FOUNDATIONS OF COMPUTING AND DECISION SCIENCES
Vol. 41 (2016) No. 3
DE GRUYTER OPEN
DOI: 10.1515/fcds-2016-0010
ISSN 0867-6356
e-ISSN 2300-3405

POLYNOMIAL TIME ALGORITHMS FOR VARIANTS OF GRAPH MATCHING ON PARTIAL k-TREES

Takayuki NAGOYA *

Abstract. In this paper, we deal with two variants of graph matching, the graph isomorphism with restriction and the prefix set of graph isomorphism. The former problem is known to be NP-complete, whereas the latter problem is known to be GI-complete. We propose polynomial time exact algorithms for these problems on partial k-trees.

Keywords: graph isomorphism with restriction, partial k-trees, polynomial time algorithm

1 Introduction

Graphs can be effectively used to represent relationships between objects such as social networks, web search engines and genome sequencing. Comparison of structures of graphs has been of significant importance and has wide-spread applications. For example, the subgraph isomorphism problem is used to search for, given two graphs G and H, a subgraph of G whose structure is equal to H, and algorithms for the problem have been designed (e.g., [9, 15]).

The most basic version of graph matching is the graph isomorphism problem. We say that two graphs $G = (V, E)$ and $H = (W, F)$ are isomorphic if there is a bijection $\varphi : V \to W$ such that for every two vertices $v_1, v_2 \in V$, the edge (v_1, v_2) is in E if and only if the edge $(\varphi(v_1), \varphi(v_2))$ is in F. We call such a bijection an isomorphism. The graph isomorphism problem (GI for short) is to determine whether given two graphs are isomorphic or not. The problem is clearly in NP, but not known either to be solvable in polynomial time, or to be NP-complete. Thus, several works have been done for GI on restricted graph classes [2, 3, 11, 16, 17] and for variants of GI [10, 14].

*Tottori University of Environmental Studies, 1-1-1 Wakabadai-Kita, Tottori, Tottori 689-1111, Japan (email:nagoya@kankyo-u.ac.jp)

We consider two variants of graph isomorphism. The first one is the graph isomorphism with restriction (GIR for short). This problem is to determine whether for given two graphs G and H and a binary relation $R \subseteq V \times W$, there is an isomorphism between G and H that avoids any element of R. This problem can be applicable to the situation in that we want to decide a mapping between vertices of two graphs G and H under the constraint that some vertices of G are not allowed to be mapped to some vertices of H. The second one is the prefix set of graph isomorphism (PrefixGI for short). This problem is to determine whether for given two graphs G and H and an isomorphism φ from an induced subgraph of G to an induced subgraph of H, there is an isomorphism between G and H that is an extension of φ. This problem can applicable to the situation in that we want to decide a mapping between vertices of two graphs G and H under the constraint that some vertices of G have to mapped to some vertices of H according to φ.

The above two problems are generalizations of GI and that have a more wide application than GI. However, unfortunately, NP-completeness of GIR have shown by A. Lubiw [10]. Also, J. Köbler et al [8] have shown that PrefixGI is GI-complete. Namely, PrefixGI and GI is of the same difficulty. Thus, exact algorithm for these problems is not likely to be found.

In this paper, we propose polynomial time exact algorithms for GIR and PrefixGI on partial k-trees (i.e. subgraphs of k-trees). For a natural number k, k-trees is a class of graphs with tree structure. Roughly speaking, a k-tree is a tree with width k and that is a generalization of tree. For example, 1-tree is the set of all trees. It is known that the class of partial k-trees is a large set of graphs. Thus, our algorithms could be powerful tools for several graph problems.

We finish this section by explaining theoretical aspects of our result. Bodlaender [3] designed a polynomial time algorithm for GI on partial k-trees. Also, GIR for chordal graphs with bounded clique size can be solved in polynomial time [13]. Since GIR is harder than or equal to GI and partial k-trees is a super class of chordal graphs with bounded clique size, our result extends those results.

2 Preliminary

Throughout this paper, we suppose that all graphs are undirected and simple. Let $G = (V, E)$ be a graph. We denote by $G[U]$ the subgraph that is induced by $U \subseteq V$. We often identify a graph and its vertex set.

Definition 1. *The class of k-trees is defined recursively as follows:*

1. *A clique (i.e., a complete graph) with k vertices is a k-tree.*
2. *If $G = (V, E)$ is a k-tree, and $x \notin V$, and $v_1, \cdots, v_k$ form a clique in G with k vertices, then $H = (V \cup \{x\}, E \cup \{(x, v_i) : 1 \leq i \leq k\})$ is a k-tree.*
3. *All k-trees can be formed with rules 1 and 2.*

Definition 2. *A graph is a partial k-tree, if and only if it is a subgraph of a k-tree.*

Figure 1 is an example of a 4-tree. According to Definition 1, this graph is constructed recursively as follows. First, we construct a complete graph with vertex set $\{v_1, \cdots, v_4\}$. Next, we add the new vertex x and the edges that connect x and each vertex of $\{v_1, v_2, v_3, v_4\}$. After that, we add the new vertex x_1 and the edges that connect x_1 and each vertex of $\{v_1, v_3, v_4, x\}$. We add vertices x_2, x_3, and x_4 by iterating similar procedures. Then, the obtained graph is a 4-tree. By Definition 2, any subgraph of this graph is a partial 4-tree.

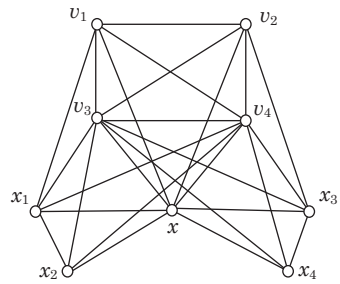

Figure 1: An example of a 4-tree

Let $G = (V, E)$ and $H = (W, F)$ be two graphs. For two partial mappings $\varphi : V \to W$ and $\psi : V \to W$, we say that φ does not contradict to ψ if for any $v \in Dom(\varphi) \cap Dom(\psi)$, $\varphi(v) = \psi(v)$. We say that φ extends ψ if φ does not contradict to ψ and $Dom(\varphi) \supseteq Dom(\psi)$. For φ and ψ that do not contradict to each other, we define the partial mapping $\varphi \sqcup \psi$ as follows.

$$\varphi \sqcup \psi(v) = \begin{cases} \varphi(v) \,\textbf{if}\, v \in Dom(\varphi) \\ \psi(v) \,\textbf{if}\, v \in Dom(\psi) \end{cases} \tag{1}$$

In the following discussion, we consider partial mappings that avoid every pair of a binary relation between V and W. This notion is defined precisely as follows.

Definition 3. *For a binary relation $R \subseteq V \times W$, we say that a partial mapping $\varphi : V \mapsto W$ is an R-mapping if for any $v \in Dom(\varphi)$, $(v, \varphi(v)) \notin R$ is satisfied.*

Any partial mapping that does not contradict to another partial mapping is defined by using the above notion. Precisely, for a partial mapping φ from V to W, we define binary relation $r(\varphi) \subseteq V \times W$ as follows.

$$\begin{aligned} r(\varphi) \quad = \quad & \{(v, w) : v \in Dom(\varphi) \text{ and } w \in W - \{\varphi(v)\}\} \\ & \cup \{(v, w) : v \notin Dom(\varphi) \text{ and } w \in Im(\varphi)\} \end{aligned} \tag{2}$$

Then, any $r(\varphi)$-mapping maps any vertex of $Dom(\varphi)$ according to φ and maps any vertex of $V - Dom(\varphi)$ to a vertex other than vertices of $Im(\varphi)$. Thus, we have that a partial mapping ψ is an $r(\varphi)$-mapping if and only if ψ does not contradict to φ.

In the remaining sections, we deal with a partial mapping that avoids every pair of R and that does not contradict to φ. In other words, we consider a partial mapping that is an R-mapping and also an $r(\varphi)$-mapping. Such a mapping can be represented as an $(R \cup r(\varphi))$-mapping by using the union of R and $r(\varphi)$.

The followings are some basic facts about above defined notions.

Fact 1. *Let $R \subseteq V \times W$ be a binary relation and let φ and ψ be two partial mappings from V to W. Then, we have the following facts.*

1. *If φ is an R-mapping then the mapping $\varphi|_C$, the restriction of φ whose domain is $C \subset Dom(\varphi)$, is also an R-mapping.*
2. *If φ and ψ are R-mappings and do not contradict to each other, then $\varphi \sqcup \psi$ is also R-mapping.*

We finish this section by state two problems that are dealt with in this paper. For two graphs $G = (V, E)$ and $H = (W, F)$ and a binary relation $R \subseteq V \times W$, a bijection $\varphi : V \to W$ is an R-isomorphism if φ is an isomorphism from G to H and is an R-mapping from V to W. We say that G is R-isomorphic to H if there exists an R-isomorphism from G to H. We write $G \cong^R H$ in this case. The graph isomorphism with restriction (GIR for short) is a problem of determining whether given two graphs are R-isomorphic or not. The prefix set of graph isomorphism (PrefixGI for short) is a problem to decide whether given two graphs G and H and an isomorphism φ from an induced subgraph of G to an induced subgraph of H, there exists an isomorphism from G to H that extends φ, namely $r(\varphi)$-isomorphism from G to H.

The difference of these two problems is that GIR requires isomorphisms to avoid the elements of R whereas PrefixGI requires isomorphisms to include the elements of R. In general, this difference seems to be significant. As stated in introduction, GIR is known to be NP-complete. However we have strong indications that GI is not NP-complete. Thus, GIR seems to be more difficult than GI. On the other hand, it is known that PrefixGI is polynomial time equivalent to GI. Therefore, it is expected that GIR is more difficult than PrefixGI. If this expectation is correct, this difference arises from the difference between avoidance or inclusion.

3 Algorithm for the graph isomorphism with restriction

Arnborg et al [1] designed $\mathcal{O}(n^{k+2})$ time algorithm for recognizing partial k-trees. Their algorithm is based on the fact that a partial k-tree is decomposed into a collection of pairs of a k-vertex separator and a connected component. Bodlaender [3] proposed a polynomial time algorithm to solves GI on partial k-trees. The algorithm is based on Arnborg et al's result. In this section, we show that GIR on partial k-trees can be solved in polynomial time by using these idea.

3.1 Bodlaender's algorithm

We briefly describe Arnborg et al's result and Bodlaender's algorithm for GI on partial k-trees. In the following, we let $G = (V, E)$ and $H = (W, F)$ be connected graphs.

Arnborg et al's recognition algorithm for partial k-trees is depend on two lemmas. We state one of the lemmas as Lemma 1. Lemma 2 is obtained by slight modification of another, and it can be proved in the same way as Arnborg et al's proof. In these lemmas, for a k-vertex separator C of G and a connected component X of $V - C$, the graph $G(C, X)$ is defined by vertex set $C \cup X$ and the edge set $\{(v_1, v_2) : v_1, v_2 \in C\} \cup \{(v_1, v_2) : v_1, v_2 \in X \text{ and } (v_1, v_2) \in E\} \cup \{(v_1, v_2) : v_1 \in C \text{ and } v_2 \in X \text{ and } (v_1, v_2) \in E\}$.

Lemma 1. *[1] Suppose $n \geq k + 2$. G is a partial k-tree, if and only if there exists a k-vertex separator C_r, such that for all connected components X_i of $V - C_r$, $G(C_r, X_i)$ is a partial k-tree.*

Lemma 2. *[1]Let C_p is a k-vertex separator of G and X_i is a connected component of $V - C_p$. Then, the graph $G(C_p, X_i)$ with at least $k+2$ vertices is a partial k-tree if and only if there exists a vertex $x \in X_i$, such that for each connected component X_i^s of the graph, obtained by removing x from X_i, there is a vertex $u^s \in C_p$ such that*

1. *$C_p^s = C_p - \{u^s\} \cup \{x\}$ is a k-vertex separator of G.*
2. *No vertex in X_i^s is adjacent to the vertex u^s.*
3. *$G(C_p^s, X_i^s)$ is a partial k-tree.*

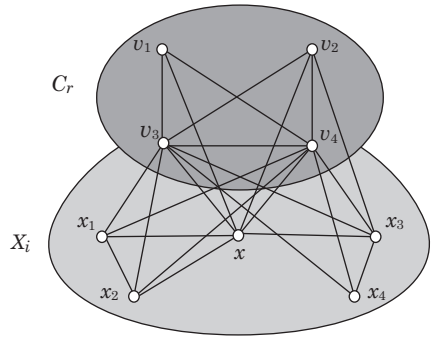

Figure 2: An illustrative example of Lemma 1

The above two lemmas are based on the recursive definition of k-trees and on the fact that a partial k-tree is a subgraph of a k-tree. Figure 2 shows an example of a partial k-tree that is obtained by removing some edges from the k-tree in Figure 1. We can apply Lemma 1 to the graph as follows. Let $C_r = \{v_1, \cdots, v_4\}$ and let $X_i = \{x, x_1, \cdots x_4\}$ be a connected component of $V - C_r$. Since C_r is a clique in the original k-tree, $G(C_r, X_i)$ is a subgraph of the original k-tree. Therefore, $G(C_r, X_i)$

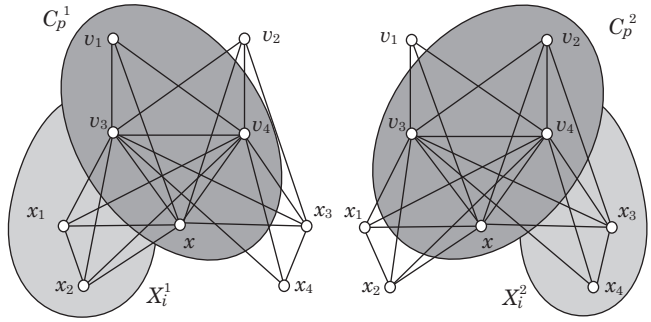

Figure 3: An illustrative example of Lemma 2

is a partial k-tree. Figure 3 is an illustrative example of Lemma 2 with $p = r$. $X_i^1 = \{x_1, x_2\}$ and $X_i^2 = \{x_3, x_4\}$ are connected components that are obtained by removing x from X_i. For $s = 1$, we let $u^1 = v_2$. Then, $C_p^1 = \{v_1, v_3, v_4, x\}$ and $X_i^1 = \{x_1, x_2\}$. Since C_p^1 is a k-vertex separator in the original k-tree, it is also a k-vertex separator in the graph of Figure 3. According to recursive construction of the k-tree of Figure 1, no vertex of X_i^1 is adjacent to u^1. In addition, since $G[C_p^1 \cup X_i^1]$ is a subgraph of the original k-tree and C_p^1 is a clique of the original k-tree, $G(C_p^1, X_i^1)$ is a partial k-tree. We have that the conditions 1, 2, and 3 are satisfied for $s = 1$. For $s = 2$, we let $u^2 = v_1$. Then we can verify that the conditions 1, 2, and 3 are satisfied in the same manner.

According to Lemma 1 and Lemma 2, a partial k-tree can be decomposed to a collection of pairs of k-vertex separator and connected component. Bodlaender designed $\mathcal{O}(n^{k+4.5})$ time algorithm for GI on partial k-trees by using this fact. In the algorithm, we first store information of the decomposition of G by executing Arnborg et al's algorithm for G. By using the stored information, we can decide whether the given two partial k-trees are isomorphic or not. He called the information "representation of G as partial k-tree".

3.2 Polynomial time algorithm for GIR on partial k-trees

In this section, we propose a polynomial time algorithm for GIR on partial k-trees. We let G, H, and R be inputs for GIR, where $G = (V, E)$ and $H = (W, F)$ are connected partial k-trees with $|V| = |W| = n$ and $R \subseteq V \times W$. In the our algorithm, we first computes "representation of G as partial k-tree". Then, we decide whether G and H are R-isomorphic or not by using dynamic programming. In the following discussion, we assume that "representation of G as partial k-tree" is already computed.

Let C_r be the k-vertex separator of G in the representation of G as partial k-tree that satisfies Lemma 1. Let also $X_1, \cdots, X_m$ be all of connected components of $V - C_r$. Then, we have the next lemma.

Lemma 3. *$G \cong^R H$ if and only if there exists a k-vertex separator D_q of H and*

R-isomorphism φ from C_r to D_q such that the following conditions hold:

1. *Let $Y_1, \cdots, Y_{m'}$ be all of connected components of $W - D_q$. Then $m = m'$.*
2. *There exists a bijection τ on $\{1, \cdots, m\}$ such that for any $i \in \{1, \cdots, m\}$, $G[C_r \cup X_i] \cong^{R \cup r(\varphi)} H[D_q \cup Y_{\tau(i)}]$.*

Proof. $\Rightarrow$) Suppose that ξ is an R-isomorphism from G to H. Let $D_q = \xi(C_r)$ and let φ be the restriction of ξ whose domain is C_r. Then, since C_r is a k-vertex separator of G and ξ is an isomorphism, D_q is also a k-vertex separator of H. In addition, since φ is a restriction of R-mapping ξ, φ is also an R-mapping from C_r to D_q (1 of Fact 1). Furthermore, since φ is a restriction of an isomorphism ξ, φ is also an isomorphism from C_r to D_q. Thus, φ is an R-isomorphism from C_r to D_q. Then, we can show that the condition 1 and 2 of Lemma 3 are satisfied as follows.

1. Since ξ is an isomorphism, the number of connected components of $V - C_r$ is equal to the number of connected components of $W - \xi(C_r)$. Since $D_q = \xi(C_r)$, the condition 1 holds.
2. Since ξ is an isomorphism and X_i's and Y_j's are connected components of $V - C_r$ and $W - D_q$ respectively, there exists a bijection τ on $\{1, \cdots, m\}$ such that for each $i \in \{1, \cdots, m\}$, $\xi(X_i) = Y_{\tau(i)}$. We below show that for each $i \in \{1, \cdots, m\}$, $\xi|_{C_r \cup X_i}$ is an $(R \cup r(\varphi))$-isomorphism from $G[C_r \cup X_i]$ to $H[D_q \cup Y_{\tau(i)}]$. Since $\xi|_{C_r \cup X_i}$ is a restriction of R-mapping ξ, $\xi|_{C_r \cup X_i}$ is also an R-mapping (1 of Fact 1). In addition, $\xi|_{C_r \cup X_i}$ does not contradict to φ, because both $\xi|_{C_r \cup X_i}$ and φ are restrictions of ξ. Thus, $\xi|_{C_r \cup X_i}$ is an $r(\varphi)$-mapping. Furthermore, since $\xi|_{C_r \cup X_i}$ is a restriction of an isomorphism ξ, $\xi|_{C_r \cup X_i}$ is also an isomorphism. As a result, we have that $\xi|_{C_r \cup X_i}$ is an $(R \cup r(\varphi))$-isomorphism form $G[C_r \cup X_i]$ to $H[D_q \cup Y_{\tau(i)}]$, the condition 2 holds.

$\Leftarrow$) Suppose that there exists D_q, φ, and τ that satisfy the condition 1 and 2. We below show that there is an R-isomorphism from G to H. Suppose that ξ_i, $i \in \{1, \cdots, m\}$ is an $(R \cup r(\varphi))$-isomorphism from $G[C_r \cup X_i]$ to $H[D_q \cup Y_{\tau(i)}]$ that satisfies the condition 2. For any two mappings ξ_i and ξ_j, the intersection of domains of them is C_r. Furthermore, since any ξ_i is an $r(\varphi)$-mapping, $\xi_i(v) = \varphi(v)$ is satisfied for each $v \in C_r = Dom(\varphi)$. Thus, these mappings do not contradict to each other. Therefore, we can define the mapping $\xi = \xi_1 \sqcup \cdots \sqcup \xi_m$. Since each ξ_i is an R-mapping, ξ is also an R-mapping (2 of Fact 1). Furthermore, since any X_i and Y_j are connected components of $V - C_r$ and $W - D_q$ respectively and any ξ_i is an isomorphism, ξ is an isomorphism from G to H. We conclude that ξ is an R-isomorphism from G to H, and the lemma holds.

□

According to Lemma 3, we can solve GIR by deciding whether there exist a k-vertex separator D_q of H, an R-isomorphism φ from C_r to D_q, and a bijection τ on $\{1, \cdots, m\}$ such that for each $i \in \{1, \cdots, m\}$, $G[C_r \cup X_i]$ is an $(R \cup r(\varphi))$-isomorphic to $H[D_q \cup Y_{\tau(i)}]$. This can be done as follows. For each k-vertex separator D_q and each R-isomorphism φ, we construct a bipartite graph $B = (V_B^1, V_B^2, E_B)$ where

$V_B^1 = \{X_1, \ldots, X_m\}$, $V_B^2 = \{Y_1, \ldots, Y_m\}$, and $(X_i, Y_j) \in E_B$ if and only if $G[C_r \cup X_i]$ is $(R \cup r(\varphi))$-isomorphic to $H[D_q \cup Y_j]$. Then, we easily see that B has a perfect matching if and only if a bijection τ that satisfies the condition 2 exists. Furthermore, if there exists D_q and φ such that the condition 1 of Lemma 3 holds and B has a perfect matching, then G is R-isomorphic to H.

The next lemma shows that we can recursively decide whether $G[C_r \cup X_i]$ is $(R \cup r(\varphi))$-isomorphic to $H[D_q \cup Y_j]$. In the lemma, (C_p, X_i) is any pair of k-vertex separator C_p and connected component X_i that are in "representation of G as partial k-tree". Also, a vertex $x \in X_i$, connected components $X_i^1, \cdots, X_i^m$ of $X_i - \{x\}$, a vertex u^s for each s, and $C_p^s = C_p - \{u^s\} \cup \{x\}$ for each s are recursive information about (C_p, X_i) in the representation of G as partial k-tree that satisfy Lemma 2. In addition, we let (D_q, Y_j) be any pair of a k-vertex separator D_q of H and a connected component Y_j of $W - D_q$ and let φ be any R-isomorphism from C_p to D_q.

Lemma 4. *$G[C_p \cup X_i] \cong^{R \cup r(\varphi)} H[D_q \cup Y_j]$ is satisfied if and only if there exists a vertex $y \in Y_j$ such that the following conditions hold.*

1. *$Y_j - \{y\}$ has m connected components $Y_j^1, \cdots, Y_j^m$.*
2. *For each $s \in \{1, \cdots, m\}$, $(u^s, x) \in E \Leftrightarrow (\varphi(u^s), y) \in F$*
3. *There exists a bijection τ on $\{1, \cdots, m\}$ such that for each $s \in \{1, \cdots, m\}$, the following conditions hold.*
   (a) *There is no vertex in $Y_j^{\tau(s)}$ that adjacent to $\varphi(u^s)$.*
   (b) *We denote a mapping $\varphi|_{C_p - \{u^s\}} \sqcup \{x \to y\}$ by φ_s. Then, φ_s is an $(R \cup r(\varphi))$-isomorphism from C_p^s to $D_q^s = D_q - \{\varphi(u^s)\} \cup \{y\}$, and $G[C_p^s \cup X_i^s] \cong^{R \cup r(\varphi_s)} H[D_q^s \cup Y_j^{\tau(s)}]$ holds.*

Proof. $\Rightarrow$) Suppose that there exists an $(R \cup r(\varphi))$-isomorphism ξ from $G[C_p \cup X_i]$ to $H[D_q \cup Y_j]$. Let $y = \xi(x)$. Then, we below prove that the conditions 1, 2, and 3 hold. The top of Figure 4 is an illustrative example when $y = \xi(x)$.

1. Since ξ is an $r(\varphi)$-mapping, ξ maps C_p to D_q. Thus, ξ maps X_i to Y_j. Furthermore, since ξ is an isomorphism, the number of connected components of $X_i - \{x\}$ is equal to the number of connected components of $Y_j - \{\xi(x)\}$. Because of $y = \xi(x)$, the condition 1 holds.

2. Since ξ is an isomorphism, $(u^s, x) \in E \Leftrightarrow (\xi(u^s), \xi(x)) \in F$ is satisfied (see Figure 4 below). Where, $\xi(u^s) = \varphi(u^s)$ is satisfied, because ξ is a φ-mapping and $u^s \in Dom(\varphi)$. Since $\xi(x) = y$, $(u^s, x) \in E \Leftrightarrow (\varphi(u^s), y) \in F$ is satisfied. We have that the condition 2 holds.

3. Since ξ is an isomorphism and $\{X_i^1, \cdots, X_i^m\}$ and $\{Y_j^1 \cdots, Y_j^m\}$ are connected components of $X_i - \{x\}$ and $Y_j - \{\xi(x)\}$ respectively, there exists a bijection τ on $\{1, \cdots, m\}$ such that for each $s \in \{1, \cdots, m\}$, $\xi(X_i^s) = Y_j^{\tau(s)}$. We below show that such τ satisfies the conditions (a) and (b).

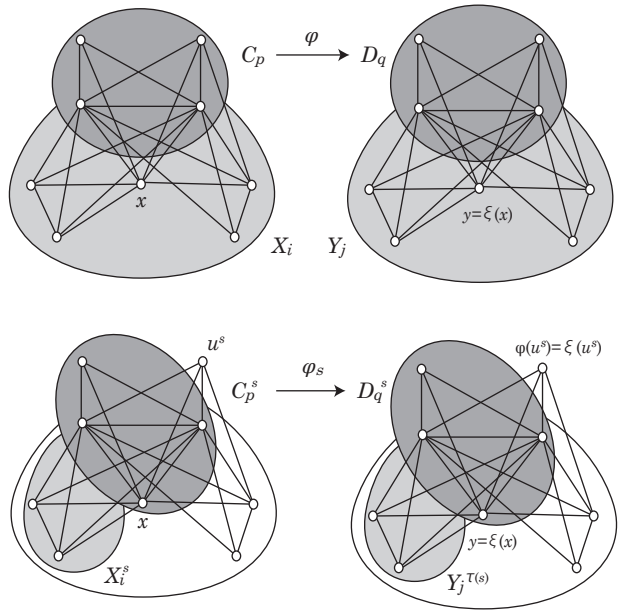

Figure 4: An illustrative example of Lemma 4

(a) By Lemma 2, there is no vertex in X_i^s that adjacent to u^s (see Figure 4 below). Since $Y_j^{\tau(s)}$ and $\xi(u^s)$ are images of X_i^s and u^s by the isomorphism ξ, there is no vertex in $Y_j^{\tau(s)}$ that adjacent to $\xi(u^s)$. Because of $\xi(u^s) = \varphi(u^s)$, $\varphi(u^s)$ does not adjacent to any vertex of $Y_j^{\tau(s)}$, the condition (a) holds.

(b) We first show that φ_s is an $(R \cup r(\varphi))$-isomorphism. By the definition of φ_s, φ_s maps each vertex of $C_p - \{u^s\}$ according to φ. In addition, since ξ is an $r(\varphi)$-isomorphism, ξ also maps each vertex of $C_p - \{u^s\}$ according to φ. Thus, we have that for each vertex $v \in C_p - \{u^s\}$, $\varphi_s(v) = \xi(v)$. In addition, we know that $\varphi_s(x) = y = \xi(x)$. Thus, φ_s maps each vertex of $C_p^s = C_p - \{u^s\} \cup \{x\}$ according to ξ. Therefore, φ_s is a restriction of ξ whose domain is C_p^s. Since ξ is an $(R \cup r(\varphi))$-isomorphism, φ_s is also $(R \cup r(\varphi))$-isomorphism (1 of Fact 1).

We next show that $G[C_p^s \cup X_i^s] \cong^{R \cup r(\varphi_s)} H[D_q^s \cup Y_j^{\tau(s)}]$ holds. We denote $\xi|_{C_p^s \cup X_i^s}$ by ξ_s. We below show that ξ_s is an R-mapping, $r(\varphi_s)$-mapping, and an isomorphism from $C_p^s \cup X_i^s$ to $D_q^s \cup Y_j^{\tau(s)}$. Since ξ_s is a restriction of R-mapping ξ, ξ_s is also R-mapping (1 of Fact 1). Both φ_s and ξ_s are restrictions of ξ. Thus, ξ_s does not contradict to φ_s. In other words, ξ_s is an $r(\varphi_s)$-mapping. Since ξ_s is a restriction of an isomorphism ξ, ξ_s is also an isomorphism. We conclude that ξ_s is an $(R \cup r(\varphi_s))$-isomorphism, the condition (b) holds.

$\Leftarrow$) Suppose that there exists a vertex $y \in Y_j$ that satisfies the condition 1, 2, and 3. By the condition 3-(b), for each $s \in \{1, \cdots, m\}$, there exists an $(R \cup r(\varphi_s))$-isomorphism ξ_s that maps $G[C_p^s \cup X_i^s]$ to $H[D_q^s \cup Y_j^{\tau(s)}]$. We denote $\xi_s \sqcup \{u^s \to \varphi(u^s)\}$ by ξ'_s.

We first show that ξ'_s is an $(R \cup r(\varphi))$-isomorphism that maps $G[C_p \cup \{x\} \cup X_i^s]$ to $H[D_q \cup \{y\} \cup Y_j^{\tau(s)}]$. Since $\{u^s \to \varphi(u^s)\}$ is a restriction of an R-mapping φ, $\{u^s \to \varphi(u^s)\}$ is also an R-mapping (1 of Fact 1). Since ξ'_s is a composition of two R-mappings ξ_s and $\{u^s \to \varphi(u^s)\}$, ξ'_s is also an R-mapping (2 of Fact 1). Since ξ_s is an $r(\varphi_s)$-mapping and φ_s is an $r(\varphi)$-mapping, ξ_s is an $r(\varphi)$-mapping. Therefore, since ξ'_s is a composition of two $r(\varphi)$-mappings ξ_s and $\{u^s \to \varphi(u^s)\}$, ξ'_s is also an $r(\varphi)$-mapping (2 of Fact 1). Now, we have that ξ'_s is an $(R \cup r(\varphi))$-mapping. We below show that ξ'_s is an isomorphism. Since ξ_s is an isomorphism, ξ'_s preserves adjacency relation between v_1 and v_2 for any two vertices $v_1, v_2 \in Dom(\xi_s) = C_p^s \cup X_i^s$. In other words, $(v_1, v_2) \in E \Leftrightarrow (\xi'_s(v_1), \xi'_s(v_2)) \in F$ holds for any two vertices $v_1, v_2 \in Dom(\xi_s) = C_p^s \cup X_i^s$. Thus, in order to show that ξ'_s is an isomorphism, it is enough to verify that ξ'_s preserves adjacency relation between (1) u^s and any vertex of $C_p - \{u^s\}$, (2) u^s and x, and (3) u^s and any vertex of X_i^s. We below prove that these three conditions hold.

(1) Since u^s and $C_p - \{u^s\}$ are in the domain of the isomorphism φ, adjacency relation between u^s and any vertex of $C_p - \{u^s\}$ is preserved by φ. By the above, we know that ξ'_s is an $r(\varphi)$-mapping. Thus, adjacency relation between u^s and any vertex of $C_p - \{u^s\}$ is also preserved by ξ'_s.

(2) By the condition 2, $(u^s, x) \in E \Leftrightarrow (\varphi(u^s), y) \in F$ holds. Where, by the definition of ξ'_s, $\varphi(u^s) = \xi'_s(u^s)$. Furthermore, since ξ_s is an $r(\varphi_s)$-mapping and φ_s maps x to y, ξ_s maps x to y. Thus, ξ'_s also maps x to y. Therefore, we have that $(u^s, x) \in E \Leftrightarrow (\varphi(u^s), y) \in F \Leftrightarrow (\xi'_s(u^s), \xi'_s(x)) \in F$ holds. We conclude that ξ'_s preserves adjacency relation between u^s and x.

(3) By Lemma 2, u^s is not adjacent to any vertex of X_i^s. Thus, in order to show that ξ'_i preserves adjacency relation between u^s and any vertex of X_i^s, it is enough to show that $\xi'_i(u^s)$ does not adjacent to any vertex of $\xi'_i(X_i^s)$. Since ξ'_s is an $r(\varphi)$-mapping, $\xi'_s(u^s) = \varphi(u^s)$. By the definition of ξ'_s, $\xi'_s(X_i^s) = Y_j^{\tau(s)}$. In addition, by the condition 3-(a), $\varphi(u^s)$ does not adjacent to any vertex of $Y_j^{\tau(s)}$. Therefore, we have that $\xi'_i(u^s)$ does not adjacent to any vertex of $\xi'_i(X_i^s)$.

By the above argument, we have that ξ'_s is an isomorphism. Since we already know that ξ'_s is an $(R \cup r(\varphi))$-mapping, we have that ξ'_s is an $(R \cup r(\varphi))$-isomorphism.

Finally, we show that we can define a mapping $\xi = \xi'_1 \sqcup \cdots \sqcup \xi'_m$ and ξ is an $(R \cup r(\varphi))$-isomorphism from $G[C_p \cup X_i]$ to $H[D_q \cup Y_j]$. Since the domain of ξ'_s is $C_p \cup \{x\} \cup X_i^s$ for every s, the intersection of domains of any two ξ'_s's is $C_p \cup \{x\}$. Thus, since any ξ'_s is $r(\varphi)$-mapping and that maps x to y, any two ξ'_s's do not contradict to each other. Therefore, we can define the mapping $\xi = \xi'_1 \sqcup \cdots \sqcup \xi'_m$ that maps the vertices of $C_p \cup X_i$ to the vertices of $C_q \cup Y_j$. Since any ξ'_s is an $(R \cup r(\varphi))$-mapping, ξ is also an $(R \cup r(\varphi))$-mapping (2 of Fact 1). The intersection of domains of these

mappings, namely $C_p \cup \{x\}$, is a separator, and $X_i^1, \cdots, X_i^m$ are connected components that are obtained by removing $C_p \cup \{x\}$. Thus, since any ξ'_s is an isomorphism, ξ is also an isomorphism. We conclude that ξ is an $(R \cup r(\varphi))$-isomorphism, and the lemma holds. □

According to Lemma 4, we can decide whether $G[C_p \cup X_i]$ is $(R \cup r(\varphi))$-isomorphic to $H[D_q \cup Y_j]$ or not with the dynamic programming. The computation is proceed in ascending order of the size of Y_j. For each vertex $y \in Y_j$ that satisfies the condition 1 and 2 of Lemma 4, we decide whether there exists a bijection τ on $\{1, \cdots, m\}$ that satisfies the conditions 3-(a) and 3-(b) of Lemma 4. This can be done by constructing a bipartite graph B_y with two vertex sets $V_{B_y}^1 = \{X_i^1, \ldots, X_i^m\}$ and $V_{B_y}^2 = \{Y_j^1, \ldots, Y_j^m\}$ and an edge set E_{B_y}. Where, we create an edge (X_i^s, Y_j^t) if the vertex $\varphi(u^s)$ does not adjacent to any vertex of Y_j^t and there is an $(R \cup r(\varphi_s))$-isomorphism from $G[C_p^s \cup X_i^s]$ to $H[D_q^s \cup Y_j^t]$. We can easily decide whether the first condition is satisfied. In addition, since $|Y_j| > |Y_j^t|$, we can decide whether the second condition is satisfied by looking up the table of dynamic programming. Thus, we can construct the bipartite graph B_y efficiently. Furthermore, we can decide whether there exists a bijection τ that satisfies the condition 3-(a) and 3-(b) by solving the perfect matching problem for B_y. Therefore, we can conclude that $G[C_p \cup X_i]$ is $(R \cup r(\varphi))$-isomorphic to $H[D_q \cup Y_j]$ if there exists a vertex $y \in Y_j$ such that the condition 1 and condition 2 is satisfied and B_y has a perfect matching.

Algorithm 1 is the pseudo-code of the algorithm. By the next theorem, the worst-case time complexity of the Algorithm is polynomial for any partial k-tree with fixed k.

Theorem 1. *There is an $\mathcal{O}(k!k^2n^{k+2.5} + k!n^{k+4.5})$-time algorithm to solve the graph isomorphism with restriction on partial k-trees.*

Proof. In the first of the algorithm, we create a table $R(v, w), v \in V, w \in W$ such that $R(v, w) = 1$ if $(v, w) \in R$, $R(v, w) = 0$ otherwise. This enable us to decide whether $(v, w) \in R$ holds or not in constant time. The table can be created in $\mathcal{O}(n^2)$ time. A representation of G can be computed in $\mathcal{O}(n^{k+2})$ time [1]. The computation of the line 3 takes $\mathcal{O}(\binom{n}{k} n \log(\binom{n}{k} n)) = \mathcal{O}(n^{k+1} \log n^{k+1}) = \mathcal{O}(n^{k+2})$ time. $GIR(C_p, X_i, D_q, Y_j, \varphi)$ is a table of dynamic programming. Initial values of each element of the table is 0. We will set the value of $GIR(C_p, X_i, D_q, Y_j, \varphi)$ to be 1 if $G[C_p \cup X_i] \cong^{R \cup r(\varphi)} H[D_q \cup Y_j]$ is satisfied. Since the numbers of $(C_p, X_i)'s$, $(D_q, Y_j)'s$, and $\varphi' s$ are $\mathcal{O}(n)$, $\mathcal{O}(n^{k+1})$, and $\mathcal{O}(k!)$ respectively, The size of the talbe is $\mathcal{O}(k!n^{k+2})$. Thus, the lines 1-4 take $\mathcal{O}(k!n^{k+2})$ time.

In the line 6 to 10, we decide whether, for each $(D_q, Y_j), |Y_j| = 1$, $(C_r, X_i), |X_i| = 1$, and R-isomorphism $\varphi : C_p \mapsto D_q$, $G[C_p \cup X_i] \cong^{R \cup r(\varphi)} H[D_q \cup Y_j]$ is satisfied. We can decide whether φ is an R-mapping in $\mathcal{O}(|Dom(\varphi)|) = \mathcal{O}(k)$ time. We can also decide whether φ is an isomorphism in $\mathcal{O}(k^2)$ time. Since $|X_i| = |Y_j| = 1$, only $\varphi \sqcup \{v \mapsto w\}, v \in X_i, w \in Y_j$ could be an $(R \cup r(\varphi))$-isomorphism. It takes $\mathcal{O}(k)$ time to decide this mapping is an $(R \cup r(\varphi))$-isomorphism. Since the numbers of (D_q, Y_j)'s with $|Y_j| = 1$, (C_r, X_i)'s with $|X_i| = 1$, and φ's are $\mathcal{O}(n^{k+1})$, $\mathcal{O}(n)$, and $\mathcal{O}(k!)$ respectively, the complexity of the lines 6-10 is $\mathcal{O}(k^2 k! n^{k+2})$.

Algorithm 1 : Graph Isomorphism with Restriction

Require: two partial k-trees G and H and a binary relation $R \subseteq V \times W$
Ensure: answer $G \cong^R H$ or $G \ncong^R H$

1: create a table $R(v, w)$, $v \in V, w \in W$
2: compute a representation of G as partial k-tree
3: compute all pairs (D_q, Y_j) and sort them to increasing size
4: create a table $GIR(C_p, X_i, D_q, Y_j, \varphi)$, and set value of each element to 0
5: ▷ initial step of dynamic programming
6: **for all** (D_q, Y_j) and (C_p, X_i) with $|Y_j| = |X_i| = 1$ and bijection $\varphi : C_p \to D_q$ **do**
7: **if** φ is an R-isomorphism and $G[C_p \cup X_i] \cong^{R \cup r(\varphi)} H[D_q \cup Y_j]$ **then**
8: $GIR(C_p, X_i, D_q, Y_j, \varphi) \leftarrow 1$
9: **end if**
10: **end for**
11: ▷ decide whether $G[C_p \cup X_i] \cong^{R \cup r(\varphi)} H[D_q \cup Y_j]$ according to Lemma 4
12: **for all** (D_q, Y_j) and (C_p, X_i) with $|Y_j| = |X_j| \geq 2$ in order of increasing size **do**
13: **for all** R-isomorphism φ from C_p to D_q and $y \in Y_j$ **do**
14: **if** conditions 1 and 2 of Lemma 4 are satisfied **then**
15: constructs a bipartite graph B_y
16: **if** B has a perfect matching **then**
17: $GIR(C_p, X_i, D_q, Y_j, \varphi) \leftarrow 1$
18: **end if**
19: **end if**
20: **end for**
21: **end for**
22: ▷ decide whether $G \cong^R H$ according to Lemma 3
23: **for all** k-vertex separator D_q of H and R-isomorphism φ from C_r to D_q **do**
24: constructs a bipartite graph B
25: **if** B has a perfect matching **then**
26: return $G \cong^R H$
27: **end if**
28: **end for**
29: return $G \ncong^R H$

In the lines 12-21, we decide whether, for each (D_q, Y_j), (C_p, X_i), and bijection $\varphi : C_p \mapsto D_q$, $G[C_p \cup X_i] \cong^{R \cup r(\varphi)} H[D_q \cup Y_j]$ is satisfied according to Lemma 4.

In the lines 14-19, we construct bipartite graph B_y and compute a perfect matching of B_y for each $y \in Y_j$. We can check whether the conditions 1 and 2 of Lemma 4 are satisfied in $\mathcal{O}(|Y_j|^2 + m) = \mathcal{O}(|Y_j|^2)$ time because $m \leq |Y_j|$. B_y can be constructed in $\mathcal{O}(\sum_{Y_j^t} \sum_{X_j^s} |Y_j^t|) = \mathcal{O}(\sum_{X_j^s} |Y_j|) = \mathcal{O}(|Y_j|^2)$ time. A perfect matching of B_y can be computed in $\mathcal{O}(m^{2.5}) = \mathcal{O}(|Y_j|^{2.5})$ time (see e.g. [7]). We have that the complexity of the lines 14-19 is $\mathcal{O}\left(|Y_j|^{2.5}\right)$. Thus, the complexity of the lines 12-21 is as follows.

$$\mathcal{O}\left(\sum_{D_q}\sum_{Y_j}\sum_{(C_p,X_i)}\sum_{\varphi}\sum_{y\in Y_j}|Y_j|^{2.5}\right)=\mathcal{O}\left(k!\sum_{D_q}\sum_{(C_p,X_i)}\sum_{Y_j}|Y_j|^{3.5}\right) \tag{3}$$

Because of $\sum_{Y_j}|Y_j|=\mathcal{O}(n)$ for each D_q, we have the following.

$$\mathcal{O}\left(k!\sum_{D_q}\sum_{(C_p,X_i)}n^{3.5}\right)=\mathcal{O}\left(k!n^{k+4.5}\right) \tag{4}$$

Since the complexity of the computation of the lines 23-28 is $\mathcal{O}(\sum_{D_q}\sum_{\varphi}k^2(n^2+n^{2.5}))=\mathcal{O}(k!k^2n^{k+2.5})$, we conclude that the complexity of Algorithm 1 is $\mathcal{O}(k!k^2n^{k+2.5}+k!n^{k+4.5})$. □

4 Faster implementation for some binary relations

In the algorithm 1, we compute a perfect matching of a bipartite graph. This takes $\mathcal{O}(m^{2.5})$ time. However, if a given binary relation R has the following property, we can compute a perfect matching more efficiently.

Property 1. *Let G_1 and G_2 be any induced subgraphs of G and let H_1 and H_2 be any induced subgraphs of H. For any R-isomorphism ξ_{11} from G_1 to H_1, any R-isomorphism ξ_{21} from G_2 to H_1, and any R-isomorphism ξ_{22} from G_2 to H_2, the mapping ξ from G_1 to H_2 that is denoted by $\xi(v)=\xi_{22}(\xi_{21}^{-1}(\xi_{11}(v)))$ is an R-isomorphism.*

If the property holds then any connected component of a bipartite graph is a complete bipartite graph. In such case, we have that the bipartite graph has a perfect matching if and only if cardinality of two vertex sets of any connected component are equal. Thus, we can compute a perfect matching in $\mathcal{O}(m^2)$ time by using a basic graph search algorithm.

If R is empty, any isomorphism is an R-isomorphism and vice versa. Thus, the property clearly holds because of the transitivity of isomorphisms. On the other hand, it is not always true that the property holds. Figure 5 is an example in which the property does not hold. In the figure, let $R=\{(v_1,w_3),(v_1,w_4)\}$. Then, ξ is not an R-isomorphism for any R-isomorphisms ξ_{11},ξ_{21}, and ξ_{22}.

The next lemma shows an example of binary relations that satisfy the property.

Lemma 5. *Let φ be an isomorphism from an induced subgraph of G to an induced subgraph of H. Then the binary relation $r(\varphi)$ satisfies Property 1.*

Proof. Since ξ_{11},ξ_{21}, and ξ_{22} are isomorphism, ξ is also an isomorphism by the transitivity of isomorphisms. Thus, we below show ξ is an $r(\varphi)$-mapping.

We first consider the case where there is a vertex $v\in Dom(\varphi)\cap G_1$. In this case, since $G_1\cong^{r(\varphi)}H_1$, $G_2\cong^{r(\varphi)}H_1$, and $G_2\cong^{r(\varphi)}H_2$, v is also a vertex of G_2 and $\varphi(v)$ is a vertex of $H_1\cap H_2$. Since ξ_{11}, ξ_{21}, and ξ_{22} are $r(\varphi)$-mapping and

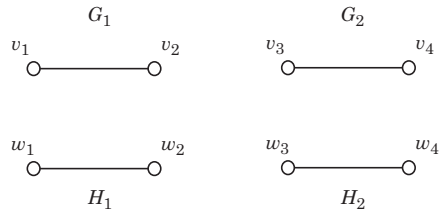

Figure 5: An example of an R that does not satisfy Property 1

$v \in Dom(\varphi)$, $\xi(v) = \xi_{22}(\xi_{21}^{-1}(\xi_{11}(v))) = \xi_{22}(\xi_{21}^{-1}(\varphi(v))) = \xi_{22}(v) = \varphi(v)$. Thus, ξ is an $r(\varphi)$-mapping.

We next consider the case where $Dom(\varphi) \cap G_1 = \emptyset$. In this case, since $G_1 \cong^{r(\varphi)} H_1$, $G_2 \cong^{r(\varphi)} H_1$, and $G_2 \cong^{r(\varphi)} H_2$, we have that $Dom(\varphi) \cap G_2 = \emptyset$, $Im(\varphi) \cap H_1 = \emptyset$ and $Im(\varphi) \cap H_2 = \emptyset$. In such case, any mapping between G_1 and H_2 is an $r(\varphi)$-mapping. Thus, ξ is an R-mapping.

We conclude that ξ is an R-isomorphism, and Property 1 holds. □

The next lemma is derived from Lemma 5. By the lemma, for any binary relation R that satisfies Property 1, we can solve GIR on partial k-trees more efficiently.

Corollary 1. *Algorithm 1 can be implemented in $\mathcal{O}(k!k^2n^{k+2} + k!n^{k+4})$ time if an input binary relation satisfies Property 1.*

Proof. Any mapping that is used to constructs bipartite graphs in Algorithm 1 is an isomorphism from an induced subgraph of G to an induced subgraph of H. According to Lemma 5, binary relations that correspond to such mappings have Property 1. In addition, a binary relation that is a combination of such binary relations also have Property 1. Thus, all binary relations that are used in the algorithm (i.e., an input binary relation, binary relations that correspond to isomorphisms between induced subgraphs, and combinations of them) satisfies Property 1. Therefore any computation of perfect matching in Algorithm 1 is implemented in $\mathcal{O}(m^2)$. We have that $|Y_j|^{2.5}$ of the equation 3 can be replaced by $|Y_j|^2$, and the total complexity of the algorithm is $\mathcal{O}(k!k^2n^{k+2} + k!n^{k+4})$. □

5 Algorithm for the prefix set of graph isomorphism

By Corollary 1, we can solve PrefixGI for partial k-trees in $\mathcal{O}(k!k^2n^{k+2} + k!n^{k+4})$ time, because PrefixGI with inputs G, H, and φ is reduced to GIR with inputs G, H, and $r(\varphi)$. However, an execution of the algorithm spends as much time as the case where the size of domain of φ is small whenever $|Dom(\varphi)|$ is close to the size of the given graphs.

In this section, we propose more efficient algorithm to solve PrefixGI. If $|Dom(\varphi)|$ is large, this algorithm runs faster than Algorithm 1. The algorithm is based on

a reduction to GIR on disconnected partial k-trees. Given inputs G, H, and φ of PrefixGI, we construct inputs G', H', and R' of GIR as follows. Let $G' = (V', E')$ be a subgraph of G that is induced by $V - Dom(\varphi)$. Please remind that this graph could be disconnected (i.e., a set of connected graphs). Also, let $H' = (W', F')$ be a subgraph of H that is induced by $W - Im(\varphi)$. Since any induced subgraph of a partial k-tree is also partial k-tree, each of these connected components is a partial k-tree. We assume that, without loss of generality, G' and H' are sets of m connected components $\{G'_1, \cdots, G'_m\}$ and $\{H'_1, \cdots, H'_m\}$ respectively. We also define $R'(v) \subseteq W'$ for each $v \in V'$ and define binary relation $R' \subseteq V' \times W'$ as follows.

$$R'(v) = \{w : w \in W' \text{ and } \forall z \in Dom(\varphi)[(v, z) \in E \Leftrightarrow (w, \varphi(z)) \in F]\} \tag{5}$$

$$R' = \{(v, w) : v \in V' \text{ and } w \in W' \text{ and } w \notin R'(v)\} \tag{6}$$

We easily see that G is $r(\varphi)$-isomorphic to H if and only if G' is R'-isomorphic to H'. We can decide whether G' is R'-isomorphic to H' as follows.

We construct a bipartite graph $B = (V_B^1, V_B^2, E_B)$ where $V_B^1 = \{G'_1, \ldots, G'_m\}$, $V_B^2 = \{H'_1, \ldots, H'_m\}$, and $(G'_i, H'_j) \in E_B$ if and only if G'_i is R'-isomorphic to H'_j. Then, G' is R'-isomorphic to H' if and only if there is a perfect matching of B.

The bipartite graph B can be constructed by using Algorithm 1 iteratively. Furthermore, we can decide whether a perfect matching exists in $\mathcal{O}(m^2)$ time by using the next lemma.

Lemma 6. *R' satisfies Property 1.*

Proof. Let G'_1 and G'_2 be any two connected components of G' and let H'_1 and H'_2 be any two connected components of H'. Let ξ_{11}, ξ_{21}, and ξ_{22} be R'-isomorphisms from G'_1 to H'_1, from G'_2 to H'_1, and from G'_2 to H'_2 respectively. Then, the mapping ξ from G_1 to H_2 that is defined by $\xi(v) = \xi_{22}(\xi_{21}^{-1}(\xi_{11}(v)))$ is an isomorphism by the transitivity of isomorphism. We below prove ξ is an R'-mapping.

Let v be any vertex of G'_1. Since ξ_{11} is an R'-mapping, $(v, \xi_{11}(v)) \notin R'$ holds. Therefore, we have that $\xi_{11}(v) \in R'(v)$. According to the definitions of $R'(v)$, we have the following.

$$\forall z \in Dom(\varphi)[(v, z) \in E \Leftrightarrow (\xi_{11}(v), \varphi(z)) \in F] \tag{7}$$

Since the same statement holds for any vertex of G'_2 and ξ_{21}, we have the following fact.

$$\forall z \in Dom(\varphi)[(v, z) \in E \Leftrightarrow (\xi_{11}(v), \varphi(z)) \in F \Leftrightarrow (\xi_{21}^{-1}(\xi_{11}(v)), z) \in E] \tag{8}$$

In addition, since the same statement holds for any vertex of G'_2 and ξ_{22}, we have the following fact.

$$\forall z \in Dom(\varphi)[(v, z) \in E \Leftrightarrow (\xi_{21}^{-1}(\xi_{11}(v)), z) \in E \Leftrightarrow (\xi_{22}(\xi_{21}^{-1}(\xi_{11}(v))), \varphi(z) \in F] \tag{9}$$

According to the equation (7), (8), and (9), we have the following fact.

$$\forall z \in Dom(\varphi)[(v,z) \in E \Leftrightarrow (\xi(v), \varphi(z)) \in F] \tag{10}$$

Therefore, $\xi(v) \in R'(v)$ is satisfied. As a result, we have that $(v, \xi(v)) \notin R'$. This indicates that ξ is an R'-mapping, and we complete the proof of the lemma. □

Now, we have the main theorem in this section.

Theorem 2. *There is an $\mathcal{O}(n^2 + d(n-d)^2 + nf(n-d))$ time algorithm to solve the prefix set of graph isomorphism on partial k-trees, where d is the size of the domain of φ and $f(n)$ is the function of the time complexity of Corollary 1.*

Proof. We can construct G' and H' in $\mathcal{O}(n^2)$ time by using ordinary graph search algorithm. $R'(v)$ for every $v \in G'$ can be computed by, for any $w \in H'$ and $z \in Dom(\varphi)$, checking adjacency relation between (v,z) and $(w, \varphi(z))$. This takes $\mathcal{O}(d(n-d)^2)$ time. R' can be computed by using $R'(v)$ in $\mathcal{O}((n-d)^2)$ time. Since $|G'_1| + \cdots + |G'_m| = n-d$, a bipartite graph B can be computed in $\mathcal{O}(mf(|G'_1|) + \cdots + mf(|G'_m|)) = \mathcal{O}(mf(n-d))$ time. According to Lemma 6, we can decide whether there exists a perfect matching of a bipartite graph in $\mathcal{O}(m^2)$ time. Because of $m \leq n$, the total complexity is $\mathcal{O}(n^2 + d(n-d)^2 + nf(n-d))$ time. □

6 Conclusions and future works

In this paper, we designed polynomial time exact algorithms for extensions of GI (e.g., GIR and PrefixGI) on partial k-trees with bounded k. Since we can control image of isomorphisms, these problems have more wide application than GI. We below discuss some observation and future works.

As stated earlier, k-trees is a class of graphs with tree structure. Chordal graphs also are graphs with tree structure. These two graph classes are known to be kinds of intersection graphs. GI for some intersection graph classes are known to be solved in polynomial time, e.g., trees[12], Interval Graphs[11], rooted directed path graphs[2], permutation graphs[6] and ptolemaic graphs[17]. See [5] for a good survey on these graph classes. The inclusion relation between trees, k-trees and chordal graphs is as follows.

$$\text{Trees} \subset k\text{-trees} \subset \text{Chordal graphs}$$

The definition of chordal graphs is similar to that of k-trees, except that any clique of chordal graphs has arbitrary size. Precisely, chrdal graphs are defined as follows.

1. A clique is a chordal graph.
2. If $G = (V, E)$ is a chordal graph, and $x \notin V$, and $C \subseteq V$ is a clique in G, then $H = (V \cup \{x\}, E \cup \{(x,v) : v \in C\})$ is a chordal graph
3. All chordal graph can be formed with rules 1 and 2.

By the definition, although its width is not bounded, chordal graphs also have a tree structure. For example, Figure 6 is a chordal graph and its tree structure.

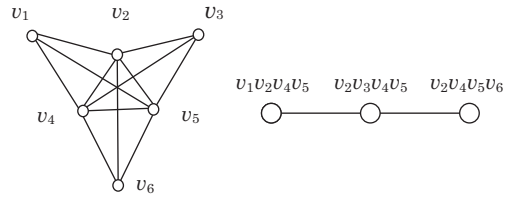

Figure 6: An example of a chordal graph and its tree structure

It is well known that GI for labeled or unlabeled trees are solved in polynomial time. GI for chordal graphs seems to be solvable by using the algorithm for the tree isomorphism problem by collapsing each clique to a node of a tree. However, this approach fails. The chordal graph of Figure 7 has the same tree structure with the graph of Figure 6, but these graphs are not isomorphic. We can attach a label for each node of tree structure as shown in the figures. However, in order to decide whether or not given two chordal graphs are isomorphic, we have to decide not only one-to-one correspondence between nodes of tree structures but also one-to-one correspondence between alphabets of labels (i.e. vertex names of given chordal graphs). It seems that the structure of labeled trees is insufficient to represents chordal graphs. Actually, it is unlikely that GI for chordal graphs is reducible to GI for trees because GI for chordal graphs is known to be GI-complete [11]. A good survey on GI-complete problems appears in [4].

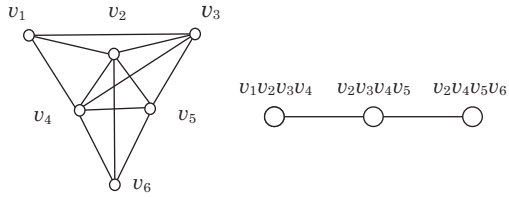

Figure 7: Another example of a chordal graph and its tree structure

Partial k-trees is a class of graphs that are subgraphs of a k-trees. Thus, the class of partial k-trees includes the class of k-trees, whereas it is incomparable with the class of chordal graphs. In contrast with the case of chordal graph, vertex sets of nodes of tree structure of partial k-trees could be arbitrary graphs. Therefore, it seems more difficult to reduce GI for partial k-trees to GI for labeled trees.

The time complexity of algorithms that we proposed in this paper are polynomial for bounded k, but it has large degree. Thus, it is interesting to design more efficient algorithm for GI, GIR, and PrefixGI for partial k-trees.

Acknowledgment

This work was supported by JSPS KAKENHI Grant Number 26330018.

References

[1] Arnborg S., Corneil D., and Proskurowski A., Complexity of finding embeddings in a k-tree, *SIAM J. Alg. Disc. Meth.*, 8, 1987, 277–284.

[2] Babel L., Ponomarenko I. N. and Tinhofer G., The Isomorphism Problems for directed Path Graphs and for Rooted Directed Path Graphs, *J. Algorithms*, 21, 1996, 542–564.

[3] Bodlaender H. L., Polynomial algorithms for graph isomorphism and chromatic index on partial k-trees, *J. Algorithms*, 11, 1990, 631–643.

[4] Booth K. S. and Colbourn C. J., Problems polynomially equivalent to graph isomorphism, Technical Report CS-77-04, Computer Science Department, University of Waterloo, 1979

[5] Brandstädt A., Le V. B. and Spinrad J. P., *Graph Classeses: A Survey*, SIAM, 1999.

[6] Colbourn G.J., On testing isomorphism of permutation graphs, *Networks*, 11, 1981, 13–21.

[7] Hopcroft J. and Karp R. M., An $n^{5/2}$ algorithm for maximum matching in bipartite graphs, *SIAM J. Comput.*, 4, 1975, 225–231.

[8] Köbler J. and Schöning U. and ToranJ., *The Graph Isomorphism Problem: Its Structural Complexity*, Birkhauser, 1993.

[9] Lee J., Han W. S., Kasperovics R., and Lee J. H., An indepth comparison of subgraph isomorphism algorithms in graph databases, *Proceedings of the 39th international conference on Very Large Data Bases*. VLDB Endowment, 2012, 133–144.

[10] Lubiw A., Some NP-complete problems similar to graph isomorphism, *SIAM J. Comput.*, 10(1), 1981, 11–21.

[11] Lueker G. S. and Booth K. S., A Linear Time Algorithm for deciding interval graph isomorphism, *J. ACM*, 26, 1979, 183–195.

[12] Matura D., Subtree isomorphism in $O(n^{5/2})$, *Annals of Discrete Mathematics*, 2, 1978, 91–106.

[13] Nagoya T., Algorithms for Graph Isomorphism with Restriction on Chordal Graphs with Bounded Clique Size, *IEICE Trans. Inf. & Sys.*, J95-D(11), 2012, 1889–1896.

[14] Nagoya T. and Toda S., Computational Complexity of Computing a Partial Solution for the Graph Automorphism Problems, *Theor. Comput. Sci.*, 410, 2009, 2064–2071.

[15] Saltz M., Jain A., Kothari A., Fard A., Miller J. A., Ramaswamy L., An Algorithm for Subgraph Pattern Matching on Very Large Labeled Graphs, *IEEE International Congress on Big Data*, 2014.

[16] Toda S., Computing Automorphism Groups of Chordal Graphs Whose Simplicial Components Are of Small Size, *IEICE Trans. Inf. & Sys.*, E89-D(8), 2006, 2388–2401.

[17] Uehara R. and Uno Y., Laminar Structure of Ptolemaic Graphs with Applications. *Disc. Appl. Math.*, 157, 2009, 1533–1543.

This is an extended version of the paper presented at the International Conference on Big Data Intelligence and Computing (DataCom 2015), Chengdu, China, December 19-21, 2015