴリズムが知られている．このアルゴリズムは無順序木 1 つあたり $O(1)$ 時間で計算するアルゴリズムである．従って，列挙アルゴリズムの効率を図る尺度では，最も望ましいものの一つであるといえる．また，浅井ら [1] は，このアルゴリズムをラベル付き順序木を列挙するアルゴリズムに拡張している．浅井ら [1] のアルゴリズムもうまくデータ構造を構築することで，ラベル付き順序木を 1 つあたり $O(1)$ 時間で計算する．

d-TC-パターン照合問題に対する FPT アルゴリズムの提案に加えて，本論文では，浅井ら [1] のラベル付き順序木を列挙するアルゴリズムを基に，d-TC-パターンの列挙アルゴリズムを構築する．d-TC-パターンは頂点ラベル・辺ラベルを無視しても縮約可能頂点と縮約不能頂点の 2 種類の頂点があり，さらに，縮約可能頂点にはその子頂点の数に定数制限があるため，効率良く列挙するためには工夫が必要である．本論文では，d-TC-パターンを列挙する最悪多項式時間遅延列挙アルゴリズムを提案する．さらに，そのアルゴリズムを実装し，列挙に必要な計算時間を検証する．また，d-TC-パターンを糖鎖データのクラス分類問題に適用し，d-TC-パターンの有効性を確かめる．

2. TC-パターン

グラフ G の頂点集合，辺集合をそれぞれ $V(G)$，$E(G)$ と記す．頂点集合 $V(G)$ の部分集合 V' に対して，V' の誘導部分グラフを $G[V']$ と表す．

定義 1 G と H を連結グラフとする．H-証拠構造とは，次の条件を満たす H の頂点集合の部分集合族 $\mathcal{W} = \{W(u) \mid u \in V(H)\}$ である．

(1) 任意の $u \in V(H)$ に対して，$G[W(u)]$ が連結である．
(2) 任意の $u, u' \in V(H)$ $(u \neq u')$ に対して，$\{u, u'\} \in E(H)$ であるとき，かつそのときに限り $\{v, v'\} \in E(G)$ が存在して，$v \in W(u)$ かつ $v' \in W(u')$ が成り立つ．

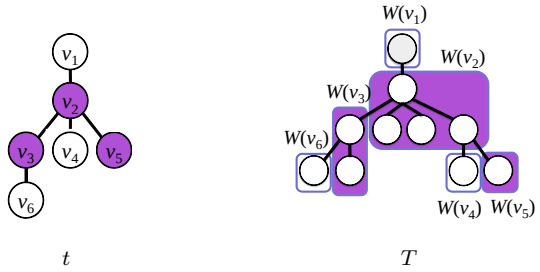

図 1 3-TC-pattern $t = (V(t), E(t), U(t))$ とそれに照合する無順序木 $T = (V(T), E(T))$．$\mathcal{W} = \{W(v_1), \ldots, W(v_6)\}$ は T の t-証拠構造である．

H-証拠構造 $\mathcal{W}$ に属す各集合 $W(u) \in \mathcal{W}$ を，u の H-証拠集合とよぶ．グラフ G が H-証拠構造を持つとき，G は H へ，各 H-証拠集合をひとつの頂点へと辺縮約することによって，変換できる．グラフ縮約パターン (GC-パターン) とは，3 つ組 $h = (V, E, U)$ で，(V, E) は連結グラフを表し，U は V の部分集合である．Σ を有限アルファベットとする．GC-パターン h において，h の頂点から Σ の要素への関数 (頂点ラベリングとよぶ) を $\varphi_h : V \to \Sigma$ とおく．V の部分集合 V' に対して，$h[V']$ を V' から誘導されるグラフ縮約部分パターン (GC-部分パターン) を表す．すなわち，$h[V'] = (V', E \cap \{\{v, w\} \mid v, w \in V'\}, U \cap V')$ である．U に属す頂点を縮約可能頂点，$V(h) \setminus U$ の頂点を縮約不能頂点とよぶ．

GC-パターン h に対して，$V(h)$ で h の頂点集合を，$E(h)$ で h の辺集合を，$U(h)$ で h の縮約可能頂点全体を表す．

定義 2 h と g を GC-パターンとし，h と g の頂点ラベリングをそれぞれ φ_h と φ_g とする．$H = (V(h), E(h))$，$G = (V(g), E(g))$ とおく．もし G の H-証拠構造 $\mathcal{W} = \{W(u) \mid u \in V(h)\}$ が存在して，全ての $v \in V(h) \setminus U(h)$ に対して，$W(v)$ がちょうど 1 つの頂点 $u \in V(g) \setminus U(g)$ を含み，さらに $\varphi_g(u) = \varphi_h(v)$ ならば，$h \preceq g$ と書く．このとき，G の H-証拠構造 $\mathcal{W}$ を，g の h-証拠構造とよぶ．

通常のグラフ G は縮約可能頂点全体が空であるような GC-パターンであるとみなす．このとき，GC-パターン h と連結グラフ G に対して，G が h に照合するとは，$h \preceq G$ が成り立つときをいう．

定義 3 GC-パターン t が TC-パターンであるとは，$(V(t), E(t))$ が根付き無順序木であるときをいう．根 r_T を持つ無順序木 T が，根 r_t を持つ TC-パターン t に照合するとは，T の t-証拠構造 $\mathcal{W} = \{W(u) \mid u \in V(t)\}$ が存在して，$r_T \in W(r_t)$ が成り立つときをいう．

例えば，図 1 において，TC-パターン t と無順序木 T は，定義 3 の条件を満たす T の t-証拠構造 $\mathcal{W} = \{W(v_1),\ldots,W(v_6)\}$ を持つ．$\mathcal{W}$ の証拠集合は四角で囲まれた領域で示されている．すなわち T は t に照合する．

2.1 集合族による集合被覆問題

TC-パターンの照合アルゴリズムを構築するにあたって，次の計算問題は重要である．

集合族による集合被覆問題 (SET COVER BY COLLECTIONS, SCC)

入力: 要素数 d の集合 X と m 個の集合族 $\mathcal{C}_1,\ldots,\mathcal{C}_m$．各 $\mathcal{C}_i$ は X の部分集合からなる集合族である $(1 \le i \le m)$.

問題: m 個の集合族の中から，次の条件を満たすいくつかの集合族 $\mathcal{C}_{j_1},\ldots,\mathcal{C}_{j_r}$ $(1 \le j_1 < \cdots < j_r \le m)$ を選択せよ: 各集合族 $\mathcal{C}_{j_1}$ からひとつだけ集合 X_{j_ℓ} を選べば $\bigcup_{\ell=1}^{r} X_{j_\ell} = X$ となる．

本問題を **SCC** と略す．この問題は NP 完全である．実際，よく知られた NP 完全問題のひとつである **X3C** の入力 $(X,\mathcal{C})$ $(|X| = 3q)$ から多項式時間帰着を次のように簡単に構成できる．全ての j $(1 \le j \le m)$ に対して，$m = q$ かつ $\mathcal{C}_j = \mathcal{C}$ とする．このとき，入力 $(X,\mathcal{C})$ に対する **X3C** の答えが *yes* のとき，かつそのときに限り入力 $(X,\mathcal{C}_1,\ldots,\mathcal{C}_m)$ に対する **SCC** の答えが *yes* となる．

SCC について次の補題を与える．

補題 1 **SCC** の入力を $(X,\mathcal{C}_1,\ldots,\mathcal{C}_m)$ $(|X| = d)$ とする．

(1) もし全ての $\mathcal{C}_j$ $(1 \le j \le m)$ が X の単集合 (singleton set) から構成されていれば，**SCC** は $O(d\sum_{j=1}^{m}|\mathcal{C}_j| + dm\sqrt{d+m})$ 時間で計算できる．

(2) d が定数であれば，**SCC** は $O(\sum_{j=1}^{m}|\mathcal{C}_j| + m\sqrt{m})$ 時間で計算できる．

証明

(1) 次のような二部グラフ G を構成する．G の頂点集合を $V(G) = X \cup \{\mathcal{C}_1,\ldots,\mathcal{C}_m\}$ とし，辺集合を $E(G) = \{\{x,\mathcal{C}_j\} \mid x \in X \text{ and } \{x\} \in \mathcal{C}_j\ (1 \le j \le m)\}$ とする．**SCC** の入力から，このような二部グラフ G を構成するには $O(d\sum_{j=1}^{m}|\mathcal{C}_j|)$ 時間が必要である．この二部グラフ G に対して，最大二部グラフマッチングを求めることにより，**SCC** の答えが得られる．これには Hopcroft と Karp のアルゴリズム [2] を実行すればよい．従って，全体の計算時間は $O(E(G)\sqrt{V(G)}) = O(dm\sqrt{d+m})$ 時間である．得られた最大二部グラフマッチングの要素数が d のとき，かつそのときに限り **SCC** の答えは *yes* である．

(2) 任意の d' $(d' = 1,\ldots,d)$ と任意の X の分割 $\{X_1,\ldots,X_{d'}\}$ に対して，二部グラフ G を構成する．G の頂点集合を $V(G) = \{X_1,\ldots,X_{d'}\} \cup \{\mathcal{C}_1,\ldots,\mathcal{C}_m\}$ とし，辺集合を $E(G) = \{\{X_i,\mathcal{C}_j\} \mid \exists C \in \mathcal{C}_j \text{ s.t. } X_i \subseteq C\}$ とする．このように G を構成するために $O(\sum_{i=1}^{d'}\sum_{j=1}^{m}|X_i||\mathcal{C}_j|) = O(d\sum_{j=1}^{m}|\mathcal{C}_j|)$ 時間が必要である．次に G の最大二部グラフマッチングを求める．このとき，**SCC** の答えが *yes* のとき，かつそのときに限り，X の分割 $\{X_1,\ldots,X_{d'}\}$ が存在して，最大二部グラフマッチングの要素数が d' となる．X $(|X| = d)$ の分割の総数は d 番目のベル数 B_d であるので，全体の計算時間は $O(B_d \cdot (d\sum_{j=1}^{m}|\mathcal{C}_j| + dm\sqrt{d+m}))$ 時間となる．d を定数とみなすことが出来れば，この計算時間は $O(\sum_{j=1}^{m}|\mathcal{C}_j| + m\sqrt{m})$ となる．

□

以降，$p_{scc}(X,\mathcal{C}_1,\ldots,\mathcal{C}_m)$ は次のような述語を表す．$p_{scc}(X,\mathcal{C}_1,\ldots,\mathcal{C}_m) = \mathit{true}$ のとき，かつそのときに限り **SCC** の入力 $(X,\mathcal{C}_1,\ldots,\mathcal{C}_m)$ に対する答えは *yes* である．

2.2 d-TC-パターン照合アルゴリズム

t を d-TC-パターン，T を無順序木とする．t の根を r_t とする．t の頂点 v に対して，t における v の子頂点数を $d_t(v)$ と書く．また，t の頂点 v に対して，$t[v]$ で v とその子孫から誘導される t の誘導部分 d-TC-パターンを表す．

本節では，与えられた無順序木 T と d-TC-パターン t が照合するか否かを判定する TC-パターン照合アルゴリズムを与える．t の各頂点 v は異なる識別子を持つとし，それを $I_t(v)$ で表す．$c_{v,1},\ldots,c_{v,d_t(v)}$ を頂点 v の t における子頂点全体とし，$X_t(v) = \{I_t(c_{v,1}),\ldots,I_t(c_{v,d_t(v)})\}$ とする．

提案するアルゴリズムは，T の各頂点 u に集合族 $\mathcal{I}_T(u)$ と $\mathcal{J}_T(u)$ を計算して割り当てる過程である．$\mathcal{I}_T(u)$ と $\mathcal{J}_T(u)$ は次のような集合から計算される．$\mathcal{C}_T(u) = \mathcal{I}_T(u) \cup \mathcal{J}_T(u)$ とする．$\mathcal{I}_T(u) \subseteq \{\{I_t(v)\} \mid v \in V(t)\}$ であり，$\mathcal{J}_T(u) \subseteq \{P \subsetneq X_t(v) \mid v \in U(t)\}$ である．

アルゴリズム MATCHING;

入力: d-TC-パターン t と無順序木 T.

ステップ 1: (初期化) 全ての $u \in V(T)$ に対して，もし u が T の葉であれば，$\mathcal{I}_T(u) := \{\{I(v)\} \mid v \text{ は } t \text{ の葉}\}$ とし，$\mathcal{J}_T(u) := \emptyset$ とする．F そうでなければ，$\mathcal{I}_T(u) := \emptyset$，$\mathcal{J}_T(u) := \emptyset$ とする．$h := h_T - 1$ とする．ただし，h_T は T の高さである．

ステップ 2: (Iteration) 高さ h の全ての $u \in V(T)$ に対して，ステップ 2.1 から 2.4 までを行う:

2.1 全ての頂点 $v \in V(t) \setminus U(t)$ に対して，
if $p_{scc}(X_t(v),\mathcal{I}_T(c_{u,1}),\ldots,\mathcal{I}_T(c_{u,d_T(u)})) = \mathit{true} \wedge |X_t(v)| = d_T(u)$ **then** $\mathcal{I}_T(u) := \mathcal{I}_T(u) \cup \{\{I_t(v)\}\}$;

2.2 全ての頂点 $v \in U(t)$ に対して，
if $\exists j \in \{1,\ldots,d_T(u)\}$ s.t. $\{I_t(v)\} \in \mathcal{I}_T(c_{u,j})$ **then** $\mathcal{I}_T(u) := \mathcal{I}_T(u) \cup \{\{I_t(v)\}\}$;

2.3 全ての頂点 $v \in U(t)$ に対して，
if $p_{scc}(X_t(v),\mathcal{C}_T(c_{u,1}),\ldots,\mathcal{C}_T(c_{u,d_T(u)})) = \mathit{true}$

then $\mathcal{I}_T(u) := \mathcal{I}_T(u) \cup \{\{I_t(v)\}\}$;

2.4 全ての頂点 $v \in U(t)$ と全ての $X_t(v)$ の部分集合 X' $(1 \leq |X'| < |X_t(v)|)$ に対して，
if $p_{scc}(X', \mathcal{C}_T(c_{u,1}), \ldots, \mathcal{C}_T(c_{u,d_T(u)})) = true$ **then** $\mathcal{J}_T(u) := \mathcal{J}_T(u) \cup \{X'\}$;

ステップ 3: (判定) もし $h > 0$ ならば $h := h - 1$ としてステップ 2 に戻る．もし $\{I_t(r_t)\} \in \mathcal{I}_T(r_T)$ ならば T は t に照合する．そうでなければ T は t に照合しない．
(アルゴリズム終わり)

図 2 にアルゴリズム Matching の動きの例をあげる．次に本アルゴリズムの正当性と時間計算量について述べる．ページ数の都合上，補題 2–4 の証明は省略する．

補題 2 t を d-TC-パターン，T を無順序木とする．頂点 $u \in V(T)$ に対して，$X' = \{I_1, \ldots, I_{d'}\} \in \mathcal{J}_T(u)$ のとき，かつそのときに限り頂点 $v \in U(t)$ が存在して次の 2 条件が成り立つ．

(1) $X' \subsetneq X_t(v)$，かつ
(2) u の異なる真の子孫 $u_1, \ldots, u_{d'}$ があって，任意の 2 つの頂点は互いに子孫関係になく，かつ $\{I_i\} \in \mathcal{I}_T(u_i)$ である．

補題 3 t を d-TC-パターン，T を無順序木とする．任意の $v \in V(t)$ と $u \in V(T)$ に対して，$\{I_t(v)\} \in \mathcal{I}_T(u)$ であるとき，かつそのときに限り次の 4 つの条件のいずれかが成り立つ．

(1) $v \in V(t) \setminus U(t)$ であり，u と v はそれぞれ T と t の葉である．
(2) $v \in U(t)$ であり，v は t の葉である．
(3) $v \in V(t) \setminus U(t)$ であり，全単射 $\psi : \{1, \ldots, d_t(v)\} \to \{1, \ldots, d_T(u)\}$ が存在して，全ての i $(1 \leq i \leq d_t(v))$ に対して，$\{I_t(c_{v,i})\} \in \mathcal{I}_T(c_{u,\psi(i)})$ が成り立つ．
(4) $v \in U(t)$ であり，u の異なる真の子孫が $d_t(v)$ 個存在して，それを $u_1, \ldots, u_{d_t(v)}$ とするとき，任意の 2 つの頂点は互いに子孫関係になく，かつ $\{I_t(c_{v,i})\} \in \mathcal{I}_T(u_i)$ $(1 \leq i \leq d_t(v))$ となる．

補題 4 t を d-TC-パターン，T を無順序木とする．任意の組 $(v, u) \in V(t) \times V(T)$ に対して，$\{I_t(v)\} \in \mathcal{I}_T(u)$ のとき，かつそのときに限り $T[u]$ は $t[v]$ に照合する．

最後に，次の定理を得る．補題 4 から，$\{I_t(r_t)\} \in \mathcal{I}_T(r_T)$ のとき，かつそのときに限り T が t に照合することがわかる．また補題 1 より，アルゴリズム Algorithm の時間計算量も解析いできる．詳細な証明は省略する．

定理 1 d-TC-パターン照合問題は，d-TC-パターン t と無順序木 T を入力とするとき，$O(nN(n + \sqrt{N}))$ 時間で計算できる．ただし，$n = |V(t)|$，$N = |V(T)|$ とする．

3. d-TC-パターンの列挙アルゴリズム

本節では d-TC-パターンの列挙アルゴリズムを提案し，その計算量について述べる．列挙の際は，完全性と重複がないことを確認しなければならない．完全性とは，列挙の過程ですべての d-TC-パターンが必ず現れることをいい，重複がないとは，同型な d-TC-パターンが 2 回以上現れないことをいう．列挙アルゴリズムに対する入力のサイズを n，そのときの出力の数を N とする．このとき，列挙アルゴリズムの効率の良さを次のように分類する [5]．出力多項式時間列挙アルゴリズム (output-polynomial enumeration algorithm) とは，時間計算量が n と N に関する多項式で抑えられるアルゴリズムである．ならし多項式時間遅延列挙アルゴリズム (amortized polynomial-time-delay algorithm) とは，時間計算量が n に関する多項式，N に関する一次式で抑えられるアルゴリズムである．最悪時多項式時間遅延列挙アルゴリズム (worst-case polynomial-time-delay algorithm) とは，次の 3 つがどれも n に関する多項式で抑えられる列挙アルゴリズムである．

(1) アルゴリズムの実行開始から一つ目の出力が得られるまでの時間．
(2) 任意の i $(1 \leq i \leq N-1)$ に対して，i 番目の出力が得られてから，$i+1$ 番目の出力が得られるまでの時間．
(3) N 番目の出力が得られてから，アルゴリズムが停止するまでの時間．

3.1 d-TC-パターンの深さ列と正規形表現

順序木の深さ列とは，その順序木の頂点を深さ優先順に探索し，訪れた順に深さを書き並べて得られる数列のこととする．順序木は深さ列と一対一対応なので，任意の無順序木 T に対して兄弟関係を無視した際に得られる順序木 (T と同型) で，深さ列のある特定の条件を満たす木を考え標準形とする．$C(T)$ で順序木 T の深さ列を表す．d-TC-パターン $t = (V(t), E(t), U(t))$ に対し，順序木としての $(V(t), E(t))$ の深さ列を $(i_1, i_2, \ldots, i_{|V(t)|})$ とする．このとき，t の深さ列 $(i'_1, i'_2, \ldots, i'_{|V(t)|})$ を次のように定める．任意の j $(1 \leq j \leq |V(t)|)$ に対して，i_j に対応する頂点が $U(t)$ に属していれば，$i'_j = i_j + 0.5$, そうでなければ，$i'_j = i_j$ とする．

定義 4 (d-TC-パターンの左荷重条件) $t = (V(t), E(t), U(t))$ の任意の頂点 $v_1, v_2 \in V(t)$ に対して，v_1 と v_2 が兄弟関係ならば $C(t(v_1)) \geq_{lex} C(t(v_2))$ が成立するとき，d-TC-パターン t は左荷重条件を満たしているという．ここで $t(v)$ は，v の子孫からなる t の誘導部分 d-TC-パターンである．ここで，$\geq_{lex}$ は有理数列間の辞書式順序を意味する．

左荷重条件を満たす順序木 T を，順序木の兄弟関係を無視して得られる無順序木として同型な順序木の代表であると定めこれを正規形表現とよぶ．図 3 に 3 つの順序木の例を与える．これら 3 つの順序木は兄弟関係を無視して得られる無順序木としては同型である．左端の順序木の深さ列は，他の 2 つの順序木の深さ列にたいして辞書式順序で大

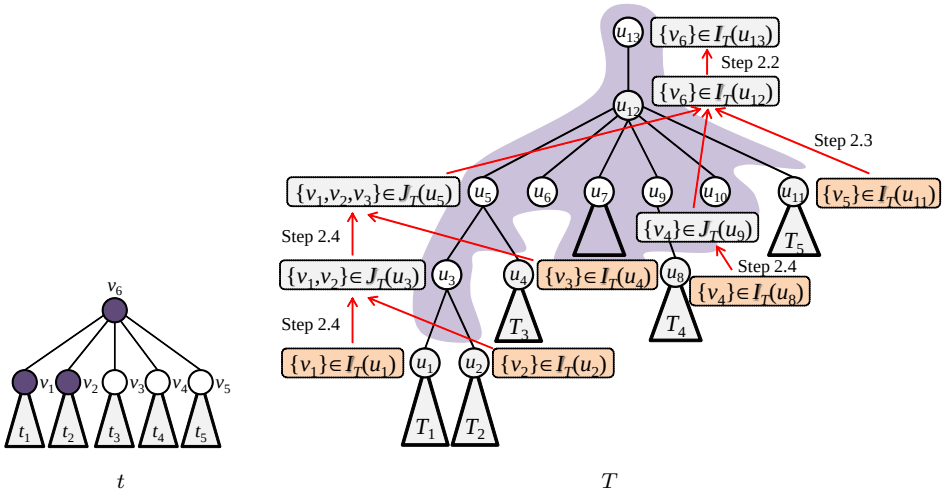

図 2 アルゴリズム MATCHING の処理の例: 部分順序木 $T_1, \ldots, T_5$ は d-TC-部分パターン $t_1, \ldots, t_5$ とそれぞれ照合すると仮定する．T の背景の濃色領域は v_6 の証拠集合を示す．

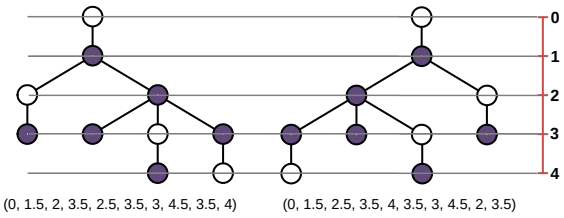

図 3 これらの 2-TC-パターンは順序木は同型である．右の 2-TC-パターンの深さ列は左の 2-TC-パターンの深さ列より辞書式順序で大きい．実際，右の順序木は，無順序木として同型な順序木の中での正規形表現である．

きい．実際，これらの兄弟関係を無視して得られる無順序木が同型な順序木のうち，最も大きい深さ列を持つ順序木が左端の順序木であり，それらのうちでこの順序木を正規形表現と定める．

中野と宇野 [4] によって開発された無順序木の最右拡張法とは，順序木の列挙のならし多項式時間遅延アルゴリズムである．これは同型の集合の中から一つの代表（正規形表現）のみを出力できるようなアルゴリズムで，逆探索を用いて根から頂点を追加していくことで無順序木の正規形表現を 1 つあたり定数時間で完全に，重複なく列挙することができる．同型の集合の中から一つの代表（正規形表現）のみを出力できるような条件で列挙していく．

正規形表現である d-TC-パターン t の頂点がアクティブであるとは，以下の 3 つの条件を満たしているときをいう．平面に d-TC-パターン t を兄弟が小さい順に左から並べるように描くとき，最も右側に来る根から葉へ至る道を最右道という．

(1) 最右道 $r_0, r_1, \ldots, r_k$ に深さ i の頂点 r_i がある．
(2) r_i は 2 つ以上の子を持つ．さらに
(3) $C(t(s)) = (a_1, a_2, \ldots, a_k, a_{k+1}, \ldots, a_m)$ とするとき，$C(t(r_{i+1})) = (a_1, a_2, \ldots, a_k)$ である．ここで s は r_{i+1} のすぐ左の兄弟である．

アクティブな頂点の中で一番深さが浅いものをコピー深さと呼ぶ．このコピー深さの子の状態により新しく追加する頂点の場所を変更する．以下コピー深さの頂点を $cd(t)$ で表し，最右道上の頂点をコピー深さより k だけ深い頂点を，$r_{cd(t)+k}$ とする．また，s を $cd_{(t)}$ の最も右から 2 番目の子とし，s とその子孫から成る部分 d-TC-パターン t を $t(s)$ とする．右部分 d-TC-パターン t を $t(r_{cd(t)+1})$ とする．

d-TC-パターンは縮約可能頂点と縮約不能頂点を頂点ラベルが異なる頂点であるとみなせば，浅井ら [1] のアルゴリズムを適用できる．ただし，縮約可能頂点は定数 d 以下の子頂点しか持てないので，アルゴリズムの適切な箇所で，その条件を満たすよう調整しなければならない．d-TC-パターンの正規形表現 t と t のコピー深さ k が与えられたとき，族木における全ての子を列挙する手続きを図 4 に，それを使って頂点数 n の d-TC-パターンを全て列挙するアルゴリズムを図 5 に示す．

図 6 に d-TC-パターンを列挙する際にたどる族木をあげる．

中野と宇野 [4] や浅井ら [1] のアルゴリズムと同様に，アルゴリズム TCP-ENUM でも，完全性と重複がないことが証明できる．

補題 5 アルゴリズム TCP-ENUM は完全に，かつ重複なく d-TC-パターンを列挙する．

従って，次の定理を得る．

定理 2 アルゴリズム TCP-ENUM は，n 頂点までの d-TC-パターンを完全に重複なく列挙する最悪時多項式時間遅延列挙アルゴリズムである．

証明 アルゴリズム TCP-ENUM の正当性は補題 5 から得られる．同アルゴリズムが最悪時多項式時間遅延 $O(nN)$ 時間で列挙することは，最右路の長さが高々 n であることから明らかである．ここで，n はアルゴリズム TCP-ENUM の入力で，列挙する d-TC-パターンの最大頂点数を表し，

手続き d-TCP-UpdateQueue(q);
入力: d-TC-パターンの正規形表現とそのコピー深さの組を
　格納したキュー q, 縮約可能頂点の最大子数 d;
begin
　$(C,k) := dequeue(q)$; C の最右路を $r_0, r_1, \ldots$ とする;
　$L_k := r_k$ の右から 2 つ目の子を根とする部分木;
　$R_k := r_k$ の最も右の子を根とする部分木;
　if $C(L_k) = C(R_k)$ **then**
　　for $i := 1$ to $k+1$ **do**
　　　if (r_{i-1} は子の数 $d-1$ 以下の縮約可能頂点)
　　　　$\vee$(r_{i-1} は縮約不能頂点) **then**
　　　　if r_i は縮約可能頂点 **then begin**
　　　　　$enqueue((C\&(i+0.5), i-1), q)$;
　　　　　$enqueue((C\&(i), i), q)$
　　　　end else $enqueue((C\&(i), i-1), q)$;
　else if $C(L_k) \neq C(R_k)$ **then begin**
　　$m := |C(R_k)| + 1$;
　　ℓ を $C(L_k)$ の m 番目の整数部分とし,
　　それに対応する頂点を w とする;
　　for $i := 1$ to $\ell - 1$ **do**
　　　if (r_{i-1} は子の数 $d-1$ 以下の縮約可能頂点)
　　　　$\vee$(r_{i-1} は縮約不能頂点) **then**
　　　　if r_i は縮約可能頂点 **then begin**
　　　　　$enqueue((C\&(i+0.5), i-1), q)$;
　　　　　$enqueue((C\&(i), i), q)$
　　　　end else $enqueue((C\&(i), i-1), q)$;
　　if w は縮約可能頂点 **then begin**
　　　$enqueue((C\&(\ell+0.5), k), q)$;
　　　$enqueue((C\&(i), \ell), q)$
　　end else $enqueue((C\&(\ell), k), q)$
end;

図 4　手続き d-TCP-UpdateQueue: 手続き中の q は d-TC-パターンの正規形表現とそのコピー深さの組を格納したキューである．手続き *dequeue* と *enqueue* はキューの先頭を取り出す操作とキューの最後に要素を付け足す操作を行う．

N は d-TC-パターンの出力数を表す．　□

4. 計算機実験と評価

4.1 d-TC-パターンの個数と列挙時間

第 3 節で提案した d-TC-パターン列挙アルゴリズムをプログラミング言語 C を用いて実装し，列挙した d-TC-パターンの個数と列挙時間を評価した．実験環境は Xeon E5620 (2.4GHz,12MB キャッシュ) × 2，メモリ 24GB で，実装は Red Hat Linux 上の GCC 4.4.6 を用いた．

図 7 の左グラフと図 8 の各成分上段に，提案列挙アルゴリズムの頂点数 n の d-TC-パターン 1 つあたりの計算時間を示す．第 3 章で証明したように，本論文で提案したアルゴリズムは最悪多項式時間遅延で動作する．実験結果が示すように，d-TC-パターン 1 つあたりの列挙時間は頂点数

アルゴリズム d-TCP-Enum(n);
入力: 列挙する d-TC-パターンの最大頂点数 n,
　縮約可能頂点の最大子数 d;
begin
　$q :=$ 空のキュー;
　$enqueue(((0.5, 1.5), 1), q)$;
　$enqueue(((0.5, 1), 1), q)$;
　$enqueue(((0, 1.5), 1), q)$;
　$enqueue(((0, 1), 1), q)$;
　$(C,k) := front(q)$;
　while $|C| < n$ **do**
　　d-TCP-UpdateQueue(q);
　return q
end.

図 5　アルゴリズム d-TCP-Enum: 手続き *enqueue* と $front$ はキューの最後に要素を付け足す操作とキューの先頭の要素を参照する操作を行う．

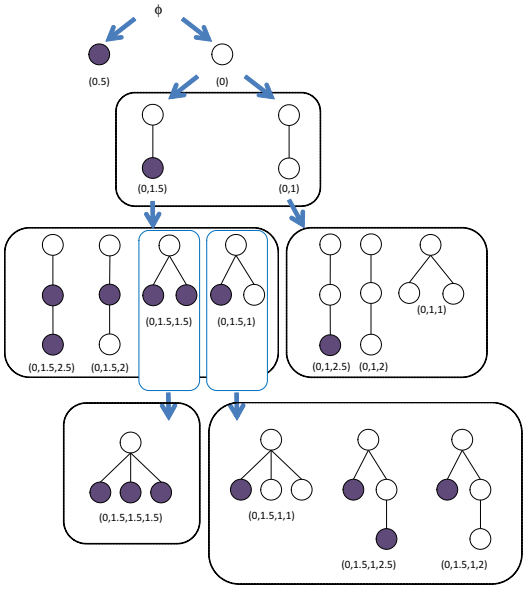

図 6　族木：頂点数 4 の 1-TC-パターンを列挙する過程の一部．

に対して増大せず，ほぼ一定の時間を保っており，定理の正しさを実験でも確かめることができた．ただし，同じ頂点数では最大子数が小さい方が，d-TC-パターン 1 つあたりの計算時間が大きい傾向がある．これは最大子数が小さいと多くの縮約可能頂点をスキップしないといけないため，オーバーヘッドが起きているものと推測される．オーバーヘッドを無くすためには新たなアイデアが必要である．

図 7 の右グラフと図 8 の各成分下段に，頂点数 n の d-TC-パターンの個数を表す．d-TC-パターンは頂点数に対して，指数関数的に増大する．一方，縮約可能頂点の最大子数に対する増加率は，頂点数に対する増加率に比べてかなり低い．今回の実験では頂点数 12 までの比較的小さな d-TC-パターンの列挙を行った．それでも d-TC-パターンの個数はかなり多い．実装したプログラムでは，列挙した d-TC-パターンを実メモリ上に保持しているため，頂点数を多く

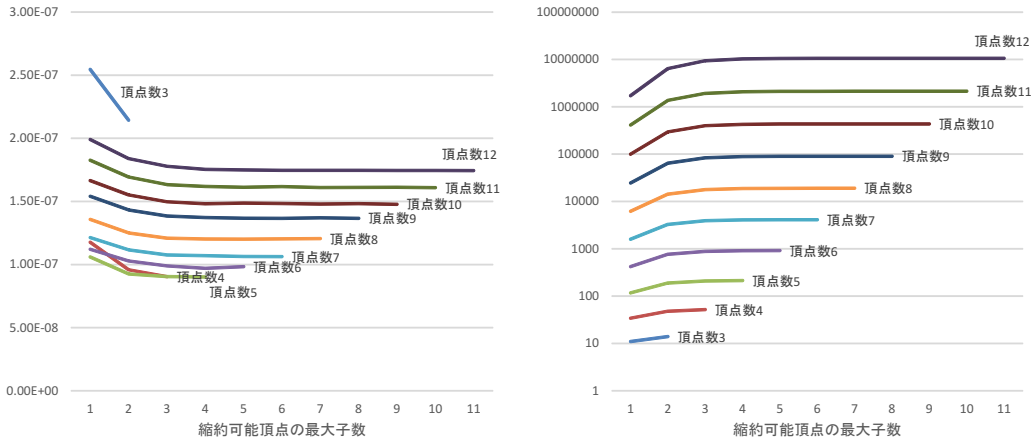

図 7 両グラフとも横軸は縮約可能頂点の最大子数 d を表す．左グラフの縦軸は d-TC-パターン 1 つあたりの計算時間を，右グラフの縦軸は d-TC-パターンの個数を表す．

することには限界があった．列挙した d-TC-パターンを外部メモリ等に保持することで，多少は頂点数を多くすることが可能であるが，それでもよりいっそう工夫が必要である．

4.2 糖鎖データに対する予備実験

本論文で提案した照合アルゴリズムと列挙アルゴリズムの評価を行うために，糖鎖データを対象にして実験を行った．糖鎖データは白血病 (leukemia) やガンの転移，糖尿病などに関係していると考えられている．糖鎖 (多糖類) は，単糖を頂点，単糖類間の結合を辺とすることでで，木構造データとみなせる．本論文では，頂点数 8 の d-TC-パターンをすべて列挙して，486 個の糖鎖データとの d-TC-パターンマッチングを行うことにより，leukemia に関与する木構造を予測する．486 のデータの内訳は，正例 (leukemia に関するデータ) は 192 個，負例 (leukemia には関していないデータ) は，294 個である．実験の精度の評価として，正解率，再現率，適合率，F 値の四つの評価値を用いる．また，d-TC-パターンの有効性を示すため，d-TC-パターンを用いて糖鎖データとの照合を行った場合と，根付き項木による照合を行った場合の精度を比較する．正解率 (accuracy) とは，予測したデータのうち正解であるものの割合，適合率 (precision) とは，正例と予測したもののうち実際に正例であるものの割合であり，再現率 (recall) とは，正例であるもののうち正例と予測したものの割合，F 値 (F-measure) とは，適合率と再現率の調和平均である．

無順序項木とは，構造的変数を導入した無順序木構造を持つパターンで，変数には任意の無順序木を代入できる．一般的には，変数ラベルが同じであれば，同じ無順序木を代入しなければならないという規則があるが，そのような場合，パターン照合問題が NP 完全となってしまうため，変数ラベルは全て異なると仮定する．詳細は [3], [7] を参照せよ．無順序項木 t が無順序木 T と照合するとは，無順序

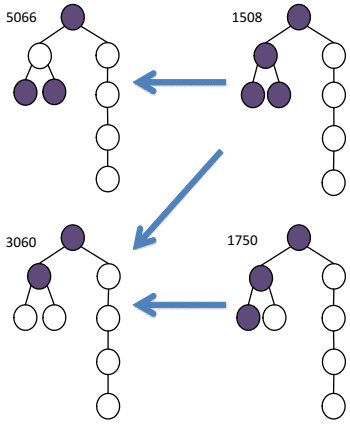

図 9 F 値が最大であった 4 つの 8 頂点の 2-TC-パターン (F 値:0.73255814)．各 2-TC-パターンの左上の数字は 8 頂点 2-TC-パターンの列挙順を示す．2-TC-パターンの間の矢印は 2-TC-パターンの特殊化・一般化の関係 [6] を表す．

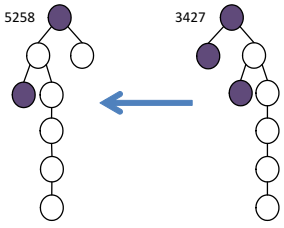

図 10 図 9 の 4 つの 2-TC-パターンを除いて F 値が最大の 2-TC-パターン (F 値:0.72886297)．

項木 t の持つ変数に適当な無順序木を代入して，T と同型な無順序木を得られるときをいう．無順序項木のパターン照合問題は変数のサイズが 2 の場合，$O(nN\sqrt{N})$ 時間のパターン照合アルゴリズムを持つ [3]．

山田は [9] で根付き項木を用いたカーネルを設計し，SVM を用いて前述した糖鎖データのクラス分類実験を行っている．その結果では，正解率，適合率，再現率，F 値の最大値が，それぞれ 0.8747, 0.9286, 0.7413, 0.8226 であった．山田の実験では，頂点ラベル付きの根付き項木を用いて実

(n,d)	1	2	3	4	5	6	7	8	9	10	11
3	255(ns) 11	214(ns) 14									
4	118(ns) 34	95.8(ns) 48	90.4(ns) 52								
5	106(ns) 117	92.6(ns) 189	90.4(ns) 209	90.2(ns) 214							
6	112(ns) 421	103(ns) 766	98.9(ns) 884	97.0(ns) 910	98.4(ns) 916						
7	121(ns) 1,590	112(ns) 3,275	108(ns) 3,923	107(ns) 4,081	106(ns) 4,113	106(ns) 4,120					
8	136(ns) 6,171	125(ns) 14,331	121(ns) 17,882	120(ns) 18,788	120(ns) 18,986	120(ns) 19,024	121(ns) 19,032				
9	154(ns) 24,571	143(ns) 64,351	138(ns) 83,543	137(ns) 88,715	137(ns) 89,867	137(ns) 90,105	137(ns) 90,149	137(ns) 90,158			
10	166(ns) 99,662	155(ns) 294,019	150(ns) 397,451	148(ns) 426,518	149(ns) 433,204	148(ns) 434,602	148(ns) 434,880	148(ns) 434,930	148(ns) 434,940		
11	183(ns) 410,683	169(ns) 1,364,225	163(ns) 1,919,821	162(ns) 2,082,513	161(ns) 2,120,671	162(ns) 2,128,851	161(ns) 2,130,495	161(ns) 2,130,813	161(ns) 2,130,869	161(ns) 2,130,880	
12	199(ns) 1,713,942	184(ns) 6,407,538	178(ns) 9,388,801	175(ns) 10,294,967	175(ns) 10,511,668	175(ns) 10,558,724	175(ns) 10,568,398	175(ns) 10,570,288	175(ns) 10,570,646	175(ns) 10,570,708	174(ns) 10,570,720

図 8 (n,d) 成分の上段は頂点数 n の d-TC-パターン 1 つあたりの計算時間 (ナノ秒 $=10^{-9}$ 秒) を表し，下段は頂点数 n の d-TC-パターンの数を表す.

験を行っている．一方，本実験では，8 頂点の頂点ラベル無しの d-TC-パターンを全て列挙し，全ての糖鎖データと照合を行い，それぞれの評価値を求めた．それらを F 値による評価を行い最大であった 4 つの構造を図 9 に示す.

図 9 の 4 つの d-TC-パターンに対する正解率，適合率，再現率，F 値は，全て同じで，それぞれ，0.8107, 0.828947, 0.656, 0.73255814 であった．また，山田の実験に対する評価として，グリーディーに頂点ラベルを付加した 4 つの評価値を求めたところ，それぞれ 0.835390947, 0.9375, 0.625, 0.75 という結果が得られた．適合率は山田の実験を上回っており，一つの d-TC-パターンによるクラス分類であることを考えれば，良い結果であると言える．図 10 の 2 つの d-TC-パターンは，図 9 の 4 つの d-TC-パターンを除いて F 値が最大の d-TC-パターン (F 値:0.72886297) である．これらの結果を表 11 にまとめる.

	正解率	適合率	再現率	F 値
無順序項木 SVM([9])	0.8747	0.9286	0.7413	0.8226
頂点ラベル無し (図 9)	0.8107	0.8290	0.6563	0.7326
頂点ラベル付き (図 9)	0.8354	0.9375	0.6250	0.7500
頂点ラベル無し (図 10)	0.8086	0.8278	0.6510	0.7289

図 11 実験結果まとめ

5. 結論

本論文では，d-TC-パターンを列挙する多項式時間遅延アルゴリズムを提案した．また，その実装をプログラミング言語 C で行い，d-TC-パターン 1 つに対する列挙時間が平均で定数時間であることを確認した．さらに，d-TC-パターンの表現能力を確かめるために，糖鎖データ上でのクラス分類実験を行った.

今後の課題は，列挙した解の保存に，容量がかなりかかるために，頂点数を増やせないために，大規模データに対して用いるにはメモリの確保について，改善する必要がある．また，縮約可能頂点の最大子数 d が小さいときのオーバーヘッドをなくす必要がある．解をより絞ることにより，メモリの問題を改善できることが予想されるので，最小一般化 d-木縮約パターンのみを列挙するアルゴリズムの開発が必要である.

d-TC-パターンを使った糖鎖データのクラス分類に関する実験では，サポートベクタマシン (SVM) を用いることで，評価値の上昇が期待できる．d-TC-パターンを使ったカーネルの開発とそれを用いたサポートベクタマシンでの評価が次の課題である.

謝辞 本研究は JSPS 科研費 23500182, MEXT 科研費 24106010 の助成を受けたものです.

参考文献

[1] 浅井 達哉, 有村 博紀, 宇野 毅明, 中野 眞一. 半構造データからの効率よい無順序木パターン発見手法. 電子情報通信学会技術研究報告. DE, データ工学 103(355), pp. 33–38, 2003.

[2] J. E. Hopcroft and R. M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing, 2*, pages 225–231, 1973.

[3] T. Miyahara, T. Shoudai, T. Uchida, K. Takahashi, and H. Ueda. Polynomial Time Matching Algorithms for Tree-Like Structured Patterns in Knowledge Discovery. *Proc. PAKDD-2000, Springer-Verlag, LNAI 1805*, pages 5–16, 2000.

[4] 中野 眞一, 宇野 毅明. 無順序木を列挙するシンプルなアルゴリズム. 情報処理学会研究報告. AL, アルゴリズム研究会報告 2003-AL-90(4), page 25–32, 2003.

[5] 岡本 吉央. 列挙の基本と基礎的なアルゴリズム. 電子情報通信学会誌 95(6), pp. 477–483, 2012.

[6] 岡本 康宏, 吉村 友太, 正代 隆義. 辺縮約に基づく木構造パターンの多項式時間学習アルゴリズム. 平成 24 年度 (第 65 回) 電気関係学会九州支部連合大会, 2012 年 9 月 24 日–25 日, 長崎大学, 2012.

[7] Y. Suzuki, T. Shoudai, S. Matsumoto, T. Uchida, and T. Miyahara. Efficient learning of ordered and unordered tree patterns with contractible variables. *Proc. ALT-2003, Springer, LNAI 2842*, pages 114–128, 2003.

[8] 宇野 毅明. 列挙アルゴリズムの高速化技法とその応用. 数理解析研究所講究録 1120, pp. 1–10, 1999.

[9] 山田 貴志. 頻出木構造パターンとサポートベクターマシンによる木構造データのクラス分類. 平成 20 年度九州大学理学部物理学科情報理学コース卒業論文, 2009.

[10] Y. Yoshimura and T. Shoudai. Learning Unordered Tree Contraction Patterns in Polynomial Time. The 22nd International Conference on Inductive Logic Programming (ILP2012), Dubrovnik, Croatia, September 17–19, 2012.