IEICE TRANS. FUNDAMENTALS, VOL.E106–A, NO.6 JUNE 2023

PAPER

Parameterized Formal Graph Systems and Their Polynomial-Time PAC Learnability

Takayoshi SHOUDAI[†a)], **Satoshi MATSUMOTO**[††], **Yusuke SUZUKI**[†††], **Tomoyuki UCHIDA**[†††], *and* **Tetsuhiro MIYAHARA**[†††], ***Members***

SUMMARY A formal graph system (FGS for short) is a logic program consisting of definite clauses whose arguments are graph patterns instead of first-order terms. The definite clauses are referred to as graph rewriting rules. An FGS is shown to be a useful unifying framework for learning graph languages. In this paper, we show the polynomial-time PAC learnability of a subclass of FGS languages defined by parameterized hereditary FGSs with bounded degree, from the viewpoint of computational learning theory. That is, we consider $\mathcal{VH}$-$\mathcal{FGSL}_{k,\Delta}(m, s, t, r, w, d)$ as the class of FGS languages consisting of graphs of treewidth at most k and of maximum degree at most Δ which is defined by variable-hereditary FGSs consisting of m graph rewriting rules having term graph patterns as arguments. The parameters s, t, and r denote the maximum numbers of variables, atoms in the body, and arguments of each predicate symbol of each graph rewriting rule in an FGS, respectively. The parameters w and d denote the maximum number of vertices of each hyperedge and the maximum degree of each vertex of term graph patterns in each graph rewriting rule in an FGS, respectively. $\mathcal{VH}$-$\mathcal{FGSL}_{k,\Delta}(m, s, t, r, w, d)$ has infinitely many languages even if all the parameters are bounded by constants. Then we prove that the class $\mathcal{VH}$-$\mathcal{FGSL}_{k,\Delta}(m, s, t, r, w, d)$ is polynomial-time PAC learnable if all m, s, t, r, w, d, Δ are constants except for k.

key words: *formal graph system, graph pattern, PAC learning, polynomial-time learnability*

1. Introduction

Frequent subgraph mining is one of the most intensively studied problems in graph mining. This is to extract local similarities that frequently occur in graph data, and there are many research results including the work of Yan et al. [23]. However, some computational problems involving graph data have a troublesome problem in that the number of solutions to be searched for a general graph can explode into a huge number due to a combinatorial explosion. Frequent subgraph mining that deals with general graphs is one such example. As an approach to avoid such combinatorial explosion, Horváth et al. [10] focused on the graph class of outerplanar graphs, a well-known class in graph theory, and proposed an efficient mining algorithm of frequent connected subgraphs. Yamasaki et al. [21], [22] defined a term graph pattern based on the class of outerplanar graphs and proposed an efficient machine learning algorithm from the viewpoint of computational learning theory. A term graph pattern is a graph pattern that can represent the global structure, such as how multiple frequent subgraphs are positioned.

This paper discusses the polynomial-time learnability of graph pattern classes designed by formal graph system (FGS for short). FGS is a kind of logic program [11] for graphs introduced by Uchida et al. [19]. That is, FGS is a logic program consisting of definite clauses whose arguments are graph patterns instead of first-order terms. Languages generated by FGS are called FGS languages. The class of trees (acyclic connected graphs), the class of ordered trees, the class of outerplanar graphs, and the class of graphs of bounded treewidth (partial k-trees for fixed k) are all FGS languages. This suggests that FGS can be used to unify conventional graph pattern design methods. In addition, by using the logic programmatic features of FGS, it is expected to reveal a polynomial-time learnable FGS language class.

Computational learning theory is a field of the mathematical study for investigating how much computational time, memory, and other computational resources are inherently required to learn a certain concept. Valiant [20] proposed PAC learning (Probably Approximately Correctly Learning) as a theoretical framework for dealing with this investigation. PAC learning model allows errors in the hypothesis produced by a learning algorithm and considers it a good hypothesis even if it does not perfectly match the target concept, as long as an occurrence where it does not match the target concept is rare.

One of the classes of polynomial-time PAC learnable formal languages is a subclass of the elementary formal system (EFS) languages [2]. The elementary formal system is a kind of logic program that can handle strings directly, whereas FGS handles graphs directly. Miyano et al. [12] showed a polynomial-time PAC learnable class of parameterized hereditary elementary formal systems. The elementary formal system is very well studied in the field of computational learning theory, with results on learnability under inductive inference models [2] and query learning models [16], in addition to results under PAC learning models. We easily see that any string can be viewed as an edge-labeled chain graph with distinguished start and end vertices by using an edge label instead of a symbol on the string. Thus, research results for EFS also hold for FGS by using chain graphs. On

Manuscript received May 10, 2022.
Manuscript revised October 5, 2022.
Manuscript publicized December 14, 2022.

[†]The author is with Department of Computer Science and Engineering, Fukuoka Institute of Technology, Fukuoka-shi, 811-0295 Japan.
[††]The author is with Department of Mathematical Sciences, Tokai University, Hiratsuka-shi, 259-1292 Japan.
[†††]The authors are with Graduate School of Information Sciences, Hiroshima City University, Hiroshima-shi, 731-3194 Japan.
a) E-mail: shodai@fit.ac.jp

DOI: 10.1587/transfun.2022EAP1052

the other hand, if we have no edge label, FGS needs to focus only on connections between vertices, i.e., graph structures. In this paper, we mainly discuss the learnability of FGS languages when the size of the alphabet representing the edge labels is 1. When we have only one edge label, since we can think of it as having no edge label, we omit the edge label in any figure.

This paper deals with a subclass of FGS languages that can be generated by variable-hereditary formal graph systems. $\mathcal{VH}$-$\mathcal{FGSL}(m,s,t,r,w,d)$ is the subclass of FGS languages consisting of graphs which is defined by variable-hereditary FGSs consisting of graph rewriting rules with the parameters m,s,t,r,w,d of the representation of each FGS (Definition 22). We consider $\mathcal{VH}$-$\mathcal{FGSL}_{k,\Delta}(m,s,t,r,w,d)$ as the subclass of $\mathcal{VH}$-$\mathcal{FGSL}(m,s,t,r,w,d)$ in which each FGS language consists of graphs of treewidth at most k and of maximum degree at most Δ (Definition 23). In this paper, we prove that $\mathcal{VH}$-$\mathcal{FGSL}_{k,\Delta}(m,s,t,r,w,d)$ is polynomial-time PAC learnable if all m,s,t,r,w,d,Δ are constants except for k. Here we should note that there are infinitely many languages definable by variable-hereditary FGSs even if all the parameters are bounded by constants.

2. Preliminaries

2.1 Term Graph Pattern

In this paper, we deal with *graphs* and *multigraphs* (please refer to [6] for the terminology). In this and the following sections, we give the technical definitions for *formal graph systems* for *graphs* [19]. Because we can easily generalize the following definitions to the case of *multigraphs*, we also give an example of *formal graph systems* for *multigraphs*.

Let S be a set or a list. We denote by $|S|$ the number of elements included in S. Let Σ and Λ be finite alphabets. Let X be an alphabet consisting of infinitely many symbols. A symbol in X is called a *variable*. Each symbol x in X has a positive integer called a *rank*. The rank of a variable $x \in X$ is expressed as $rank(x)$ ($rank(x) \geq 1$). Also, for the alphabets Σ, Λ, and X, we assume that $\Sigma \cap X = \emptyset$, and $\Lambda \cap X = \emptyset$.

Definition 1: For a triple of alphabets $\langle \Sigma, \Lambda, X \rangle$, a 7-tuple $g = (V, E, \varphi, \psi, H, \lambda, ports)$ is called a *term graph pattern*, or a *term graph*, if it satisfies the following 4 conditions:

1. (V, E) is a graph, where V is a vertex set and E is an edge set. φ is a vertex labeling function $\varphi : V \to \Sigma$, and ψ is an edge labeling function $\psi : E \to \Lambda$.
2. H is a set of non-empty subsets of V, i.e., $H \subseteq 2^V \setminus \{\emptyset\}$ (possibly $H = \emptyset$). An element in H is called a *hyperedge*.
3. λ is a hyperedge labeling function $\lambda : H \to X$, which satisfies $rank(\lambda(h)) = |h|$. For a hyperedge $h \in H$, $rank(\lambda(h))$ is called the *rank* of the hyperedge h. When $H = \emptyset$, λ is the empty function. In this paper, the empty function is denoted by $\emptyset$.
4. $ports$ is a function $ports : H \to V^*$ such that for each hyperedge $h = \{v_1, \ldots, v_\ell\} \in H$ ($\ell \geq 1$), $ports(h)$ is an ordered list $(v_{j_1}, \ldots, v_{j_\ell})$ where $\{j_1, \ldots, j_\ell\} = \{1, \ldots, \ell\}$. A vertex in $ports(h)$ is called a *port* of the hyperedge h. When $H = \emptyset$, $ports$ is the empty function $\emptyset$.

A term graph which has no hyperedge, i.e., $H = \emptyset$, is called a *ground term graph*. Ground term graph can be regarded as an ordinary labeled graph. We denote the set of all term graphs on $\langle \Sigma, \Lambda, X \rangle$ by $\mathcal{G}(\Sigma, \Lambda, X)$. In addition, we denote the set of all ground term graphs on Σ and Λ by $\mathcal{G}(\Sigma, \Lambda)$.

For a term graph $g \in \mathcal{G}(\Sigma, \Lambda, X)$, we use the following notations to express the members of g:

V_g	: the vertex set of g,
E_g	: the edge set of g,
φ_g	: the vertex labeling function $V_g \to \Sigma$,
ψ_g	: the edge labeling function $E_g \to \Lambda$,
H_g	: the hyperedge set of g,
λ_g	: the hyperedge labeling function $H_g \to X$,
$ports_g$	: the $ports$ function $H_g \to V_g^*$.

Let g be a term graph. The degree of a vertex $v \in V_g$ is defined as the sum of the number of edges in E_g that connect to v and the number of hyperedges in H_g that contain v. For a term graph g, the ranges of the labeling functions $\varphi_g, \psi_g, \lambda_g$ are described as $\varphi_g(V_g), \psi_g(E_g), \lambda_g(H_g)$ respectively.

Definition 2: Let f and g be term graphs. f and g are said to be *term graph isomorphic*, or *isomorphic*, if the following 3 conditions are satisfied:

1. (V_f, E_f) and (V_g, E_g) are isomorphic as vertex-labeled and edge-labeled graphs. Let $\pi : V_f \to V_g$ be an isomorphism from (V_f, E_f) to (V_g, E_g).
2. There exists a bijection $\omega : H_f \to H_g$ such that $ports_g(\omega(h)) = \pi^*(ports_f(h))$ for any hyperedge $h \in H_f$, where $\pi^*((v_1, \ldots, v_\ell))$ ($\ell \geq 1$) is defined as $(\pi(v_1), \ldots, \pi(v_\ell))$.
3. For $h, h' \in H_f$, $\lambda_f(h) = \lambda_f(h')$ if and only if $\lambda_g(\omega(h)) = \lambda_g(\omega(h'))$.

For term graphs f and g, we describe $f \cong g$ when f and g are term graph isomorphic.

Let g be a term graph. Let V'_g, E'_g, H'_g be subsets of V_g, E_g, H_g respectively. Let $\varphi_g|V'_g$, $\psi_g|E'_g$, and $\lambda_g|H'_g$ be the labeling functions $\varphi_g, \psi_g, \lambda_g$ whose domains are restricted to V'_g, E'_g, and H'_g, respectively. Similarly, let $ports_g|H'_g$ be a function $ports_g$ whose domain is restricted to H'_g.

Definition 3: Let g and g' be term graphs. g' is a *term subgraph* of g if g and g' satisfy that (i) $V_{g'} \subseteq V_g$, $E_{g'} \subseteq E_g$, $H_{g'} \subseteq H_g$, and (ii) $\varphi_{g'} = \varphi_g|V_{g'}$, $\psi_{g'} = \psi_g|E_{g'}$, $\lambda_{g'} = \lambda_g|H_{g'}$, $ports_{g'} = ports_g|H_{g'}$.

Definition 4: Let f and g be term graphs. f is *term subgraph isomorphic* to g if there exists a term subgraph g' of g such that f and g' are term graph isomorphic.

Definition 5: Let g be a term graph. g is called a *star term graph* of a variable x if $|V_g| = rank(x)$, $E_g = \emptyset$, and

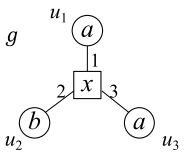


Fig. 1 Star term graph: $V_g = \{u_1, u_2, u_3\}, E_g = \emptyset, H_g = \{h\}$ where $h = V_g$, $\lambda_g(h) = x$ with $rank(x) = 3$, and $ports_g(h) = (u_1, u_2, u_3)$. The hyperedge h is represented by its variable in the square and its ports are represented by the vertices connected by the lines, on which the numbers describe the port ordering.

$H_g = \{h\}$ where $h = V_g$ and $\lambda_g(h) = x$. That is, a star term graph of a variable x is a term graph consisting of only one hyperedge which have all vertices in g and is labeled with a variable x.

Figure 1 shows an example of star term graphs of a variable. In this paper, the hyperedges of a term graph are represented by squares surrounding the variables of the hyperedges.

Definition 6: A term graph g is *connected* if for any 2 vertices $u, v \in V_g$, there exists a vertex sequence $u = u_0, u_1, \ldots, u_\ell = v$ $(u_0, u_1, \ldots, u_\ell \in V_g, \ell \geq 0)$ such that any pair of vertices u_{i-1}, u_i $(1 \leq i \leq \ell)$ is either connected by an edge in E_g or contained in a hyperedge in H_g.

2.2 Formal Graph System

Let $f_1, \ldots, f_r, g_1, \ldots, g_r$ $(r \geq 1)$ be term graphs in $\mathcal{G}(\Sigma, \Lambda, X)$. For a predicate symbol p of r arguments, $p(f_1, \ldots, f_r)$ is called an *atom*. Let $A, B_1, \ldots, B_t$ be atoms $(t \geq 0)$. A *graph rewriting rule*, or a *rule*, is a definite clause of the form $A \leftarrow B_1, \ldots, B_t$. For a graph rewriting rule $A \leftarrow B_1, \ldots, B_t$, the atom A is called the *head* of the rule, and the atoms $B_1, \ldots, B_t$ are called the *body* of the rule. For two atoms $p(f_1, \ldots, f_r)$ and $p(g_1, \ldots, g_r)$, if $f_i \cong g_i$ for all $i = 1, \ldots, r$, then we write $p(f_1, \ldots, f_r) \cong p(g_1, \ldots, g_r)$. For two graph rewriting rules $A \leftarrow B_1, \ldots, B_t$ and $A' \leftarrow B'_1, \ldots, B'_t$, if $A \cong A'$ and $B_i \cong B'_i$ for all $i = 1, \ldots, t$ then we write $(A \leftarrow B_1, \ldots, B_t) \cong (A' \leftarrow B'_1, \ldots, B'_t)$.

Definition 7: A *formal graph system* (*FGS*) is a finite set of graph rewriting rules.

Definition 8: For a term graph f and an ordered list σ of vertices of f, the form $[f, \sigma]$ is called a *term graph pattern fragment*, or *term graph fragment* shortly.

Definition 9: Let f and g be term graphs which are vertex-disjoint, i.e., $V_f \cap V_g = \emptyset$. Let $h = \{u_1, \ldots, u_\ell\}$ $(\ell \geq 1)$ be a hyperedge in H_g. Without loss of generality, we assume that $ports_g(h) = (u_1, \ldots, u_\ell)$. Let $[f, \sigma]$ be a term graph fragment where $\sigma = (v_1, \ldots, v_\ell)$ is an ordered list of ℓ vertices in V_f $(1 \leq \ell \leq |V_f|)$. The *hyperedge replacement* $h \leftarrow [f, \sigma]$ on g is the following procedure: Let f' be a vertex-disjoint copy of f $(V_f \cap V_{f'} = \emptyset)$ such that f and f' are term graph isomorphic with an isomorphism $\pi : V_f \rightarrow V_{f'}$. First we remove the hyperedge h from H_g, and then identify the ports $u_1, \ldots, u_\ell$ of h with the vertices $\pi(v_1), \ldots, \pi(v_\ell)$ of f' in this order. In this way, we attach f' to g. The new vertex label of u_i is $\varphi_g(u_i)$.

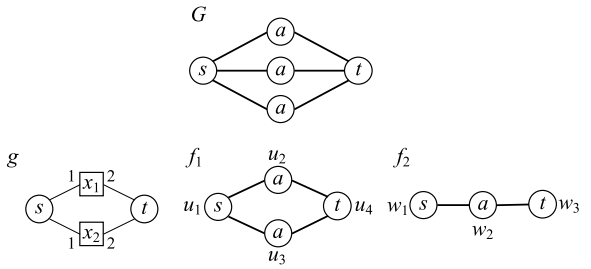


Fig. 2 Substitution: A graph G is obtained by applying the substitution $\theta = \{x_1 := [f_1, (u_1, u_4)], x_2 := [f_2, (w_1, w_3)]\}$ to the term graph g. That is, $G \cong g\theta$.

Let Υ be a non-empty finite set of hyperedge replacements on f. We denote by $f(\Upsilon)$ the term graph obtained by applying all the hyperedge replacements in Υ simultaneously.

Definition 10: Let x be a variable in X and $[f, \sigma]$ a term graph fragment, where $rank(x) = |\sigma|$. The form $x := [f, \sigma]$ is called the *binding* of x by f and σ.

Definition 11: A non-empty finite set of bindings $\theta = \{x_1 := [f_1, \sigma_1], \ldots, x_\ell := [f_\ell, \sigma_\ell]\}$ is called a *substitution* if $x_1, \ldots, x_\ell$ are mutually distinct variables in X and each f_i has no hyperedge labeled by a variable in $\{x_1, \ldots, x_\ell\}$. Let g be a term graph. For x_i $(1 \leq i \leq \ell)$, let $H_g(x_i) = \{h \in H_g \mid \lambda_g(h) = x_i\}$ and $\Upsilon_i = \{h \leftarrow [f_i, \sigma_i] \mid h \in H_g(x_i)\}$. Then let $\Upsilon_\theta = \bigcup_{i=1}^{\ell} \Upsilon_i$. The term graph $g\theta$ is defined as $g(\Upsilon_\theta)$.

For a substitution θ and an atom $p(g_1, \ldots, g_r)$, an atom $p(g_1, \ldots, g_r)\theta$ is defined as $p(g_1\theta, \ldots, g_r\theta)$. For a graph rewriting rule $A \leftarrow B_1, \ldots, B_t$, $(A \leftarrow B_1, \ldots, B_t)\theta$ is defined as $A\theta \leftarrow B_1\theta, \ldots, B_t\theta$. Figure 2 gives an example of term graphs and substitutions. Figure 3 shows an example of FGSs.

Let f and g be term graphs which are vertex-disjoint. When a substitution θ satisfies that $f\theta \cong g\theta$, the substitution θ is called a *unifier* of f and g. If there exist substitutions θ and θ' such that both $f \cong g\theta$ and $f\theta' \cong g$ hold, f (resp. g) is called a *variant* of g (resp. f). For atoms and graph rewriting rules, a unifier and a variant are similarly defined. A *goal* is a graph rewriting rule of the form $\leftarrow B_1, \ldots, B_t$ $(t \geq 0)$. If $t = 1$, the goal "$\leftarrow B_1$" is called a *single goal*. Also, if $t = 0$, it is called the *empty goal*.

For a graph rewriting rule C, let $var(C)$ be the set of all variables of term graphs appearing in C.

Definition 12: Let Γ be an FGS and D a goal. A *derivation* from D on Γ is a sequence of triples (D_i, θ_i, C_i) $(i = 0, 1, \ldots)$ satisfying the following 4 conditions:

1. D_i is a goal, θ_i is a substitution, and C_i is a variant of a graph rewriting rule in Γ.
2. D_0 is D.
3. $var(C_i) \cap var(C_j) = \emptyset$ for any j $(i \neq j)$.
4. We assume that there exists a procedure Q that selects one atom from the body of the goal. Let D_i be the

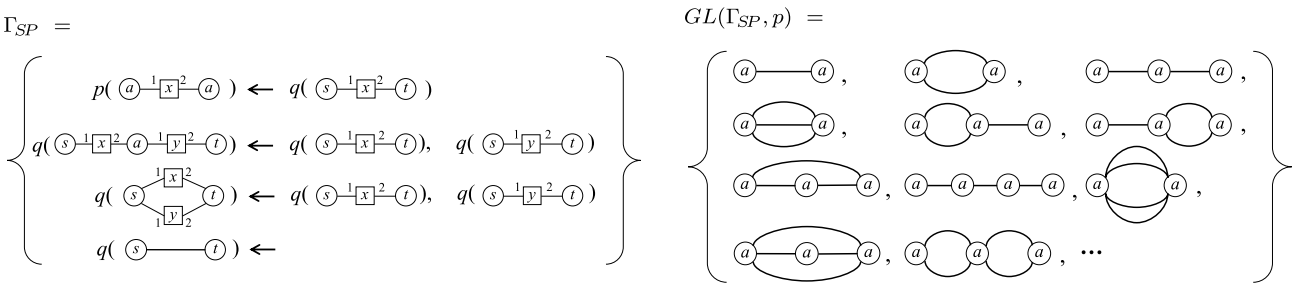


Fig. 3 Formal graph system and its FGS language: Γ_{SP} contains two unary predicate symbols p and q. The FGS language $GL(\Gamma_{SP}, q)$ is the set of all two-terminal series-parallel (TTSP) graphs. $GL(\Gamma_{SP}, p)$ consists of all series-parallel graphs.

goal $\leftarrow A_1, \ldots, A_\ell$ ($\ell \geq 1$) and C_i the graph rewriting rule $A \leftarrow B_1, \ldots, B_t$ ($t \geq 0$). Let $A_{\ell'}$ ($1 \leq \ell' \leq \ell$) be an atom of the body of D_i chosen by Q. In this case, θ_i is a unifier of A and $A_{\ell'}$, and D_{i+1} is the goal $\leftarrow A_1\theta_i, \ldots, A_{\ell'-1}\theta_i, B_1\theta_i, \ldots, B_t\theta_i, A_{\ell'+1}\theta_i, \ldots, A_\ell\theta_i$.

A *refutation* is a finite derivation that ends with the empty goal.

Definition 13: Let Γ be an FGS and D a single goal. Let $F = \{(D_i, \theta_i, C_i)\}_{0 \leq i \leq \gamma}$ be a refutation from D on Γ. The *refutation tree* of F is the rooted tree $T_{\gamma+1}$ defined as follows:

1. Each vertex is labeled with a single or the empty goal.
2. The root label is a single goal $D_0 = D$.
3. Each leaf is labeled with the empty goal.
4. T_0 is the rooted tree consisting of one vertex labeled with D. For any $i = 0, \ldots, \gamma$, T_{i+1} is the rooted tree obtained by applying (D_i, θ_i, C_i) to T_i as follows: Let D_i be the goal $\leftarrow A^i_1, \ldots, A^i_{\ell_i}$ ($\ell_i \geq 1$). We assume that T_i has a leaf labeled by $\leftarrow A^i_j$ ($1 \leq j \leq \ell_i$). Let C_i be a graph rewriting rule $A^i \leftarrow B^i_1, \ldots, B^i_{t_i}$ ($t_i \geq 0$) and θ_i a unifier of A^i and A^i_j. In this case, T_{i+1} is a rooted tree in which the vertex labeled by $\leftarrow A^i_j$ has new t_i children which are labeled by $\leftarrow B^i_j\theta_i$ ($1 \leq j \leq t_i$) respectively. Hence, the t_i children are leaves.

Figure 4 shows an example of refutation trees, which represents a refutation from a single goal.

Definition 14: Let Γ be an FGS. The relation $\Gamma \vdash C$ for a graph rewriting rule C is defined as follows:

1. If $C \in \Gamma$, then $\Gamma \vdash C$ holds.
2. If $\Gamma \vdash C$, then for any substitution θ, $\Gamma \vdash C\theta$ holds.
3. If $\Gamma \vdash A \leftarrow B_1, \ldots, B_{\ell'}, \ldots, B_\ell$ ($1 \leq \ell' \leq \ell$) and $\Gamma \vdash B_{\ell'} \leftarrow C_1, \ldots, C_t$ ($t \geq 0$), then $\Gamma \vdash A \leftarrow B_1, \ldots, B_{\ell'-1}, C_1, \ldots, C_t, B_{\ell'+1}, \ldots, B_\ell$ holds.

A graph rewriting rule C is said to be *provable* from Γ if $\Gamma \vdash C$ holds.

Hereafter, a predicate symbol with one argument is called a *unary predicate symbol*. For an FGS Γ and a unary predicate symbol p, the *FGS language* is defined as follows:

$$GL(\Gamma, p) = \{G \in \mathcal{G}(\Sigma, \Lambda) \mid \Gamma \vdash p(G) \leftarrow\}.$$

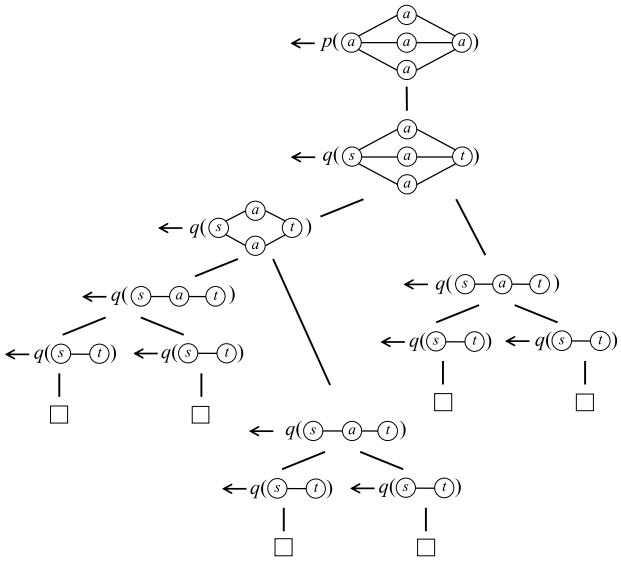


Fig. 4 Refutation tree: This is obtained by Γ_{SP} in Fig. 3 from the single goal of the root. The squares "□" in this figure represent the empty goal.

Figure 3 shows $GL(\Gamma_{SP}, p)$ as an example of the FGS languages. Moreover, the class of all FGS languages is defined as follows:

$$\mathcal{FGSL} = \{GL(\Gamma, p) \mid \Gamma \text{ is an FGS and } p \text{ is a unary predicate symbol}\}.$$

2.3 Boundary Representation of a Subgraph

Let G be a connected ground term graph in $\mathcal{G}(\Sigma, \Lambda)$. Let α be a connected term subgraph of G, i.e., we can write $\alpha = (V_\alpha, E_\alpha, \varphi_G|V_\alpha, \psi_G|E_\alpha, \emptyset, \emptyset, \emptyset)$, where the symbol $\emptyset$ represents either the empty set or a function whose domain is empty. We denote by $G \setminus \alpha$ the term subgraph of G which is induced by $E_G \setminus E_\alpha$, i.e., $G \setminus \alpha = (V_{G\setminus\alpha}, E_G \setminus E_\alpha, \varphi_G|V_{G\setminus\alpha}, \psi_G|(E_G \setminus E_\alpha), \emptyset, \emptyset, \emptyset)$, where $V_{G\setminus\alpha} = \bigcup_{\{u,v\} \in E_G \setminus E_\alpha} \{u, v\}$.

Definition 15: Let G be a connected ground term graph in $\mathcal{G}(\Sigma, \Lambda)$ and α a connected term subgraph of G. A *boundary vertex* of α is a vertex of α that is one of the endpoints of an edge of $G \setminus \alpha$. *A boundary edge* of α is an edge of α at least one of whose endpoints is a boundary vertex. A *boundary representation* of α is a 4-tuple $(bv(\alpha), be(\alpha), v, isin(v, \alpha))$.

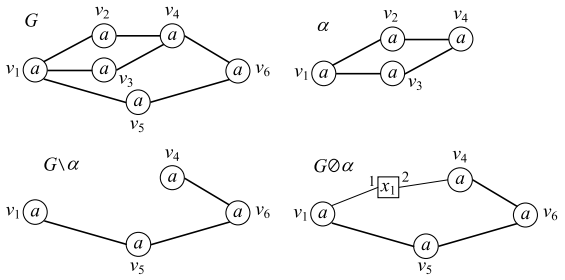


Fig. 5 The term graphs $G \setminus \alpha$ and $G \oslash \alpha$ for a term graph G and a connected term subgraph α of G: One of the boundary representations of α is $(bv(\alpha), be(\alpha), v_6, false)$, where $bv(\alpha) = \{v_1, v_4\}$ and $be(\alpha) = \{(v_1, v_2), (v_1, v_3), (v_2, v_4), (v_3, v_4)\}$.

Each component is as follows:

1. $bv(\alpha)$ is the set of all boundary vertices of α.
2. $be(\alpha)$ is the set of all boundary edges of α.
3. Let v be an arbitrary vertex of G. This vertex is called the *marker* of α.
4. $isin(v, \alpha)$ is a predicate that indicates whether the marker v is a vertex of α or not. That is, if $v \in V_\alpha$, then $isin(v, \alpha) = true$, otherwise $isin(v, \alpha) = false$.

Let G be a connected ground term graph and α a connected term subgraph of G. Let $bv(\alpha) = \{v_1, \ldots, v_\ell\}$ $(\ell \geq 1)$. For a variable $x \in X$ with $rank(x) = |bv(\alpha)|$ and an ordered list $\sigma = (v_1, \ldots, v_\ell)$ of vertices in $bv(\alpha)$, we denote by $G \oslash \alpha$ a term graph $(V_{G\setminus\alpha}, E_{G\setminus\alpha}, \varphi_{G\setminus\alpha}, \psi_{G\setminus\alpha}, \{bv(\alpha)\}, \lambda, ports)$ where $\lambda(bv(\alpha)) = x$, and $ports(bv(\alpha)) = \sigma$. Figure 5 shows an example of G, α, $G \setminus \alpha$, and $G \oslash \alpha$. A boundary representation of a subgraph α is a sufficient representation to show α concisely. Let α and α' be connected subgraphs of G. Chiang et al. [5] show that if α and α' have the same boundary representation, then $\alpha \cong \alpha'$ holds.

2.4 Concept Learning

This paper deals with the FGS language class $\mathcal{FGSL}$ as a concept class. The set of all pairs (Γ, p) of an FGS Γ and a unary predicate symbol p is used to represent the concept class $\mathcal{FGSL}$. We use standard terminology in learning theory. Please see [13], [17] for detailed descriptions of the PAC learning model and formalizations of *concept classes*, *representation classes*, *examples*, and so on.

Let Ω be a finite alphabet. The set of all strings over alphabet Ω is denoted by Ω^*. For a string $w \in \Omega^*$, we denote by $|w|$ the length of string w. For an integer $n \geq 0$, let $\Omega^n = \{w \in \Omega^* \mid |w| = n\}$ and $\Omega^{[n]} = \{w \in \Omega^* \mid |w| \leq n\}$. A subset c of Ω^* is called a *concept*. A concept c is regarded as a binary function on Ω^*. That is, for any $w \in \Omega^*$, the characteristic function $\chi_c : \Omega^* \to \{0, 1\}$ returns 1 if $w \in c$, otherwise 0. A *concept class* is a set $C \subseteq 2^{\Omega^*}$ of non-empty concepts.

A pair $\langle w, \rho \rangle$ of $w \in \Omega^*$ and $\rho \in \{0, 1\}$ is called an *example*. For a set of examples $S \subseteq \Omega^* \times \{0, 1\}$, let $S_+ = \{w \mid \langle w, 1 \rangle \in S\}$ and $S_- = \{w \mid \langle w, 0 \rangle \in S\}$. A concept $c \in C$ is said to be *consistent* with a set S of examples if $\chi_c(w) = 1$ for all $w \in S_+$ and $\chi_c(w) = 0$ for all $w \in S_-$.

Let Θ be a finite alphabet. A *hypothesis representation*, or a *representation* simply, of a concept class C is a function $R : C \to 2^{\Theta^*}$ such that for any concept c, $R(c)$ is a non-empty subset of Θ^*, such that for any two different concepts c_1 and c_2, $R(c_1) \cap R(c_2) = \emptyset$ holds. Let c be a concept in C. $R(c)$ denotes the set of *names* for c. Let $\ell_{\min}(c, R)$ be the minimum length of the names for the concept c.

Definition 16: A representation R of a concept class C is *polynomial-time computable* if there exists a deterministic algorithm $\mathcal{B}$ and a polynomial $poly(\cdot)$ such that (a) and (b) below are satisfied:

(a) $\mathcal{B}$ receives $w \in \Omega^*$ and $\nu \in \Theta^*$ as inputs.
(b) If $\nu \in R(c)$ for some $c \in C$, $\mathcal{B}$ outputs $\chi_c(w)$ in $poly(|w| + |\nu|)$ time and terminates.

For any set S of examples, let $\ell_{\min}(S, R)$ be the minimum length of the names for the concepts that are consistent with S.

Definition 17: Let C be a concept class, R the representation of C, and $S \subseteq \Omega^* \times \{0, 1\}$ a finite set of examples. A deterministic algorithm $\mathcal{A}$ with S as input is *fitting* if it outputs a representation $\nu \in R(c)$ whenever there exists a concept $c \in C$ that is consistent with S. $\mathcal{A}$ is *polynomial-time fitting* if $\mathcal{A}$ runs in a polynomial time with respect to $\ell_{\min}(S, R)$ and $\sum_{\langle w,\rho\rangle \in S} |w|$.

Definition 18: Let C be a concept class and S a subset of Ω^*. C is said to *shatter* S if $\{c \cap S \mid c \in C\} = 2^S$ holds. The *Vapnik-Chervonenkis dimension*, or *VC dimension* simply, of C is the maximum cardinality of the set S shattered by C. C has a *polynomial dimension* if there exists some polynomial $poly(\cdot)$ such that the VC dimension of $C^{[n]} = \{c \cap \Omega^{[n]} \mid c \in C\}$ is less than $poly(n)$. When the VC dimension of C is not a polynomial dimension, the VC dimension of C is called to be an *exponential dimension*.

Lemma 1 ([3], [9], [14], [15]): Let C be a concept class and R a polynomial-time computable representation of C. A concept class C is polynomial-time PAC learnable with R if C has a polynomial dimension and a polynomial-time fitting algorithm with R.

Lemma 2 ([14]): A concept class C has a polynomial dimension if and only if there exists a polynomial $poly(\cdot)$ such that $\log_2 \left|C^{[n]}\right| \leq poly(n)$ holds $(n \geq 0)$.

3. Polynomial-Time Learnable Hereditary Formal Graph System

3.1 Variable-Hereditary FGS of Bounded Degree

Definition 19: A graph rewriting rule

$$q_0(g_1^0, \ldots, g_{r_0}^0) \leftarrow q_1(g_1^1, \ldots, g_{r_1}^1), \ldots, q_t(g_1^t, \ldots, g_{r_t}^t)$$

is *hereditary* if each term graph g_j^i $(1 \leq i \leq t, 1 \leq j \leq r_i)$

in the body is a connected term subgraph of some g_j^0 ($1 \leq j \leq r_0$) in the head when the vertex labels of ports of the variables are ignored. An FGS Γ is a *hereditary FGS* if all graph rewriting rules in Γ are hereditary.

Definition 20: A hereditary graph rewriting rule

$$q_0(g_1^0, \ldots, g_{r_0}^0) \leftarrow q_1(g_1^1, \ldots, g_{r_1}^1), \ldots, q_t(g_1^t, \ldots, g_{r_t}^t)$$

is *variable-hereditary* if each term graph g_j^i ($1 \leq i \leq t, 1 \leq j \leq r_i$) in the body is a star term graph of a variable appearing in some term graph g_j^0 ($1 \leq j \leq r_0$) in the head. An FGS Γ is a *variable-hereditary FGS* if all graph rewriting rules in Γ are variable-hereditary.

Definition 21: For nonnegative integers m, s, t, r, w, and d, let $\mathcal{H}\text{-}\mathcal{FGSL}(m, s, t, r, w, d)$ be the class of all FGS languages $GL(\Gamma, p)$ such that p is a unary predicate symbol and Γ is a hereditary FGS that satisfies the following 6 conditions:

(1) The FGS Γ consists of at most m graph rewriting rules ($m \geq 1$).
(2) The number of occurrences of variables in the head of a graph rewriting rule in Γ is at most s ($s \geq 0$).
(3) The number of atoms in the body of a graph rewriting rule in Γ is at most t ($t \geq 0$).
(4) The number of arguments for each predicate symbol appearing in Γ is at most r ($r \geq 1$).
(5) The rank of each variable of term graphs appearing in Γ is at most w ($w \geq 1$).
(6) The degree of each vertex of term graphs appearing in Γ is at most d ($d \geq 0$).

The parameters m, s, t, r, w, d are related to the representation of an FGS when it is written.

Definition 22: Let $\mathcal{VH}\text{-}\mathcal{FGSL}(m, s, t, r, w, d)$ be the class of all FGS languages $GL(\Gamma, p)$ such that p is a unary predicate symbol and Γ is a variable-hereditary FGS that satisfies (1)–(6) of Definition 21.

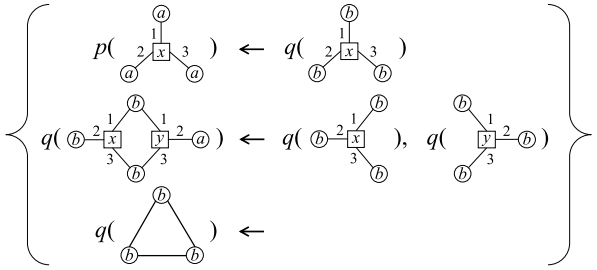
Let k be a positive integer. A *k-tree* is an undirected graph which is defined inductively as follows: The complete graph on $k+1$ vertices is a k-tree. A k-tree G with $n+1$ vertices ($n \geq k+1$) can be constructed from a k-tree H with n vertices by adding a new vertex and k new edges connecting the new vertex to the vertices of a clique of size k of H. A *partial k-tree* is a subgraph of a k-tree. The *treewidth* of an undirected graph G is the minimum k such that G is a partial k-tree.

Definition 23: Let $\mathcal{VH}\text{-}\mathcal{FGSL}_{k,\Delta}(m, s, t, r, w, d)$ be the subclass of $\mathcal{VH}\text{-}\mathcal{FGSL}(m, s, t, r, w, d)$ in which all FGS languages $GL(\Gamma, p)$ satisfy the following 2 conditions:

(7) The treewidth of each graph in the FGS language $GL(\Gamma, p)$ is at most k.
(8) The maximum degree of each graph in the FGS language $GL(\Gamma, p)$ is at most Δ.

We note that unlike the parameters m, s, t, r, w, d, the parameters k, Δ are related to the FGS language obtained from an FGS.

$\Gamma_{2\text{-}tree} =$



$\Gamma_{partial\_2\text{-}tree} =$

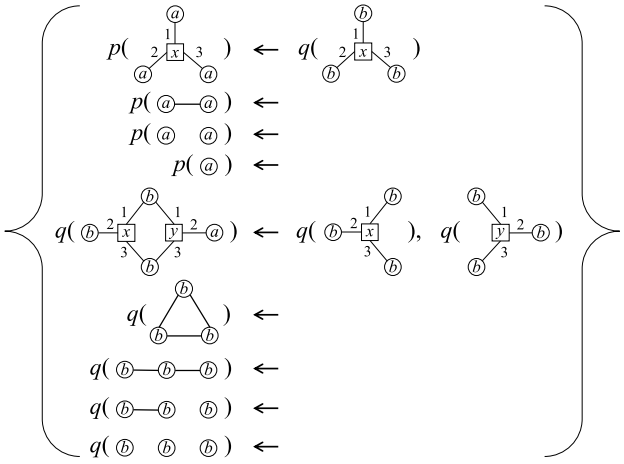


Fig. 6 The FGSs $\Gamma_{2\text{-}tree}$ and $\Gamma_{partial\_2\text{-}tree}$: These FGS languages are the sets of all 2-trees and all partial 2-trees, respectively. We see that $GL(\Gamma_{2\text{-}tree}, p) \in \mathcal{VH}\text{-}\mathcal{FGSL}(3, 2, 2, 1, 3, 2)$ and $GL(\Gamma_{partial\_2\text{-}tree}, p) \in \mathcal{VH}\text{-}\mathcal{FGSL}(9, 2, 2, 1, 3, 2)$.

We give two FGSs $\Gamma_{2\text{-}tree}$ and $\Gamma_{partial\_2\text{-}tree}$ in Fig. 6, where $GL(\Gamma_{2\text{-}tree}, p)$ and $GL(\Gamma_{partial\_2\text{-}tree}, p)$ are the set of all 2-trees and the set of all partial 2-trees (including disconnected graphs), respectively. The 2nd graph rewriting rule in $\Gamma_{2\text{-}tree}$ represents the operation of creating one new smooth tree decomposition [4] by identifying two vertices in the specified nodes of two different smooth tree decompositions. We omit the details of the proof. We can generalize this idea to construct variable-hereditary FGSs for k-trees and partial k-trees.

Let $\mathcal{G}(\Sigma, \Lambda)^n$ be the set of all ground term graphs with n vertices in $\mathcal{G}(\Sigma, \Lambda)$, and let $\mathcal{G}(\Sigma, \Lambda)^{[n]}$ be the set of all ground term graphs with at most n vertices in $\mathcal{G}(\Sigma, \Lambda)$. Also, hereditary FGS language classes are defined as follows:

$$\mathcal{H}\text{-}\mathcal{FGSL}(m, *, t, *, w, d) = \bigcup_{s \geq 0,\, r \geq 1} \mathcal{H}\text{-}\mathcal{FGSL}(m, s, t, r, w, d),$$

$$\mathcal{VH}\text{-}\mathcal{FGSL}(m, *, *, r, w, d) = \bigcup_{s \geq 0,\, t \geq 0} \mathcal{VH}\text{-}\mathcal{FGSL}(m, s, t, r, w, d),$$

$$\mathcal{VH}\text{-}\mathcal{FGSL}_{*,\Delta}(m, s, t, r, w, d)$$

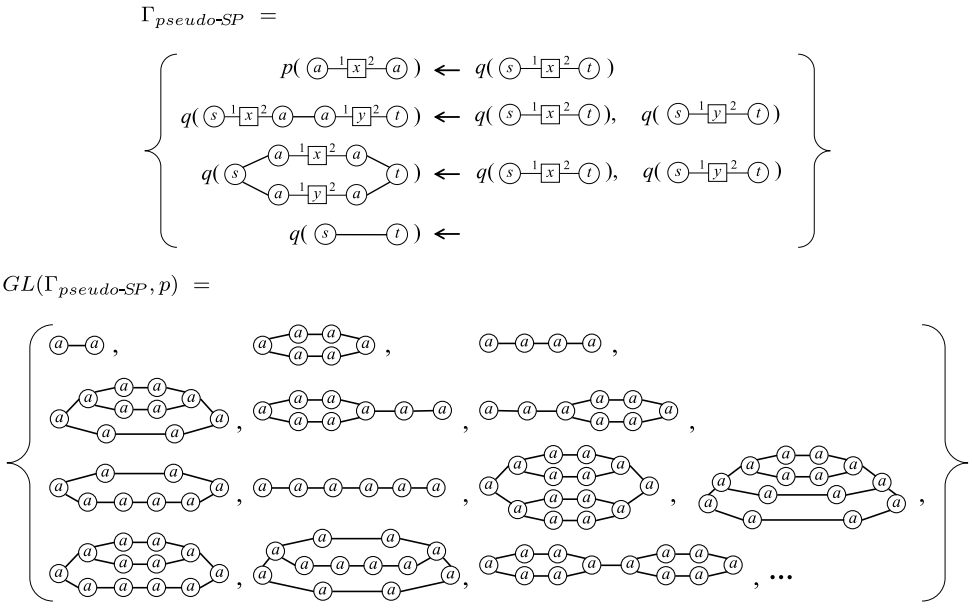


Fig. 7 Variable-hereditary FGS language: The FGS language $GL(\Gamma_{pseudo\text{-}SP}, p)$ is in $\mathcal{VH}\text{-}\mathcal{FGSL}_{2,3}(4,2,2,1,2,2)$. The treewidth of any graph in $GL(\Gamma_{pseudo\text{-}SP}, p)$ is at most 2 because each graph in it is a series-parallel graph.

$$= \bigcup_{k\geq 1} \mathcal{VH}\text{-}\mathcal{FGSL}_{k,\Delta}(m,s,t,r,w,d).$$

Figure 3 shows an example Γ_{SP} of variable-hereditary FGSs. From the graph rewriting rules of Γ_{SP}, we have $GL(\Gamma_{SP}, p) \in \mathcal{VH}\text{-}\mathcal{FGSL}(4,2,2,1,2,2)$. In Γ_{SP}, the term graph appearing at the head of the 2nd and 3rd graph rewriting rules have 2 variables that share a vertex as a port. From this, the degree of a graph in $GL(\Gamma_{SP}, p)$ is not bounded. We divide each shared vertex into at least 2 vertices and connect the vertices by edges. In this way, we have the FGS $\Gamma_{pseudo\text{-}SP}$ in Fig. 7, whose language $GL(\Gamma_{pseudo\text{-}SP}, p)$ consists of graphs of degree at most 3. Since the treewidth of any series-parallel graphs is at most 2, it is easy to see that the treewidth of graphs that are generated by $\Gamma_{pseudo\text{-}SP}$ is at most 2.

3.2 VC Dimension of Variable-Hereditary FGS

Theorem 1: $\mathcal{H}\text{-}\mathcal{FGSL}(m,*,t,*,w,d)$ has an exponential dimension when $m \geq 10, t \geq 1, w \geq 2$, and $d \geq 2$.

Proof. It follows obviously from the proof of Theorem 4.2 in [12]. We translate any string w over $\{0,1\}$ appearing on the proof into a ground term graph g which is a chain graph of the same length $|w|$ with edge labels in $\{0,1\}$. Two vertices of degree 1 of g are specified by special vertex labels which express the start and end points of the string, respectively. □

Figure 8 shows an example of FGSs with no edge label, which also realizes an idea of the proof of Theorem 4.2 in [12]. It shows that $\mathcal{H}\text{-}\mathcal{FGSL}(m,*,t,*,w,d)$ has an exponential dimension when $m \geq 10, t \geq 1, w \geq 2$, and $d \geq 5$, unless any term graph and ground term graph have more than one edge label.

Theorem 2: If m, r, w, d are constants ($m \geq 1, r \geq 1, w \geq 1, d \geq 0$), $\mathcal{VH}\text{-}\mathcal{FGSL}(m,*,*,r,w,d)$ has a polynomial dimension.

Proof. For any $n \geq 0$, let $\mathcal{VH}\text{-}\mathcal{FGSL}(m,*,*,r,w,d)^n$ be

$$\{L \cap \mathcal{G}(\Sigma,\Lambda)^n \mid L \in \mathcal{VH}\text{-}\mathcal{FGSL}(m,*,*,r,w,d)\},$$

and let $\mathcal{VH}\text{-}\mathcal{FGSL}(m,*,*,r,w,d)^{[n]}$ be

$$\{L \cap \mathcal{G}(\Sigma,\Lambda)^{[n]} \mid L \in \mathcal{VH}\text{-}\mathcal{FGSL}(m,*,*,r,w,d)\}.$$

Thereafter, the number of variable-hereditary FGS languages in $\mathcal{VH}\text{-}\mathcal{FGSL}(m,*,*,r,w,d)^n$ is evaluated. Let Γ be a variable-hereditary FGS. Since the number of graph rewriting rules does not exceed m, only m predicate symbols with at most r arguments need to be considered. Let $C = q_0(g_1^0,\ldots,g_{r_0}^0) \leftarrow q_1(g_1^1,\ldots,g_{r_1}^1),\ldots,q_t(g_1^t,\ldots,g_{r_t}^t)$ be a graph rewriting rule in Γ. We only need to consider a sufficient amount of variable-hereditary FGSs to generate a non-empty set in $\mathcal{G}(\Sigma,\Lambda)^n$. Hence, we consider graph rewriting rules whose heads contain term graphs with at most n vertices.

For a term graph of at most n vertices, considering the ordering of the ports of size at most w, the maximum number of hyperedges is

$$\sum_{\ell=1}^{w} \ell! \left(\begin{array}{c} n \\ \ell \end{array} \right) \leq w n^w.$$

Since the number of arguments of the predicate symbol q_0 of the rule C is at most r, i.e., $r_0 \leq r$, the number of possible distinct variables in C is at most rwn^w. Let $K = rwn^w$. K is the maximum number of distinct variables which appear in C.

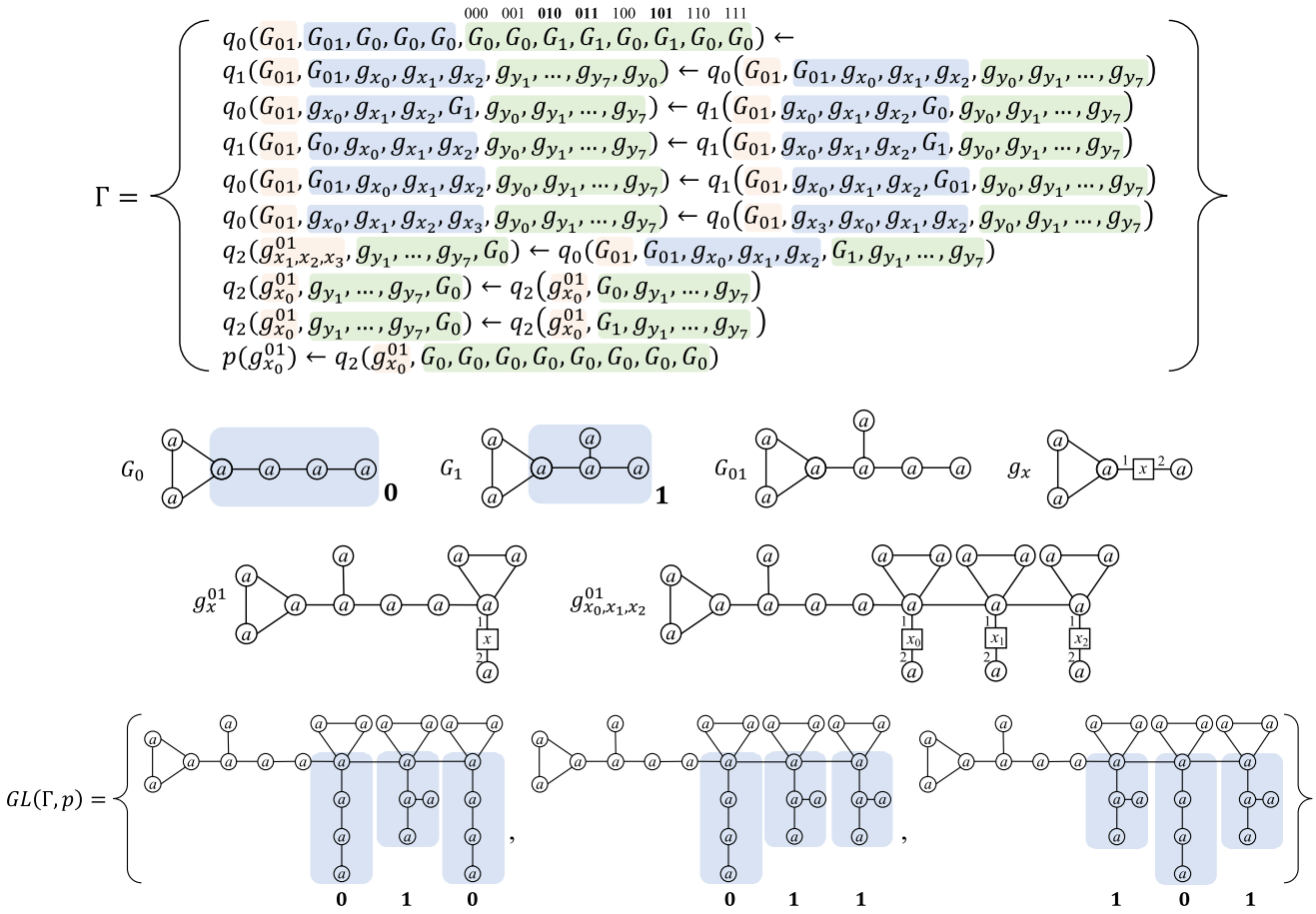


Fig. 8 Hereditary FGS language: $GL(\Gamma, p)$ is in $\mathcal{H}$-$\mathcal{FGSL}(10, *, 1, *, 2, 5)$. The FGS language $GL(\Gamma, p)$ contains three graphs that express 010, 011 and 101. In a similar way, we construct a hereditary FGS which generates a finite set of graphs which express strings in $\{0, 1\}^*$. $\mathcal{H}$-$\mathcal{FGSL}(10, *, 1, *, 2, 5)$ has an exponential dimension, even if any term graph and ground term graph have no edge label.

For a term graph of at most n vertices and maximum degree d, the number of edges is at most dn and the number of possible combinations of edges is

$$\left(\sum_{\ell=1}^{d} \binom{n-1}{\ell}\right)^n \leq n^{dn}.$$

Since the head has at most r arguments and at most m predicate symbols is sufficient for Γ, the number of possible heads for C is at most

$$\begin{aligned}\mathcal{S}_1 &= m \cdot \left((|\Sigma| + 1)^n \cdot (|\Lambda| + 1)^{dn} \cdot n^{dn} \cdot (K + 1)^{wn^w}\right)^r \\ &\leq m \left((2|\Sigma|)^n \cdot (2|\Lambda|)^{dn} \cdot n^{dn} \cdot (2K)^{wn^w}\right)^r \\ &= m \left(2^{n+dn+wn^w} |\Sigma|^n |\Lambda|^{dn} n^{dn} K^{wn^w}\right)^r.\end{aligned}$$

A term graph of the body of a variable-hereditary graph rewriting rule is the star term graph of a variable of the head of the graph rewriting rule. The number of star term graphs obtained from the head of C is at most K. Thus, the number of possible atoms appearing in the body of C is at most $m(K + 1)^r$. Since the order of atoms in a body is negligible, the number of possible bodies for the head of C is at most $2^{m(K+1)^r}$. From the above, the number of possible graph rewriting rules for C is at most

$$\begin{aligned}\mathcal{S}_2 &= \mathcal{S}_1 \cdot 2^{m(K+1)^r} \\ &\leq m \left(2^{n+dn+wn^w} |\Sigma|^n |\Lambda|^{dn} n^{dn} K^{wn^w}\right)^r 2^{m(K+1)^r}.\end{aligned}$$

Therefore, since the number of variable-hereditary FGS of m graph rewriting rules is at most $(\mathcal{S}_2 + 1)^m$, we have $\log_2 |\mathcal{VH}\text{-}\mathcal{FGSL}(m, *, *, r, w, d)^n| \leq m \log_2(\mathcal{S}_2 + 1) \leq m + m \log_2 \mathcal{S}_2$. Since

$$\begin{aligned}\log_2 \mathcal{S}_2 \leq{}& \log_2 m + rn(1 + d + \log_2 |\Sigma| + d \log_2 |\Lambda|) \\ &+ rdn \log_2 n + rwn^w \\ &+ rwn^w \log_2 K + m(K + 1)^r,\end{aligned}$$

and $K = rwn^w$, if m, r, w, d are constants, $\log_2 |\mathcal{VH}\text{-}\mathcal{FGSL}(m, *, *, r, w, d)^n| = O(n^w \log_2 n + n^{rw})$. Thus, we have $\log_2 \left|\mathcal{VH}\text{-}\mathcal{FGSL}(m, *, *, r, w, d)^{[n]}\right| = O(n^w \log_2 n + n^{rw})$ if m, r, w, d are constants. Therefore, from Lemma 2, if all m, r, w, d are constants, $\mathcal{VH}\text{-}\mathcal{FGSL}(m, *, *, r, w, d)$ has a polynomial dimension. □

Corollary 1: Let Δ be any positive integer ($\Delta \geq 1$). If m, r, w, d are constants ($m \geq 1, r \geq 1, w \geq 1, d \geq 0$), $\mathcal{VH}$-$\mathcal{FGSL}_{*,\Delta}(m, *, *, r, w, d)$ has a polynomial dimension.

3.3 Polynomial-Time Fitting for Variable-Hereditary FGS of Bounded Degree

Miyano et al. showed in [12] that a class of elementary formal systems is polynomial-time PAC learnable. In this section, we prove that a class of variable-hereditary FGSs is polynomial-time PAC learnable. The result of this section follows the result of the paper [12].

Lemma 3: Given an FGS Γ, a unary predicate symbol p, and a ground term graph G with n vertices, if Γ is a variable-hereditary FGS that satisfies the conditions (1)–(6) of Definition 21 and $GL(\Gamma, p)$ satisfies the conditions (7)–(8) of Definition 23, the problem of determining whether or not $G \in GL(\Gamma, p)$ is computable in

$$O(\xi(m, s, t, r, w, \Delta)\, n^{2s(w+1)} \cdot \mathrm{ISO}(n)) \text{ time},$$

where $\xi(m, s, t, r, w, \Delta)$ is an exponential function of s, w, Δ and $\mathrm{ISO}(n)$ is the computation time required to decide the isomorphism between two ground term graphs with n vertices.

Proof. Since G is a ground term graph and Γ is variable-hereditary, whenever we use $C \in \Gamma$ to show that $p(G)$ is provable from Γ, i.e., $\Gamma \vdash p(G) \leftarrow$, we can obtain the graph rewriting rule $C\theta$ consisting of ground term graphs by a substitution θ before applying the modus ponens. Moreover, since Γ is hereditary, the substitution only needs to consider all variables that correspond to the connected subgraph (ground term subgraph) of G with at most w boundary vertices.

Let $Sub(G)$ be the set of all connected subgraphs α of G such that $|bv(\alpha)| \leq w$ holds. Since any subgraph in $Sub(G)$ is intended to be replaced with variables, the order of boundary vertices corresponding to the ports of each variable is also considered. First we evaluate $|Sub(G)|$ by using the boundary representations of the subgraphs (Definition 15) and the condition (8) of Definition 23. Let u be a boundary vertex of a subgraph of G. The number of possible combinations of boundary edges connecting to u is at most 2^{Δ}. The maximum number of hyperedges is at most wn^w from the proof of Theorem 2. Then, we have $|Sub(G)| \leq 2 \cdot n \cdot wn^w \cdot (2^{\Delta})^w = 2^{w\Delta+1} wn^{w+1}$.

Let $\mathcal{G}(G)$ be the set of all graph rewriting rules obtained by replacing each variable of the term graphs of each graph rewriting rule in Γ with any connected subgraph in $Sub(G)$. The set $\mathcal{G}(G)$ contains sufficiently many graph rewriting rules in order to construct a derivation showing that $\Gamma \vdash p(G) \leftarrow$ holds. For each graph rewriting rule in Γ, the number of substitutions that replace all variables of the term graphs with connected subgraphs in $Sub(G)$ is at most $|Sub(G)|^s$, since every head has at most s variables. Therefore, $|\mathcal{G}(G)| \leq m|Sub(G)|^s$ holds. The length of a graph rewriting rule C, denoted by $|C|$, is defined as the sum of the representation sizes of the graphs appearing in C. For a graph rewriting rule C and an atom A, the graph rewriting rule obtained by applying modus ponens can be computed in $O(|C| + |A|)$ time. The length of each graph rewriting rule in $\mathcal{G}(G)$ is at most $|G|r(1 + t)$, where $|G|$ is the representation size of the graph G.

If $\mathcal{G}(G)$ contains a graph rewriting rule $A \leftarrow$, we remove all occurrences of A from the body of each graph rewriting rule in $\mathcal{G}(G)$. This step can be computed in $O(tr\,|\mathcal{G}(G)| \cdot \mathrm{ISO}(n))$ time. This step is repeated until an atom $p(G)$ is obtained or no new atom is obtained anymore. Since at most $|\mathcal{G}(G)|$ steps are sufficient for this task, the total computation is completed in

$$\begin{aligned}
&O(tr\,|\mathcal{G}(G)|^2 \cdot \mathrm{ISO}(n)) \\
&= O(tr(m|Sub(G)|^s)^2 \cdot \mathrm{ISO}(n)) \\
&= O(tr(m(2^{w\Delta+1} wn^{w+1})^s)^2 \cdot \mathrm{ISO}(n)) \\
&= O(\xi(m, s, t, r, w, \Delta)\, n^{2s(w+1)} \cdot \mathrm{ISO}(n)) \text{ time},
\end{aligned}$$

where $\xi(m, s, t, r, w, \Delta)$ is an exponential function of s, w, Δ. □

Lemma 4 ([7]): Given two graphs G_1 and G_2 with n vertices of maximum degree at most Δ, the problem of determining whether or not G_1 and G_2 are isomorphic is computable in $n^{O((\log \Delta)^c)}$ time for an absolute constant c.

Lemma 5: Given a ground term graph G with n vertices and an FGS $\Gamma \in \mathcal{VH}$-$\mathcal{FGSL}_{k,\Delta}(m, s, t, r, w, d)$, the problem of determining whether $G \in GL(\Gamma, p)$ is computable in polynomial time with respect to m, t, r and n, if s, w and Δ are constants.

Proof. This is proved from Lemmas 3 and 4. □

Lemma 6: If m, s, t, r, w, d, Δ are constants ($m \geq 1$, $s \geq 0$, $t \geq 0$, $r \geq 1$, $w \geq 1$, $d \geq 0$, $\Delta \geq 1$), $\mathcal{VH}$-$\mathcal{FGSL}_{*,\Delta}(m, s, t, r, w, d)$ has a polynomial-time fitting.

Proof. Let S be a finite set of examples of a concept in $\mathcal{VH}$-$\mathcal{FGSL}_{*,\Delta}(m, s, t, r, w, d)$. We note that S is a subset of $\mathcal{G}(\Sigma, \Lambda) \times \{0, 1\}$. If S_+ is empty, we choose one ground term graph $G \in \mathcal{G}(\Sigma, \Lambda)$ of treewidth at most w that is not isomorphic to any element of S_-, and then we set $\Gamma = \{p(G) \leftarrow\}$. Clearly, Γ is a variable-hereditary FGS consisting of term graphs of treewidth at most w, and $GL(\Gamma, p)$ is consistent with S. Hence, we assume that S_+ is non-empty.

Let $\mathcal{G}(m, s, t, r, w, d, S_+)$ be the set of all pairs (Γ, p) of an FGS Γ and a unary predicate symbol p that satisfy the following 2 conditions:

(1) Γ is a variable-hereditary FGS which satisfies conditions (1)–(8) of Definitions 21 and 23.
(2) For each term graph g of the heads of graph rewriting rules in Γ, there exist a substitution θ and a ground term graph $G \in S_+$ such that $g\theta$ is term subgraph isomorphic to G.

Claim 1. There exists a pair $(\Gamma, p) \in \mathcal{G}(m, s, t, r, w, d, S_+)$

such that $GL(\Gamma, p)$ and S are consistent.

Proof of Claim 1. Since S is a set of examples of a concept c in $\mathcal{VH}\text{-}\mathcal{FGSL}_{*,\Delta}(m,s,t,r,w,d)$, there is a pair (Γ_0, p) of an FGS Γ_0 and a unary predicate symbol p that satisfy condition (1) and $GL(\Gamma_0, p) = c$. Without loss of generality, we assume that Γ_0 contains only graph rewriting rules necessary to generate the examples in S_+. Therefore, Γ_0 satisfies condition (2). (*End of Proof of Claim* 1)

Claim 2. $|\mathcal{G}(m,s,t,r,w,d,S_+)|$ is a polynomial with respect to $|S_+|$ and $n = \max\{|V_G| \mid G \in S_+\}$.

Proof of Claim 2. Let $\Pi(s, S_+)$ be the set of term graphs g satisfying the following 2 conditions:

(1) g is a term graph with at most s occurrences of variables in X. We note that g may have a variable that occurs more than once.
(2) There exists a substitution θ such that $g\theta$ is a term subgraph isomorphic to at least one example $G \in S_+$.

Let $Sub(G)$ be the set of all connected subgraphs α of G such that $|bv(\alpha)| \leq w$ holds. From the proof of Lemma 3, we have $Sub(G) \leq 2^{w\Delta+1} w n^{w+1}$. We note that a term subgraph $g\theta$ that satisfies the condition (2) is a subgraph in $Sub(G)$ since any graph rewriting rule is variable-hereditary. Since at most s variables appear in g and the rank of each variable is between 1 and w, and moreover since the scope of each variable is limited to the graph rewriting rule in which the variable appears, without loss of generality, the number of distinct variables which are used in Γ is assumed to be at most sw. Thus, the number of possible bindings obtained from G is at most $sw|Sub(G)|$. Since the number of possible substitutions θ is at most $(sw|Sub(G)|)^\ell$ for a term graph g having ℓ occurrences of variables ($0 \leq \ell \leq s$) and the vertex labels of ports of the variables are ignored, the following inequality holds:

$$\begin{aligned}
|\Pi(s, S_+)| &\leq \sum_{G \in S_+} \left(\sum_{\ell=0}^{s} |Sub(G)| |\Sigma|^{\ell w} (sw|Sub(G)|)^\ell \right) \\
&\leq \sum_{G \in S_+} (sw|\Sigma|^w)^{s+1} |Sub(G)|^{s+2} \\
&\leq |S_+| (sw|\Sigma|^w)^{s+1} \left(2^{w\Delta+1} w n^{w+1}\right)^{s+2} \\
&= \xi_1(s,r,w,\Delta) |S_+| n^{(s+2)(w+1)},
\end{aligned}$$

where $\xi_1(s,r,w,\Delta)$ is an exponential function of s, w, Δ. The number of possible heads is at most $m|\Pi(s,S_+)|^r$. Also, the number of possible body atoms is at most $m(s|\Sigma|^w)^r$. This is because each term graph of the body is a star term graph of a variable appearing in the term graph of the head. Therefore, the following equation holds:

$$\begin{aligned}
&|\mathcal{G}(m,s,t,r,w,d,S_+)| \\
&\leq \left(m|\Pi(s,S_+)|^r (m(s|\Sigma|^w)^r)^t\right)^m \\
&\leq \left(m\left(\xi_1(s,r,w,\Delta) |S_+| n^{(s+2)(w+1)}\right)^r (m(s|\Sigma|^w)^r)^t\right)^m \\
&= \xi_2(m,s,t,r,w,\Delta) |S_+|^{mr} n^{mr(s+2)(w+1)},
\end{aligned}$$

where $\xi_2(m,s,t,r,w,\Delta)$ is an exponential function of m, s, t, r, w, Δ. Therefore, $|\mathcal{G}(m,s,t,r,w,d,S_+)|$ is a polynomial with respect to $|S_+|$ and n if all the parameters m, s, t, r, w, and Δ are constants. (*End of Proof of Claim* 2)

From the above, the following algorithm finds a desired variable-hereditary FGS in polynomial time: Firstly we enumerate the pairs (Γ, p) in $\mathcal{G}(m,s,t,r,w,d,S_+)$. For each pair, we check whether or not $G \in GL(\Gamma, p)$ holds for all $G \in S_+$ and $G \notin GL(\Gamma, p)$ holds for all $G \in S_-$. This check is polynomial-time computable from Lemma 5. If a pair is found for which all of these checks hold, the algorithm outputs the pair as a hypothesis. □

3.4 Main Theorem

Theorem 3: If m, s, t, r, w, d, and Δ are constants ($m \geq 1$, $s \geq 0$, $t \geq 0$, $r \geq 1$, $w \geq 1$, $d \geq 0$, $\Delta \geq 1$), $\mathcal{VH}\text{-}\mathcal{FGSL}_{*,\Delta}(m,s,t,r,w,d)$ is polynomial-time PAC learnable.

Proof. It follows from Lemma 1, Corollary 1, and Lemma 6. □

4. Conclusion

In this paper, we defined $\mathcal{VH}\text{-}\mathcal{FGSL}_{k,\Delta}(m,s,t,r,w,d)$ as the class of variable-hereditary FGS languages under some parameters, which is a subclass of the class $\mathcal{FGSL}$ of FGS languages, and proved that the class $\mathcal{VH}\text{-}\mathcal{FGSL}_{*,\Delta}(m,s,t,r,w,d)$ is polynomial-time PAC learnable if the parameters m, s, t, r, w, d, Δ are constants. In future work, we will discuss the learnability of the class $\mathcal{VH}\text{-}\mathcal{FGSL}_{k,\Delta}(m,s,t,r,w,d)$ under the minimally adequate teacher (MAT) learning model. The MAT learning model is a computational learning model that learns a concept in a specified concept class using only membership and equivalence queries as oracles, which was introduced by Angluin [1].

In addition to the PAC learning model discussed in this paper, other known learning models in computational learning theory include the query learning model, which is an extension of MAT learning model, and the polynomial-time inductive inference model from positive examples. Shibata et al. [18] showed that a subclass of the context-deterministic context-free languages is polynomial-time learnable. Hara et al. [8] showed that the class of graph languages defined by the context-deterministic regular FGSs is polynomial-time MAT learnable. Based on these results, we are studying the polynomial-time learnability of context-deterministic regular FGS language classes and the polynomial-time learnability of concept classes based on the size restricted FGSs introduced by Uchida et al. [19].

Acknowledgments

This work was partially supported by Grant-in-Aid for Sci-

entific Research (C) (Grant Numbers 19K12102, 19K12103, 21K12021) from Japan Society for the Promotion of Science (JSPS).

References

[1] D. Angluin, "Learning regular sets from queries and counterexamples," Information and Computation, vol.75, no.2, pp.87–106, 1987.
[2] S. Arikawa, T. Shinohara, and A. Yamamoto, "Learning elementary formal systems," Theoretical Computer Science, vol.95, pp.97–113, 1992.
[3] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth, "Learnability and the Vapnik-Chervonenkis Dimension," J. ACM, vol.36, no.4, pp.929–965, 1989.
[4] H.L. Bodlaender, "A linear-time algorithm for finding tree-decompositions of small treewidth," SIAM J. Comput., vol.25, pp.1305–1317, 1990.
[5] 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, 2013.
[6] R. Diestel, Graph Theory, 5th ed., Graduate Texts in Mathematics, vol.173, Springer, 2017.
[7] M. Grohe, D. Neuen, and P. Schweitzer, "A faster isomorphism test for graphs of small degree," SIAM J. Comput., pp.FOCS18-1–FOCS18-36, 2020.
[8] S. Hara and T. Shoudai, "Polynomial time MAT learning of C-deterministic regular formal graph systems," Proc. 3rd International Conference on Advanced Applied Informatics, pp.204–211, 2014.
[9] D. Haussler, M. Kearns, N. Littlestone, and M.K. Warmuth, "Equivalence of models for polynomial learnability," Information and Computation, vol.95, no.2, pp.129–161, 1991.
[10] T. Horváth, J. Ramon, and S. Wrobel, "Frequent subgraph mining in outerplanar graphs," Data Min. Knowl. Disc., vol.21, pp.472–508, 2010.
[11] J.W. Lloyd, Foundations of Logic Programming, 2nd extended ed., Springer, 1987.
[12] S. Miyano, A. Shinohara, and T. Shinohara, "Polynomial-time learning of elementary formal systems," New Gener. Comput., vol.18, pp.217–242, 2000.
[13] M. Mohri, et al., Foundation of Machine Learning, 2nd ed., MIT Press, 2018.
[14] B. Natarajan, "On learning sets and functions," Mach. Learn., vol.4, no.1, pp.67–97, 1989.
[15] B. Natarajan, Machine Learning — A Theoretical Approach, Morgan Kaufmann Publishers, 1991.
[16] H. Sakamoto, K. Hirata, and H. Arimura, "Learning elementary formal systems with queries," Theoretical Computer Science, vol.298, no.1, pp.21–50, 2003.
[17] S. Shalev-Shwartz and S. Ben-David, Understanding Machine Learning: From Theory to Algorithms, Cambridge University Press, 2014.
[18] C. Shibata and R. Yoshinaka, "PAC learning of some subclasses of context-free grammars with basic distributional properties from positive data," Proc. 24th International Conference on Algorithmic Learning Theory, Springer, Lecture Notes in Artificial Intelligence, vol.8139, pp.143–157, 2013.
[19] T. Uchida, T. Shoudai, and S. Miyano, "Parallel algorithms for refutation tree problem on formal graph systems," IEICE Trans. Inf. & Syst., vol.78, no.2, pp.99–112, 1995.
[20] L. Valiant, "A theory of the learnable," Commun. ACM, vol.27, no.11, pp.1134–1142, 1984.
[21] H. Yamasaki and T. Shoudai, "Mining of frequent externally extensible outerplanar graph patterns," Proc. 7th International Conference on Machine Learning and Applications, pp.871–876, 2008.
[22] H. Yamasaki, Y. Sasaki, T. Shoudai, T. Uchida, and Y. Suzuki, "Learning block-preserving graph patterns and its application to data mining," Mach. Learn., vol.76, no.1, pp.137–173, 2009.
[23] X. Yan and J. Han, "gSpan: Graph-based substructure pattern mining," Proc. 2002 IEEE International Conference on Data Mining, pp.721–724, 2002.

Takayoshi Shoudai received the B.S. in 1986, the M.S. degree in 1988 in Mathematics and the Dr. Sci. in 1993 in Information Science all from Kyushu University. Currently, he is a professor of Department of Computer Science and Engineering, Fukuoka Institute of Technology. His research interests include graph algorithms, computational learning theory and machine learning.

Satoshi Matsumoto is a professor of Department of Mathematical Sciences, Tokai University, Kanagawa, Japan. He received the B.S. degree in Mathematics, the M.S. and Dr. Sci. degrees in Information Systems all from Kyushu University, Fukuoka, Japan in 1993, 1995 and 1998, respectively. His research interests include algorithmic learning theory.

Yusuke Suzuki received the B.S. degree in Physics, the M.S. and Dr. Sci. degrees in Informatics all from Kyushu University, in 2000, 2002 and 2007, respectively. He is currently a lecturer of Graduate School of Information Sciences, Hiroshima City University, Hiroshima, Japan. His research interests include machine learning and data mining.

Tomoyuki Uchida received the B.S. degree in Mathematics, the M.S. and Dr. Sci. degrees in Information Systems all from Kyushu University, in 1989, 1991 and 1994, respectively. Currently, he is a professor of Graduate School of Information Sciences, Hiroshima City University. His research interests include data mining from semistructured data, algorithmic graph theory and algorithmic learning theory.

Tetsuhiro Miyahara is an associate professor of Graduate School of Information Sciences, Hiroshima City University, Hiroshima, Japan. He received the B.S. degree in Mathematics, the M.S. and Dr. Sci. degrees in Information Systems all from Kyushu University, Fukuoka, Japan in 1984, 1986 and 1996, respectively. His research interests include algorithmic learning theory, knowledge discovery and machine learning.