Distributional Learning of Regular Formal Graph System of Bounded Degree

Takayoshi Shoudai[1⋆], Satoshi Matsumoto[2], and Yusuke Suzuki[3]

[1] Faculty of International Studies, Kyushu International University, Japan
shoudai@isb.kiu.ac.jp
[2] Faculty of Science, Tokai University, Japan
[3] Graduate School of Information Sciences, Hiroshima City University, Japan

Abstract. In this paper, we describe how distributional learning techniques can be applied to formal graph system (FGS) languages. Formal graph system is a logic program that deals with term graphs instead of the terms of first-order predicate logic. We show that the FGS languages of bounded degree with the 1-finite context property (1-FCP) and the bounded treewidth property can be learned from positive data and membership queries.

1 Introduction

In the field of algorithmic learning theory, many models and algorithmic techniques for learning from examples have been developed. Distributional learning was proposed firstly by Clark and Eyraud [2] to learn a subclass of context-free grammars efficiently. Recently distributional learning have been developed in learning of various subclasses of context-free grammars [7]. Those techniques were extended to languages that have more complex structures [4].

On the other hand, graph grammar has been developed as an extension to graphs from a string of the concept of grammatical forms. Graph grammar has been applied to a wide range of fields including pattern recognition, image analysis, and so on. Uchida et al. [6] introduced a framework called formal graph system (FGS, for short) as one of the graph grammars. An FGS is a logic program that deals with term graphs, which can be considered to be a kind of hypergraphs, instead of the terms of first-order predicate logic.

For the learning of graph grammar, Okada et al. [5] showed that some classes of graph pattern languages is MAT learnable in polynomial time. There are some early studies in other, but discussions on computational learning of graph grammars are not sufficient yet. In this paper, by the existing distributional learning techniques [7], we show that the FGS languages of bounded degree with the 1-finite context property (1-FCP) [3] and the bounded treewidth property can be learned from positive data and membership queries.

⋆ This work was partially supported by JSPS KAKENHI (26280087) and MEXT KAKENHI (24106010).

2 Preliminaries

For a set or a list S, $|S|$ denotes the number of all elements that are contained in S. For a set V, V^* denotes the set of all finite lists consisting of elements in V. For a list S and an integer i ($1 \leq i \leq |S|$), $S[i]$ denotes the i-th member of S. Let Σ and Λ be finite alphabets. Let X be an infinite alphabet, whose elements are called *variables*. We assume that each symbol $x \in X$ has a nonnegative integer $rank(x)$, $\Sigma \cap X = \emptyset$ and $\Lambda \cap X = \emptyset$.

Definition 1 (Term graph). *A term graph $g = (V, E, \varphi, \psi, H, \lambda, ports)$ is defined as follows:*

1. *(V, E) is a vertex- and edge-labeled (directed or undirected) graph,*
2. *$\varphi : V \to \Sigma$ and $\psi : E \to \Lambda$ are vertex- and edge-labeling functions,*
3. *H is a finite multiset of hyperedges that are elements in 2^V,*
4. *$\lambda : H \to X$ is a variable-labeling function, and*
5. *$ports : H \to V^*$ is a mapping s.t. for every $h \in H$, $ports(h)$ is a list of $rank(\lambda(h))$ distinct vertices in V. These vertices are called the ports of h.*

We denote the 7-tuple of a term graph g by $(V_g, E_g, \varphi_g, \psi_g, H_g, \lambda_g, ports_g)$. A term graph g is called *ground* if $H_g = \emptyset$. λ_g and $ports_g$ are empty functions $\emptyset$. For $\sigma \in V_g^*$, $\varphi_g(\sigma)$ denotes $(\varphi_g(v_1), \ldots, \varphi_g(v_\ell))$. Let d be a nonnegative integer. $\mathcal{G}(\Sigma, \Lambda, X)$ (resp. $\mathcal{G}_d(\Sigma, \Lambda, X)$) denotes the set of all term graphs (resp. all term graphs of maximum degree d) over $\langle \Sigma, \Lambda, X \rangle$. Moreover, $\mathcal{G}(\Sigma, \Lambda)$ (resp. $\mathcal{G}_d(\Sigma, \Lambda)$) denotes the set of all ground term graphs (resp. all gournd term graphs of maximum degree d). A term graph g is a *star term graph* if $E_g = \emptyset$ and $H_g = \{h\}$ for some hyperedge h s.t. $h = V_g$. For a star term graph g, h_g denotes the unique hyperedge of g.

Definition 2 (Formal graph system (FGS)). *Let $g_1, \ldots, g_n \in \mathcal{G}(\Sigma, \Lambda, X)$ $(n \geq 1)$. Let Π_n be a finite set of n-ary predicate symbols. Let $\Pi = \bigcup_{i \geq 0} \Pi_i$. For $p \in \Pi_n$, we say that $p(g_1, \ldots, g_n)$ is an atom. Let $A, B_1, \ldots, B_m$ be atoms $(m \geq 0)$. A graph rewriting rule over $\langle \Sigma, \Lambda, X, \Pi \rangle$ is a clause of the form $A \leftarrow B_1, \ldots, B_m$. A formal graph system (FGS) is denoted by $S = (\Sigma, \Lambda, X, \Pi, \Gamma)$, where Γ a finite set of graph rewriting rules over $\langle \Sigma, \Lambda, X, \Pi \rangle$.*

Let f be a term graph in $\mathcal{G}(\Sigma, \Lambda, X)$ and σ an ordered list of ℓ distinct vertices in V_f ($0 \leq \ell \leq |V_f|$). A pair $[f, \sigma]$ is called a *term graph fragment*. If f is a ground term graph, we call it a *ground term graph fragment*. Let $\mathcal{F}(\Sigma, \Lambda)$ be the set of all ground term graph fragments. For a nonnegative integer d, $\mathcal{F}_d(\Sigma, \Lambda) = \{[f, \sigma] \in \mathcal{F}(\Sigma, \Lambda) \mid f \in \mathcal{G}_d(\Sigma, \Lambda)\}$. For a term graph fragment $[f, \sigma]$ and a variable $x \in X$ with $rank(x) = |\sigma|$. Let $\sigma = (v_1, \ldots, v_\ell)$ ($\ell \geq 1$). The *binding* $x := [f, \sigma]$ is defined to be an operation on a term graph g that works in the following way: For each $h \in H_g$ with $\lambda_g(h) = x$, let $f' = (V_{f'}, E_{f'}, \varphi_{f'}, \psi_{f'}, H_{f'}, \lambda_{f'}, ports_{f'})$ be a copy of f. For a vertex $v \in V_f$, we denote the corresponding copy vertex of f' by v'. We attach f' to g by removing the hyperedge h from H_g and by identifying the ports $u_1, \ldots, u_\ell$ of h in g with $v'_1, \ldots, v'_\ell$ in f', respectively. We set the new vertex-label of u_i to be the original vertex-label of u_i, i.e., $\varphi_g(u_i)$.

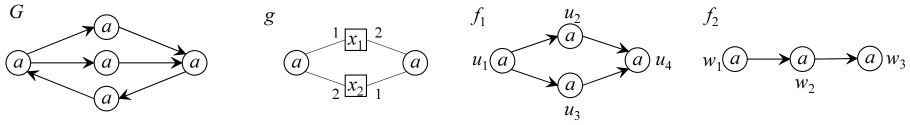

Fig. 1. A graph G can be obtained from g by applying a substitution $\theta = \{x_1 := [f_1, (u_1, u_4)], x_2 := [f_2, (w_1, w_3)]\}$, i.e., $g\theta$ is isomorphic to G.

A *substitution* θ is a finite set of bindings $\{x_1 := [f_1, \sigma_1], \ldots, x_n := [f_n, \sigma_n]\}$, where x_i's are mutually distinct variables in X and each f_i has no hyperedge labeled with a variable in $\{x_1, \ldots, x_n\}$. For a substitution θ and an atom $p(g_1, \ldots, g_n)$, we define $p(g_1, \ldots, g_n)\theta$ to be $p(g_1\theta, \ldots, g_n\theta)$. For a graph rewriting rule $A \leftarrow B_1, \ldots, B_m$, we also define $(A \leftarrow B_1, \ldots, B_m)\theta$ to be $A\theta \leftarrow B_1\theta, \ldots, B_m\theta$. We give an example of term graphs and substitutions in Fig. 1. For the term graphs in those figures, a hyperedge is represented by a box with lines to its ports. The order of the ports is indicated by digits at these lines.

Let $S = (\Sigma, \Lambda, X, \Pi, \Gamma)$ be an FGS. Let δ be a function from a predicate symbol $p \in \Pi$ to a finite list of distinct symbols $(a_1, \ldots, a_k) \in \Sigma^*$ for some k ($k \geq 0$). Not only the symbols a_i ($0 \leq i \leq k$) but also the length k depend on p. We call the function δ a *pointer* of predicate symbols. Let $\delta(\Pi)$ be the set of all symbols appearing in $\delta(p)$ for all predicates $p \in \Pi$.

Definition 3 (Regular FGS). *We say that an FGS $S = (\Sigma, \Lambda, X, \Pi, \Gamma)$ is regular with a pointer δ if all graph rewriting rules in Γ are of the form $q_0(g_0) \leftarrow q_1(g_1), \ldots, q_m(g_m)$ that satisfies the following conditions:*

1. *All $q_i \in \Pi$ ($0 \leq i \leq m$) are 1-ary predicate symbols.*
2. *Each g_i ($i = 1, \ldots, m$) is a star term graph s.t. $\varphi_{g_i}(ports_{g_i}(h_{g_i})) = \delta(q_i)$.*
3. *There is a list $(v_1, \ldots, v_{|\delta(q_0)|}) \in V_{g_0}^{|\delta(q_0)|}$ s.t. $\varphi_{g_0}(v_1, \ldots, v_{|\delta(q_0)|}) = \delta(q_0)$ and for any $u \in V_{g_0} \setminus \{v_1, \ldots, v_{|\delta(q_0)|}\}$, $\varphi_{g_0}(u) \in \Sigma \setminus \delta(\Pi)$.*
4. *$\lambda_{g_0}(H_{g_0}) = \bigcup_{i=1}^{m} \lambda_{g_i}(H_{g_i})$ and $\lambda_{g_i}(H_{g_i}) \cap \lambda_{g_j}(H_{g_j}) = \emptyset$ for $1 \leq i < j \leq m$.*
5. *For every $h_1, h_2 \in H_{g_0}$, $h_1 \neq h_2$ iff $\lambda_{g_0}(h_1) \neq \lambda_{g_0}(h_2)$.*

A regular FGS $S = (\Sigma, \Lambda, X, \Pi, \Gamma)$ with a pointer δ is denoted by (S, δ) or $((\Sigma, \Lambda, X, \Pi, \Gamma), \delta)$.

Let f_0 be a ground term graph of one vertex or two vertices with one edge, f_1 a term graph with two hyperedges and no edge, and f_2, f_3 star term graphs. Let p_0, p_1, p_2, p_3 be unary predicate symbols. A regular FGS (S, δ) is in *Chomsky normal form* (CNF) if every graph rewriting rule of S is of the form.

- Terminal rule: $p_0(f_0) \leftarrow$,
- Binary rule: $p_1(f_1) \leftarrow p_2(f_2),\ p_3(f_3)$.

Definition 4 (Regular FGS language). *Let $S = (\Sigma, \Lambda, X, \Pi, \Gamma)$ be an FGS. A relation $\Gamma \vdash C$ is defined recursively in the following way:*

1. *If $C \in \Gamma$, then $\Gamma \vdash C$ holds.*
2. *If $\Gamma \vdash C$, then $\Gamma \vdash C\theta$ for an arbitrary substitution θ.*
3. *If $\Gamma \vdash A \leftarrow B_1, \ldots, B_n$ and for some i ($1 \leq i \leq n$), $\Gamma \vdash B_i \leftarrow C_1, \ldots, C_m$, then $\Gamma \vdash A \leftarrow B_1, \ldots, B_{i-1}, C_1, \ldots, C_m, B_{i+1}, \ldots, B_n$ holds.*

Let (S, δ) be a regular FGS and p a unary predicate symbol in Π. We define the graph language of (S, δ, p) as $GL(S, \delta, p) = \{g \in \mathcal{G}(\Sigma, \Lambda) \mid \Gamma \vdash p(g) \leftarrow\}$. We say that a graph language $L \subseteq \mathcal{G}(\Sigma, \Lambda)$ is definable by regular FGS or a regular FGS language if such a triplet (S, δ, p) exists.

3 Learning Regular FGS with 1-Finite Context Property

We assume that our learner has an access to an oracle Mem_{L_*} who answers membership queries. The queries ask whether or not an arbitrary ground term graph $g \in \mathcal{G}(\Sigma, \Lambda)$ is included in a target graph language L_*. The answer is, denoted by $Mem_{L_*}(g)$, either *yes* or *no*.

Let $g = (V_g, E_g, \varphi_g, \psi_g, \emptyset, \emptyset, \emptyset)$ be a ground term graph and $\sigma = (v_1, \ldots, v_\ell)$ a list of vertices in V_g ($1 \leq \ell \leq |V_g|$). Let x be a variable label in X that does not appear anywhere. For the term graph fragment $[g, \sigma]$, we denote by $g(\sigma)$ the term graph $(V_g, E_g, \varphi_g, \psi_g, \{h\}, \lambda_g, ports_g)$ where $h = \{v_1, \ldots, v_\ell\}$, $\lambda_g(h) = x$, and $ports_g(h) = \sigma$. In order to make the argument easier, we assume that g has no isolated vertex. Let $\{E_\alpha, E_\beta\}$ be a partition of E_g. Let V_α be the set of all endpoints of edges in E_α and V_β the set of all endpoints of edges in E_β. Let σ be one of the ordered list consisting of all the vertices in $V_\alpha \cap V_\beta$. Then, we obtain two term graph fragments $[\alpha, \sigma]$ and $[\beta, \sigma]$. We easily see that $\alpha(\sigma)\{x := [\beta, \sigma]\}$ and $\beta(\sigma)\{x := [\alpha, \sigma]\}$ are isomorphic to g.

For $[\alpha, \sigma_\alpha], [\beta, \sigma_\beta] \in \mathcal{F}(\Sigma, \Lambda)$, we define an operation $\odot$ as follows:

$$[\alpha, \sigma_\alpha] \odot [\beta, \sigma_\beta] = \begin{cases} \alpha(\sigma_\alpha)\{x := [\beta, \sigma_\beta]\} & \text{if } |\sigma_\alpha| = |\sigma_\beta|, \\ \mathit{undefined} & \text{otherwise.} \end{cases}$$

Note that in general, $[\alpha, \sigma_\alpha] \odot [\beta, \sigma_\beta]$ is not always equivalent to $[\beta, \sigma_\beta] \odot [\alpha, \sigma_\alpha]$, because the vertex labels in the first operand always survive by any binding. If $\varphi_\alpha(\sigma_\alpha) = \varphi_\beta(\sigma_\beta)$, $[\alpha, \sigma_\alpha] \odot [\beta, \sigma_\beta] = [\beta, \sigma_\beta] \odot [\alpha, \sigma_\alpha]$ holds.

Let d be a nonnegative integer. For a nonempty finite set of ground term graphs $D \subseteq \mathcal{G}_d(\Sigma, \Lambda)$, let

$$Sub(D) = \{[\beta, \sigma_\beta] \in \mathcal{F}_d(\Sigma, \Lambda) \mid \exists [\alpha, \sigma_\alpha] \in \mathcal{F}_d(\Sigma, \Lambda)[\,[\alpha, \sigma_\alpha] \odot [\beta, \sigma_\beta] \in D\,]\},$$
$$Con(D) = \{[\alpha, \sigma_\alpha] \in \mathcal{F}_d(\Sigma, \Lambda) \mid \exists [\beta, \sigma_\beta] \in \mathcal{F}_d(\Sigma, \Lambda)[\,[\alpha, \sigma_\alpha] \odot [\beta, \sigma_\beta] \in D\,]\}.$$

Note that both $|Sub(D)|$ and $|Con(D)|$ are of polynomial size w.r.t. $|D|$. If $[\beta, \sigma_\beta] \in Sub(D)$ with $\sigma_\beta = (v_1, \ldots, v_\ell)$, any ground term graph fragment $[(V_\beta, E_\beta, \varphi'_\beta, \psi_\beta, \emptyset, \emptyset, \emptyset), \sigma_\beta]$ with $\varphi'_\beta|(V_\beta \setminus \{v_1, \ldots, v_\ell\}) = \varphi_\beta|(V_\beta \setminus \{v_1, \ldots, v_\ell\})$ is also in $Sub(D)$. Therefore, $Con(D) \subseteq Sub(D)$ holds.

Let $(S, \delta) = ((\Sigma, \Lambda, X, \Pi, \Gamma), \delta)$ be a regular FGS. For a term graph f and $q \in \Pi$, if for all i ($1 \leq i \leq |\delta(q)|$), $\exists v_i \in V_f$ s.t. $\varphi_f(v_i) = \delta(q)[i]$, let $\varphi_f^{-1}(\delta(q)) = (v_1, \ldots, v_{|\delta(q)|})$, otherwise let $\varphi_f^{-1}(\delta(q)) = ()$.

Algorithm 1 1-FCP-$\mathcal{RFGSL}$

Let $K := \emptyset$, $F := \emptyset$;
for $n = 1, 2, 3, \ldots$ **do**
  Let $D = \{g_1, g_2, \ldots, g_n\}$;
  if $D \not\subseteq GL(S(F, K), \delta, p)$ **then**
  /* The above line can be determined by the parsing algorithm in [1]. */
    Let $F := Con(D)$;
  end if
  Let $K := Sub(D)$;
  output $(S(F, K), \delta, p)$;
end for

Definition 5 (1-FCP). *For a regular FGS* $(S, \delta) = ((\Sigma, \Lambda, X, \Pi, \Gamma), \delta)$, $p, q \in \Pi$ *and a ground term graph fragment* $[g, \sigma_g] \in \mathcal{F}_d(\Sigma, \Lambda)$, *we let*

$$C(S, \delta, p, q, g) = \{f \in \mathcal{G}_d(\Sigma, \Lambda) \mid [g, \sigma_g] \odot [f, \varphi_f^{-1}(\delta(q))] \in GL(S, \delta, p)\}.$$

We say that (S, δ) *and* p *have 1-FCP iff for every* $q \in \Pi$, *there is a ground term graph fragment* $[g, \sigma_g] \in \mathcal{F}_d(\Sigma, \Lambda)$ *s.t.* $C(S, \delta, p, q, g) = GL(S, \delta, q)$.

Definition 6 (1-FCP regular FGS language class). 1-*FCP-*$\mathcal{RFGSL}(w, d)$ *denotes the set of all regular FGS languages* $L \subseteq \mathcal{G}_d(\Sigma, \Lambda)$ *that satisfies the following conditions:*

1. *L is definable by $((\Sigma, \Lambda, X, \Pi, \Gamma), \delta, p)$ that has the* 1*-FCP,*
2. *Γ is written in Chomsky normal form, and*
3. *The treewidth of the term graphs in Γ is at most w. Therefore, the maximum length of ports of the hyperedges in Γ is also at most w.*

Let $L_* \subseteq \mathcal{G}(\Sigma, \Lambda)$ be a target regular FGS language. In Algorithm 1 (1-FCP-$\mathcal{RFGSL}$), we construct a regular FGS $S(F, K) = (\Sigma, \Lambda, X, \Pi, \Gamma)$, pointer δ, and initial predicate p in the following way:

- $\Sigma = \Sigma' \cup \{s_1, \ldots, s_w\}$, where $\Sigma' = \{a \mid \exists g_i \in D, \exists v \in V_{g_i}[\ \varphi_{g_i}(v) = a\ ]\}$ and $\Sigma' \cap \{s_1, \ldots, s_w\} = \emptyset$.
- $\Lambda = \{a \mid \exists g_i \in D, \exists e \in E_{g_i}[\ \psi_{g_i}(e) = a\ ]\}$.
- X: We use a new variable label only when it needed.
- $\Pi = \{[\![\alpha, \sigma_\alpha]\!] \mid [\alpha, \sigma_\alpha] \in F \subseteq Con(D)\}$. Let $[\![\emptyset, ()]\!]$ be the initial predicate p.
- δ, Γ: In Table 1, we describe the pointer $\delta(q)$ for each predicate q in Π and the graph rewriting rules in Γ. In the table, we use the following notations. Let k, ℓ be two positive integers ($k \leq \ell$) and $\mathcal{P}_{k,\ell}$ the set of all list of k distinct positive integers that are less than or equal to ℓ. Let $\sigma = (a_1, \ldots, a_k) \in \mathcal{P}_{k,\ell}$. For a list of elements $\nu = (v_1, \ldots, v_\ell)$ ($k \leq \ell$), $\chi_\sigma(\nu)$ denotes $(v_{a_1}, \ldots, v_{a_k})$ and $\bar{\chi}_\sigma(\nu)$ denotes the list obtained from ν by deleting $v_{a_1}, \ldots, v_{a_k}$.

Theorem 1. *Let w and d be constant integers greater than zero. The class* 1-*FCP-*$\mathcal{RFGSL}(w, d)$ *is identifiable in the limit with polynomial time update by using membership queries.*

Terminal rules $p_0(f_0) \leftarrow$ in $(S(F,K),\delta,p)$:

p_0	$\delta(p_0)$	f_0	Condition
$\llbracket g_0, \sigma_{g_0} \rrbracket$ $\lvert\sigma_g\rvert = 1$	(s_1)	$(\{v_1\}, \emptyset, \varphi, \emptyset, \emptyset, \emptyset, ())$ $\varphi(v_1) = s_1$	$[g_0, \sigma_{g_0}] \odot [f_0, (v_1)] \in L_*$
$\llbracket g_0, \sigma_{g_0} \rrbracket$ $\lvert\sigma_g\rvert = 1$	(s_1)	$(\{v_1, v_2\}, \{(v_1, v_2)\}, \varphi, \psi, \emptyset, \emptyset, ())$ $\varphi(v_1) = s_1, \varphi(v_2) \in \Sigma'$	$[g_0, \sigma_{g_0}] \odot [f_0, (v_1)] \in L_*$
$\llbracket g_0, \sigma_{g_0} \rrbracket$ $\lvert\sigma_g\rvert = 2$	(s_1, s_2)	$(\{v_1, v_2\}, \{(v_1, v_2)\}, \varphi, \psi, \emptyset, \emptyset, ())$ $\varphi(v_1) = s_1, \varphi(v_2) = s_2$	$[g_0, \sigma_{g_0}] \odot [f_0, (v_1, v_2)] \in L_*$

Binary rules $p_1(f_1) \leftarrow p_2(f_2),\ p_3(f_3)$ in $(S(F,K),\delta,p)$

$p_i\ (i = 2, 3)$	$\delta(p_i)\ (i = 2, 3)$	$f_i\ (i = 2, 3, j = 1, \ldots, \ell_i)$
$\llbracket g_i, \sigma_{g_i} \rrbracket$ $\lvert\sigma_{g_i}\rvert = \ell_i$	$(s_1, \ldots, s_{\ell_i})$	$f_i = (\{v_{i,1}, \ldots, v_{i,\ell_i}\}, \emptyset, \varphi_i, \emptyset, \{h_i\}, \lambda_i, port_i)$, where $\varphi_i(v_{i,j}) = s_j, \lambda_2(h_2) \neq \lambda_3(h_3), ports_i(h_i)[j] = v_{i,j}$.
p_1	$\delta(p_1)$	f_1
$\llbracket g_1, \sigma_{g_1} \rrbracket$ $\lvert\sigma_{g_1}\rvert = \ell_1$	$(s_1, \ldots, s_{\ell_1})$	$f_1 = [f_2, \chi_{\sigma_2}(ports_2(h_2))] \odot [f_3, \chi_{\sigma_3}(ports_3(h_3))]$, where $\sigma_i \in \mathcal{P}_{k,\lvert ports_i(h_i)\rvert}$ $(i = 2, 3)$ for some k. Let $\nu = ports_{f_2}(h_2) \cdot \bar{\chi}_{\sigma_3}(ports_{f_3}(h_3))$ and $\sigma_1 \in \mathcal{P}_{\ell_1,\lvert\nu\rvert}$. The vertices in ν are relabeled so that $\chi_{\sigma_1}(\nu) = (s_1, \ldots, s_{\ell_1})$ and $\bar{\chi}_\sigma(\nu) \in \Sigma'^{\lvert\nu\rvert - \ell_1}$.

Condition
For $\forall[\tau_2, \sigma_{\tau_2}], [\tau_3, \sigma_{\tau_3}] \in F$, if $[g_2, \sigma_{g_2}] \odot [\tau_2, \sigma_{\tau_2}] \in L_*$ and $[g_3, \sigma_{g_3}] \odot [\tau_3, \sigma_{\tau_3}] \in L_*$, then $[g_1, \sigma_{g_1}] \odot [[\tau_2, \chi_{\sigma_2}(\sigma_{\tau_2})] \odot [\tau_3, \chi_{\sigma_3}(\sigma_{\tau_3})], \xi] \in L_*$, where $\xi = \chi_{\sigma_1}(\chi_{\sigma_2}(\sigma_{\tau_2}) \cdot \bar{\chi}_{\sigma_3}(\sigma_{\tau_3}))$.

Table 1. $(S(F,K),\delta,p)$: There are three types of terminal rules and one type of binary rule. Each graph rewriting rule is created if the corresponding condition is satisfied. All conditions can be determined by asking to the membership oracle Mem_{L_*}.

References

1. D. Chiang, J. Andreas, D. Bauer, K. M. Hermann, B. Jones, and K. Knight. Parsing graphs with hyperedge replacement grammars. *Proc. 51st Meeting of the ACL. Marie-Catherine de Marneffe (ACL 2013)*, 2013.
2. A. Clark and R. Eyraud. Polynomial identification in the limit of substitutable context-free languages. JMLR 8, pp. 1725–1745, 2007.
3. A. Clark. A learnable representation for syntax using residuated lattice. *Proc. FG2009, LNAI 5591*, pp. 183–198, 2009.
4. A. Kasprzik and R. Yoshinaka. Distributional learning of simple context-free tree grammars. *Proc. ALT2011, LNAI 6925*, Springer, pp. 398–412, 2011.
5. R. Okada, S. Matsumoto, T. Uchida, Y. Suzuki, and T. Shoudai. Exact learning of finite unions of graph patterns from queries. *Proc. ALT 2007, LNAI 4754*, Springer, pp. 298–312, 2007.
6. T. Uchida, T. Shoudai, and S. Miyano. Parallel algorithms for refutation tree problem on formal graph systems. IEICE Trans. Inform. Syst. E78-D(2), pp. 99–112, 1995.
7. R. Yoshinaka. Integration of the dual approaches in the distributional learning of context-free grammars. *Proc. LATA2012, LNCS 7183*, Springer, pp. 538–550, 2012.