Learning of Primitive Formal Systems Defining Labeled Ordered Tree Languages via Queries

Tomoyuki Uchida[1], Satoshi Matsumoto[2], Takayoshi Shoudai[3], Yusuke Suzuki[1], and Tetsuhiro Miyahara[1]

[1] Graduate School of Information Sciences, Hiroshima City University, Japan
uchida@hiroshima-cu.ac.jp
[2] Faculty of Science, Tokai University, Japan
[3] Faculty of Contemporary Business, Kyushu International University, Japan

Abstract. Formal Graph System (FGS) is a logic programming system that deals with term graph patterns instead of terms of first-order predicate logic. In this paper, based on FGS, we introduce primitive Formal Ordered Tree System (pFOTS) as formal system defining labeled ordered tree languages. A pFOTS is defined as a set of graph rewriting rules. The query learning model is an established learning model in computational learning theory. In the query learning model, we discuss the learnability of one or more graph rewriting rules under background knowledge defined by a pFOTS using a polynomial number of queries.

1 Introduction

Recently, many graph structured data become accessible on Internet. Especially, graph structured data such as HTML/XML files having tree structures are called tree structured data. Graph grammar has been applied to a wide range of fields including pattern recognition and image analysis. Uchida et al. [7] introduced a framework called Formal Graph System (FGS) as a graph grammar. FGS is a logic programming system that deals with term graph patterns instead of terms of first-order predicate logic. In this paper, based on FGS, we introduce primitive Formal Ordered Tree System (pFOTS) as formal system defining labeled ordered tree languages. A pFOTS is defined as a finite set of graph rewriting rules that replace variables of ordered term tree patterns with specified ordered trees. In Fig.1, we give a pFOTS $\Gamma_{\mathcal{OT}}$, a rule α and the language $\mathcal{L}(\Gamma_{\mathcal{OT}} \cup \{\alpha\}, r)$ defined by the pFOTS $\Gamma_{\mathcal{OT}} \cup \{\alpha\}$ and the predicate symbol r as examples.

In the query learning model [1], a learning algorithm accesses to oracles, which answer specific kinds of queries, and collects information about a target. In this paper, firstly we discuss the learnability of one or more graph rewriting rules under background knowledge defined by a pFOTS in the query learning model. Secondly, we show that if background knowledge is given by a pFOTS which has at least two edge labels and one predicate symbol, then one graph rewriting rule is exactly learnable with one positive example and a polynomial number of membership queries w.r.t. the size of the positive example.

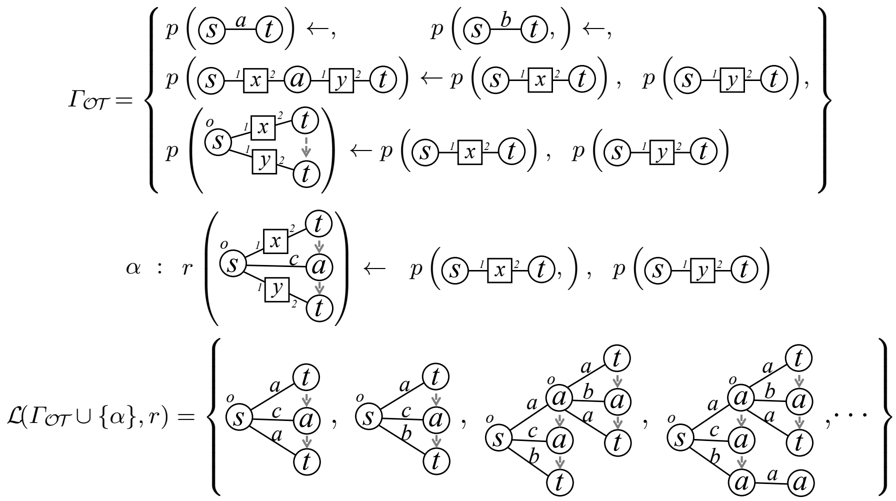

Fig. 1. A pFOTS $\Gamma_{\mathcal{OT}}$, a rule α and the labeled ordered tree language $\mathcal{L}(\Gamma_{\mathcal{OT}} \cup \{\alpha\}, r)$ defined by the pFOTS $\Gamma_{\mathcal{OT}} \cup \{\alpha\}$ and the predicate symbol r. The symbol o over internal nodes indicates that the nodes has children in order shown by broken arrows. Variables are drawn by squares each of which connects to two nodes ordered by numbers on lines.

For the learning of graph grammar, Okada et al. [4] showed that some classes of graph pattern languages are exactly learnable with a polynomial number of equivalence and restricted subset queries. Shoudai et al. [5] showed that the regular FGS languages of bounded degree with the 1-finite context property (1-FCP) and bounded treewidth property can be learned from positive data and membership queries with current distributional learning techniques [2].

2 Formal Ordered Tree Systems

Let Σ and Λ be finite alphabets whose elements are called *node-labels* and *edge-labels*, respectively. Let $\mathcal{X}$ be an infinite alphabet whose element is called a *variable label*. We assume that $\Lambda \cap \mathcal{X} = \emptyset$. A node- and edge-labeled ordered tree $t = (V_t, E_t)$ over $\langle \Sigma, \Lambda \cup \mathcal{X} \rangle$ is an ordered tree that has a node labeling function $\psi_t : V_t \to \Sigma$ and an edge labeling function $\varphi_t : E_t \to \Lambda \cup \mathcal{X}$. An edge labeled with a variable label in $\mathcal{X}$ is called a *variable*. For any $x \in \mathcal{X}$, $o(t, x)$ denotes the number of variables in t labeled with x. A node- and edge-labeled ordered tree t over $\langle \Sigma, \Lambda \cup \mathcal{X} \rangle$ is said to be a *linear term tree pattern* over $\langle \Sigma, \Lambda \cup \mathcal{X} \rangle$ if, for any $x \in \mathcal{X}$, $o(t, x) \leq 1$ holds. For a node $v \in V_t$ except the root of t and the parent $u \in V_t$ of v, the edge e between u and v is denoted by (u, v) if $\varphi_t(e) \in \Lambda$ and by $\langle u, v \rangle$ if $\varphi_t(e) \in \mathcal{X}$, respectively. Hereafter, an edge of t means an edge whose label is in Λ. A linear term tree pattern over $\langle \Sigma, \Lambda \cup \mathcal{X} \rangle$ with no variable is a node- and edge-labeled ordered tree over $\langle \Sigma, \Lambda \rangle$ and is simply called a *tree* over $\langle \Sigma, \Lambda \rangle$. $\mathcal{OT}$ denotes the set of all trees over $\langle \Sigma, \Lambda \rangle$. Hereafter, a linear term tree pattern over $\langle \Sigma, \Lambda \cup \mathcal{X} \rangle$ is simply called a *term tree pattern.* A term tree pattern is said to be *primitive* if it consists of two nodes and one variable between them.

Let f and g be term tree patterns with at least two nodes. Let $\sigma = [u, v]$ be a pair of the root u and a leaf v of g and x a variable label in $\mathcal{X}$. The form $x := [g, \sigma]$ is called a *binding* over $\langle \Sigma, \Lambda \cup \mathcal{X} \rangle$. A new term tree pattern, denoted by $f\{x := [g, \sigma]\}$, is obtained by applying the binding $x := [g, \sigma]$ to f in the following way: Let $e = \langle s, t \rangle$ be a variable in f with the variable label x, i.e., $\varphi_f(e) = x$. Let g' be a copy of g, and u' and v' be the nodes of g' corresponding to u and v of g, respectively. For $e = \langle s, t \rangle$, we attach g' to f by removing e from f and then identifying s with u' and t with v'. In order that all internal nodes of $f\{x := [g, \sigma]\}$ have the ordered children, the children of s needs to be reordered (please refer to [6]). A *substitution* θ is a finite set of bindings $\{x_1 := [g_1, \sigma_1],\ x_2 := [g_2, \sigma_2], \ldots,\ x_n := [g_n, \sigma_n]\}$, where x_i's are mutually distinct variable labels in $\mathcal{X}$. For a term tree pattern f and a substitution θ, we denote by $f\theta$ the term tree pattern obtained from f and θ by applying all bindings in θ to f simultaneously.

Let Π be a set of unary predicate symbols. We assume that each predicate symbol $p \in \Pi$ is assigned a pair of distinct two symbols in Σ. The pair is denoted by $pointer(p)$. Let p be a unary predicate symbol in Π and t a term tree pattern. An *atom* is an expression of the form $p(t)$. Let $A, B_1, B_2, \ldots, B_n$ be atoms, where $n \geq 0$. A *graph rewriting rule* (*rule*, for short) over $\langle \Pi, \Sigma, \Lambda \cup \mathcal{X} \rangle$ is a clause of the form $A \leftarrow B_1, B_2, \ldots, B_n$. The atom A is called the *head* and the right part $B_1, B_2, \ldots, B_n$ is called the *body* of the rule. If $n = 0$, the rule is called a *fact.* A rule $p(t) \leftarrow q_1(f_1),\ q_2(f_2), \ldots,\ q_n(f_n)$ is said to be *primitive* if the following conditions (1)–(3) hold: (1) if $n \geq 1$, f_i are primitive term tree patterns over $\langle \{a \mid a \in pointer(q_i)\}, \mathcal{X} \rangle$ for all i $(1 \leq i \leq n)$, (2) if $n = 0$, t is a tree over $\langle \{a \mid a \in pointer(p)\}, \Lambda \rangle$ consisting of two nodes and the edge between them, otherwise t is a term tree pattern such that each symbol in $pointer(p)$ appears in t, and (3) for any variable $x \in \mathcal{X}$, $o(t, x) = 1$ if and only if $o(f_1, x) + o(f_2, x) + \cdots + o(f_n, x) = 1$. For example, the rule α in Fig. 1 is primitive. $p\mathcal{GRR}$ denotes the set of all primitive rules over $\langle \Pi, \Sigma, \Lambda \cup \mathcal{X} \rangle$. For a *rule* $\alpha =$'$p(t) \leftarrow q_1(f_1),\ q_2(f_2), \ldots,\ q_n(f_n)$', let $\Pi^h(\alpha) = \{p\}$ and $\Pi^b(\alpha) = \{q_1, q_2, \ldots, q_n\}$. A finite set Γ of primitive rules is called a *primitive Formal Ordered Tree System* (*pFOTS*, for short) if Γ has only one predicate symbol in Π. As an example, we give a pFOTS $\Gamma_{\mathcal{OT}} \cup \{\alpha\}$ in Fig. 1.

For an atom $p(g)$, a rule $A \leftarrow B_1, \ldots, B_n$ and a substitution θ, we define $p(g)\theta = p(g\theta)$ and $(A \leftarrow B_1, \ldots, B_n)\theta = A\theta \leftarrow B_1\theta, \ldots, B_n\theta$. Let Γ be a pFOTS. The relation $\Gamma \vdash C$ for a rule C is inductively defined as follows. (1) If $C \in \Gamma$, then $\Gamma \vdash C$. (2) If $\Gamma \vdash C$, then $\Gamma \vdash C\theta$ for any substitution θ. (3) If $\Gamma \vdash A \leftarrow B_1, \ldots, B_i, \ldots, B_n$ and $\Gamma \vdash B_i \leftarrow C_1, \ldots, C_m$,
then $\Gamma \vdash A \leftarrow B_1, \ldots, B_{i-1}, C_1, \ldots, C_m, B_{i+1}, \ldots, B_n$. For a pFOTS Γ and its predicate symbol p in Π, $\mathcal{L}(\Gamma, p)$ denotes the subset $\{g \in \mathcal{OT} \mid \Gamma \vdash p(g) \leftarrow\}$ of $\mathcal{OT}$. We say that a subset $L \subseteq \mathcal{OT}$ is a *pFOTS language* if there exists a pFOTS Γ and its predicate symbol p such that $L = \mathcal{L}(\Gamma, p)$ holds. In Fig. 1, we give the pFOTS language $\mathcal{L}(\Gamma_{\mathcal{OT}} \cup \{\alpha\}, r)$ defined by pFOTS $\Gamma_{\mathcal{OT}} \cup \{\alpha\}$ as an example.

3 Learning of Primitive Formal Ordered Tree Systems via Queries

For a pFOTS Γ, $\Pi^h(\Gamma) = \bigcup_{\alpha\in\Gamma} \Pi^h(\alpha)$, $\Pi^b(\Gamma) = \bigcup_{\alpha\in\Gamma} \Pi^b(\alpha)$ and $\Pi(\Gamma) = \Pi^h(\Gamma) \cup \Pi^b(\Gamma)$. For a predicate symbol $r \in \Pi \setminus \Pi(\Gamma)$, let $p\mathcal{GRR}(\Gamma, r)$ be the set of all primitive rules α such that $\Pi^h(\alpha) = \{r\}$ and $\Pi^b(\alpha) \subseteq \Pi^h(\Gamma)$. We assume that our learning algorithm knows a fixed pFOTS Γ, which is referred to as BK, in advance as background knowledge. Then, we consider the learnability of the class $\{\mathcal{L}(\Gamma \cup \{\alpha\}, r) \mid \alpha \in p\mathcal{GRR}(\Gamma, r)\}$ via queries, where $r \in \Pi \setminus \Pi(\Gamma)$. The rule α_* denotes the rule in $p\mathcal{GRR}(\Gamma, q)$ to be identified, which is called the *target*, where $q \in \Pi \setminus \Pi(\Gamma)$. Any tree in $\mathcal{L}(\Gamma \cup \{\alpha_*\}, q)$ is said to be a *positive example*. In the query learning model [1], a learning algorithm can access to *oracles* that will answer queries about the target α_*. We consider the following three queries. Let $r, q \in \Pi \setminus \Pi(\Gamma)$. **Membership query(MQ)**: The input is a tree $t \in \mathcal{OT}$. The output is `yes` if $t \in \mathcal{L}(\Gamma \cup \{\alpha_*\}, r)$, otherwise `no`. **Restricted subset query(rSQ)**: The input is a rule $\alpha \in p\mathcal{GRR}(\Gamma, r)$. The output of a restricted subset query is `yes` if $\mathcal{L}(\Gamma \cup \{\alpha\}, r) \subseteq \mathcal{L}(\Gamma \cup \{\alpha_*\}, q)$, otherwise `no`. **Equivalence query(EQ)**: The input is a rule α in $p\mathcal{GRR}(\Gamma, r)$. The output of an equivalence query is `yes` if $\mathcal{L}(\Gamma \cup \{\alpha\}, r) = \mathcal{L}(\Gamma \cup \{\alpha_*\}, q)$, otherwise a tree, called a *counterexample*, in $(\mathcal{L}(\Gamma \cup \{\alpha\}, r) \cup \mathcal{L}(\Gamma \cup \{\alpha_*\}, q)) \setminus (\mathcal{L}(\Gamma \cup \{\alpha\}, r) \cap \mathcal{L}(\Gamma \cup \{\alpha_*\}, q))$. A learning algorithm A is said to *exactly identify the target* $\alpha_* \in p\mathcal{GRR}(\Gamma, q)$ if A outputs a rule $\alpha \in p\mathcal{GRR}(\Gamma, r)$ such that $r \in \Pi \setminus \Pi(\Gamma)$ and $\mathcal{L}(\Gamma \cup \{\alpha\}, r) = \mathcal{L}(\Gamma \cup \{\alpha_*\}, q)$. In case that a target is a finite subset of $p\mathcal{GRR}(\Gamma, r)$, the above queries and the exact identifications are defined in a similar way.

We summarize related work and the results of this paper in Table 1. PII and ILPU stand for the learning models "polynomial-time inductive inference from positive data" and "identification in the limit by polynomial-time update from positive data," respectively. OP means that the learning algorithm in the result needs exactly one positive example. We say that a pFOTS Γ satisfies the Condition 1 if Γ has the 1-finite context property with Chomsky normal form and constant degree property [5]. Furthermore we say that Γ over $\langle \Pi, \Sigma, \Lambda \cup \mathcal{X} \rangle$ satisfies Condition 2 if for each predicate symbol $p \in \Pi(\Gamma)$, there exist two edge labels $a_1, a_2 \in \Lambda$ $(a_1 \neq a_2)$ such that $P(a_1) \cap P(a_2) = \{p\}$, where $P(a) = \{q \in \Pi(\Gamma) \mid \text{there is a fact } \alpha \text{ over } \langle \Pi(\Gamma), \Sigma, \{a\} \rangle \text{ s.t. } q \in \Pi^h(\alpha)\}$. We can extend pFOTS to deal with the class of labeled ordered tree languages in case of $|\Lambda| = \infty$ by using the special atom deciding whether or not the edge label is in Λ.

Let Γ be a pFOTS with $\Pi(\Gamma) = \{p\}$. $\mathcal{F}(\Gamma)$ denotes the set of all facts in Γ. For a predicate symbol $r \in \Pi \setminus \{p\}$ and a rule $\alpha =$ '$p(s) \leftarrow p(s_1), \ldots, p(s_n)$' $\in \Gamma$, let $Tr_r^p(\alpha) =$ '$r(s) \leftarrow p(s_1), \ldots, p(s_n)$'. Let f, g be trees and $e = (u, v)$ an edge in f with $\varphi_f(e) \in \Lambda$. Let f' be the term tree pattern obtained from f by changing the edge label $\varphi_f(e)$ of e to a variable label $x \in \mathcal{X}$, i.e., $\varphi_{f'}(e) = x$. Then, we write $g \rhd_\Gamma^e f$ if there exists a substitution $\theta = \{x := [t, \sigma_t]\}$ such that the following three conditions hold: (1) $f'\theta$ is isomorphic to g, (2) t is in $\bigcup_{\alpha \in W} \mathcal{L}(\mathcal{F}(\Gamma) \cup \{Tr_r^p(\alpha)\}, r)$, where $W = \Gamma \setminus \mathcal{F}(\Gamma)$, and (3) σ_t is the pair (u, v) of two nodes u, v of t with $(\psi_t(u), \psi_t(v)) = pointer(p)$. Moreover, we write $g \rhd_\Gamma f$ if there exists an edge e in f such that $g \rhd_\Gamma^e f$ holds.

Table 1. Summary on the learnability of pFOTSs.

	BK: $\Gamma_{\mathcal{OT}}$, Target : rule α	BK: Γ, Target: rule α	
		$\lvert\Pi(\Gamma)\rvert = 1$	$\lvert\Pi(\Gamma)\rvert \geq 2$
$\lvert\Lambda\rvert = 1$	PII [6]	Open	
$2 \leq \lvert\Lambda\rvert < \infty$		OP & MQ [This paper; Th. 1]	OP & MQ if Cond. 2 holds [This paper; Cor. 1]
$\lvert\Lambda\rvert = \infty$			

	BK: $\Gamma_1 \cup \cdots \cup \Gamma_K$, Target: rules $\alpha_1, \ldots, \alpha_N$, where $\lvert\Pi(\Gamma_k)\rvert = 1$ $(1 \leq k \leq K)$ and $\Pi(\Gamma_i) \cap \Pi(\Gamma_j) = \emptyset$ $(1 \leq i < j \leq K)$		BK: none Target: Γ
	$K > 1, N = 1$	$K = 1, N > 1$	
$\lvert\Lambda\rvert = 1$	Open		ILPU & MQ if Cond. 1 holds [5]
$2 \leq \lvert\Lambda\rvert < \infty$	OP & MQ if Cond. 2 holds [This paper; Cor. 2]	Open	
$\lvert\Lambda\rvert = \infty$		EQ & rSQ [3, 4]	

Theorem 1. *Let $|\Lambda| \geq 2$. Let Γ be a fixed pFOTS over $\langle \Pi, \Sigma, \Lambda \cup \mathcal{X} \rangle$ and r a predicate symbol in $\Pi \setminus \Pi(\Gamma)$. Given one positive example $t \in \mathcal{OT}$, the algorithm LEARNING_pFOTS in Fig 1 exactly identifies the target in $p\mathcal{GRR}(\Gamma, r)$ using $O(N^2)$ membership queries, where N is the number of edges in t.*

Corollary 1. *Let Γ be a fixed pFOTS over $\langle \Pi, \Sigma, \Lambda \cup \mathcal{X} \rangle$ which satisfies Condition 2, $|\Lambda| \geq 2$ and $|\Pi(\Gamma)| \geq 2$ and r a predicate symbol in $\Pi \setminus \Pi(\Gamma)$. Given one positive example $t \in \mathcal{OT}$, there exists a learning algorithm which exactly identifies the target in $p\mathcal{GRR}(\Gamma, r)$ using $O(N^4)$ membership queries, where N is the number of edges in t.*

Corollary 2. *Let $\Gamma_1, \ldots, \Gamma_K$ be fixed pFOTSs over $\langle \Pi, \Sigma, \Lambda \cup \mathcal{X} \rangle$ such that $\bigcup_{1 \leq j \leq K} \Gamma_j$ satisfies Condition 2, $|\Lambda| \geq 2$, $|\Pi(\Gamma_i)| = 1$ for any i $(1 \leq i \leq K)$ and $\Pi(\Gamma_i) \cap \Pi(\Gamma_j) = \emptyset$ for any i, j $(1 \leq i < j \leq K)$. Let r be a predicate symbol in $\Pi \setminus \bigcup_{1 \leq i \leq K} \Pi(\Gamma_i)$. Given one positive example $t \in \mathcal{OT}$, there exists a learning algorithm which exactly identifies the target in $p\mathcal{GRR}(\Gamma_1 \cup \cdots \cup \Gamma_K, r)$ using $O(N^2)$ membership queries, where N is the number of edges of t.*

4 Conclusions

In this paper, based on FGS, we have introduced primitive Formal Ordered Tree System (pFOTS) defining the labeled ordered tree languages. We have discussed the learnability of one or more graph rewriting rules under background knowledge defined by a pFOTS in the query learning model. As future work, we

Algorithm 1 *LEARNING_pFOTS*

Input: A tree $t \in \mathcal{OT}$.
Output: A rule $\alpha \in p\mathcal{GRR}(\Gamma, r)$ such that $\mathcal{L}(\Gamma \cup \{\alpha\}, r) = \mathcal{L}(\Gamma \cup \{\alpha_*\}, q)$.
Assumption: A pFOTS Γ is known as BK. Predicate symbols r and q are in $\Pi \setminus \Pi(\Gamma)$ and the target α_* is in $p\mathcal{GRR}(\Gamma, q)$.

repeat
  for all $f \in \{t' \in \mathcal{OT} \mid t \rhd_\Gamma t'\}$ **do**
    if $\mathbf{MQ}(f) = \mathtt{yes}$ **then**
      $t := f$; **break**;
    end if
  end for
until t does not change;
$f := t$; $n := 0$;
for all edges e_f in f **do**
  Let μ be an edge label which is different from the edge label of e_f;
  Let f' be a tree obtained from f by replacing e_f with an edge labeled with μ;
  if $\mathbf{MQ}(f') = \mathtt{yes}$ **then**
    $n := n + 1$;
    Replace the edge e_t of t that corresponds to e_f with a new variable h_n;
  end if
end for
// We regard the variable h_i as a primitive term tree pattern consisting of h_i $(i = 1, \ldots, n)$.
output $\alpha := r(t) \leftarrow p(h_1), p(h_2), \ldots, p(h_n)$;

consider the learnability of one or more graph rewriting rules under background knowledge which has only one edge label. This work was partially supported by JSPS KAKENHI Grant Numbers 15K00312, 15K00313, 17K00321.

References

1. Angluin, D.: Queries and concept learning. Machine Learning 2(4), 319–342 (1988)
2. Clark, A., Yoshinaka, R.: Distributional learning of context-free and multiple context-free grammars. In: Topics in Grammatical Inference, pp. 143–172. Springer-Verlag Berlin Heidelberg (2016)
3. Matsumoto, S., Shoudai, T., Uchida, T., Miyahara, T., Suzuki, Y.: Learning of finite unions of tree patterns with internal structured variables from queries. IEICE Trans. Inf. & Syst. E91-D(2), 222–230 (2008)
4. Okada, R., Matsumoto, S., Uchida, T., Suzuki, Y., Shoudai, T.: Exact learning of finite unions of graph patterns from queries. In: Proc. ALT 2007, LNAI 4754. pp. 298–312. Springer (2007)
5. Shoudai, T., Matsumoto, S., Suzuki, Y.: Distributional learning of regular formal graph system of bounded degree. In: Proc. ILP 2016. Springer (To appear)
6. Suzuki, Y., Shoudai, T., Uchida, T., Miyahara, T.: Ordered term tree languages which are polynomial time inductively inferable from positive data. Theoretical Computer Science 350(1), 63–90 (2006)
7. Uchida, T., Shoudai, T., Miyano, S.: Parallel algorithms for refutation tree problem on formal graph systems. IEICE Trans. Inf. & Syst. E78-D(2), 99–112 (1995)