PAPER *Special Section on Foundations of Computer Science — Frontiers of Theoretical Computer Science —*

Polynomial Time Learnability of Graph Pattern Languages Defined by Cographs

Takayoshi SHOUDAI[†a)], ***Member***, **Yuta YOSHIMURA**[††*], ***Nonmember***, **Yusuke SUZUKI**[†††], **Tomoyuki UCHIDA**[†††], ***and*** **Tetsuhiro MIYAHARA**[†††], ***Members***

SUMMARY A cograph (complement reducible graph) is a graph which can be generated by disjoint union and complement operations on graphs, starting with a single vertex graph. Cographs arise in many areas of computer science and are studied extensively. With the goal of developing an effective data mining method for graph structured data, in this paper we introduce a graph pattern expression, called a *cograph pattern*, which is a special type of cograph having structured variables. Firstly, we show that a problem whether or not a given cograph pattern g matches a given cograph G is NP-complete. From this result, we consider the polynomial time learnability of cograph pattern languages defined by cograph patterns having variables labeled with mutually different labels, called *linear cograph patterns*. Secondly, we present a polynomial time matching algorithm for linear cograph patterns. Next, we give a polynomial time algorithm for obtaining a minimally generalized linear cograph pattern which explains given positive data. Finally, we show that the class of linear cograph pattern languages is polynomial time inductively inferable from positive data.

key words: *graph pattern matching, cograph pattern, polynomial time algorithm, inductive inference, computational learning theory*

1. Introduction

We shall consider the problem of learning graph patterns from positive graph structured data. To apply such learnability to effective data mining from a graph database, graph structured data and graph patterns need to have rich expressive power and computational tractability. Cographs, which we use as graph structured data, and cograph patterns, which we introduce here as a new kind of graph patterns, have these properties. A cograph (complement reducible graph) [5] is a graph which can be generated by disjoint union and complement operations on graphs, starting with a single vertex graph. Any cograph is also generated by disjoint union and join operations on appropriate cographs, where a join operation is an operation on graphs that makes the disjoint union and adds an edge between every two vertices in different cographs. Cographs are P_4-free, that is, graphs which do not contain any chain consisting of 4 vertices as an induced subgraph [5]. Let Σ be an alphabet for vertex labels. In Fig. 1, we give examples of cographs whose vertex labels are in $\Sigma = \{A, B, C, D, E\}$.

In this paper, we introduce a *cograph pattern* which is an expression for common structures in a graph database. A cograph pattern is a graph pattern which is a special type of cographs having structured variables. Structured variables in a cograph pattern can be replaced with arbitrary cographs. Thus, cograph patterns have rich expressive power. A polynomial time matching algorithm for cograph patterns, which we present in Sect. 4, ensures computational tractability. Let $\mathcal{X}$ be an infinite alphabet for variable labels. A *variable* in a cograph pattern g is a vertex of g that is labeled with a variable label in $\mathcal{X}$. We replace a variable label $x \in \mathcal{X}$ in g with a cograph G in the following way. For each variable h labeled with x, we make a copy of G, say G_h, and the new edges between all adjacent vertices to h and the vertices of G_h, and remove h. For a cograph pattern g and a cograph G, g is said to *match* G if G can be obtained from g by certain variable replacements. For example, the cograph pattern g in Fig. 2 matches the cographs G_1, G_2, and G_3 in Fig. 1

We denote by $\mathcal{CG}(\Sigma)$ and $\mathcal{CGP}(\Sigma, \mathcal{X})$ the set of all cographs and the set of all cograph patterns, respectively. For a cograph pattern g in $\mathcal{CGP}(\Sigma, \mathcal{X})$, the *cograph pattern language* of g, denoted by $L(g)$, is the set of all cographs $G \in \mathcal{CG}(\Sigma)$ such that g matches G. Let $\mathcal{P}$ be a subset of $\mathcal{CGP}(\Sigma, \mathcal{X})$. The class of all cograph pattern languages of $\mathcal{P}$

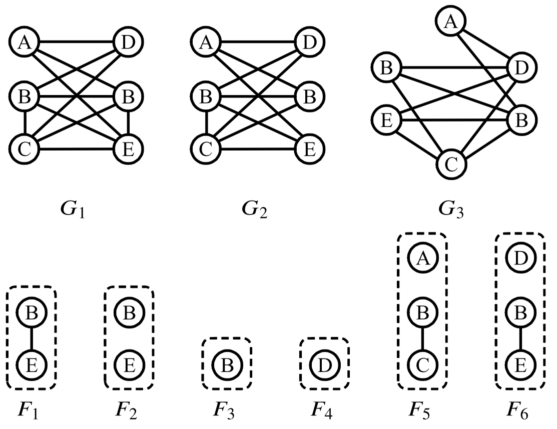


Fig. 1 Cographs G_1, G_2, G_3, $F_1, \ldots, F_6$. F_6 is obtained by disjoint union operation on F_1 and F_4, F_1 is obtained by complement operation on F_2, and G_1 is obtained by join operation on F_5 and F_6.

Manuscript received March 23, 2017.
Manuscript revised July 21, 2017.
Manuscript publicized December 19, 2017.

[†]The author is with Faculty of Contemporary Business, Kyushu International University, Kitakyushu-shi, 805–8512 Japan.
[††]The author was with Department of Informatics, Kyushu University, Fukuoka-shi, 819–0395 Japan.
[†††]The authors are with Faculty of Information Sciences, Hiroshima City University, Hiroshima-shi, 731–3194 Japan.
[*]Presently, with CAE Solutions Department, Engineering Solutions Unit, Fujitsu Kyushu System Services Limited.
a) E-mail: shoudai@cb.kiu.ac.jp

DOI: 10.1587/transinf.2017FCP0005

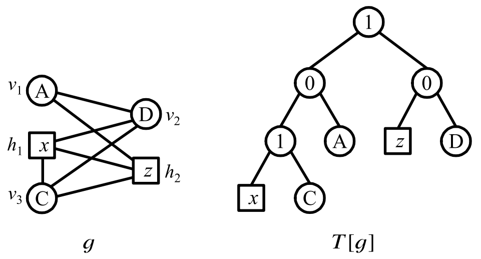


Fig. 2 A cograph pattern g and the cotree pattern $T[g]$ of g. We use square boxes to describe variables of cograph patterns. The labels ⓪ and ① of internal nodes in $T[g]$ mean applying disjoint union and join operations for cographs corresponding to subtrees, respectively. G_1 in Fig. 1 is obtained from g by replacing x and z with F_3 and F_1 in Fig. 1.

is defined as $\mathcal{L}_{\mathcal{P}} = \{L(g) \mid g \in \mathcal{P}\}$. Polynomial time inductive inference from positive data, which is a method used in computational learning theory, is an important type of learnability which ensures efficient learning from a database. Angluin [2] and Shinohara [12] showed that, if a class of languages C has finite thickness, and the membership problem and the minimal language problem for C are solvable in polynomial time, then the class C is polynomial time inductively inferable from positive data.

Firstly, we show that for any nonempty set S of cographs, the cardinality of $\{L \in \mathcal{L}_{\mathcal{CGP}(\Sigma,\mathcal{X})} \mid S \subseteq L\}$ is finite, that is, the class $\mathcal{L}_{\mathcal{CGP}(\Sigma,\mathcal{X})}$ has finite thickness. The *membership problem* for $\mathcal{L}_{\mathcal{P}}$ is to decide, given a cograph $G \in \mathcal{CG}(\Sigma)$ and a cograph pattern $g \in \mathcal{P}$, whether $G \in L(g)$. Secondly, we show that the membership problem for $\mathcal{L}_{\mathcal{CGP}(\Sigma,\mathcal{X})}$ is NP-complete. If NP≠P, this result indicates that it is very hard to show that $\mathcal{L}_{\mathcal{CGP}(\Sigma,\mathcal{X})}$ is polynomial time inductively inferable from positive data. Hence, we introduce a *linear cograph pattern*, which is a cograph pattern whose variable labels are mutually distinct. The set of all linear cograph patterns is denoted by $\mathcal{LCGP}(\Sigma,\mathcal{X})$. Since $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})} \subseteq \mathcal{L}_{\mathcal{CGP}(\Sigma,\mathcal{X})}$, $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$ has also finite thickness. Next, we show that the membership problem for $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$ is solvable in polynomial time. This result indicates that the computational tractability of the membership problem for cograph patterns depends on the existence of variables labeled with the same label. The *minimal language* (*MINL*) *problem* for $\mathcal{L}_{\mathcal{P}}$ is to find, given a finite set $S \subseteq \mathcal{CG}(\Sigma)$, a cograph pattern $g \in \mathcal{P}$ such that $S \subseteq L(g)$ and there is no cograph pattern $g' \in \mathcal{P}$ with $S \subseteq L(g') \subsetneq L(g)$. Thirdly, we show that the MINL problem for $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$ is solvable in polynomial time. Finally, as our main result, we show that the class $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$ is polynomial time inductively inferable from positive data.

We considered the inference of ordered term tree patterns [13], TTSP graph patterns [14], and interval graph patterns [15]. Since cographs are P_4-free, cograph patterns have expressive power incomparable with these patterns. Cograph patterns are used as a data model that these patterns cannot represent. From the definition of a cograph, a cograph pattern has a unique representation of its parse structure, called a *cotree pattern*, such that any variable in a cograph pattern appears in a leaf of its cotree pattern. In Fig. 2, we give the cotree pattern $T[g]$ of the cograph pattern g as an example. In [9], [10], we introduced unordered term tree patterns, in which the variables are defined as hyperedges with variable labels. We showed that even if a given unordered term tree pattern is linear, the membership problem for unordered term tree patterns is NP-complete if each variable consists of 4 vertices as a hyperedge [10]. However it is solvable in polynomial time if a given unordered term tree pattern is linear and each variable consists of 2 vertices [9]. Unlike cograph patterns, the computational tractability of the membership problem for unordered term tree patterns also depends on the number of vertices constituting a variable.

A cograph has a unique representation of its parse structure, called a *cotree*. By using the cotree representation, several problems which are intractable for general graphs, such as the graph isomorphism problem, graph coloring problem, and Hamiltonian cycle problem, are solvable in polynomial time for cographs [6]. In this paper, we also use the cotree representation to show that $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$ is polynomial time inductively inferable from positive data. Jamison and Olariu [8] investigated the notion of p-connectedness of a graph, and proposed a unique tree representation for arbitrary graphs, called the *homogeneous decomposition tree*. It is a natural extension of the cotree representation. Some superclasses of cographs were defined in terms of the number and structure of its induced P_4's [3], [4]. These classes are known to have the property of admitting a unique tree representation of a graph G that can be computed in polynomial time w.r.t. the size of G. The result in this paper will give a foundation for further studies of the polynomial time learnability of the graph classes that have such unique tree representations.

This paper is organized as follows. In Sect. 2, we define a cograph pattern and its graph language. In Sect. 3, we define the membership problem and the MINL problem for $\mathcal{L}_{\mathcal{CGP}(\Sigma,\mathcal{X})}$, and show that the membership problem for $\mathcal{L}_{\mathcal{CGP}(\Sigma,\mathcal{X})}$ is NP-complete. In Sect. 4, we present a polynomial time algorithm for solving the membership problem for $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$, and in Sect. 5, we present a polynomial time algorithm for solving the MINL problem for $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$. From these results, we can conclude that $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$ is polynomial time inductively inferable from positive data. In Sect. 6, we conclude this paper by discussing related research problems. This paper is the full version of the paper [16], with complete definitions and proofs.

2. Preliminaries

In this section, we formally define a cograph pattern as a new graph pattern, which can be generated by disjoint union and complement operations on graph patterns, starting with a single vertex or a single structured variable. We define a cograph pattern language as a language of a cograph pattern.

Let Σ be an alphabet for vertex labels. A vertex labeling of a graph $G = (V, E)$ is a function φ from V to Σ. In this paper, a graph means a vertex-labeled undirected graph without multi-edges or self-loops. For a graph G, the vertex and edge sets of G are denoted by $V(G)$ and $E(G)$, respectively. For a subset U of $V(G)$, the *induced subgraph* of G w.r.t. U, denoted by $G[U]$, is the subgraph F of G such that $V(F) = U$ and $E(F) = \{\{u, v\} \in E(G) \mid u, v \in U\}$. Let $G_1, G_2 \ldots, G_k$ be graphs with $V(G_i) \cap V(G_j) = \emptyset$ for each i, j $(1 \leq i < j \leq k)$. The *disjoint union graph* of $G_1, G_2, \ldots, G_k$, denoted by $G_1 ⓪ G_2 ⓪ \cdots ⓪ G_k$, is the graph having the vertex set $V(G_1) \cup V(G_2) \cup \cdots \cup V(G_k)$ and the edge set $E(G_1) \cup E(G_2) \cup \cdots \cup E(G_k)$. The *complement graph* of G, denoted by $\bar{G}$, is the graph having the vertex set $V(G)$ and the edge set $\{\{u, v\} \mid u, v \in V(G), \{u, v\} \notin E(G)\}$. The *join graph* of $G_1, G_2, \ldots, G_k$, denoted by $G_1 ① G_2 ① \cdots ① G_k$, is the graph having the vertex set $V(G_1) \cup V(G_2) \cup \cdots \cup V(G_k)$ and the edge set $E(G_1) \cup E(G_2) \cup \cdots \cup E(G_k) \cup \{\{u, v\} \mid u \in V(G_i),\ v \in V(G_j)\ (1 \leq i < j \leq k)\}$.

Definition 1 (Cograph): A *cograph* G is a vertex-labeled undirected graph over Σ recursively defined as follows.

1. A single vertex labeled with an element in Σ is a cograph.
2. If G is a cograph, then the complement graph $\bar{G}$ is a cograph.
3. If G_1 and G_2 are cographs, then the disjoint union graph $G_1 ⓪ G_2$ is a cograph.

Definition 2 (Cograph pattern): Let Σ be an alphabet and $\mathcal{X}$ an infinite alphabet, where $\Sigma \cap \mathcal{X} = \emptyset$. A *cograph pattern* is a cograph $g = (V, E)$ with a vertex labeling $\varphi : V \to \Sigma \cup \mathcal{X}$. An element of $\mathcal{X}$ is called a *variable label*. A vertex labeled with a variable label is called a *variable*.

For a cograph pattern $g = (V, E)$ with vertex labeling φ, $H(g)$ denotes the set of variables, i.e., $H(g) = \{v \in V \mid \varphi(v) \in \mathcal{X}\}$, and $\mathcal{X}(g)$ denotes the set of all variable labels in g, i.e., $\mathcal{X}(g) = \{x \in \mathcal{X} \mid \exists v \in V \text{ s.t. } \varphi(v) \in \mathcal{X}\}$. We give an example in Fig. 1. Let $\Sigma = \{\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}, \mathrm{E}\}$ and $\mathcal{X} = \{x, y, z, \ldots\}$. For a cograph pattern g in Fig. 2, $V(g) = \{v_1, v_2, v_3, h_1, h_2\}$, $E(g) = \{\{v_1, v_2\}, \{v_1, h_2\}, \{h_1, v_2\}, \{h_1, h_2\}, \{h_1, v_3\}, \{v_3, v_2\}, \{v_3, h_2\}\}$, $H(g) = \{h_1, h_2\}$, and $\mathcal{X}(g) = \{x, z\}$. Since G_1 in Fig. 1 has no variable, G_1 is a cograph. $\mathcal{CG}(\Sigma)$ denotes the set of all cographs whose vertices are labeled with elements in Σ. $\mathcal{CGP}(\Sigma, \mathcal{X})$ denotes the set of all cograph patterns whose vertices are labeled with elements in $\Sigma \cup \mathcal{X}$.

Proposition 1 ([5]): The following properties hold for cograph patterns.

1. g is a cograph pattern if and only if there is no subset U of $V(g)$ such that the induced subgraph $g[U]$ is isomorphic to P_4, where P_4 is the chain consisting of 4 vertices.
2. Let g be a cograph pattern. For any subset $U \subsetneq V(g)$, the induced subgraph $g[U]$ is a cograph pattern.
3. Let g be a cograph pattern. g can be generated by disjoint union ⓪ and join ① on graphs, starting with a single vertex.

A cograph pattern g_1 is said to be *isomorphic* to a cograph pattern g_2, denoted by $g_1 \cong g_2$, if there exists a bijection $\psi : V(g_1) \to V(g_2)$ satisfying the following three conditions. Let φ_1 and φ_2 be the vertex labelings of g_1 and g_2, respectively. (1) For any vertices $u, v \in V(g_1)$, $\{u, v\} \in E(g_1)$ if and only if $\{\psi(u), \psi(v)\} \in E(g_2)$. (2) For any vertex $u \in V(g_1) \setminus H(g_1)$, $\varphi_1(u) = \varphi_2(\psi(u))$. (3) For any variables $h, h' \in H(g_1)$, $\varphi_1(h) = \varphi_1(h')$ if and only if $\varphi_2(\psi(h)) = \varphi_2(\psi(h'))$.

Let g be a cograph pattern in $\mathcal{CGP}(\Sigma, \mathcal{X})$ with a vertex labeling φ. For a vertex u in $V(g)$, let $N_g(u) = \{v \in V(g) \mid \{u, v\} \in E(g)\}$. For $X \subsetneq \mathcal{X}$, let $H_g(X) = \{h \in H(g) \mid \varphi(h) \in X\}$. For a cograph pattern g and a subset $V' \subsetneq V(g)$, we denote by $g - V'$ the cograph pattern obtained from g by removing all vertices in V', i.e., $g - V' = (V(g) \setminus V', E(g) \setminus \{\{u, v\} \in E(g) \mid u \in V' \text{ or } v \in V'\})$. Furthermore, for two cograph pattern f and f', we denote by $f \cup f'$ the cograph pattern $(V(f) \cup V(f'), E(f) \cup E(f'))$. Below, we denote a single vertex graph by its vertex simply.

Let x be a variable label in $\mathcal{X}$ and f a cograph in $\mathcal{CG}(\Sigma)$ or a cograph pattern in $\mathcal{CGP}(\Sigma, \mathcal{X})$ such that $V(g) \cap V(f) = \emptyset$. The form x/f is called a *variable replacement* of x by f. A new graph $g\{x/f\}$ can be constructed by replacing each variable h in $H(g)$ having the variable label x with a copy of f simultaneously and updating the neighboring relation. We define it formally as follows. For each $h \in H_g(\{x\})$, we make a copy of f, denoted by $f_{x,h}$.

$$\begin{aligned} g\{x/f\} = {} & (g - H_g(\{x\})) \\ & \cup \bigcup_{h \in H_g(\{x\})} f_{x,h} \\ & \cup \bigcup_{h \in H_g(\{x\})} \bigcup_{u \in N_g(h) \setminus H_g(\{x\})} f_{x,h} ① u \\ & \cup \bigcup_{h \in H_g(\{x\})} \bigcup_{h' \in N_g(h) \cap H_g(\{x\})} f_{x,h} ① f_{x,h'}. \end{aligned}$$

In Fig. 3, we give an example of the graph pattern obtained from g by a variable replacement of x by f.

Proposition 2: For cograph patterns $g, f \in \mathcal{CGP}(\Sigma, \mathcal{X})$, $g\{x/f\}$ is a cograph pattern in $\mathcal{CGP}(\Sigma, \mathcal{X})$.

Proof. Let h be a variable labeled with x. Since, in the graph $g\{x/f\}$, each vertex in $N_g(h)$ is adjacent to any vertex in $V(f_{x,h})$, we see that there is no subset U of $V(g\{x/f\})$ such

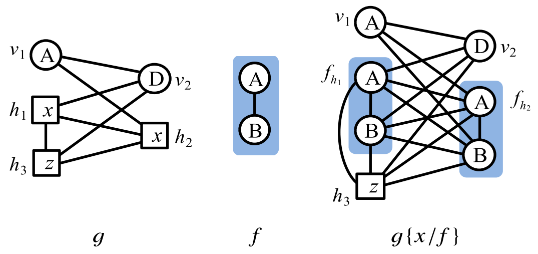


Fig. 3 The graph pattern $g\{x/f\}$ obtained from g by a variable replacement of x by f.

that the induced subgraph of $g\{x/f\}$ w.r.t. U is isomorphic to P_4. From Proposition 1, the statement holds. □

Let g be a cograph pattern in $\mathcal{CGP}(\Sigma,\mathcal{X})$ and $f_1,\dots,f_n$ ($n \geq 1$) be cographs in $\mathcal{CG}(\Sigma)$ or cograph patterns in $\mathcal{CGP}(\Sigma,\mathcal{X})$ such that $V(g)\cap V(f_i)=\emptyset$ $(1\leq i\leq n)$ and $V(f_i)\cap V(f_j)=\emptyset$ $(1\leq i<j\leq n)$. Let $x_1,\dots,x_n$ ($n\geq 1$) be mutually distinct variable labels in $\mathcal{X}$. A *substitution* is a finite collection of variable replacements $\{x_1/f_1,x_2/f_2,\dots,x_n/f_n\}$. For a substitution $\theta=\{x_1/f_1,x_2/f_2,\dots,x_n/f_n\}$, a new cograph pattern $g\theta$ is obtained by applying all variable replacements x_i/f_i in θ to g simultaneously. We formally define $g\theta$ as follows. Let $X=\{x_1,x_2,\dots,x_n\}$. For each $h\in H_g(\{x_i\})$ $(1\leq i\leq n)$, we make a copy of f_i, denoted by $f_{x_i,h}$. Note that if $h\in H_g(\{x_i\})$ $(1\leq i\leq n)$, $f_{x_i,h}$ is the same as $f_{\varphi_g(h),h}$.

$$\begin{aligned} g\theta = {} & (g - H_g(X)) \\ & \cup \bigcup_{h\in H_g(X)} f_{\varphi_g(h),h} \\ & \cup \bigcup_{h\in H_g(X)} \bigcup_{u\in N_g(h)\setminus H_g(X)} f_{\varphi_g(h),h} \textcircled{1} u \\ & \cup \bigcup_{h\in H_g(X)} \bigcup_{h'\in N_g(h)\cap H_g(X)} f_{\varphi_g(h),h} \textcircled{1} f_{\varphi_g(h'),h'}. \end{aligned}$$

For example, the cograph G_1 in Fig. 1 is obtained from g in Fig. 2 by replacing x and z with F_3 and F_1, respectively. That is, $G_1\cong g\{x/F_3,z/F_1\}$. Also $G_2\cong g\{x/F_3,z/F_2\}$ and $G_3\cong g\{x/F_2,z/F_3\}$.

We say that a cograph pattern g in $\mathcal{CGP}(\Sigma,\mathcal{X})$ is *linear* if all variables in g have mutually distinct variable labels in $\mathcal{X}$. $\mathcal{LCGP}(\Sigma,\mathcal{X})$ denotes the set of all linear cograph patterns whose vertices are labeled with elements in $\Sigma\cup\mathcal{X}$.

Definition 3 (Cograph pattern language): Let $\mathcal{P}$ be a subset of $\mathcal{CGP}(\Sigma,\mathcal{X})$. Let g be a cograph pattern in $\mathcal{P}$. The *cograph pattern language* of g, denoted by $L(g)$, is defined as the set $\{G\in\mathcal{CG}(\Sigma)\mid G\cong g\theta$ for some substitution $\theta\}$. The class of all cograph pattern languages of $\mathcal{P}$ is defined as $\mathcal{L}_{\mathcal{P}}=\{L(g)\mid g\in\mathcal{P}\}$.

3. Inductive Inference of Cograph Pattern Languages

In this section, we formally define the membership problem and the minimal language problem for cograph pattern languages. In Sect. 4 and Sect. 5, we will discuss these problems in detail. Here, we summarize the results of this paper.

Angluin [2] and Shinohara [12] showed that if a class of languages C has finite thickness, and the membership problem and the minimal language problem for C are solvable in polynomial time, then C is polynomial time inductively inferable from positive data. We consider the class $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$ as a target of inductive inference.

For a set S, $|S|$ denotes the number of elements in S. A class $C\subseteq\mathcal{L}_{\mathcal{CGP}(\Sigma,\mathcal{X})}$ is said to have *finite thickness* if, for any nonempty finite set $S\subseteq\mathcal{CG}(\Sigma)$, the number of cograph pattern languages in C that contain S is finite, i.e., $|\{L\in C\mid S\subseteq L\}|<\infty$.

Lemma 1: The class $\mathcal{L}_{\mathcal{CGP}(\Sigma,\mathcal{X})}$ has finite thickness.

Proof. It is sufficient to prove that for any nonempty finite set $S\subseteq\mathcal{CG}(\Sigma)$, the cardinality of the set $\{g\in\mathcal{CGP}(\Sigma,\mathcal{X})\mid S\subseteq L(g)\}$ is finite. For any cograph pattern g and any substitution θ such that $g\theta$ has no variable, $|V(g)|\leq|V(g\theta)|$ holds. Therefore, if $S\subseteq L(g)$ for a cograph pattern g, then $|V(g)|\leq\min\{|V(G)|\mid G\in S\}$. The number of vertex labels in Σ of g is equal to or less than $\min\{|V(G)|\mid G\in S\}$. Furthermore, since any two isomorphic cograph patterns define the same cograph pattern language, the number of variable labels in $\mathcal{X}$ that are needed for defining the language is equal to or less than $\min\{|V(G)|\mid G\in S\}$. Hence the number of cograph patterns g in $\mathcal{CGP}(\Sigma,\mathcal{X})$ such that $S\subseteq L(g)$ is finite, that is, $\mathcal{L}_{\mathcal{CGP}(\Sigma,\mathcal{X})}$ has finite thickness. □

Since $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}\subsetneq\mathcal{L}_{\mathcal{CGP}(\Sigma,\mathcal{X})}$, from Lemma 1, we have the next corollary.

Corollary 1: The class $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$ has finite thickness.

For a subset $\mathcal{P}\subseteq\mathcal{CGP}(\Sigma,\mathcal{X})$, the membership problem for $\mathcal{L}_{\mathcal{P}}$ is formally defined as follows.

Membership Problem for $\mathcal{L}_{\mathcal{P}}$
Instance: A cograph pattern $g\in\mathcal{P}$ and a cograph $G\in\mathcal{CG}(\Sigma)$.
Question: Does $L(g)$ contain G?

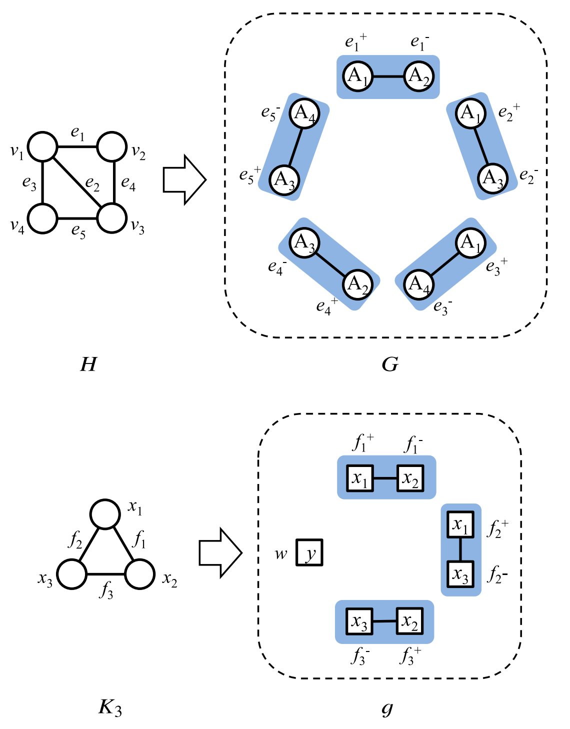
Unfortunately, we have the next theorem.

Theorem 1: Membership Problem for $\mathcal{L}_{\mathcal{CGP}(\Sigma,\mathcal{X})}$ is NP-complete.

Proof. It is obvious that Membership Problem for $\mathcal{L}_{\mathcal{CGP}(\Sigma,\mathcal{X})}$ is in NP. We will reduce CLIQUE, i.e., the problem of deciding whether or not an unlabeled graph H has a clique of size k, to this problem. Without loss of generality, we can assume that H has no isolated vertex. The idea of the reduction is similar to the proof of NP-completeness of the unordered tree pattern matching problem in [1].

Let $V(H)=\{v_1,v_2,\dots,v_n\}$ $(n\geq 1)$ and $E(H)=\{e_1,e_2,\dots,e_m\}$ $(m\geq 1)$. Let K_k be the clique of size k. Let $V(K_k)=\{u_1,u_2,\dots,u_k\}$ and $E(K_k)=\{f_1,f_2,\dots,f_{k'}\}$ where $k'=\frac{k(k-1)}{2}$. Let $\Sigma=\{A_1,A_2,\dots,A_n\}$ and $\mathcal{X}=\{x_1,x_2,\dots,x_k,y,\dots\}$, where $y\notin\{x_1,x_2,\dots,x_k\}$.

First, we construct a graph G with vertex labeling $\varphi_G: V(G)\to\Sigma$ as follows. For each $e\in E(H)$, we use e^+,e^- as vertices of G. By using them, let $V(G)=\{e_1^+,e_1^-,e_2^+,e_2^-,\dots,e_m^+,e_m^-\}$. If $e_\ell=\{v_i,v_j\}$ $(1\leq\ell\leq m,\ 1\leq i<j\leq n)$, let $\varphi_G(e_\ell^+)=v_i$ and $\varphi_G(e_\ell^-)=v_j$. We define G as $(e_1^+\textcircled{1}e_1^-)\textcircled{0}(e_2^+\textcircled{1}e_2^-)\textcircled{0}\cdots\textcircled{0}(e_m^+\textcircled{1}e_m^-)$. Next, we construct a cograph pattern g with a vertex labeling $\varphi_g: V(g)\to\Sigma\cup\mathcal{X}$ as follows. For each edge $f\in E(K_k)$, we use f^+,f^- as vertices of g. Furthermore, we use w as another vertex of g. Let $V(g)=\{f_1^+,f_1^-,f_2^+,f_2^-,\dots,f_{k'}^+,f_{k'}^-\}\cup\{w\}$. If $f_\ell=\{u_i,u_j\}$ $(1\leq\ell\leq k',\ 1\leq i<j\leq k)$, let $\varphi_g(f_\ell^+)=x_i$ and $\varphi_g(f_\ell^-)=x_j$. Furthermore, let $\varphi_g(w)=y$. We define g as $(f_1^+\textcircled{1}f_1^-)\textcircled{0}(f_2^+\textcircled{1}f_2^-)\textcircled{0}\cdots\textcircled{0}(f_{k'}^+\textcircled{1}f_{k'}^-)\textcircled{0}w$. We give an example of this reduction in Fig. 4. Since $|V(G)|+|E(G)|=$



v_1 v_2 v_3 v_4
e_1 e_2 e_3 e_4 e_5
e_1^+ e_1^- e_2^+ e_2^- e_3^+ e_3^- e_4^+ e_4^- e_5^+ e_5^-
A_1 A_2 A_3 A_4
H
G
x_1 x_2 x_3
f_1 f_2 f_3
f_1^+ f_1^- f_2^+ f_2^- f_3^+ f_3^-
w y
K_3
g

Fig. 4 A reduction of the NP-completeness proof of Th. 1. The left graph H and clique of size 3 are transformed to the right cograph G and cograph pattern g.

$O(m)$ and $|V(K_k)| + |E(K_k)| = O(k^2)$, we compute this transformation in polynomial time w.r.t. m and k.

We assume that there is a subgraph C of H such that $C \cong K_k$ holds. Let $V(C) = \{v_{p_1}, v_{p_2}, \ldots, v_{p_k}\}$ $(1 \leq p_1 < p_2 < \cdots < p_k \leq n)$. Let $a_1, a_2, \ldots, a_k$ be new k vertices, where the label of a_ℓ is A_{p_ℓ} $(1 \leq \ell \leq k)$. We define a substitution θ as $\{x_1/a_1, x_2/a_2, \ldots, x_k/a_k, y/G'\}$ where $G' = G[\{e^+, e^- \in V(G) \mid e \in E(H) \setminus E(C)\}]$. By this substitution θ, we have $g\theta \cong G$. Thus, $L(g)$ contains G. Conversely, we assume that $L(g)$ contains G, i.e., there is a substitution θ such that $g\theta \cong G$ holds. If the graph with which x_i $(1 \leq i \leq k)$ is replaced has more than one vertex, $g\theta$ has a vertex of degree at least two. It contradicts that G has no vertex of degree more than one. Therefore, the graph with which x_i is replaced is a single vertex graph. Let $A_{p_1}, A_{p_2}, \ldots, A_{p_k}$ $(1 \leq p_1 < p_2 < \cdots < p_k \leq n)$ be the vertex labels of the single vertex graphs with which $x_1, x_2, \ldots, x_k$ are replaced, respectively. Let $V' = \{v_{p_1}, v_{p_2}, \ldots, v_{p_k}\}$. From the constructions of G and g, for any pair of labels A_{p_i} and A_{p_j} $(i \neq j)$, two vertices v_{p_i} and v_{p_j} are adjacent. Thus, $H[V'] \cong K_k$ holds, i.e., H has a clique of size k. □

Below we consider the linear cograph patterns only.

Theorem 2: Membership Problem for $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$ is solvable in polynomial time.

In Sect. 4, we will prove Theorem 2 by presenting a polynomial time algorithm for solving Membership Problem for $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$.

Let $\mathcal{P}$ be a subset of $\mathcal{CGP}(\Sigma, \mathcal{X})$. A *minimally generalized cograph pattern w.r.t. $\mathcal{P}$ explaining* a given set of graphs $S \subsetneq \mathcal{CG}(\Sigma)$ is a cograph pattern $g \in \mathcal{P}$ such that $S \subsetneq L(g)$ and there is no cograph pattern $g' \in \mathcal{P}$ with $S \subseteq L(g') \subsetneq L(g)$. The minimal language problem for $\mathcal{L}_{\mathcal{P}}$ is defined as follows.

Minimal Language (MINL) Problem for $\mathcal{L}_{\mathcal{P}}$
Instance: A nonempty finite set of cographs $S \subsetneq \mathcal{CG}(\Sigma)$.
Question: Find a minimally generalized cograph pattern w.r.t. $\mathcal{P}$ explaining S.

In Sect. 5, we will prove Theorem 3 by presenting a polynomial time algorithm for solving MINL Problem for $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$.

Theorem 3: MINL Problem for $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$ is solvable in polynomial time.

From the results of Angluin [2] and Shinohara [12] and Corollary 1, Theorems 2 and 3, we have the following main result.

Theorem 4: The class $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$ is polynomial time inductively inferable from positive data.

4. Polynomial Time Algorithm for Solving the Membership Problem for Linear Cograph Patterns

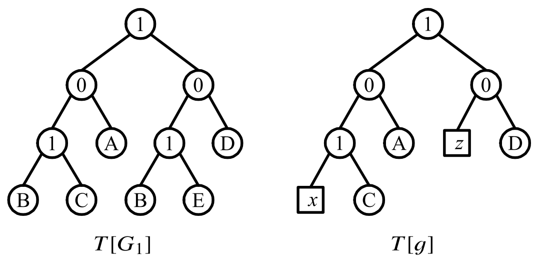
In this section, we present a polynomial time algorithm for solving Membership Problem for $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$ by giving a pattern matching algorithm for linear cotree patterns, which are tree representations of parse structures of cograph patterns.

4.1 Polynomial Time Matching Algorithm for Linear Cotree Patterns

A cotree pattern is defined as follows. Below, a vertex of any tree representation is called a *node*.

Definition 4 (Cotree pattern): A *cotree pattern* is a node-labeled unordered tree in which the internal nodes are labeled with ⓪ or ①. The leaves of a cotree pattern are the vertices and variables of the corresponding cograph pattern. A subtree rooted at a node labeled with ⓪ (disjoint union operation) corresponds to the cograph pattern $g_1$⓪$g_2$⓪$\cdots$⓪g_n of the subgraphs $g_1, g_2, \ldots, g_n$ defined by the subtrees rooted at the children. A subtree rooted at a node labeled with ① (join operation) corresponds to the cograph pattern $g_1$①$g_2$①$\cdots$①g_n of the subgraphs $g_1, g_2, \ldots, g_n$ defined by the subtrees rooted at the children.

An internal node of a cotree pattern labeled with ⓪ (resp., ①) is called a ⓪*-node* (resp., ①*-node*). A leaf labeled with an element in Σ is called a Σ*-node*. A leaf labeled with an element in $\mathcal{X}$ is called an $\mathcal{X}$*-node*. A *cotree* is a cotree pattern with no $\mathcal{X}$-node. We say that a cotree pattern t is *linear* if for each variable $x \in \mathcal{X}$, the number of $\mathcal{X}$-nodes of t labeled with x is at most one. In this paper, we deal with



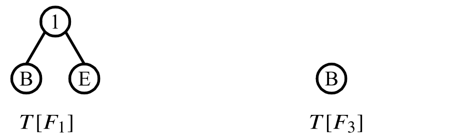


Fig. 5 $T[G_1]$, $T[g]$, $T[F_1]$, and $T[F_3]$ are linear cotree patterns of the linear cograph patterns G_1, g, F_1 and F_3 in Fig. 1, respectively. Square boxes describe $\mathcal{X}$-nodes of linear cotree patterns.

linear cotree patterns only. The set of all linear cotree patterns is denoted by $\mathcal{LCTP}(\Sigma, \mathcal{X})$, and the set of all cotrees is denoted by $\mathcal{CT}(\Sigma)$.

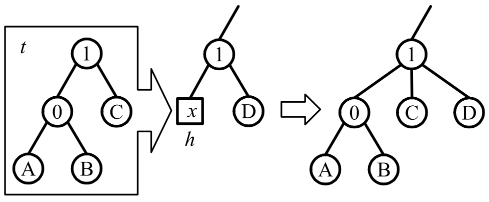
For a node v, we denote the depth of v by $d(v)$ and the parent of v by $p(v)$. We denote the label of v by $\lambda(v)$ and the number of children of v by $ch(v)$. For a cotree or a cotree pattern t, the vertex and edge sets of t are denoted by $V(t)$ and $E(t)$, respectively. For two cotree patterns s and t, s and t are *isomorphic*, denoted by $s \equiv t$, if a bijection $\psi : V(s) \rightarrow V(t)$ exists such that (1) the root of s is mapped to the root of t by ψ, (2) for any node $v \in V(s)$ that is not an $\mathcal{X}$-node, $\lambda(\psi(v)) = \lambda(v)$, (3) $\{u, v\} \in E(s)$ if and only if $\{\psi(u), \psi(v)\} \in E(t)$, and (4) for any $\mathcal{X}$-nodes $v, v' \in V(s)$, $\lambda(v) = \lambda(v')$ if and only if $\lambda(\psi(v)) = \lambda(\psi(v'))$.

Since a cotree is a unique representation of a cograph [5], we have the following proposition.

Proposition 3 ([5]): The cotree pattern for a cograph pattern in $\mathcal{CGP}(\Sigma, \mathcal{X})$ is unique up to isomorphism.

For a cograph pattern $g \in \mathcal{CGP}(\Sigma, \mathcal{X})$, $T[g]$ denotes the cotree pattern for g. A naive algorithm for constructing the cotree pattern $T[g]$ from a given cograph g is as follows. (1) If g is either a single vertex or a single variable, then the cotree is a single Σ-node or $\mathcal{X}$-node corresponding to g. (2) If g is disconnected, then make the ⓪-node the root and continue recursively on each connected component. (3) If g is connected, then make the ①-node the root, form $\bar{g} = g_1 ⓪ g_2 ⓪ \cdots ⓪ g_k$ $(k > 1)$, where the g_i's $(1 \le i \le k)$ are the connected components of $\bar{g}$, and continue recursively on each $\bar{g}_i$. For example, $T[G_1]$, $T[g]$, $T[F_1]$, and $T[F_3]$ in Fig. 5 are the linear cotree patterns of the linear cograph patterns G_1, g, F_1, and F_3 in Fig. 1. We can construct the cotree pattern $T[f]$ for a cograph pattern f in linear time, by applying the linear time algorithm in [6] to f.

Let g be a cograph pattern in $\mathcal{CGP}(\Sigma, \mathcal{X})$. For two leaves u, v of $T[g]$, the *lowest common ancestor* of u and v, denoted by $lca_{T[g]}(u, v)$, is the node that is the farthest from the root of $T[g]$ among the common ancestors of u and v.



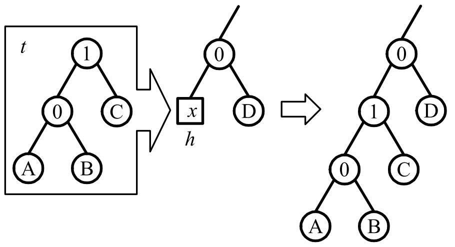


Fig. 6 Two cases of the bindings. Let x/t be a binding and h an $\mathcal{X}$-node of label x. Ex. 1 shows the case that the label of the root of t is equal to that of $p(h)$. Ex. 2 shows the other case.

For a leaf u of $T[g]$, the corresponding vertex of u of g is denoted by the same symbol u.

Proposition 4 ([5]): Let g be a cograph pattern in $\mathcal{CGP}(\Sigma, \mathcal{X})$. For two leaves u, v of $T[g]$, $lca_{T[g]}(u, v)$ is a ①-node if and only if u and v are adjacent in g.

Let s be a cotree pattern and h an $\mathcal{X}$-node of s with variable label $x \in \mathcal{X}$. Let t be a cotree pattern in $\mathcal{LCTP}(\Sigma, \mathcal{X})$ having r as its root. Then the form x/t is called a *binding* for x. A new cotree pattern $s\{x/t\}$ can be obtained by applying the binding x/t to s in the following way. (1) If $\lambda(r) = \lambda(p(h))$, then remove h and identify r with $p(h)$ (See Ex. 1 in Fig. 6). (2) Otherwise, remove h and connect r directly to $p(h)$ (See Ex. 2 in Fig. 6). A *substitution* $\theta = \{x_1/t_1, \ldots, x_n/t_n\}$ is a finite collection of bindings such that for any i, j $(1 \le i < j \le n)$, the variable labels x_i and x_j are distinct. The cotree pattern $s\theta$ is obtained by simultaneously applying all bindings in θ to s. For example, in Fig. 5, the cotree $T[G_1]$ is obtained from $T[g]$ by substituting $T[F_3]$ and $T[F_1]$ for x and z, respectively. That is, $T[G_1] \equiv T[g]\{x/T[F_3], z/T[F_1]\}$.

For a cotree T and a cotree pattern t, t is said to *match* T if there exists a substitution θ such that $T \equiv t\theta$. The matching problem for $\mathcal{LCTP}(\Sigma, \mathcal{X})$ is defined as follows.

Matching Problem for $\mathcal{LCTP}(\Sigma, \mathcal{X})$
Instance: A cotree pattern $t \in \mathcal{LCTP}(\Sigma, \mathcal{X})$ and a cotree $T \in \mathcal{CT}(\Sigma)$.
Question: Does t match T?

In Procedure 1, we present a procedure, called Matching-$\mathcal{LCTP}$, for solving Matching Problem for $\mathcal{LCTP}(\Sigma, \mathcal{X})$. For a cotree or a cotree pattern t and its node u, $t[u]$ denotes the rooted subtree of t induced by the descendants of u. Note that u is a descendant of itself.

Procedure 1 MATCHING-$\mathcal{LCTP}(t, T)$

```
Input: t: a cotree pattern in $\mathcal{LCTP}(\Sigma, \mathcal{X})$, T: a cotree in $\mathcal{CT}(\Sigma)$
Output: "yes" or "no"
 1: Let r and R be the roots of t and T, respectively
 2: for d :=the height of t to 0 do
 3:     for each node u of t such that $d(u) = d$ do
 4:         $CS(u) := \emptyset$
 5:         if u is a $\Sigma$-node of t then
 6:             for each leaf $\ell$ of T such that $d(\ell) = d$ and $\lambda(u) = \lambda(\ell)$ do
 7:                 $CS(u) := CS(u) \cup \{\ell\}$
 8:             end for
 9:         end if
10:         if u is an $\mathcal{X}$-node of t then
11:             for each node v of T with $d(v) = d$ do
12:                 $CS(u) := CS(u) \cup \{v\}$
13:             end for
14:         end if
15:         if u is an internal node of t then
16:             $CS(u)$ :=INODE-CSSET(u, T) (Procedure 2)
17:         end if
18:     end for
19: end for
20: if $R \in CS(r)$ then
21:     return "yes"
22: else
23:     return "no"
24: end if
```

Procedure 2 INODE-CSSET(u, T)

```
Input: u: an internal node of a cotree pattern t in $\mathcal{LCTP}(\Sigma, \mathcal{X})$, T: a cotree in $\mathcal{CT}(\Sigma)$
Output: CS: a correspondence set
 1: $CS := \emptyset$
 2: Let $C_u = \{u_1, \dots, u_{ch(u)}\}$ be the set of children of u
 3: for each internal node v of T such that $d(u) = d(v)$ and $\lambda(u) = \lambda(v)$ do
 4:     Let $C_v = \{v_1, \dots, v_{ch(v)}\}$ be the set of children of v
 5:     Construct a bipartite graph $G = (X, Y, E)$, where $X = C_u$, $Y = C_v$,
        and $E = \{\{u_i, v_j\} \mid 1 \le i \le ch(u), 1 \le j \le ch(v), v_j \in CS(u_i)\}$
 6:     Let m be the size of maximum bipartite graph matching in G
 7:     if there is a child u' of u such that u' is an $\mathcal{X}$-node then
 8:         if $m = ch(u)$ then
 9:             $CS := CS \cup \{v\}$
10:         end if
11:     else if $m = ch(u) = ch(v)$ then
12:         $CS := CS \cup \{v\}$
13:     end if
14: end for
15: return CS
```

Definition 5: Let t be a cotree pattern in $\mathcal{LCTP}(\Sigma, \mathcal{X})$ and T a cotree in $\mathcal{CT}(\Sigma)$. The correspondence set of a node $u \in V(t)$, denoted by $CS(u)$, is defined as $\{v \in V(T) \mid t[u]$ matches $T[v]$ and $d(u) = d(v)\}$.

Procedure MATCHING-$\mathcal{LCTP}$ (Procedure 1) computes $CS(u)$ for each node u of a given cotree pattern t by using $CS(c_1), \dots, CS(c_{ch(u)})$ where $c_1, \dots, c_{ch(u)}$ are all children of u. The procedure assigns a correspondence set to each node of t and terminates when a correspondence set is assigned to the root of t. From the definition of a correspondence set, $CS(r)$ contains the root of T if and only if t matches T, where r is the root of t.

Lemma 2: Given a cotree pattern $t \in \mathcal{LCTP}(\Sigma, \mathcal{X})$ and a cotree $T \in \mathcal{CT}(\Sigma)$, MATCHING-$\mathcal{LCTP}$ correctly computes the correspondence sets of all nodes in $V(t)$.

Proof. For each $u \in V(t)$, let $CS(u)$ be a set of nodes of T in MATCHING-$\mathcal{LCTP}$. According to Definition 5, we will prove that when MATCHING-$\mathcal{LCTP}$ terminates, for any $u \in V(t)$, $v \in CS(u)$ if and only if $t[u]$ matches $T[v]$ and $d(u) = d(v)$. This is shown by backward induction on the depth of a node u of t.

Basis: When $d(u)$ is equal to the height of t, u is either a Σ-node or an $\mathcal{X}$-node.

If u is a Σ-node, at the lines 5–9 of MATCHING-$\mathcal{LCTP}$, $CS(u)$ is computed as the set of all leaves ℓ of T such that $d(u) = d(\ell)$ and $\lambda(u) = \lambda(\ell)$. If $\ell \in CS(u)$, since both $t[u]$ and $T[\ell]$ consist of one Σ-node only and $\lambda(u) = \lambda(\ell)$, $t[u] \equiv T[\ell]$ holds, i.e., $t[u]$ matches $T[\ell]$ and $d(u) = d(\ell)$. Conversely, for any $v \in V(T)$, if $t[u]$ matches $T[v]$ and $d(u) = d(v)$, since $t[u]$ consists of one Σ-node, v must be a leaf of T with $\lambda(u) = \lambda(v)$. Therefore $v \in CS(u)$ holds.

If u is an $\mathcal{X}$-node, at the lines 10–14 of MATCHING-$\mathcal{LCTP}$, $CS(u)$ is computed as the set of all nodes v of T with $d(u) = d(v)$. Since $t[u]$ consists of one $\mathcal{X}$-node only, for any cotree T, $t[u]\{x/T'\} \equiv T$ holds, where x is the variable label of u and T' is a copy of T. Thus, if $v \in CS(u)$, then $t[u]$ matches $T[v]$ and $d(u) = d(v)$. The converse is obvious.

Inductive Step: We assume that for all $u' \in V(t)$ with $d(u') > d(u)$, $v \in CS(u')$ if and only if $t[u']$ matches $T[v]$ and $d(u') = d(v)$. The node u is either a Σ-node or an $\mathcal{X}$-node or an internal node (i.e., a ⓪-node or a ①-node). If u is either a Σ-node or an $\mathcal{X}$-node, the proof is the same as Basis. Then we will prove that when u is an internal node, $v \in CS(u)$ if and only if $t[u]$ matches $T[v]$ and $d(u) = d(v)$.

If u is an internal node, at the lines 15–17 of MATCHING-$\mathcal{LCTP}$, $CS(u)$ is computed as the set of all internal nodes v of T satisfying the following conditions.

1. $d(u) = d(v)$ and $\lambda(u) = \lambda(v)$.
2. If there is a child w of u such that w is an $\mathcal{X}$-node, then $ch(v) \ge ch(u)$, otherwise $ch(v) = ch(u)$.
3. Let $u_1, \dots, u_{ch(u)}$ and $v_1, \dots, v_{ch(v)}$ be the children of u and v, respectively. Then there is an injection $\xi : \{u_1, \dots, u_{ch(u)}\} \to \{v_1, \dots, v_{ch(v)}\}$ such that $\xi(u_i) \in CS(u_i)$ for all i $(1 \le i \le ch(u))$.

The condition 3 is decided by computing the maximum graph matching for a bipartite graph $B = (U, V, E)$ where $U = \{u_1, \dots, u_{ch(u)}\}$, $V = \{v_1, \dots, v_{ch(v)}\}$, and $E = \{(u_i, v_j) \mid 1 \le i \le ch(u), 1 \le j \le ch(v), v_j \in CS(u_i)\}$ in INODE-CSSET (Procedure 2). The size of maximum graph matching for B is equal to $ch(u)$ if and only if the condition 3 is satisfied.

Let k be the number of $\mathcal{X}$-nodes in $u_1, \dots, u_{ch(u)}$. We will prove the statement when $k > 1$ and $ch(v) > ch(u)$. The other cases are easy or similar. Without loss of generality, we suppose that $u_1, \dots, u_k$ are $\mathcal{X}$-nodes and $u_{k+1}, \dots, u_{ch(u)}$ are not an $\mathcal{X}$-node. Let $x_1, \dots, x_k$ be the variable labels of $u_1, \dots, u_k$, respectively.

For any $v \in V(T)$, if $v \in CS(u)$, then $d(u) = d(v)$ and there is an injection $\xi : \{u_1, \dots, u_{ch(u)}\} \to \{v_1, \dots, v_{ch(v)}\}$ such that $\xi(u_i) \in CS(u_i)$ for all i $(1 \leq i \leq ch(u))$. Without loss of generality, we suppose that $\xi(u_1) = v_1, \ldots, \xi(u_{ch(u)}) = v_{ch(u)}$. Let $T'_1, \ldots, T'_{k-1}$ be copies of $T[v_1], \ldots, T[v_{k-1}]$, respectively, and let T'_k be a copy of the subtree of $T[v]$ induced by $\{v\} \cup V(T[v_k]) \cup V(T[v_{ch(u)+1}]) \cup \cdots \cup V(T[v_{ch(v)}])$. From the induction hypothesis and the definition of binding, $t[u]\{x_1/T'_1, \ldots, x_k/T'_k\} \equiv T[v]$ holds, i.e., $t[v]$ matches $T[v]$.

Conversely, for any $v \in V(T)$, we assume that $t[u]$ matches $T[v]$ and $d(u) = d(v)$. Then there is a substitution $\theta = \{x_1/T'_1, \ldots, x_k/T'_k\}$ such that $t[u]\theta \equiv T[v]$ holds. Let ψ be an isomorphism from $t[u]\theta$ to $T[v]$. By using ψ, we define a function $\xi : \{u_1, \ldots, u_{ch(u)}\} \to \{v_1, \ldots, v_{ch(v)}\}$ as follows. According to the label of u, we have two cases. We assume that u is a ⓪-node. For any u_i $(1 \leq i \leq ch(u))$, if $1 \leq i \leq k$ and the root of T'_i is a ⓪-node, from the definition of binding, the children of the root of T'_i are mapped into $\{v_1, \ldots, v_{ch(v)}\}$ by ψ. Then we define $\xi(u_i)$ as one of the nodes in $\{\psi(w) \in V(T) \mid w \text{ is a child of the root of } T'_i\}$. Thus $d(u_i) = d(\xi(u_i)) > d(u)$ holds. Since u_i is an $\mathcal{X}$-node, $t[u_i]$ matches $T[\xi(u_i)]$. For any u_i $(1 \leq i \leq ch(u))$, if $k+1 \leq i \leq ch(u)$ or the root of T'_i is a ①-node, since u_i is a child of the root of $t[u]\theta$, we define $\xi(u_i) = \psi(u_i)$. Since $t[u]\theta$ is isomorphic to $T[v]$ by the isomorphism ψ, $t[u_i]$ matches $T[\xi(u_i)]$. In both cases, from the induction hypothesis, $\xi(u_i) \in CS(u_i)$ holds. Since ξ defined above is an injection from $\{u_1, \ldots, u_{ch(u)}\}$ to $\{v_1, \ldots, v_{ch(v)}\}$, the conditions 1–3 hold, and then we have $v \in CS(u)$. When u is a ①-node, in a similar way, we show that $v \in CS(u)$ holds.

Finally we conclude that for any $u \in V(t)$, $v \in CS(u)$ if and only if $t[u]$ matches $T[v]$ and $d(u) = d(v)$. □

Lemma 3: Given a cotree pattern $t \in \mathcal{LCTP}(\Sigma, \mathcal{X})$ and a cotree $T \in \mathcal{CT}(\Sigma)$, Matching Problem for $\mathcal{LCTP}(\Sigma, \mathcal{X})$ is solvable in $O(nN^{1.5})$ time, where $n = |V(t)|$ and $N = |V(T)|$.

Proof. From Lemma 2, Matching-$\mathcal{LCTP}$ correctly computes the correspondence sets of all nodes in $V(t)$. Here, we show the time complexity of Matching-$\mathcal{LCTP}$. Let n_i and N_i be the numbers of nodes of depth i of t and T, respectively. For a node $u \in V(t)$ of depth i, if u is either a Σ-node or an $\mathcal{X}$-node, lines 5–14 of Matching-$\mathcal{LCTP}$ work in $O(N_i)$ time to compute the set $CS(u)$. If u is an internal node, we construct a bipartite graph and compute a maximum graph matching of it. Hopcroft and Karp [7] presented a maximum graph matching algorithm which runs in $O(|E(G)|\sqrt{|V(G)|})$ time for a given bipartite graph G. By using their algorithm, we need $O(ch(u)ch(v)\sqrt{ch(u)+ch(v)})$ time to decide whether or not an internal node $v \in V(T)$ is in $CS(u)$. Let $K_{i,max} = \max\{ch(v) \mid v \text{ is an internal node of depth } i \text{ in } V(T)\}$. Accordingly, the time complexity of Inode-CSset is $O(ch(u)N_{i+1}\sqrt{K_{i,max}})$. Therefore, we need $O(n_{i+1}N_{i+1}\sqrt{K_{i,max}}) + O(N_i)$ time to compute correspondence sets of all nodes of depth i of t. Let d be the height of t. Since a node of depth d of t is either a Σ-node or an $\mathcal{X}$-node, the total time for computing $CS(u)$ for all nodes $u \in V(t)$ is $O(\sum_{i=0}^{d-1}(n_{i+1}N_{i+1}\sqrt{K_{i,max}}+N_i))$ time. Since $\sum_{i=0}^{d} n_i = n$, $\sum_{i=0}^{d} N_i \leq N$, $\max\{K_{i,max} \mid 0 \leq i \leq d\} \leq N$, we need $O(nN^{1.5})$ time to compute the Matching Problem for $\mathcal{LCTP}(\Sigma, \mathcal{X})$. □

Algorithm 3 Matching-$\mathcal{LCGP}(g, G)$

```
Input: g: a cograph pattern in $\mathcal{LCGP}(\Sigma,\mathcal{X})$, G: a cograph in $\mathcal{CG}(\Sigma)$
Output: "yes" or "no"
1: if $|V(g)| > |V(G)|$ or $|E(g)| > |E(G)|$ then
2:     return "no"
3: end if
4: Construct a cotree pattern $T[g]$ of $g$
5: Construct a cotree $T[G]$ of $G$
6: return Matching-$\mathcal{LCTP}(T[g], T[G])$
```

4.2 Polynomial Time Matching Algorithm for Linear Cograph Patterns

In Algorithm 3, we give a polynomial time algorithm Matching-$\mathcal{LCGP}$ for solving Membership Problem for $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$, which calls Matching-$\mathcal{LCTP}$ as a procedure. Firstly, we will prove the following lemma.

Lemma 4: For a cograph pattern $g \in \mathcal{LCGP}(\Sigma, \mathcal{X})$ and a substitution $\theta = \{x_1/g_1, \ldots, x_n/g_n\}$, $T[g\theta] \equiv T[g]\theta_T$ holds, where $\theta_T = \{x_1/T[g_1], \ldots, x_n/T[g_n]\}$.

Proof. From Propositions 3 and 4, it is sufficient to show that the next equation holds for any two vertices u and v in $(V(g) \setminus H(g)) \cup V(g_1) \cup \cdots \cup V(g_n)$:

$$\lambda(lca_{T[g\theta]}(u,v)) = \lambda(lca_{T[g]\theta_T}(u,v)) \tag{1}$$

We have four cases.

1. $u, v \in V(g) \setminus H(g)$: It is easy to see that $\lambda(lca_{T[g\theta]}(u,v)) = \lambda(lca_{T[g]}(u,v)) = \lambda(lca_{T[g]\theta_T}(u,v))$.
2. $u, v \in V(g_i)$ for some i $(1 \leq i \leq n)$: Since $\{u,v\} \in E(g_i)$ if and only if $\{u,v\} \in E(g\theta)$, $\lambda(lca_{T[g\theta]}(u,v)) = \lambda(lca_{T[g_i]}(u,v))$ holds. From the definition of a binding, $\lambda(lca_{T[g_i]}(u,v)) = \lambda(lca_{T[g]\theta_T}(u,v))$ holds. This leads to Eq. (1).
3. $u \in V(g) \setminus H(g)$ and $v \in V(g_i)$ for some i $(1 \leq i \leq n)$: For a binding $x_i/T[g_i]$ in θ_T, let h_i be an $\mathcal{X}$-node with $\lambda(h_i) = x_i$. From the definition of a variable replacement, $\{u, h_i\} \in E(g)$ if and only if $\{u,v\} \in E(g\theta)$. Thus, $\lambda(lca_{T[g]}(u,h_i)) = \lambda(lca_{T[g\theta]}(u,v))$. We have $\lambda(lca_{T[g]}(u,h_i)) = \lambda(lca_{T[g]\theta_T}(u,v))$, since all leaves of $T[g_i]$ are descendants of $p(h_i)$. Accordingly, we get Eq. (1).
4. $u \in V(g_i)$ and $v \in V(g_j)$ for some i and j $(1 \leq i \neq j \leq n)$: For bindings $x_i/T[g_i], x_j/T[g_j]$ in θ_T, let h_i and h_j be $\mathcal{X}$-nodes with $\lambda(h_i) = x_i$ and $\lambda(h_j) = x_j$, respectively. From the definition of a variable replacement, $\{h_i, h_j\} \in E(g)$ if and only if $\{u,v\} \in E(g\theta)$. Thus, $\lambda(lca_{T[g\theta]}(u,v)) = \lambda(lca_{T[g]}(h_i,h_j))$. Since all leaves of $T[g_i]$ and $T[g_j]$ are descendants of $p(h_i)$ and $p(h_j)$, respectively, we have $\lambda(lca_{T[g]}(h_i,h_j)) = \lambda(lca_{T[g]\theta_T}(u,v))$. Accordingly, we get Eq. (1).

Therefore, we conclude that $T[g\theta] \equiv T[g]\theta_T$ holds. □

We can directly prove the following lemma from Lemma 4.

Lemma 5: For a cograph pattern $g \in \mathcal{LCGP}(\Sigma, \mathcal{X})$ and a cograph $G \in \mathcal{CG}(\Sigma)$, $L(g)$ contains G if and only if the cotree pattern $T[g]$ matches the cotree $T[G]$.

The following lemmas show the time complexity of Matching-$\mathcal{LCGP}$.

Theorem 5 ([6]): For a cograph $G = (V(G), E(G))$, $T[G]$ can be constructed from G in $O(|V(G)| + |E(G)|)$ time.

Lemma 6: Given a cograph pattern $g \in \mathcal{LCGP}(\Sigma, \mathcal{X})$ and a cograph $G \in \mathcal{CG}(\Sigma)$, Algorithm Matching-$\mathcal{LCGP}$ solves Membership Problem for $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$ in $O(nN^{1.5} + M)$ time, where $n = |V(g)|$, $N = |V(G)|$, and $M = |E(G)|$.

Proof. From Lemma 5, Matching-$\mathcal{LCGP}$ correctly decides whether or not $G \in L(g)$. From Theorem 5, we can construct $T[G]$ from G and $T[g]$ from g in $O(N + M)$ time. Since $|V(T[G])| = O(N)$ and $|V(T[g])| = O(n)$, by using Matching-$\mathcal{LCTP}(\pm, \mathcal{X})$ (Procedure 1), line 4 of Matching-$\mathcal{LCGP}$ is executed in $O(nN^{1.5})$ time from Lemma 3. Hence, Matching-$\mathcal{LCGP}$ decides whether or not $G \in L(g)$ in $O(nN^{1.5} + M)$ time. □

Hence, we have proven Theorem 2 in Sect. 3.

5. Polynomial Time Algorithm for Solving MINL Problem for Linear Cograph Patterns

In this section, we assume that $|\Sigma| = \infty$. For $g \in \mathcal{CGP}(\Sigma, \mathcal{X})$, $C(g)$ denotes the number of connected components of a cograph pattern $(V(g), E(g))$. For example, in Fig. 7, $C(g_1) = 1$ and $C(g_0) = 2$. For any vertex label $a \in \Sigma$, we denote by $G_a = (\{v\}, \emptyset)$ a *single-vertex cograph pattern* such that the label of v is a. We call a cograph pattern consisting of only one single variable a *single-variable cograph pattern*. We have the following proposition.

Proposition 5: Let θ be any substitution for $g \in \mathcal{LCGP}(\Sigma, \mathcal{X})$. Then $C(g\theta) \geq C(g)$. If there is no single-variable cograph pattern in the connected components of g, then $C(g\theta) = C(g)$.

Proof. It is sufficient to prove that $C(g\{x/f\}) \geq C(g)$ for any $x/f \in \theta$. Since g is linear, there is a unique variable h whose variable label is x. If there is a vertex $u \in V(g)$ such that $\{u, h\} \in E(g)$, from the definition of a variable replacement, $C(g\{x/f\}) = C(g)$ holds. If there is no vertex in $V(g)$ that is adjacent to h, $C(g\{x/f\}) = C(g - \{h\}) + C(f) \geq C(g)$ holds. Accordingly, $C(g\theta) \geq C(g)$ holds. □

Algorithm MINL-$\mathcal{LCGP}$ (Algorithm 4) solves MINL problem for $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$. Lines 5–15 extend a cograph pattern g by adding variables as much as possible while $S \subseteq L(g)$ holds (Fig. 7). Lines 18–25 try to replace each variable in g with a labeled vertex if it is possible.

Lemma 7: Let $g \in \mathcal{LCGP}(\Sigma, \mathcal{X})$ be the output of Algorithm MINL-$\mathcal{LCGP}(\pm, \mathcal{X})$ for an input S. Let g' be a cograph pattern in $\mathcal{LCGP}(\Sigma, \mathcal{X})$ satisfying $S \subseteq L(g') \subseteq L(g)$.

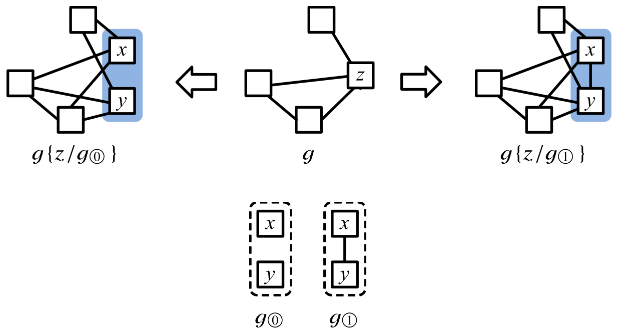


Fig. 7 Two refinement operators on Algorithm MINL-$\mathcal{LCGP}$ (Algorithm 4).

Algorithm 4 MINL-$\mathcal{LCGP}(S)$

```
Input: $S$: a set of cographs in $\mathcal{CG}(\Sigma)$
Output: $g$: a minimally generalized cograph pattern w.r.t. $\mathcal{LCGP}(\Sigma, \mathcal{X})$
    explaining $S$
 1: Let $g$ be a cograph pattern with one unmarked variable
 2: if $S$ contains both connected and unconnected cographs then
 3:     return $g$
 4: end if
 5: for unmarked variable $h$ in $g$ do
 6:     Let $x$ be the variable label of $h$
 7:     Let $g_{①}$ (resp., $g_{⓪}$) be a connected (resp., unconnected) cograph
        with two new unmarked variables (Fig. 7)
 8:     if Matching-$\mathcal{LCGP}(g\{x/g_{①}\}, G)$="yes" for any $G \in S$ then
 9:         $g := g\{x/g_{①}\}$
10:     else if Matching-$\mathcal{LCGP}(g\{x/g_{⓪}\}, G)$="yes" for any $G \in S$ then
11:         $g := g\{x/g_{⓪}\}$
12:     else
13:         mark $h$
14:     end if
15: end for
16: Unmark all variables of $g$
17: Let $\Sigma(S)$ be the set of all labels in $\Sigma$ that appear in $S$
18: for each unmarked variable $h$ in $g$ do
19:     Let $x$ be the variable label of $h$
20:     if there is a label $a \in \Sigma(S)$ such that Matching-$\mathcal{LCGP}(g\{x/G_a\}, G)$
        ="yes" for any $G \in S$, where $G_a = (\{v\}, \emptyset, \emptyset)$ such that the label of $v$
        is $a$ then
21:         $g := g\{x/G_a\}$
22:     else
23:         mark $h$
24:     end if
25: end for
26: return $g$
```

Then, $g' \cong g$.

Proof. It is easy to see that $|V(g')| \geq |V(g)|$, since if it is not the case, $L(g)$ does not contain the cograph obtained from g' by replacing all variables in g' with single vertices. Let $m = \min_{G \in S} C(G)$.

Claim 1. $C(g) = m$.
Proof of Claim 1. From Proposition 5, if $C(g) > m$, $L(g)$ cannot contain a cograph $G \in S$ with $C(G) = m$. If $C(g) < m$, g must contain a single-variable cograph pattern as a connected component. This contradicts the fact that line 7 of MINL-$\mathcal{LCGP}(\pm, \mathcal{X})$ increases $C(g)$ by $g_{⓪}$. (*End of Proof of Claim* 1)

Claim 2. $C(g') = C(g) = m$.
Proof of Claim 2. It is straightforwardly proven from Proposition 5 and Claim 1. (*End of Proof of Claim* 2)

Claim 3. Let $g_1, \ldots, g_m$ be the connected components of g, and $g'_1, \ldots, g'_m$ those of g'. There is a permutation π of $(1, \ldots, m)$ such that $L(g'_i) \subsetneqq L(g_{\pi(i)})$ for all i $(1 \le i \le m)$.
Proof of Claim 3. Since $|\Sigma| = \infty$, there is a vertex label c that does not appear in S. Let σ' be a substitution $\bigcup_{x' \in \mathcal{X}(g')}\{x'/G_c\}$ for g'. Since $L(g') \subsetneqq L(g)$, there is a permutation π of $(1, \ldots, m)$ such that $g'_i\sigma' \in L(g_{\pi(i)})$ for all i $(1 \le i \le m)$. Therefore, there is a substitution $\sigma = \bigcup_{x \in \mathcal{X}(g)}\{x/F_x\}$ where each F_x is a (possibly single-vertex) cograph such that $g'_i\sigma' \cong g_{\pi(i)}\sigma$. Since there is no occurrence of c in $g_{\pi(i)}$, all c's must appear in F_x's. Let $f_x = (V(F_x), E(F_x))$ and $H(f_x) = \{v \in V(F_x) \mid \varphi(v) = c\}$. We assume that the variables in $H(f_x)$ of f_x are labeled with mutually distinct new variable labels in $\mathcal{X}$. Then we see that $g'_i \cong g_{\pi(i)}\rho$ holds, where $\rho = \bigcup_{x \in \mathcal{X}(g)}\{x/f_x\}$. Therefore, $L(g'_i) \subsetneqq L(g_{\pi(i)})$ for all i $(1 \le i \le m)$. (*End of Proof of Claim* 3)

Claim 4. Let π be the permutation of Claim 3. Then $g'_i \cong g_{\pi(i)}$ holds for any i $(1 \le i \le m)$.
Proof of Claim 4. We consider variable replacements x/f_x in ρ of Claim 3. If $|V(f_x)| = 1$ for all $x \in \mathcal{X}(g)$, then $g'_i \cong g_{\pi(i)}$ $(1 \le i \le m)$ holds. Otherwise, there is a cograph pattern f_x for some $x \in \mathcal{X}(g)$ such that $|V(f_x)| > 1$. If f_x is not connected, $L(g'_i) \subsetneqq L(g_{\pi(i)}\{x/g_{⓪}\})$ holds. Then we assume that f_x is connected. Since f_x is a cograph pattern, it is constructed from some cographs $f_{x,1}, \ldots, f_{x,k}$ $(k \ge 2)$ by join operations. Therefore, for any pair of i and j $(1 \le i < j \le k)$, all vertices in $f_{x,i}$ connect to all vertices in $f_{x,j}$ in f_x. Thus, $f_x \in L(g_{①})$ holds, and therefore $L(g'_i) \subsetneqq L(g_{\pi(i)}\{x/g_{①}\})$ holds. Consequently, since $S \subsetneqq L(g') \subsetneqq L(g)$, we have either $S \subsetneqq L(g\{x/g_{①}\})$ or $S \subsetneqq L(g\{x/g_{⓪}\})$. Since g is an output of Algorithm MINL-$\mathcal{LCGP}$, this contradicts lines 6 and 7 of the algorithm. (*End of Proof of Claim* 4)

From Claim 4, we conclude that $g' \cong g$. □

Lemma 8: Given a set of cographs $S \subsetneqq \mathcal{CG}(\Sigma)$, Algorithm MINL-$\mathcal{LCGP}$ solves MINL Problem for $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$ in $O(N_{max}^{4.5}|S|)$ time, where $N_{max} = \max_{G \in S} |V(G)|$.

Proof. Let g be an output of MINL-$\mathcal{LCGP}(\pm, \mathcal{X})$. Let $N_{min} = \min_{G \in S} |V(G)|$, $M_{min} = \min_{G \in S} |E(G)|$, and $M_{max} = \max_{G \in S} |E(G)|$. Since $|V(g)| \le N_{min}$ and the lines 5–15 of MINL-$\mathcal{LCGP}(\pm, \mathcal{X})$ try to divide one variable into two variables, Matching-$\mathcal{LCGP}(\pm, \mathcal{X})$ is called $O(N_{min}|S|)$ times at lines 5–15. Let $\Sigma(S)$ be the set of all labels that appear in all cographs in S. Since $|\Sigma(S)| \le N_{min}$, Matching-$\mathcal{LCGP}(\pm, \mathcal{X})$ is called $O(N_{min}^2|S|)$ times at lines 18–25. From the proof of Lemma 6, one call for Matching-$\mathcal{LCGP}(\pm, \mathcal{X})$ takes $O(N_{min}N_{max}^{1.5})$ time, except for the time needed for constructing the cotree and the cotree pattern. The total time complexity except for the constructions of the cotrees and cotree patterns is $O(N_{max}^{4.5}|S|)$ time. From Theorem 5, constructing the cotrees of all cographs in S takes $O((N_{max} + M_{max})|S|) = O(N_{max}^2|S|)$ time. Moreover, we need to construct the cotree pattern of a temporary cograph pattern at most $O(N_{min}^2)$ times. We need totally $O(N_{min}^2(N_{min} + M_{min}))$ time to construct all temporary cograph patterns. Thus, we conclude that the whole algorithm runs in $O(N_{max}^{4.5}|S|)$ time. □

Theorem 3 in Sect. 3 follows from Lemmas 7 and 8.

6. Conclusion and Future Work

In this paper, first, we introduced a cograph pattern and a cograph pattern language, and proved that the class $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$ of linear cograph pattern languages has finite thickness. Next, we gave two polynomial time algorithms for solving the membership problem for $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$ and the MINL problem for $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$. Finally, by using these results, we showed that the class $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$ is polynomial time inductively inferable from positive data.

As future work, we will improve our algorithms in order to propose fully effective data mining methods for graph structured data. For example, the membership problem for $\mathcal{L}_{\mathcal{LCGP}(\Sigma,\mathcal{X})}$ might be solved faster than the running time of our algorithm by using an idea in [11]. Several practical applications in computer science and computational linguistics suggest the study of graphs with few P_4's, and some NP-complete problems for general graphs can be solved efficiently for these graphs [3], [4]. We are now studying the polynomial time learnability problems for the classes of graph languages defined by graphs with few P_4's. Furthermore, we are developing general data mining techniques for various real world data that can be modeled by these graph classes.

Acknowledgments

We thank the anonymous referees for helpful comments and detailed suggestions that helped to improve this article. This work was partially supported by Grant-in-Aid for Scientific Research (C) (Grant Numbers JP15K00312, JP15K00313, JP17K00321) from Japan Society for the Promotion of Science (JSPS).

References

[1] T.R. Amoth, P. Cull, and P. Tadepalli, "On Exact Learning of Unordered Tree Patterns," Machine Learning, vol.44, no.3, pp.211–243, 2001.
[2] D. Angluin, "Inductive Inference of Formal Languages from Positive Data," Information and Control, vol.45, no.2, pp.117–135, 1980.
[3] L. Babel and S. Olariu, "On the structure of graphs with few P4s," Discrete Applied Mathematics, vol.84, no.1-3, pp.1–13, 1998.
[4] L. Babel, T. Kloks, J. Kratochvíl, D. Kratsch, H. Müller, and S. Olariu, "Efficient algorithms for graphs with few P_4's," Discrete Mathematics, vol.235, no.1-3, pp.29–51, 2001.
[5] D.G. Corneil, H. Lerchs, and L.S. Burlingham, "Complement Reducible Graph," Discrete Applied Mathematics, vol.3, no.3, pp.163–174, 1981.
[6] D.G. Corneil, Y. Perl, and L.K. Stewart, "A Linear Recognition Algorithm for Cographs," SIAM Journal on Computing, vol.14, no.4, pp.926–934, 1985.
[7] J.E. Hopcroft and R.M. Karp, "An $n^{5/2}$ algorithm for maximum matching in bipartite graphs," SIAM Journal on Computing, vol.2,

no.4, pp.225–231, 1973.
[8] B. Jamison and S. Olariu, "P-Components and the Homogeneous Decomposition of Graphs," SIAM Journal on Discrete Mathematics, vol.8, no.3, pp.448–463, 1995.
[9] T. Miyahara, T. Shoudai, T. Uchida, T. Kuboyama, K. Takahashi, and H. Ueda, "Discovering New Knowledge from Graph Data Using Inductive Logic Programming," Proc. 9th Int. Conf. Inductive Logic Programming (ILP-99), Springer, LNAI, vol.1634, pp.222–233, 1999.
[10] T. Miyahara, T. Shoudai, T. Uchida, K. Takahashi, and H. Ueda, "Polynomial Time Matching Algorithms for Tree-like Structured Patterns in Knowledge Discovery," Proc. 4th Pacific-Asia Conf. Knowledge Dicovery and Data Mining (PAKDD 2000), Springer, LNAI, vol.1805, pp.5–16, 2000.
[11] R. Shamir and D. Tsur, "Faster Subtree Isomorphism," Journal of Algorithms, vol.33, no.2, pp.267–280, 1999.
[12] T. Shinohara, "Polynomial Time Inductive Inference of Extended Regular Pattern Languages," RIMS Symposia on Software Science and Engineering, LNCS, vol.147, pp.115–127, 1982.
[13] Y. Suzuki, T. Shoudai, T. Uchida, and T. Miyahara, "Ordered Term Tree Languages Which Are Polynomial Time Inductively Inferable from Positive Data," Theoretical Computer Science, vol.350, no.1, pp.63–90, 2006.
[14] R. Takami, Y. Suzuki, T. Uchida, and T. Shoudai, "Polynomial Time Inductive Inference of TTSP Graph Languages from Positive Data," IEICE Trans. Inf. & Syst., vol.E92-D, no.2, pp.181–190, 2009.
[15] H. Yamasaki and T. Shoudai, "A Polynomial Time Algorithm for Finding a Minimally Generalized Linear Interval Graph Pattern," IEICE Trans. Inf. & Syst., vol.E92-D, no.2, pp.120–129, 2009.
[16] Y. Yoshimura, T. Shoudai, Y. Suzuki, T. Uchida, and T. Miyahara, "Polynomial Time Inductive Inference of Cograph Pattern Languages from Positive Data," Proc. 21st Int. Conf. Inductive Logic Programming (ILP2011), Springer, LNAI, vol.7207, pp.389–404, 2012.

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 Faculty of Contemporary Business, Kyushu International University. His research interests include graph algorithms, computational learning theory, and data mining.

Yuta Yoshimura received the B.S. and the M.S. degrees in Informatics all from Kyushu University, in 2011 and 2013, respectively. He is currently with the CAE Solutions Department, Engineering Solutions Unit, Fujitsu Kyushu System Services Limited. His research interests include graph algorithms and machine learning.

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 research associate 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 an associate 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.