Vol.2018-MPS-117 No.5
2018/3/1

Enumeration of Maximally Frequent Ordered Tree Patterns with Height-Constrained Variables for Trees

Yusuke Suzuki[1,a)] Tetsuhiro Miyahara[1,b)] Takayoshi Shoudai[2,c)] Tomoyuki Uchida[1,d)]
Satoshi Matsumoto[3,e)] Tetsuji Kuboyama[4,f)]

Abstract: We consider representing tree structured features of structured data which are represented by rooted trees with ordered children. As representations of tree structured features, we propose height-constrained ordered wildcard tree patterns, which are ordered tree patterns having height-constrained structured variables and wildcards for edge labels. An (i, j)-height-constrained variable can be replaced with any rooted ordered tree whose trunk length is at least i and whose height is at most j. First we show that it is hard to compute an optimum frequent height-constrained ordered wildcard tree patterns. Then we present an algorithm for enumerating all maximally frequent height-constrained ordered wildcard tree patterns. Finally we consider extended height-constrained ordered wildcard tree patterns, called height-constrained ordered tag tree patterns, which have height-constrained variables, wildcards, tags and keywords, and an algorithm for enumerating all maximally frequent height-constrained ordered tag tree patterns.

Keywords: ordered tree pattern, height-constrained variable, enumeration, wildcard, maximal frequency

1. Introduction

In this paper we present new refined models for representing tree structured features, by extending our previous models of tree structured features, maximally frequent ordered wildcard tree patterns and maximally frequent ordered tag tree patterns [7]. We consider models of tree structured features in two aspects, i.e., representing power of tree structured patterns and the desired properties that the tree structured patterns must satisfy. We focus on the maximal frequency of the tree structured patterns as the desired property. Tree structured data are semistructured data whose structures are modeled by rooted trees with ordered children, based on Object Exchange Model [1]. Among tree structured data we consider are XML files, some biological data such as the secondary structure data of RNA or glycan data, and parse trees in natural language processing.

As a refined model of tree structured features concerning representing power of tree structured patterns, we propose *height-constrained wildcard tree patterns* (or simply called an *HC-wildcard tree pattern*), which are ordered tree patterns with height-constrained structured variables and wildcards, and match whole trees. A wildcard matches any edge label. A height-constrained structured variable [8] can be replaced with an arbitrary rooted ordered tree having height-constraint and a structured variable having no height-constraint [7] can be replaced with an arbitrary rooted ordered tree having no height-constraint. An (i, j)-*height-constrained variable* (or simply called an (i, j)-*HC-variable*) can be replaced with any tree such that the minimum length of the path (called the *trunk* of the tree), corresponding to the variable (Section 2), of the tree is i and the maximum height of the tree is j. Thus Ordered tree patterns with HC-variables are more accurate models of tree structured features than ordered tree patterns with variables having no height-constraint.

In this work, the maximal frequency of wildcard tree patterns is the desired property that the tree structured patterns must satisfy, as in our previous work [7]. An HC-wildcard tree pattern t is said to be *maximally frequent* w.r.t. a set D of given tree structured data, if t can explain more data in D than a user-specified threshold and any HC-wildcard tree pattern more specific than t cannot. A maximally frequent tree pattern is considered a least generalized tree pattern w.r.t. a set of tree structured data.

For example in Fig. 1, we consider finding one of the least generalized HC-wildcard tree patterns explaining at least two trees in the set $\{T_1, T_2, T_3\}$ of trees. The HC-wildcard tree pattern t_0 can explain all trees in $\{T_1, T_2, T_3\}$. But t_0 can explain all trees whose height is at most 5, so t_0 is an overgeneralized and meaningless pattern. On the other hand, the HC-wildcard tree pattern t_1 is one of the least generalized HC-wildcard tree patterns explaining two trees T_1 and T_2 but not T_3. The wildcard tree pattern t_3, which is an ordered tree pattern having no height-constraint considered in our previous [7], is one of the least generalized wildcard tree patterns explaining two trees T_1 and T_2 but not T_3.

1 Graduate School of Information Sciences, Hiroshima City University, Hiroshima 731-3194, Japan
2 Faculty of Contemporary Business, Kyushu International University, Kitakyushu 805-8512, Japan
3 Faculty of Science, Tokai University, Hiratsuka 259-1292, Japan
4 Computer Centre, Gakushuin University, Tokyo 171-8588, Japan
a) y-suzuki@info.hiroshima-cu.ac.jp
b) miyares18@info.hiroshima-cu.ac.jp
c) shoudai@cb.kiu.ac.jp
d) uchida@info.hiroshima-cu.ac.jp
e) matsumoto@tsc.u-tokai.ac.jp
f) ori-mps18@tk.cc.gakushuin.ac.jp

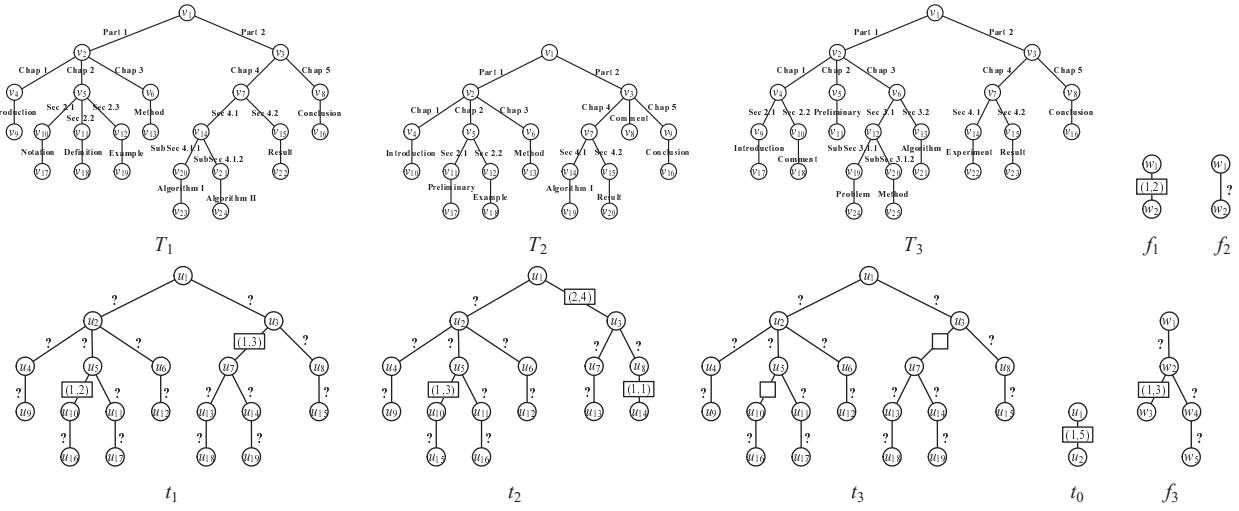


Fig. 1 A variable is represented by a box with lines to its elements. A box with a notation (i, j) shows an (i, j)-HC-variable A box without a notation (i, j) shows a variable having no height-constraint. The HC-wildcard tree pattern t_1 is maximally σ-frequent w.r.t. $\mathcal{D} = \{T_1, T_2, T_3\}$, where $\sigma = 0.5$.

In this paper, we consider two computational problems, **Maximally Frequent HC-Wildcard Tree Pattern of Maximum Tree-size** and **All Maximally Frequent HC-Wildcard Tree Patterns** over HC-wildcard tree patterns. Maximally Frequent HC-Wildcard Tree Pattern of Maximum Tree-size is the problem of finding the maximum HC-wildcard tree pattern t with respect to the number of vertices such that t can explain more data of input data than a user-specified threshold and t is minimally generalized. Firstly, we show that Maximally Frequent HC-Wildcard Tree Pattern of Maximum Tree-size is NP-complete. This indicates that it is hard to find an optimum HC-wildcard tree pattern representing given data. Next, we consider All Maximally Frequent HC-Wildcard Tree Patterns, which is the problem of generating all maximally frequent HC-wildcard tree patterns. This problem is based on the idea that meaningless tree patterns are excluded and all possible useful tree patterns are not missed. We present an algorithm for solving All Maximally Frequent HC-Wildcard Tree Patterns, i.e., an algorithm for enumerating maximally frequent HC-wildcard tree patterns, and show the correctness of the algorithm. Finally, as an application of the algorithm for solving All Maximally Frequent HC-Wildcard Tree Patterns, we present an algorithm for solving All Maximally Frequent HC-Tag Tree Patterns, which is the problem of enumerating all maximally frequent HC-tag tree patterns.

We discuss related work. In [12], Wang and Liu presented the algorithm for finding maximally frequent tree-expression patterns from tree structured data. In [2], Asai et al. presented an efficient algorithm for finding frequent substructures from a large collection of tree structured data. Recent research on tree structure patterns are reported [3], [4], [6], [11].

In [8], we gave an efficient pattern matching algorithm for HC-ordered term tree patterns, the extended algorithms of which we use in this paper for calculating the matching relation of HC-wildcard tree patterns and trees, and the matching relation of HC-tag tree patters and trees. In [8], also we considered finding a minimally generalized HC-ordered term tree pattern. The work [5] gave an algorithm for enumerating all maximal tree patterns,

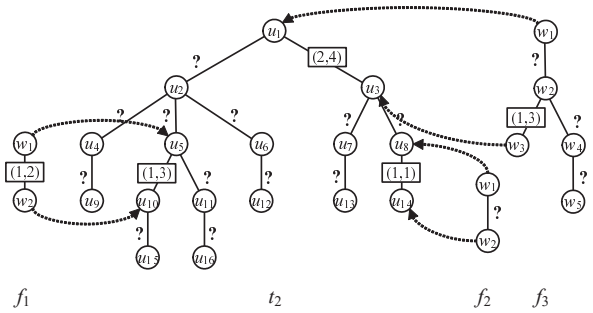


Fig. 2 Examples of $\mathcal{OWTP}^H$-bindings and $\mathcal{OWTP}^H$-substitution. Let t_2, f_1, f_2 and f_3 be HC-wildcard tree patterns described in Fig. 1. This figure displays the process of applying the $\mathcal{OWTP}^H$-bindings.

which are different tree patterns of frequency 1.0. This paper is a complete version of our previous results on HC-tag tree patterns [9].

2. Preliminaries

2.1 Height-Constrained Ordered Wildcard Tree Patterns

We explain height-constrained ordered wildcard tree patterns as tree structured patterns. Let Λ be a language which consists of infinitely or finitely many words. Let "?" be a special symbol, called a *wildcard*, such that "?" $\notin \Lambda$. Let $\Lambda_{\{?\}}$ be a proper subset of Λ. The symbol "?" is a wildcard for any word in $\Lambda_{\{?\}}$. For a set S, the number of elements in S is denoted by $|S|$. In this paper, a *tree* means a rooted ordered tree with ordered children such that each edge is labeled with an element in Λ. Let X be an infinite alphabet. We assume that $\Lambda \cap X = \emptyset$.

Definition 1 Let $X^{\mathcal{H}}$ be an infinite subset of X. For two positive integers i, j $(i \leq j)$, let $X^{\mathcal{H}(i,j)}$ be an infinite subset of $X^{\mathcal{H}}$. We assume that $X^{\mathcal{H}} = \bigcup_{1 \leq i \leq j} X^{\mathcal{H}(i,j)}$ and $X^{\mathcal{H}(i,j)} \cap X^{\mathcal{H}(i',j')} = \emptyset$ for $(i, j) \neq (i', j')$. An element of $X^{\mathcal{H}(i,j)}$ is called an (i, j)-*height-constrained variable label* or (i, j)-*HC variable label* for short.

Definition 2 Let $T = (V_T, E_T)$ be a tree which has a set V_T of vertices and a set E_T of edges with an edge labeling function $\mu_T : E_T \to \{\text{"?"}\} \cup X^{\mathcal{H}}$. Let $E_g = \{e \in E_T \mid \mu_T(e) = \text{"?"}\}$ and $H_g = \{h \in E_T \mid \mu_T(h) \in X^{\mathcal{H}}\}$ be a partition of E_T, i.e.,

$E_g \cup H_g = E_T$ and $E_g \cap H_g = \emptyset$. And let $V_g = V_T$. A *height-constrained ordered wildcard tree pattern* (or simply called an *HC-wildcard tree pattern*) is a triplet $g = (V_g, E_g, H_g)$. The root of g is the root of T. Each element in V_g, E_g and H_g is called a *vertex*, an *edge* and a *variable*, respectively. In particular, if h is a variable labeled with an element in $X^{\mathcal{H}(i,j)}$, h is called an (i,j)-*height-constrained variable* (an (i,j)-*HC variable*, for short). The notation $h^{(i,j)}$ means that h is an (i,j)-*HC-variable*. If the pair (i,j) of an (i,j)-HC-variable need not to be specified, an (i,j)-HC-variable is simply called an *HC-variable*.

For an HC-wildcard tree pattern g, $V(g)$, $E(g)$, and $H(g)$ denote the vertex set, the edge set, and the variable set of g, respectively. For a tree T, $V(T)$ and $E(T)$ denote the vertex set and the edge set of T. For an HC-wildcard tree pattern g and its vertices v_1 and v_n, a *path* from v_1 to v_n is a sequence $v_1, v_2, \ldots, v_n$ of distinct vertices of g such that for any k with $1 \leq k < n$, there exists an edge or a variable which consists of v_k and v_{k+1}. Let $H(v_1, v_n)$ be the set of all HC-variables in the path $v_1, v_2, \ldots, v_n$ of g and $E(v_1, v_n)$ the set of all edges in the path $v_1, v_2, \ldots, v_n$ of g. $len_{min}(v_1, v_n)$ is defined as the integer $\sum_{h^{(i,j)} \in H(v_1,v_n)} i + |E(v_1, v_n)|$. $len_{max}(v_1, v_n)$ is defined as the integer $\sum_{h^{(i,j)} \in H(v_1,v_n)} j + |E(v_1, v_n)|$. If there is an edge or a variable which consists of v and v' such that v lies on the path from the root to v', then v is said to be the *parent* of v' and v' is a *child* of v. We use a notation (v, v') (resp. $[v, v']$) to represent an edge (resp. a variable) such that v is the parent of v'. Then we call v the *parent port* of $[v, v']$ and v' the *child port* of $[v, v']$. The *height* of g is, denoted by $height(g)$, defined as $max\{len_{max}(r, \ell) \mid r$ is the root and ℓ a leaf $\}$. An HC-wildcard tree pattern g has a total ordering on all children of every internal vertex u. The ordering on the children of u is denoted by $<_u^g$. That is, for any two children u' and u'' of u, $u' <_u^g u''$ denotes that u' is a left sibling of u'' in g.

Definition 3 Let g be an HC-wildcard tree pattern. Let a sequence $u_0, u_1, \ldots, u_k$ be a path of g such that its length is more than one and for every $u_{\ell-1}$ and u_ℓ $(1 \leq \ell \leq k)$, $[u_{\ell-1}, u_\ell]^{(i_\ell, j_\ell)}$ is an (i_ℓ, j_ℓ)-HC variable of g for certain integers i_ℓ and j_ℓ. Then, $u_0, u_1, \ldots, u_k$ is said to be a *variable-chain* of g if u_ℓ is the only child of $u_{\ell-1}$ for any $0 \leq \ell \leq k$. If g has no variable-chain, g is called *variable-chain free*.

Definition 4 $\mathcal{OT}$ denotes the set of all trees whose edge labels are in Λ. A tree T is a *word tree* if $|V_T| = 2$ and $|E_T| = 1$. For a word $w \in \Lambda$, $T(w)$ denotes the word tree whose edge is labeled with the word w. For a subset $\Lambda' \subseteqq \Lambda$, we define the set of word trees $\mathcal{WT}_{\Lambda'} = \bigcup_{w \in \Lambda'} \{T(w)\}$. Note that for any set $\Lambda' \subseteqq \Lambda$, $\mathcal{WT}_{\Lambda'} \subsetneqq \mathcal{OT}$. $\mathcal{OWTP}^H$ denotes the set of all HC-wildcard tree patterns whose variable labels are in $X^{\mathcal{H}}$. $\mathcal{OWTP}^h$ denotes the set of all variable-chain free HC-wildcard tree patterns whose variable labels are in $X^{\mathcal{H}}$.

Let g be an HC-wildcard tree pattern or a tree with at least two vertices. Let $\tau = [w_0, w_1]$ be a list of two distinct vertices in g where w_0 is the root of g and w_1 is a leaf of g. The trunk length of $\tau = [w_0, w_1]$ is defined as $len_{min}(w_0, w_1)$. Let f be an HC-wildcard tree pattern with at least two vertices and e a variable or an edge of f. The form $e := \langle g, \tau \rangle$ is called a *binding* for e. A new HC-wildcard tree pattern or a new tree f' is obtained by applying the binding $e := \langle g, \tau \rangle$ for f in the following way. Let $e = [v_0, v_1]$ (resp. $e = (v_0, v_1)$) be a variable (resp. an edge) in f. Let g' be one copy of g and w'_0, w'_1 the vertices of g' corresponding to w_0, w_1 of g, respectively. For the variable or the edge e, we attach g' to f by removing e from $E(f) \cup H(f)$ and by identifying the vertices v_0, v_1 with the vertices w'_0, w'_1 of g', respectively. Further we define a new total ordering $<_u^{f'}$ on every vertex u of f' in a natural way. Suppose that u has more than one child and let u' and u'' be two children of u of f'. We have the following three cases. *Case 1*: If $u, u', u'' \in V(f)$ and $u' <_u^f u''$, then $u' <_u^{f'} u''$. *Case 2*: If $u, u', u'' \in V(g)$ and $u' <_u^g u''$, then $u' <_u^{f'} u''$. *Case 3*: If $u = v_0$, $u' \in V(g)$, $u'' \in V(f)$, and $v_1 <_u^f u''$ (resp. $u'' <_u^f v_1$), then $u' <_u^{f'} u''$ (resp. $u'' <_u^{f'} u'$). A *substitution* θ for f is a finite collection of bindings $\{e_1 := \langle g_1, \tau_1 \rangle, \ldots, e_n := \langle g_n, \tau_n \rangle\}$, where e_i's are mutually distinct variables or edges in f. The new HC-wildcard tree pattern or the new tree $f\theta$, called the *instance* of f by θ, is obtained by applying the all bindings $e_i := \langle g_i, \tau_i \rangle$ to f simultaneously. We note that the root of $f\theta$ is the root of f.

For an (i,j)-HC-variable e $(1 \leq i \leq j)$, a binding $e := \langle g, \tau \rangle$ is called an $\mathcal{OWTP}^H$-*binding* for e if the following conditions hold. (1) $g \in \mathcal{OWTP}^H$, (2) the trunk length of τ is at least i, (3) the height of g is at most j. For an HC-wildcard tree pattern f and a substitution $\theta = \{e_1 := \langle g_1, \tau_1 \rangle, \ldots, e_n := \langle g_n, \tau_n \rangle\}$ for f, θ is called an $\mathcal{OWTP}^H$-substitution for f if all bindings in θ are $\mathcal{OWTP}^H$-bindings. For an $\mathcal{OWTP}^H$-substitution θ, the new HC-wildcard tree pattern $f\theta$ is called the $\mathcal{OWTP}^H$-*instance* of f by θ. For an HC-variable e or an edge e, a binding $e := \langle g, \tau \rangle$ is called an $\mathcal{OT}$-*binding* for e if the following two conditions hold. (1) if e is an edge, then $g \in \mathcal{WT}_{\Lambda_{(?)}}$, (2) if e is an (i,j)-HC-variable $(1 \leq i \leq j)$ then (i) $g \in \mathcal{OT}$, (ii) the trunk length of τ is at least i and (iii) the height of g is at most j. For an HC-wildcard tree pattern f and a substitution $\theta = \{e_1 := \langle g_1, \tau_1 \rangle, \ldots, e_n := \langle g_n, \tau_n \rangle\}$ for f, θ is called an $\mathcal{OT}$-substitution for f if the following two conditions hold. (1) $\{e_1, \ldots, e_n\} = E(f) \cup H(f)$, (2) all bindings in θ are $\mathcal{OT}$-bindings. For an $\mathcal{OT}$-substitution θ, the new tree $f\theta$ is called the $\mathcal{OT}$-*instance* of f by θ.

Let f and g (resp. f and g) be two HC-wildcard tree patterns (resp. two trees). We say that f and g are *isomorphic*, denoted by $f \cong g$, if there is a bijection φ from $V(f)$ to $V(g)$ such that (1) the root of f is mapped to the root of g by φ, (2) $(u, v) \in E(f)$ if and only if $(\varphi(u), \varphi(v)) \in E(g)$ and the two edges have the same edge label, (3)For any i, j $(1 \leq i \leq j)$, $[u, v]^{(i,j)} \in H(f)$ if and only if $[\varphi(u), \varphi(v)]^{(i,j)} \in H(g)$, and (4) for any internal vertex u in f which has more than one child, and for any two children u' and u'' of u, $u' <_u^f u''$ if and only if $\varphi(u') <_{\varphi(u)}^g \varphi(u'')$.

An HC-wildcard tree pattern t *matches* a tree T if there exists an $\mathcal{OT}$-substitution θ such that $t\theta \cong T$.

Definition 5 The *language* $L_\Lambda(t)$ of an HC-wildcard tree pattern t is $\{s \in \mathcal{OT} \mid s \cong t\theta$ for an $\mathcal{OT}$-substitution $\theta\}$.

Example 1 In Fig. 2, we describe the process of applying the above $\mathcal{OWTP}^H$-bindings in the $\mathcal{OWTP}$-substitution θ' for t_2. The HC-wildcard tree pattern $t_2\theta'$ is isomorphic to t_1 in Fig. 1.

Let $\mathcal{D} = \{T_1, T_2, \ldots, T_m\} \subseteqq \mathcal{OT}$ be a nonempty finite set of trees. The *matching count* of an HC-wildcard tree pattern $\pi \in \mathcal{OWTP}^H$ w.r.t. $\mathcal{D}$, denoted by $match_{\mathcal{D}}(\pi)$, is the number of trees $T_i \in \mathcal{D}$ $(1 \leq i \leq m)$ such that π matches T_i. Then the *frequency* of π w.r.t. $\mathcal{D}$ is defined by $supp_{\mathcal{D}}(\pi) = match_{\mathcal{D}}(\pi)/m$. Let σ be a real number where $0 < \sigma \leq 1$. An HC-wildcard tree

pattern $\pi \in \mathcal{OWTP}^H$ is σ-*frequent* w.r.t. $\mathcal{D}$ if $supp_{\mathcal{D}}(\pi) \geq \sigma$. An HC-wildcard tree pattern $\pi \in \mathcal{OWTP}^h$ is *maximally* σ-*frequent* w.r.t. $\mathcal{D}$ in $\mathcal{OWTP}^h$ if (1) π is σ-frequent w.r.t. $\mathcal{D}$, and (2) if $L_\Lambda(\pi') \subsetneqq L_\Lambda(\pi)$ then π' is not σ-frequent w.r.t. $\mathcal{D}$ for any HC-wildcard tree pattern π' in $\mathcal{OWTP}^h$.

2.2 Hardness Results of Finding an Optimum Frequent Height-Constrained Wildcard Tree Pattern

In this section, we give a hardness result of computing an optimum HC-wildcard tree pattern. We show that it is hard to compute a maximally frequent HC-wildcard tree pattern of maximum tree-size w.r.t. a nonempty finite set of trees. The formal definition of the problem is as follows.

Maximally Frequent HC-Wildcard Tree Pattern of Maximum Tree-size

Instance: A nonempty finite set of trees $\mathcal{D} = \{T_1, T_2, \ldots, T_m\}$, a real number σ $(0 < \sigma \leq 1)$ and a positive integer K.

Question: Is there a maximally σ-frequent HC-wildcard tree pattern π w.r.t. $\mathcal{D}$ in $\mathcal{OWTP}^h$ with $|V(\pi)| \geq K$?

Theorem 1 **Maximally Frequent HC-Wildcard Tree Pattern of Maximum Tree-size** is NP-complete.

3. Enumeration of Maximally Frequent Wildcard Tree Patterns with HC-Variables

In this section, we consider the following problem.

All Maximally Frequent HC-Wildcard Tree Patterns (MFOWTPH)

Input: A nonempty finite set $\mathcal{D} \subseteqq \mathcal{OT}$ of trees, a real number σ $(0 < \sigma \leq 1)$.

Assumption: (1) $\Lambda_{\{?\}} \subsetneqq \Lambda$, and (2) there exists an algorithm for deciding whether or not any word in Λ is in $\Lambda_{\{?\}}$.

Problem: Enumerate all maximally σ-frequent HC-wildcard tree patterns w.r.t. $\mathcal{D}$ in $\mathcal{OWTP}^h$.

We give an algorithm Gen-MFOWTPH (Algorithm 1) which generates all maximally σ-frequent HC-wildcard tree patterns in $\mathcal{OWTP}^h$. Let $\mathcal{D} \subseteqq \mathcal{OT}$ be a nonempty finite set of trees. In the algorithm Gen-MFOWTPH, we decide whether or not a candidate HC-wildcard tree pattern is σ-frequent w.r.t. $\mathcal{D}$, by using a matching algorithm which decides whether or not an HC-wildcard tree pattern matches a tree. This matching algorithm is an extended version of the efficient pattern matching algorithm [8] for an ordered term tree pattern with HC-variables and a tree. Let D_h be the maximum height of trees in $\mathcal{D}$. A $(1, D_h)$-*HC-variable-only tree pattern* is an HC-wildcard tree pattern consisting of only vertices and $(1, D_h)$-HC-variables.

In the procedure EnumFreqTP, we use an algorithm using a *rightmost expansion technique* in order to enumerate all $(1, D_h)$-HC-variable-only tree patterns in a similar way to our previous work [7]. The rightmost expansion technique and an algorithm using this technique for enumerating all trees are developed by Asai et al. [2]. For two $(1, D_h)$-HC-variable-only tree patterns π and π', if π' is obtained from π by applying the rightmost expansion technique, then π' is called a *child tree pattern* of π and π is called the *parent tree pattern* of π'. By using the same parent-child relation as in our previous work [7], we enumerate without any duplicate all $(1, D_h)$-HC-variable-only tree patterns in a way of depth first search. We can prove the following theorem. Due to the space limit, we omit the proof.

Theorem 2 Algorithm Gen-MFOWTPH outputs the set of all maximally σ-frequent HC-wildcard tree patterns w.r.t. $\mathcal{D}$ in $\mathcal{OWTP}^h$.

Algorithm 1 Gen-MFOWTPH

Input: A nonempty finite set $\mathcal{D} \subseteqq \mathcal{OT}$ of trees and a real number σ $(0 < \sigma \leq 1)$;
Output: The set $\Pi(\sigma)$ of all maximally σ-frequent HC-wildcard tree patterns w.r.t. $\mathcal{D}$ in $\mathcal{OWTP}^h$;
1: $\Pi_1(\sigma)$:=EnumFreqTP$(\mathcal{D}, \sigma)$ (Procedure 2)
2: $\Pi_2(\sigma)$:=ReplaceEdge$(\mathcal{D}, \sigma, \Pi_1(\sigma))$ (Procedure 4)
3: $\Pi_3(\sigma)$:=MergeVariable$(\mathcal{D}, \sigma, \Pi_2(\sigma))$ (Procedure 6)
4: $\Pi_4(\sigma)$:=ConstrainVariable$(\mathcal{D}, \sigma, \Pi_3(\sigma))$ (Procedure 7)
5: $\Pi(\sigma)$:=TestMaximality$(\mathcal{D}, \sigma, \Pi_4(\sigma))$ (Procedure 9)
6: **return** $\Pi(\sigma)$

Procedure 2 EnumFreqTP

Input: A nonempty finite set $\mathcal{D} \subseteqq \mathcal{OT}$ of trees and a real number σ $(0 < \sigma \leq 1)$;
Output: A set Π_{out} of HC-variable-only tree patterns;
1: Let D_h be the maximum height of trees in $\mathcal{D}$
2: $\pi := (\{u, v\}, \emptyset, \{[u, v]^{(1, D_h)}\})$
3: Π_{out} :=EnumFreqTPSub$(\mathcal{D}, \sigma, \pi)$ (Procedure 3)
4: **return** Π_{out}

Procedure 3 EnumFreqTPSub

Input: A nonempty finite set $\mathcal{D} \subseteqq \mathcal{OT}$ of trees, a real number σ $(0 < \sigma \leq 1)$, and an HC-variable-only tree pattern π;
Output: A set Π_{out} of HC-variable-only tree patterns;
1: **if** π is not σ-frequent w.r.t. $\mathcal{D}$ **then return** $\emptyset$
2: $\Pi_{out} := \{\pi\}$
3: **for** each child tree pattern π' of π **do**
4: $\quad \Pi_{out} := \Pi_{out} \cup$EnumFreqTPSub$(\mathcal{D}, \sigma, \pi')$
5: **end for**
6: **return** Π_{out}

Procedure 4 ReplaceEdge

Input: A nonempty finite set $\mathcal{D} \subseteqq \mathcal{OT}$ of trees, a real number σ $(0 < \sigma \leq 1)$, and a set Π_{in} of HC-variable-only tree patterns;
Output: A set Π_{out} of HC-wildcard tree patterns;
1: $\Pi_{out} := \Pi_{in}$
2: **for** each HC-wildcard tree pattern $\pi \in \Pi_{in}$ **do**
3: $\quad \Pi_{out} := \Pi_{out} \cup$ ReplaceEdgeSub$(\mathcal{D}, \sigma, \pi, 1)$ (Procedure 5)
4: **end for**
5: **return** Π_{out}

4. Application to Enumeration of Maximally Frequent Tree Patterns with Tags and Keywords

Definition 6 Let Λ_{Tag} be a language consisting of infinitely or finitely many words in Λ. Let Λ_{KW} be a language consisting of infinitely or finitely many words of the form "$/k/$" for words k in Λ, where we assume that "/" $\notin \Lambda$ holds. We call a word in Λ_{Tag} a *tag* and a word in Λ_{KW} a *keyword*. For a keyword $/k/ \in \Lambda_{KW}$, we define the set $\Lambda_{\{/k/\}} = \{w \in \Lambda \mid k \text{ is a substring of } w\}$. Let $T = (V_T, E_T)$ be a tree which has a set V_T of vertices and a set E_T

Procedure 5 ReplaceEdgeSub

Input: A nonempty finite set $\mathcal{D} \subseteq \mathcal{OT}$ of trees, a real number σ $(0 < \sigma \leq 1)$, an HC-wildcard tree pattern π and a positive integer p;
Output: A set Π_{out} of HC-wildcard tree patterns;
1: **if** $p > |E(\pi) \cup H(\pi)|$ **then return** $\emptyset$
2: $\Pi_{out} := \emptyset$
3: Let T_0("?") be the HC-wildcard tree pattern in Fig.3.
4: Let $h^{(1,D_h)}$ be the p-th variable in the DFS order of all edges and variables of π.
5: $\pi_? := \pi\{h^{(1,D_h)} := \langle T_0(\text{"?"}), [R_0, L_0]\rangle\}$
6: **if** $\pi_?$ is σ-frequent w.r.t. $\mathcal{D}$ **then** $\Pi_{out} := \{\pi_?\}$
7: $\Pi_{tmp} := \Pi_{out} \cup \{\pi\}$
8: **for** each HC-wildcard tree pattern $\pi' \in \Pi_{tmp}$ **do**
9: $\Pi_{out} := \Pi_{out} \cup$ ReplaceEdgeSub$(\mathcal{D}, \sigma, \pi', p+1)$
10: **end for**
11: **return** Π_{out}

Procedure 6 MergeVariable

Input: A nonempty finite set $\mathcal{D} \subseteq \mathcal{OT}$ of trees, a real number σ $(0 < \sigma \leq 1)$, and a set Π_{in} of HC-wildcard tree patterns;
Output: A set Π_{out} of HC-wildcard tree patterns;
1: $\Pi_{out} := \emptyset$
2: **for** each HC-wildcard tree pattern $\pi \in \Pi_{in}$ **do**
3: **while** π is not variable-chain free **do**
4: Let u_1, u_2, u_3 be a variable-chain in π such that $[u_1, u_2]^{(i_1, D_h)}$ is an (i_1, D_h)-HC variable and $[u_2, u_3]^{(i_2, D_h)}$ is an (i_2, D_h)-HC variable
5: $\pi' := (V(\pi) \setminus \{u_2\}, E(\pi), H(\pi) \cup \{[u_1, u_3]^{(i_1+i_2, D_h)}\} \setminus \{[u_1, u_2]^{(i_1, D_h)}, [u_2, u_3]^{(i_2, D_h)}\})$
6: **end while**
7: $\Pi_{out} := \Pi_{out} \cup \{\pi'\}$
8: **end for**
9: **return** Π_{out}

Procedure 7 ConstrainVariable

Input: A nonempty finite set $\mathcal{D} \subseteq \mathcal{OT}$ of trees, a real number σ $(0 < \sigma \leq 1)$, and a set Π_{in} of HC-wildcard tree patterns;
Output: A set Π_{out} of HC-wildcard tree patterns;
1: $\Pi_{out} := \Pi_{in}$
2: **for** each HC-wildcard tree pattern $\pi \in \Pi_{in}$ **do**
3: $\Pi_{out} := \Pi_{out} \cup$ ConstrainVariableSub$(\mathcal{D}, \sigma, \pi, 1)$ (Procedure 8)
4: **end for**
5: **return** Π_{out}

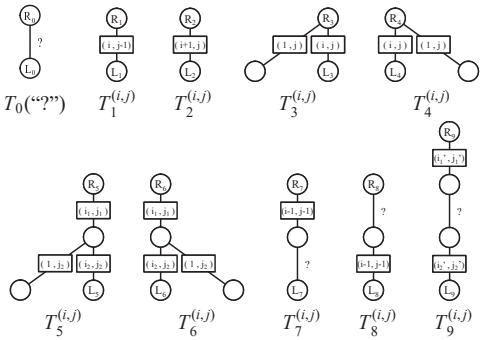


Fig. 3 HC-Wildcard tree patterns T_0("?") and $T_1^{(i,j)}, \ldots, T_9^{(i,j)}$. For HC-wildcard tree patterns $T_5^{(i,j)}, T_6^{(i,j)}$, we assume $i_1 + i_2 = i$ and $j_1 + j_2 = j$. For HC-wildcard tree pattern $T_9^{(i,j)}$, we assume $i'_1 + i'_2 + 1 = i$ and $j'_1 + j'_2 + 1 = j$.

of edges. Let E_g and H_g be a partition of E_T, i.e., $E_g \cup H_g = E_T$ and $E_g \cap H_g = \emptyset$. And let $V_g = V_T$. A *height-constrained ordered tag tree pattern* (or simply called an *HC-tag tree pattern*) is a triplet $g = (V_g, E_g, H_g)$ such that each element in E_g is labeled with any of a tag, a keyword and the symbol "?". Each element in V_g, E_g and H_g is called a *vertex*, an *edge* and an *HC-variable*, respectively.

Procedure 8 ConstrainVariableSub

Input: A nonempty finite set $\mathcal{D} \subseteq \mathcal{OT}$ of trees, a real number σ $(0 < \sigma \leq 1)$, an HC-wildcard tree pattern π and a positive integer p;
Output: A set Π_{out} of HC-wildcard tree patterns;
1: **if** $p > |H(\pi)|$ **then return** $\emptyset$
2: $\Pi_{out} := \emptyset$
3: Let $h^{(i,j)} = [u, v]^{(i,j)}$ be the p-th variable in the DFS order of HC-variables of π.
4: **for** $k := j - 1$ **downto** i **do**
5: Let π' be an HC-wildcard tree pattern obtained from π by replacing $[u, v]^{(i,j)}$ of π with a variable $[u, v]^{(i,k)}$
6: **if** π' is σ-frequent w.r.t. $\mathcal{D}$ **then** $\Pi_{out} := \Pi_{out} \cup \{\pi'\}$
7: **end for**
8: $\Pi_{tmp} := \Pi_{out} \cup \{\pi\}$
9: **for** each HC-wildcard tree pattern $\pi' \in \Pi_{tmp}$ **do**
10: $\Pi_{out} := \Pi_{out} \cup$ ConstrainVariableSub$(\mathcal{D}, \sigma, \pi', p+1)$
11: **end for**
12: **return** Π_{out}

Procedure 9 TestMaximality

Input: A nonempty finite set $\mathcal{D} \subseteq \mathcal{OT}$ of trees, a real number σ $(0 < \sigma \leq 1)$, and a set Π_{in} of HC-wildcard tree patterns;
Output: A set Π_{out} of HC-wildcard tree patterns;
1: $\Pi_{out} := \Pi_{in}$
2: **for** each HC-wildcard tree pattern $\pi \in \Pi_{out}$ **do**
3: **for** each HC-variable $h^{(1,j)}$ in π **do**
4: Let T_0("?") be the HC-wildcard tree pattern in Fig.3.
5: **if** $\pi\{h^{(1,j)} := \langle T_0(\text{"?"}), [R_0, L_0]\rangle\}$ is σ-frequent w.r.t. $\mathcal{D}$ **then**
6: $\Pi_{out} := \Pi_{out} \setminus \{\pi\}$
7: **end if**
8: **end for**
9: **for** each HC-variable $h^{(i,j)}$ in π **do**
10: Let $T_1^{(i,j)}, \ldots, T_9^{(i,j)}$ be the HC-wildcard tree patterns in Fig.3.
11: **if** there exists a $K \in \{1, \ldots, 9\}$ such that $\pi\{h^{(i,j)} := \langle T_K^{(i,j)}, [R_K, L_K]\rangle\}$ is σ-frequent w.r.t. $\mathcal{D}$ **then**
12: $\Pi_{out} := \Pi_{out} \setminus \{\pi\}$
13: **end if**
14: **end for**
15: **end for**
16: **return** Π_{out}

Two HC-tag tree patterns f and g are *isomorphic* if f and g are isomorphic as HC-wildcard tree patterns by regarding the symbol "?", tags and keywords as edge labels. A substitution for an HC-tag tree pattern is an extended form of a substitution for an HC-wildcard tree pattern, where a binding $e := \langle g, \tau\rangle$ for an edge e labeled with a keyword $/k/$ can replace the edge e with any word tree $g \in \mathcal{WT}_{\Lambda_{\{/k/\}}}$, a binding $e := \langle g, \tau\rangle$ for an edge e labeled with the symbol "?" can replace the edge e with any word tree $g \in \mathcal{WT}_{\Lambda_{\{?\}}}$, and a binding $e := \langle g, \tau\rangle$ for an (i, j)-HC-variable e can replace the (i, j)-HC-variable e with any tree g whose trunk length is at least i and whose height is at most j. An HC-tag tree pattern t is said to *match* a tree T if there exists a substitution θ such that $T \cong t\theta$ holds. An edge e of an HC-tag tree pattern is said to *match* an edge e' of a tree if there exists

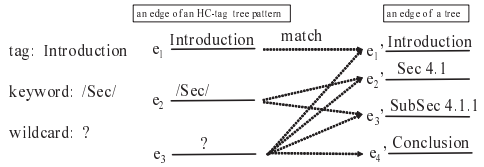
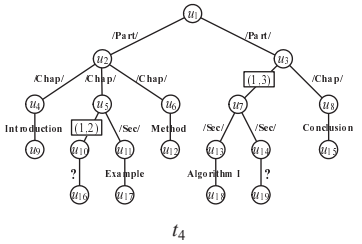


Fig. 4 The matching relation of an edge of an HC-tag tree pattern and an edge of a tree.



t_4

Fig. 5 A maximally σ-frequent HC-tag tree pattern t_4 w.r.t. $\mathcal{D} = \{T_1, T_2, T_3\}$ given in Fig.1, where Tag = {*Introduction*, *Method*, *Example*, *Algorithm I*, *Conclusion*}, KW = {/*Part*/, /*Chap*/, /*Sec*/, /*SubSec*/} and $\sigma = 0.5$.

a substitution θ such that the edge label of e after the replacement by θ equals the edge label of e'. $\mathcal{OTTP}_{(\Lambda_{Tag},\Lambda_{KW})}$ denotes the set of all HC-tag tree patterns with tags in Λ_{Tag} and keywords in Λ_{KW}. For t in $\mathcal{OTTP}_{(\Lambda_{Tag},\Lambda_{KW})}$, the *language* $L_\Lambda(t)$ is defined as {a tree T in $\mathcal{OT}$ | t matches T}.

Example 2 We explain the matching relation of an edge of an HC-tag tree pattern and an edge of a tree in Fig. 4. Let "Introduction" be a tag, and "/Sec/" a keyword. We assume that {*Introduction*, Sec4.1, $SubSec$4.1.1, *Conclusion*} $\subseteqq \Lambda_{\{?\}}$. In an HC-tag tree pattern, let us consider an edge e_1 with a label "Introduction", an edge e_2 with a label "/Sec/" and an edge e_3 with a label "?". In a tree, let us consider an edge e'_1 with a label "Introduction", an edge e'_2 with a label "Sec 4.1", an edge e'_3 with a label "SubSec 4.1.1" and an edge e'_4 with a label "Conclusion". Then we have the following. e_1 matches e'_1. e_2 matches e'_2 and e'_3. e_3 matches e'_1, e'_2, e'_3 and e'_4.

Let $\mathcal{D} = \{T_1, T_2, \ldots, T_m\}$ be a nonempty finite set of trees. The *matching count* of an HC-tag tree pattern π w.r.t. $\mathcal{D}$, denoted by $match_{\mathcal{D}}(\pi)$, is the number of trees $T_i \in \mathcal{D}$ $(1 \leq i \leq m)$ such that π matches T_i. Then the *frequency* of π w.r.t. $\mathcal{D}$ is defined by $supp_{\mathcal{D}}(\pi) = match_{\mathcal{D}}(\pi)/m$. Let σ be a real number where $0 < \sigma \leq 1$. An HC-tag tree pattern π is *σ-frequent* w.r.t. $\mathcal{D}$ if $supp_{\mathcal{D}}(\pi) \geq \sigma$. Let Tag be a finite subset of Λ_{Tag} and KW a finite subset of Λ_{KW}. Let $\Lambda(Tag, KW) = Tag \cup \bigcup_{/k/\in KW} \Lambda_{\{/k/\}}$. We denote by $\mathcal{OTTP}(Tag, KW)$ the set of all HC-tag tree patterns π with the tags of π in Tag and the keywords of π in KW. An HC-tag tree pattern π in $\mathcal{OTTP}(Tag, KW)$ is *maximally σ-frequent* w.r.t. $\mathcal{D}$ if (1) π is σ-frequent, and (2) if $L_\Lambda(\pi') \subsetneqq L_\Lambda(\pi)$ then π' is not σ-frequent for any HC-tag tree pattern π' in $\mathcal{OTTP}(Tag, KW)$.

Example 3 Let t_4 be an HC-tag tree pattern in $\mathcal{OTTP}(Tag, KW)$, which is described in Fig. 5. The HC-tag tree pattern t_4 is a maximally σ-frequent w.r.t. $\mathcal{D}$, where $\sigma = 0.5$, $\mathcal{D} = \{T_1, T_2, T_3\}$ given in Fig. 1.

All Maximally Frequent HC-Tag Tree Patterns (**MFOTTPH**)
Input: A nonempty finite set $\mathcal{D} \subseteqq \mathcal{OT}$ of trees, a real number σ $(0 < \sigma \leq 1)$, a finite set Tag of tags, and a finite set KW of keywords.
Assumption: (1) $\Lambda(Tag, KW) \subsetneqq \Lambda_{\{?\}} \subsetneqq \Lambda$, (2) $Tag \cap \bigcup_{/k/\in KW} \Lambda_{\{/k/\}} = \emptyset$, and (3) there exists an algorithm for deciding whether or not any word in Λ is in $\Lambda_{\{?\}}$.
Problem: Generate all maximally σ-frequent HC-tag tree patterns w.r.t. $\mathcal{D}$ in $\mathcal{OTTP}(Tag, KW)$.

We give an algorithm which generates all maximally σ-frequent HC-tag tree patterns by extending the algorithm GenMFOWTPH in Section 3.

5. Conclusions

We have proposed height-constrained wildcard tree patterns, which are ordered tree patterns having height-constrained structured variables and wildcards for edge labels, and match whole data trees. A height-constrained structured variable can be replaced with an arbitrary rooted ordered tree having height-constraint. First we have shown that it is hard to compute a maximally frequent HC-wildcard tree pattern of maximum-tree size. Then we have presented an algorithm for enumerating all maximally frequent HC-wildcard tree patterns. Finally, as an application we have presented an algorithm for enumerating all maximally frequent HC-tag tree patterns. This work was partially supported by Grant-in-Aid for Scientific Research (C) (Grant Numbers 15K00312, 15K00313, 17K00321) from Japan Society for the Promotion of Science (JSPS).

References

[1] Abiteboul, S., Buneman, P. and Suciu, D.: *Data on the Web: From Relations to Semistructured Data and XML*, Morgan Kaufmann (2000).
[2] Asai, T., Abe, K., Kawasoe, S., Sakamoto, H., Arimura, H. and Arikawa, S.: Efficient substructure discovery from large semi-structured data, *IEICE Trans. Inf. Syst.*, Vol. E87-D(12), pp. 2754–2763 (2004).
[3] Chehreghani, M. H. and Bruynooghe, M.: Mining rooted ordered trees under subtree homeomorphism, *Data Mining and Knowledge Discovery*, Vol. 30, No. 5, pp. 1249–1272 (2016).
[4] Doshi, M. and Roy, B.: Enhanced data processing using positive negative association mining on AJAX data, *Proc. of 2014 International Conference on Circuits, Systems, Communication and Information Technology Applications (CSCITA-2014),*, pp. 386–390 (2014).
[5] Itokawa, Y. and Uchida, T. Sano, M.: An Algorithm for Enumerating All Maximal Tree Patterns Without Duplication Using Succinct Data Structure, *Proc. IMECS 2014*, pp. 156–161 (2014).
[6] Jiang, C., Coenen, F. and Zito, M.: A survey of frequent subgraph mining algorithms, *The Knowledge Engineering Review*, Vol. 28, No. 01, pp. 75–105 (2013).
[7] Miyahara, T., Suzuki, Y., Shoudai, T., Uchida, T. and Kuboyama, T.: Enumeration of Maximally Frequent Ordered Tree Patterns with Wildcards for Edge Labels, *IPSJ Trans. Math. Model. Appl.(TOM)*, Vol. 10(2), pp. 59–69 (2017).
[8] Shoudai, T., Aikoh, K., Suzuki, Y., Matsumoto, S., Miyahara, T. and Uchida, T.: Polynomial Time Inductive Inference of Languages of Ordered Term Tree Patterns with Height-Constrained Variables from Positive Data, *IEICE Trans. Fund.*, Vol. E100-A(3), pp. 785–802 (2017).
[9] Suzuki, Y., Miyahara, T., Shoudai, T., Uchida, T. and Nakamura, Y.: Discovery of Maximally Frequent Tag Tree Patterns with Height-Constrained Variables from Semistructured Web Documents, *Proc. of International Workshop on Challenges in Web Information Retrieval and Integration (WIRI-2005)*, pp. 107–115 (2005).
[10] Suzuki, Y., Shoudai, T., Uchida, T. and Miyahara, T.: An Efficient Pattern Matching Algorithm for Ordered Term Tree Patterns, *IEICE Trans. Inf. Syst.*, Vol. E98-A(6), pp. 1197–1211 (2015).
[11] Wang, J., Liu, Z., Li, W. and Li, X.: Research on a frequent maximal induced subtrees mining method based on the compression tree sequence, *Expert Systems with Applications*, Vol. 42, No. 1, pp. 94–100 (2015).
[12] Wang, K. and Liu, H.: Discovering structural association of semistructured data, *IEEE Trans. Knowledge and Data Engineering*, Vol. 12(3), pp. 353–371 (2000).