On the Laminar Structure of Ptolemaic and Distance Hereditary Graphs

Ryuhei Uehara[*] Yushi Uno[†]

February 25, 2005

Abstract

Ptolemaic graphs are graphs that satisfy ptolemaic inequality for any four vertices. The graph class coincides with the intersection of chordal graphs and distance hereditary graphs. The graph class can also be seen as a natural generalization of block graphs (and hence trees). In this paper, a new characterization of ptolemaic graphs is presented. It is a canonical tree representation based on a laminar structure of cliques. The tree representation is constructed in linear time from a perfect elimination ordering obtained by the lexicographic breadth first search. Hence recognition and graph isomorphism for ptolemaic graphs can be solved in linear time. The tree representation also gives a simple intersection model for ptolemaic graphs. The results are also extended to distance hereditary graphs.

Keywords: algorithmic graph theory, data structure, distance hereditary graphs, intersection model, ptolemaic graphs.

1 Introduction

Recently, many graph classes have been proposed and studied [3, 13]. Among them, the class of chordal graphs is classic and widely investigated. One of the reasons is that the class has a natural intersection model and hence a concise tree representation; a graph is chordal if and only if it is the intersection graph of subtrees of a tree. The tree representation can be constructed in linear time, and the tree is called a clique tree since each node of the tree corresponds to a maximal clique of the chordal graph (see [23]). Another reason is that the class is characterized by a vertex ordering, which is called a perfect elimination ordering. The ordering can also be computed in linear time, and typical way to find it is called the lexicographic breadth first search (LBFS) introduced by Rose, Tarjan, and Lueker [22]. The LBFS is also widely investigated as a tool for recognizing several graph classes (see a comprehensive survey by Corneil [8]). Using those characterizations, many efficient algorithms have been found for chordal graphs; to list a few of them, the maximum weighted clique problem, the maximum weighted independent set problem, the minimum coloring problem [12], the minimum maximal independent set problem [11], and so on. There are also parallel algorithms to solve some of these problems efficiently [18].

Distance in graphs is one of the most important topics in algorithmic graph theory. The class of distance hereditary graphs was introduced by Howorka to deal with the distance property called isometric [15]. Some characterizations of distance hereditary graphs are given by Bandelt and Mulder [1], D'Atri and Moscarini [10], and Hammer and Maffray [14]. Especially, Bandelt and Mulder showed that any distance hereditary graph can be obtained from K_2 by a sequence of extensions called "adding a pendant vertex" and "splitting a vertex." Using the characterizations, many efficient algorithms have been found for distance hereditary graphs [6, 2, 5, 21, 17, 7]. However, the recognition of distance hereditary graphs in linear time is not so simple; Hammer and Maffray's algorithm [14] fails in some cases, and Damiand, Habib, and Paul's algorithm [9] requires to build a cotree in linear time (see [9, Chapter 4] for further details). The cotree can be constructed in linear time by using recent algorithm based on multisweep LBFS approach by Bretscher, Corneil, Habib, and Paul [4].

In this paper, we first focus on the class of ptolemaic graphs. Ptolemaic graphs are graphs that satisfy the ptolemaic inequality $d(x,y)d(z,w) \le d(x,z)d(y,w) + d(x,w)d(y,z)$ for any four vertices x, y, z, w. Howorka showed

[*]School of Information Science, Japan Advanced Institute of Science and Technology (JAIST), Ishikawa, Japan. uehara@jaist.ac.jp

[†]Department of Mathematics and Information Sciences, College of Integrated Arts and Sciences, Osaka Prefecture University, Sakai, Japan. uno@mi.cias.osakafu-u.ac.jp

that the class of ptolemaic graphs coincides with the intersection of the class of chordal graphs and the class of distance hereditary graphs [16]. Hence the results for chordal graphs and distance hereditary graphs can be applied to ptolemaic graphs. On the other hand, the class of ptolemaic graphs is a natural generalization of block graphs, and hence trees (see [25] for the relationships between related graph classes). However, there are relatively few known results specified to ptolemaic graphs. The reason seems that the class of ptolemaic graphs has no useful characterizations from the viewpoint of the algorithmic graph theory. We propose a tree representation of ptolemaic graphs, which is based on the laminar structure of cliques of a ptolemaic graph. The tree representation also gives a natural intersection model for ptolemaic graphs, which is defined over directed trees. The tree representation can be constructed in linear time for a ptolemaic graph. The construction algorithm can also be modified to a recognition algorithm which runs in linear time. In the construction and the recognition, the ordering of the vertices produced by the LBFS plays an important role. Therefore, our result adds the class of ptolemaic graphs to the list of graph classes that can be recognized efficiently using the LBFS. Moreover, the tree representation is canonical up to isomorphism. Hence, using the tree representation, we can solve the graph isomorphism problem for ptolemaic graphs in linear time. (We note that a clique tree of a chordal graph is not canonical and the graph isomorphism problem for chordal graphs is graph isomorphism complete.)

Next, we extend the results for the class of ptolemaic graphs to the class of distance hereditary graphs. We show new characterizations of the class of distance hereditary graphs. The tree representation of the ptolemaic graphs can be extended to the distance hereditary graphs, and that gives a geometric model for distance hereditary graphs. As far as the authors know, there were no known geometric model for distance hereditary graphs.

2 Preliminaries

The *neighborhood* of a vertex v in a graph $G = (V, E)$ is the set $N_G(v) = \{u \in V \mid \{u, v\} \in E\}$, and the *degree* of a vertex v is $|N_G(v)|$ and is denoted by $\deg_G(v)$. For a subset U of V, we denote by $N_G(U)$ the set $\{v \in V \mid v \in N(u) \text{ for some } u \in U\}$. If no confusion can arise we will omit the index G. Given a graph $G = (V, E)$ and a subset U of V, the induced subgraph by U, denoted by $G[U]$, is the graph (U, E'), where $E' = \{\{u, v\} \mid u, v \in U \text{ and } \{u, v\} \in E\}$. Given a graph $G = (V, E)$, its *complement* is defined by $\overline{E} = \{\{u, v\} \mid \{u, v\} \notin E\}$, and is denoted by $\overline{G} = (V, \overline{E})$. A vertex set I is an *independent set* if $G[I]$ contains no edges, and then the graph $\overline{G}[I]$ is said to be a *clique*. Two vertices u and v are said to be a *twin* if $N(u) \setminus \{v\} = N(v) \setminus \{u\}$. For a twin u and v, we say that u is a *strong sibling* of v if $\{u, v\} \in E$, and a *weak sibling* if $\{u, v\} \notin E$.

Given a graph $G = (V, E)$, a sequence of the distinct vertices $v_1, v_2, \ldots, v_l$ is a *path*, denoted by $(v_1, v_2, \ldots, v_l)$, if $\{v_j, v_{j+1}\} \in E$ for each $1 \leq j < l$. The *length* of a path is the number of edges on the path. For two vertices u and v, the *distance* of the vertices, denoted by $d(u, v)$, is the minimum length of the paths joining u and v. A *cycle* is a path beginning and ending with the same vertex.

An edge which joins two vertices of a cycle but is not itself an edge of the cycle is a *chord* of that cycle. A graph is *chordal* if each cycle of length at least 4 has a chord. Given a graph $G = (V, E)$, a vertex $v \in V$ is *simplicial* in G if $G[N(v)]$ is a clique in G. An ordering $v_1, \ldots, v_n$ of the vertices of V is a *perfect elimination ordering* (PEO) of G if the vertex v_i is simplicial in $G[\{v_i, v_{i+1}, \ldots, v_n\}]$ for all $i = 1, \ldots, n$. Once a vertex ordering is fixed, we denote $N(v_j) \cap \{v_{i+1}, \ldots, v_n\}$ by $N_{>i}(v_j)$. We also use the notions "min" and "max" to denote the first and the last vertices in an ordered set of vertices, respectively. It is known that a graph is chordal if and only if it has a PEO (see [3, Section 1.2] for further details). A typical way of finding a perfect elimination ordering of a chordal graph in linear time is the lexicographic breadth first search (LBFS), which is introduced by Rose, Tarjan, and Lueker [22], and a comprehensive survey is presented by Corneil [8].

It is also known that a graph $G = (V, E)$ is chordal if and only if it is the intersection graph of subtrees of a tree T (see [3, Section 1.2] for further details). Let T_v denote the subtree of T corresponding to the vertex v in G. Then we can assume that each node c in T corresponds to a maximal clique C of G such that C contains v on G if and only if T_v contains c on T. Such a tree T is called a *clique tree* of G. From a perfect elimination ordering of a chordal graph G, we can construct a clique tree of G in linear time [23]. We sometimes unify a node c of a clique tree T with a maximal clique (or a vertex set) C of G.

Given a graph $G = (V, E)$ and a subset U of V, an induced connected subgraph $G[U]$ is *isometric* if the distances in $G[U]$ are the same as in G. A graph G is *distance hereditary* if G is connected and every induced path in G is isometric. We will use the following characterization of distance hereditary graphs stated by Bandelt and Mulder [1]:

Theorem 1 *A graph G with at least two vertices is distance hereditary if and only if G can be obtained from K_2 by a sequence of extensions of (α) pick any vertex x in G and add x' with an edge $\{x, x'\}$, (β) pick any vertex x in*

G and add x' with edges $\{x, x'\}$ and $\{x', y\}$ for all $y \in N(x)$, or (γ) pick any vertex x in G and add x' with edges $\{x', y\}$ for all $y \in N(x)$.

In case (α), we say that the new graph is obtained by attaching a *pendant vertex* x', and in cases (β) and (γ), we say that the new graph is obtained by *splitting* the vertex x. We note that in case (β), x and x' are strong siblings, and in case (γ), x and x' are weak siblings.

A connected graph G is *ptolemaic* if for any four vertices u, v, w, x of G, $d(u, v)d(w, x) \leq d(u, w)d(v, x) + d(u, x)d(v, w)$. We will use the following characterization of ptolemaic graphs due to Howorka [16]:

Theorem 2 *The following conditions are equivalent: (1) G is ptolemaic; (2) G is distance hereditary and chordal; (3) for all distinct nondisjoint maximal cliques P, Q of G, $P \cap Q$ separates $P \setminus Q$ and $Q \setminus P$.*

Let V be a set of n vertices. Two sets X and Y are said to be *incomparable* if $X \cap Y \neq \emptyset$, $X \setminus Y \neq \emptyset$, and $Y \setminus X \neq \emptyset$. A family $\mathcal{F} \subseteq 2^V \setminus \{\{\emptyset\}\}$ is said to be *laminar* if $\mathcal{F}$ contains no incomparable sets; that is, for any pair of two distinct sets X and Y in $\mathcal{F}$ satisfy either $X \cap Y = \emptyset$, $X \subset Y$, or $Y \subset X$. Given a laminar family $\mathcal{F}$, we define *laminar digraph* $\vec{T}(\mathcal{F}) = (\mathcal{F}, \vec{E}_{\mathcal{F}})$ as follows; $\vec{E}_{\mathcal{F}}$ contains an arc (X, Y) if and only if $X \subset Y$ and there are no other subset Z such that $X \subset Z \subset Y$, for any sets X and Y. We denote the underlying graph of $\vec{T}(\mathcal{F})$ by $T(\mathcal{F}) = (\mathcal{F}, E_{\mathcal{F}})$. The following two lemmas for the laminar digraph are known (see, e.g., [20, Chapter 2.2]);

Lemma 3 *$T(\mathcal{F})$ is a forest.*

Lemma 4 *If a family $\mathcal{F} \subseteq 2^V$ is laminar, we have $|\mathcal{F}| \leq 2|V| - 1$.*

Hence, hereafter, we call $T(\mathcal{F})$ ($\vec{T}(\mathcal{F})$) a (directed) laminar forest. We regard each maximal (directed) tree in the laminar forest $T(\mathcal{F})$ ($\vec{T}(\mathcal{F})$) as a (directed) tree rooted at the maximal set, whose outdegree is 0 in $\vec{T}(\mathcal{F})$. We define a *label* of each node S_0 in $\vec{T}(\mathcal{F})$, denoted by $\ell(S_0)$, as follows: If S_0 is a leaf, $\ell(S_0) = S_0$. If S_0 is not a leaf and has children $S_1, S_2, \ldots, S_h$, $\ell(S_0) = S_0 \setminus (S_1 \cup S_2 \cup \cdots \cup S_h)$. That is, each vertex v in V appears in $\ell(S)$ where S is the minimal set containing v. Since $\mathcal{F}$ is laminar, each vertex in V appears exactly once in $\ell(S)$ for some $S \subseteq V$, and its corresponding node is uniquely determined. We note that internal nodes in $\vec{T}(\mathcal{F})$ have a label $\emptyset$ when it is partitioned completely by its subsets in $\mathcal{F}$. (For example, for $V = \{a, b\}$ and $\mathcal{F} = \{X = \{a, b\}, Y = \{a\}, Z = \{b\}\}$, we have $\ell(X) = \{\emptyset\}$, $\ell(Y) = \{a\}$, and $\ell(Z) = \{b\}$.)

3 A Tree Representation of Ptolemaic Graphs

In this section, we show that ptolemaic graphs have a canonical tree representation, and it can be constructed in linear time. We also show some applications.

3.1 A Tree Representation

For a ptolemaic graph $G = (V, E)$, let $\mathcal{M}(G)$ be the set of all maximal cliques, i.e.,

$$\mathcal{M}(G) := \{M \mid M \text{ is a maximal clique in } G\},$$

and $\mathcal{C}(G)$ be the set of nonempty vertex sets defined below:

$$\mathcal{C}(G) := \bigcup_{S \subseteq \mathcal{M}(G)} \{C \mid C = \cap_{M \in S} M, C \neq \emptyset\}.$$

Each vertex set $C \in \mathcal{C}(G)$ is a nonempty intersection of some maximal cliques. Hence, $\mathcal{C}(G)$ contains all maximal cliques, and each C in $\mathcal{C}(G)$ induces a clique. We also denote by $\mathcal{L}(G)$ the set $\mathcal{C}(G) \setminus \mathcal{M}(G)$. That is, each vertex set $L \in \mathcal{L}(G)$ is an intersection of two or more maximal cliques, and hence L is a non-maximal clique. The following properties are crucial.

Theorem 5 *Let $G = (V, E)$ be a ptolemaic graph. Let $\mathcal{F}$ be a family of sets in $\mathcal{L}(G)$ such that $\cup_{L \in \mathcal{F}} L \subset M$ for some maximal clique $M \in \mathcal{M}(G)$. Then $\mathcal{F}$ is laminar.*

Proof. To derive a contradiction, we assume that $\mathcal{F}$ is not laminar. Then we have two incomparable vertex sets L_1 and L_2 which are properly contained in the maximal clique M. Let v, v_1, v_2 be vertices in $L_1 \cap L_2$, $L_1 \setminus L_2$, and $L_2 \setminus L_1$, respectively. By definition, there are sets of maximal cliques $M_1^1, M_1^2, \ldots, M_1^a, M_2^1, M_2^2, \ldots, M_2^b$ such that $L_1 = M_1^1 \cap M_1^2 \cap \cdots \cap M_1^a$ and $L_2 = M_2^1 \cap M_2^2 \cap \cdots \cap M_2^b$. Here, if every M_1^i with $1 \leq i \leq a$ contains v_2, we have $v_2 \in L_1$. Thus, there is a maximal clique M_1^i with $v_2 \notin M_1^i$. Similarly, there is a maximal clique M_2^j with $v_1 \notin M_2^j$. Let L be $M_1^i \cap M_2^j$. Then we have $v_1, v_2 \notin L$ and $v \in L$ (hence $L \neq \emptyset$). Therefore $v_1 \in M_1^i \setminus L$ and $v_2 \in M_2^j \setminus L$. Moreover, since v_1, v_2 are in M, $\{v_1, v_2\} \in E$. Thus, $L = M_1^i \cap M_2^j$ does not separate $M_1^i \setminus L$ and $M_2^j \setminus L$, which contradicts Theorem 2(3). ∎

Lemma 6 *Let C_1, C_2 be any incomparable sets in $\mathcal{C}(G)$ for a ptolemaic graph $G = (V, E)$. Then $C_1 \cap C_2$ separates $C_1 \setminus C_2$ and $C_2 \setminus C_1$.*

Proof. Let $C := C_1 \cap C_2$. By definition of $\mathcal{C}(G)$, $C \in \mathcal{C}(G)$. Let C_i' be the sets in $\mathcal{C}(G)$ such that $C \subset C_i' \subset C_i$ and there is no other C' with $C \subset C' \subset C_i'$ for $i = 1, 2$. We first observe that C_1' and C_2' are incomparable: If $C_1' = C_2'$, we have $C_1' = C_2' \subseteq C_1 \cap C_2$ which contradicts that $C = C_1 \cap C_2$ and $C \subset C_1'$. On the other hand, if $C_1' \subset C_2'$, we have $C \subset C_1' \subset C_2'$ which contradicts the definition of C_2'.

We show that C separates C_1' and C_2'. Let M_i be maximal cliques that contains C_i' for $i = 1, 2$ such that M_1 is incomparable to C_2' and M_2 is incomparable to C_1. Let $C_c := M_1 \cap M_2$. By definition, $C \subseteq C_c$. It is sufficient to show that $C = C_c$. To derive contradictions, we assume that $v \in C_c \setminus C$. We first assume that $v \in C_1' \setminus C_2'$. In the case, since M_2 and C_1' are incomparable, $\mathcal{C}$ contains a set C'' with $(C_1' \cap C_2') \subset ((C_1' \cap C_2') \cup \{v\}) \subseteq C'' \subset C_1'$, which is a contradiction. Thus, we have $v \notin C_1'$ and $v \notin C_2'$.

By definition of C_1', there are maximal cliques $M_1^1, M_1^2, \ldots, M_1^k$ such that $C_1' = \cap_{i=1}^k M_1^i$. Since $v \notin C_1'$, there is at least one maximal clique M_1^i with $v \notin M_1^i$. Similarly, there is at least one maximal clique M_2^j with $C_2' \subseteq M_2^j$ and $v \notin M_2^j$. However, $C_1' \subseteq M_1^i$, $C_2' \subseteq M_2^j$, and $v \notin M_1^i \cup M_2^j$ imply that $M_1^i \setminus M_2^j$ and $M_2^j \setminus M_1^i$ are connected by v. This is a contradiction to Theorem 2(3). Hence $M_1^i \cap M_2^j = M_1 \cap M_2 = C_1' \cap C_2' = C_1 \cap C_2$, and it is a separator. ∎

Now we define a directed graph $\vec{T}(\mathcal{C}(G)) = (\mathcal{C}(G), A(G))$ for a given ptolemaic graph $G = (V, E)$ as follows: two nodes $C_1, C_2 \in \mathcal{C}(G)$ are joined by an arc (C_1, C_2) if and only if $C_1 \subset C_2$ and there is no other C in $\mathcal{C}(G)$ such that $C_1 \subset C \subset C_2$. We denote by $T(\mathcal{C}(G))$ the underlying graph of $\vec{T}(\mathcal{C}(G))$.

Theorem 7 *A graph $G = (V, E)$ is ptolemaic if and only if the graph $T(\mathcal{C}(G))$ is a tree.*

Proof. We first assume that G is ptolemaic and show that $T(\mathcal{C}(G))$ is a tree. It is not difficult to see that $T(\mathcal{C}(G))$ is connected. Thus, to derive contradictions, we assume that $T(\mathcal{C}(G))$ contains a cycle $(C_1, C_2, \ldots, C_k, C_1)$, which is a minimal cycle without chords on $T(\mathcal{C}(G))$. Since $C_1 \subset C_2 \subset \cdots \subset C_k \subset C_1$ (or vice versa) is impossible, there is a node C_a with $C_{a-1} \supset C_a \subset C_{a+1}$ for some a. Without loss of generality, we assume that $|C_a|$ is the smallest among such vertex sets on the cycle. Let C_x and C_y be the nodes on the cycle such that $C_{x-1} \subset C_x \supset C_{x+1} \supset \cdots \supset C_{a-1} \supset C_a \subset C_{a+1} \subset \cdots \subset C_{y-1} \subset C_y \supset C_{y+1}$. It is not difficult to see that C_{a-1} and C_{a+1}, and hence C_x and C_y are incomparable. Thus, by Lemma 6, C_a separates $C_x \setminus C_y$ and $C_y \setminus C_x$. Since C_a is a separator, we let G_x and G_y be the connected components that contain $C_x \setminus C_y$ and $C_y \setminus C_x$ on $G[V \setminus C_a]$, respectively.

Now we consider the path $\mathsf{P} = (C_x, C_{x-1}, C_{x-2}, \ldots, C_{y+2}, C_{y+1}, C_y)$ which does not contain C_a. However, since C_a is a separator, P contains at least one vertex set C_b in $\mathcal{C}$ with $C_a \cap C_b \neq \emptyset$. If $(C_x \cap C_b) \setminus C_a \neq \emptyset$ and $(C_y \cap C_b) \setminus C_a \neq \emptyset$, $C_x \setminus C_y$ and $C_y \setminus C_x$ are connected on $G[V \setminus C_a]$ since C_b is a clique. Hence each C_b with $C_a \cap C_b \neq \emptyset$ satisfies $(C_x \cap C_b) \setminus C_a = \emptyset$ or $(C_y \cap C_b) \setminus C_a = \emptyset$. Since P connects G_x and G_y through the separator C_a, we have at least two vertex sets C_b and C_b' such that $(C_y \cap C_b) \setminus C_a = \emptyset$ and $(C_x \cap C_b') \setminus C_a = \emptyset$. Moreover, since C_a separates G_x and G_y, we have $C_b \cap C_b' \subseteq C_a$. If $C_b \cap C_b' \subset C_a$, P contains smaller separator than C_a. Thus $C_b \cap C_b' = C_a$. Then P has to contain C_a between C_b and C_b', which contradicts the minimality of the cycle.

Therefore, $T(\mathcal{C}(G))$ is a tree.

It is easy to see that G is ptolemaic if $T(\mathcal{C}(G))$ is a tree; for each pair of distinct nondisjoint maximal cliques M_1 and M_2, $(M_1 \cap M_2)$ separates $T(\mathcal{C}(G))$, and hence G. ∎

Hereafter, given a ptolemaic graph $G = (V, E)$, we call $T(\mathcal{C}(G))$ ($\vec{T}(\mathcal{C}(G))$) a *(directed) clique laminar tree* of G. We extend the label of a laminar forest to the directed clique laminar tree naturally: Each node C_0 in $\mathcal{C}(G)$ has a label $\ell(C_0) := C_0 \setminus (C_1 \cup C_2 \cup \cdots \cup C_h)$, where (C_i, C_0) is an arc on $\vec{T}(\mathcal{C}(G))$ for $1 \leq i \leq h$. Intuitively, we additionally define the label of a maximal clique as follows; the label of a maximal clique is the set of vertices which are not contained in any other maximal cliques. We note that for each vertex in G its corresponding node in $T(\mathcal{C}(G))$ is uniquely determined by maximal cliques. Therefore, we can define the mapping from each vertex to the vertex set in $\mathcal{C}$ in $T(\mathcal{C}(G))$: We denote by $C(v)$ the clique C with $v \in \ell(C)$. When we know whether $C(v)$ is in $\mathcal{M}$

or $\mathcal{L}$, we specify it by writing $C_M(v)$ or $C_L(v)$. An example is given in Figure 2. In Figure 2, each single rectangle represents a non-maximal clique, each double rectangle represents a maximal clique, and each rectangle contains its label. We also note that from $\vec{T}(\mathcal{C}(G))$ with labels, we can reconstruct the original ptolemaic graph uniquely up to isomorphism. That is, two ptolemaic graphs G_1 and G_2 are isomorphic if and only if labeled $\vec{T}(\mathcal{C}(G_1))$ is isomorphic to labeled $\vec{T}(\mathcal{C}(G_2))$.

Intuitively, a clique laminar tree subdivides a clique tree of a chordal graph. For a chordal graph, maximal cliques are joined in a looser way in the sense that a clique tree for a chordal graph is not always uniquely determined up to isomorphism. The clique laminar tree subdivides the relationships between maximal cliques by using their laminar structure.

The following properties of $\vec{T}(\mathcal{C}(G))$ is easy to see, and useful from the algorithmic point of view:

Corollary 8 *If G is a ptolemaic graph, we have the following: (1) For each maximal clique M in $\mathcal{M}(G)$, $\ell(M)$ consists of simplicial vertices in M. (2) The vertices in a maximal clique M in $\mathcal{M}(G)$ induce a maximal directed subtree of $\vec{T}(\mathcal{C}(G))$ rooted at the node M. (3) Each leaf in $T(\mathcal{C}(G))$ corresponds to a maximal clique in $\mathcal{M}(G)$.*

It is well known that a graph is chordal if and only if it is the intersection graph of subtrees of a tree. By Theorem 7, we obtain an intersection model for ptolemaic graphs as follows:

Corollary 9 *Let $\vec{T}$ be any directed graph such that its underlying graph T is a tree. Let $\mathcal{T}$ be any set of subtrees $\vec{T_v}$ such that $\vec{T_v}$ consists of a root C and all vertices reachable from C in $\vec{T}$. Then the intersection graph over $\mathcal{T}$ is ptolemaic. On the other hand, for any ptolemaic graph, there exists such an intersection model.*

Proof. The directed clique laminar tree $\vec{T}(\mathcal{C}(G))$ is the base directed graph of the intersection model. For each $v \in V$, we define the root C such that $v \in \ell(C)$. ∎

3.2 A Linear Time Construction of Clique Laminar Trees

The main theorem in this section is the following:

Theorem 10 *Given a ptolemaic graph $G = (V, E)$, the directed clique laminar tree $\vec{T}(\mathcal{C}(G))$ can be constructed in $O(|V| + |E|)$ time.*

We will make the directed clique laminar tree $\vec{T}(\mathcal{C}(G))$ by separating the vertices in G into the vertex sets in $\mathcal{C}(G) = \mathcal{M}(G) \cup \mathcal{L}(G)$.

We first compute (and fix) a perfect elimination ordering $v_1, v_2, \ldots, v_n$ by the LBFS. The outline of our algorithm is similar to the algorithm for constructing a clique tree for a given chordal graph due to Spinrad in [23]. For each vertex $v_n, v_{n-1}, \ldots, v_2, v_1$, we add it into the tree and update the tree. For the current vertex v_i, let $v_j := \min\{N_{>i}(v_i)\}$. Then, in Spinrad's algorithm [23], there are two cases to consider: $N_{>i}(v_i) = C(v_j)$ or $N_{>i}(v_i) \subset C(v_j)$. The first case is easy; just add v_i into $C(v_j)$. In the second case, Spinrad's algorithm adds a new maximal clique $C(v_i)$ that consists of $N_{>i}(v_i) \cup \{v_i\}$. However, in our algorithm, involved case analysis is required. For example, in the latter case, the algorithm have to handle three vertex sets; two maximal cliques $\{v_i\} \cup N_{>i}(v_i)$ and $C(v_j)$ together with one vertex set $N_{>i}(v_i)$ shared by them. In this case, intuitively, our algorithm makes three distinct sets C_M with $\ell(C_M) = \{v_i\}$, C_L with $\ell(C_L) = N_{>i}(v_i)$, and C with $\ell(C) = C(v_j) \setminus N_{>i}(v_i)$, and adds two arcs (C_L, C_M) and (C_L, C); this means that v_i is in $C_M = N_{>i}(v_i) \cup \{v_i\}$, C is a clique $C(v_j)$, and C_L is the vertex set shared by C_M and C. However, our algorithm has to handle more complicated cases since the set $C(v_j)$ (and hence $N_{>i}(v_i)$) can already be partitioned into some vertex sets.

In $\vec{T}(\mathcal{C}(G))$, each node C stores $\ell(C)$. Hence each vertex in G appears exactly once in the tree. To represent it, each vertex v has a pointer to the node $C(v)$ in $\mathcal{C}(G) = \mathcal{M}(G) \cup \mathcal{L}(G)$. The detail of the algorithm is described as CliqueLaminarTree shown in Figure 1, and an example of the construction is depicted in Figure 2. In Figure 2, the left-hand graph gives a ptolemaic graph, and the right-hand trees are clique laminar trees constructed (a) after adding the vertices $16, 15, 14, 13, 12, 11$, (b) after adding the vertices $16, 15, 14, 13, 12, 11, 10$, (c) after adding the vertices $16, 15, 14, 13, 12, 11, 10, 9, 8$, and (d) after adding all the vertices. We show the correctness and a complexity analysis of the algorithm.

We will use the following property of a PEO found by the LBFS of a chordal graph:

Lemma 11 *Let $v_1, v_2, \ldots, v_n$ be a PEO found by the LBFS. Then $i < j$ implies* $\max\{N(v_i)\} \leq \max\{N(v_j)\}$.

Algorithm 1: CliqueLaminarTree

```
Input : A ptolemaic graph $G = (V, E)$ with a PEO $v_1, v_2, \ldots, v_n$ obtained by the LBFS,
Output: A clique laminar tree $T$.
initialize $T$ by the clique $C_M(v_n) := \{v_n\}$ and set the pointer from $v_n$ to $C_M(v_n)$;
for $i := n - 1$ down to 1 do
    let $v_j := \min\{N_{>i}(v_i)\}$;
    switch condition of $N_{>i}(v_i)$ do
        case (1) $N_{>i}(v_i) = C_M(v_j)$
            update $\ell(C_M(v_j)) := \ell(C_M(v_j)) \cup \{v_i\}$ and $\left|C_M(v_j)\right| := \left|C_M(v_j)\right| + 1$;
            set $C_M(v_i) := C_M(v_j)$;
        case (2) $N_{>i}(v_i) = C_L(v_j)$
            make a new maximal clique $C_M(v_i)$ with $\ell(C_M(v_i)) := \{v_i\}$ and $|C_M(v_i)| := \left|C_L(v_j)\right| + 1$;
            add an arc $(C_L(v_j), C_M(v_i))$;
        case (3) $N_{>i}(v_i) \subset C(v_j)$ and $\left|\ell(C(v_j))\right| = \left|C(v_j)\right|$
            update $\ell(C(v_j)) := \ell(C(v_j)) \setminus N_{>i}(v_i)$ and $\left|\ell(C(v_j))\right| := \left|\ell(C(v_j))\right| - |N_{>i}(v_i)|$;
            make a new vertex set $L := N_{>i}(v_i)$ with $\ell(L) := N_{>i}(v_i)$ and $|L| := |N_{>i}(v_i)|$;
            make a new maximal clique $C_M(v_i)$ with $\ell(C_M(v_i)) = \{v_i\}$ and $|C_M(v_i)| := |L| + 1$;
            add arcs $(L, C(v_j))$ and $(L, C_M(v_i))$;
        case (4) $N_{>i}(v_i) \subset C(v_j)$ and $\left|\ell(C(v_j))\right| < \left|C(v_j)\right|$
            make a new vertex set $L := N_{>i}(v_i)$ with $\ell(L) := N_{>i}(v_i) \cap \ell(C(v_j))$ and $|L| := |N_{>i}(v_i)|$;
            update $\ell(C(v_j)) := \ell(C(v_j)) \setminus L$ and $\left|\ell(C(v_j))\right| := \left|\ell(C(v_j))\right| - |L|$;
            make a new maximal clique $C_M(v_i)$ with $\ell(C_M(v_i)) = \{v_i\}$ and $|C_M(v_i)| = |L| + 1$;
            remove the arc $(L', C(v_j))$ with $L' \subset L$ and add an arc $(L', L)$;
            add arcs $(L, C(v_j))$ and $(L, C_M(v_i))$;
        end
    end
    set the pointer from $v_i$ to $C(v_i)$;
end
return $T$.
```

Figure 1: A linear time algorithm for the clique laminar tree T of a ptolemaic graph $G = (V, E)$.

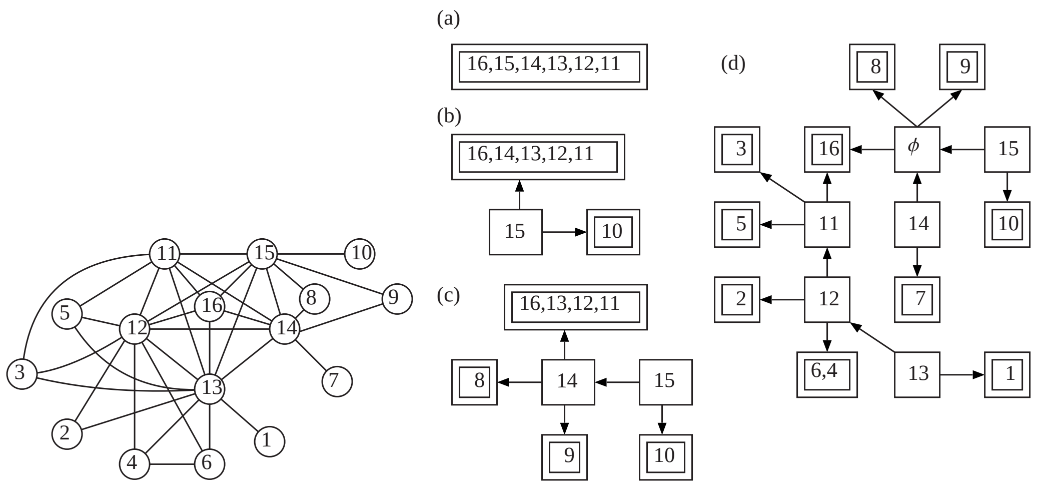

Figure 2: A ptolemaic graph and its clique laminar tree.

Proof. Let v_k be $\max\{N(v_i)\}$. If v_k is a neighbor of v_j, we have done. Hence we assume that $v_k \notin N(v_j)$. Then Theorem 1 in [8] implies that v_j should have a neighbor $v_{k'}$ with $k' > k$. ∎

We assume that Algorithm CliqueLaminarTree is going to add v_i, and let $v_j := \min\{N_{>i}(v_i)\}$. We will show that all possible cases are listed, and in each case, CliqueLaminarTree correctly manages the nodes in $C(G)$ and their labels in $O(\deg(v_i))$ time. The following lemma drastically decreases the number of possible cases, and simplifies the algorithm.

Lemma 12 *Let v_k be $\max\{N_{>i}(v_i)\}$. We moreover assume that the set $N_{>i}(v_i)$ has already been divided into some distinct vertex sets $L_1, L_2, \dots, L_h$. Then, there is an ordering of the sets such that $v_k \in L_1 \subset L_2 \subset \cdots \subset L_h$.*

Proof. We first observe that $G[\{v_i, v_{i+1}, \dots, v_n\}]$ is ptolemaic if G is ptolemaic since any vertex induced subgraph of a chordal graph is chordal, and any vertex induced subgraph of a distance hereditary graph is distance hereditary.

We assume that there is a vertex set $L \subset N_{>i}(v_i)$ such that L does not contain v_k. Then, there is a vertex $v_{i'}$ with $i' > i$ that makes the vertex set L before v_i. Since $\{v_{i'}, v_k\} \notin E$, by Lemma 11, $v_{i'}$ has another neighbor $v_{k'}$ with $k' > k$. By the property of the LBFS, it is easy to see that $G[\{v_k, \dots, v_n\}]$ is connected. Let M_i be a maximal clique $\{v_i\} \cup N_{>i}(v_i)$, and $M_{i'}$ be a maximal clique that contains $\{v_{i'}\} \cup L$. Then, $M_i \cap M_{i'} = L$ which contains no vertex in $G[\{v_k, \dots, v_n\}]$. On the other hand, we have $\{v_i, v_k\}, \{v_{i'}, v_{k'}\} \in E$. Hence, $M_i \cap M_{i'}$ does not separate $M_i \setminus M_{i'}$ and $M_{i'} \setminus M_i$. Therefore $G[\{v_i, v_{i+1}, \dots, v_n\}]$ is not ptolemaic by Theorem 2(3), which is a contradiction. Thus we have $v_k \in L$, and hence, all the vertex sets $L_1, L_2, \dots, L_h$ contain v_k. The vertex set $N_{>i}(v_i)$ is contained in a maximal clique in the ptolemaic graph $G[\{v_i, v_{i+1}, \dots, v_n\}]$. Hence by Theorem 5, $L_1, L_2, \dots, L_h$ are laminar. Therefore, we have $v_k \in L_1 \subset L_2 \subset \cdots \subset L_h$ for some suitable ordering. ∎

Since the graph G is chordal and the vertices are ordered in a perfect elimination ordering, $N_{>i}(v_i)$ induces a clique. By Lemma 12, we have three possible cases; (a) $N_{>i}(v_i) = C(v_j)$, (b) $N_{>i}(v_i) \subset C(v_j)$ and there are no vertex sets in $N_{>i}(v_i)$, and (c) $N_{>i}(v_i) \subset C(v_j)$ and there are vertex sets $L_1 \subset L_2 \subset \cdots \subset L_h \subset N_{>i}(v_i)$. In the last case, we note that $L_h \neq N_{>i}(v_i)$; otherwise, we have $v_j \in L_h$, or consequently, $L_h = C(v_j) = N_{>i}(v_i)$, which is case (a).

(a) $N_{>i}(v_i) = C(v_j)$: We have two subcases; $C(v_j)$ is a maximal clique (i.e. $N_{>i}(v_i) = C_M(v_j)$) or $C(v_j)$ is a non-maximal clique (i.e. $N_{>i}(v_i) = C_L(v_j)$). In the former case, we just update $C_M(v_j)$ by $C_M(v_j) \cup \{v_i\}$. This is case (1) in CliqueLaminarTree. In the latter case, there are other vertex set that contains $C_L(v_j)$ as a subset. Thus we add a new maximal clique $C_L(v_j) \cup \{v_i\}$. More precisely, we add a new node $C_M(v_i)$ with $\ell(C_M(v_i)) = \{v_i\}$ and $|C_M(v_i)| = |C_L(v_j)| + 1$, and a new arc $(C_L(v_j), C_M(v_i))$. This is done in case (2) of CliqueLaminarTree. We can check if $N_{>i}(v_i) = C(v_j)$ by checking if $|N_{>i}(v_i)| = |C(v_j)|$ in $O(1)$ time. Thus it is easy to see that time complexity is $O(1)$ in both cases.

(b) $N_{>i}(v_i) \subset C(v_j)$ and there are no vertex sets in $N_{>i}(v_i)$: We remove $N_{>i}(v_i)$ from $C(v_j)$ and make a new vertex set $N_{>i}(v_i)$ shared by $C(v_j)$ and $C_M(v_i) = \{v_i\} \cup N_{>i}(v_j)$. We can observe that $N_{>i}(v_i) \subset C(v_j)$ and there are no vertex sets in $N_{>i}(v_i)$ if and only if $|N_{>i}(v_i)| < |C(v_j)|$ and $|\ell(C(v_j))| = |C(v_j)|$. Thus, CliqueLaminarTree recognizes this case in $O(1)$ time, and handles it in case (3). It is easy to see that case (3) can be done in $O(|N_{>i}(v_i)|) = O(\deg(v_i))$ time. We note that, in the case, we do not mind if $C(v_j)$ is maximal or not. In any case, the property does not change for $C(v_j)$.

(c) $N_{>i}(v_i) \subset C(v_j)$ and there are vertex sets $L_1 \subset L_2 \subset \cdots \subset L_h \subset N_{>i}(v_i)$: We first observe that the nodes $L_1, L_2, \dots, L_h$, and $C(v_j)$ form a directed path in $\vec{T}$ in the case. (Hence we can recognize this case in $O(|N_{>i}(v_i)|) = O(\deg(v_i))$ time, which will be used in Theorem 13.) Thus we make a new vertex set $L := N_{>i}(v_i)$ with $\ell(L) = N_{>i}(v_i) \setminus L_h$. The set $N_{>i}(v_i) \setminus L_h$ is given by $N_{>i}(v_i) \cap \ell(C(v_j))$. Then we update $\ell(C(v_j))$ by $\ell(C(v_j)) \setminus N_{>i}(v_i)$. It is easy to add a maximal clique $C_M(v_i) = \{v_i\} \cup N_{>i}(v_i)$. Next, we have to update arcs around $C(v_j)$. By Lemma 12, this process is simple; we can find L_h in $O(\deg(v_i))$ time, and there are no other vertex set L' that has an arc $(L', C(v_j))$ which has to be updated. We note that there can be some vertex set L' with an arc $(L', C(v_j))$. But L' is independent from L in this case, and hence we do not have to mind it. Finally, we change the arc $(L_h, C(v_j))$ to (L_h, L), and add the arcs $(L, C(v_j))$ and $(L, C_M(v_i))$. Therefore the time complexity in the last case is $O(\deg(v_i))$ time.

By the above case analyses, Theorem 10 is settled.

3.3 Applications

Theorem 13 *The recognition problem for ptolemaic graphs can be solved in linear time.*

Proof.(Sketch.) Using the LBFS, we can obtain the perfect elimination ordering of G if G is chordal in linear time (and reject it if G is not chordal). For a chordal graph, we run modified CliqueLaminarTree. It is not difficult to modify CliqueLaminarTree to reject it if G is not distance hereditary. The key fact is that, if G is ptolemaic, $N_{>i}(v_i)$ corresponds to a maximal directed path in $\vec{T}(\mathcal{C}(G))$ as follows; suppose that we have vertex sets $L_1 \subset L_2 \subset \cdots \subset L_h \subset N_{>i}(v_i) \subset C(v_j)$ in case (c). In the case, (1) the nodes $L_1, L_2, \ldots, L_h, C(v_j)$ form a (connected) directed path in $T(\mathcal{C}(G))$, (2) there are no other set L with $L \subset L_1$, (3) all vertices in L_h (and hence $L_1 \cup L_2 \cup \cdots \cup L_h$) belong to $N_{>i}(v_i)$, and (4) some vertices in $C(v_j)$ may not be in $N_{>i}(v_i)$. Checking them can be done in $O(|N_{>i}(v_i)|) = O(\deg(v_i))$ time for each i, and otherwise, the vertex sets in the tree are not laminar, and hence it would be rejected. The cases (a) and (b) can be seen as special cases of the case (c). Therefore, total running time of the modified CliqueLaminarTree is still $O(n + m)$. ∎

We note that Theorem 13 is not new. A ptolemaic graph is distance hereditary and chordal [16], distance hereditary graphs are recognized in linear time [14, 9, 4], and chordal graphs are also recognized in linear time [22, 24]. Thus, combining them, we have the theorem. We dare to state Theorem 13 to show that we can recognize and then construct the clique laminar tree of G at the same time in linear time, and the algorithm is much simpler and more straightforward than the combination of known algorithms. (As noted in Introduction, the linear time algorithm for recognition of distance hereditary graphs is not so simple.)

Theorem 14 *The graph isomorphism problem for ptolemaic graphs can be solved in linear time.*

Proof. Given a ptolemaic graph $G = (V, E)$, the labeled clique laminar tree $\vec{T}(\mathcal{C}(G))$ is uniquely determined up to isomorphism by maximal cliques. Each vertex in V appears once in $\vec{T}(\mathcal{C}(G))$, and the number of nodes in $\vec{T}(\mathcal{C}(G))$ is at most $2|V| - 1$ by Lemma 4. Thus the representation of $\vec{T}(\mathcal{C}(G))$ requires $O(|V|)$ space. The graph isomorphism problem for labeled trees can be done in linear time (see, e.g., [19]), which completes the proof. ∎

4 A Tree Representation of Distance Hereditary Graphs

In Section 3.1, we defined a clique laminar tree of a ptolemaic graph G. In the clique laminar tree, a node C has a label $\ell(C)$, which means a vertex v is in $\ell(C)$ if and only if C is the minimal vertex set that contains v. When G is ptolemaic, the vertex sets C, and hence $\ell(C)$, induce cliques of the original graph G. The main idea in this section is that we loosen the condition "clique" to "clique or independent set." This idea leads us to new characterizations of distance hereditary graphs.

4.1 A Tree Representation

Given a graph $G = (V, E)$, we first define $\mathcal{M}(G)$

$$\mathcal{M}(G) := \{M \mid M \text{ is a nonempty set of vertices in } V\},$$

satisfying the following three properties; (i) $\cup M = V$, (ii) each pair of sets M and M' is incomparable or disjoint, and (iii) for any two nondisjoint sets M and M', $M \cap M'$ separates $M \setminus M'$ and $M' \setminus M$. We next define $\mathcal{C}(G)$ to be a set of non-empty vertex sets C that satisfies

$$\mathcal{C}(G) := \bigcup_{S \subseteq \mathcal{M}(G)} \{C \mid C = \cap_{M \in S} M, C \neq \emptyset\}.$$

We also define $\mathcal{L}(G) := \mathcal{C}(G) \setminus \mathcal{M}(G)$. (We note that $\mathcal{M}(G)$ is not uniquely determined for a distance hereditary graph, which will be discussed in Section 5.)

We here define the *label* $\ell(C)$ of each vertex set $C \in \mathcal{C}(G)$, and the mapping from V to the partition defined by labels in the same way. We also denote by $C(v)$, $C_M(v)$, $C_L(v)$ for each $v \in V$ similarly.

We further require that a given graph $G = (V, E)$ with the set $\mathcal{M}(G)$ has the following properties: (iv) Each partition, or label $\ell(C)$ for some C, has an attribute either "clique" or "independent" which means the vertex set $\ell(C)$ induces a clique or an independent set of G. (v) Each vertex set C in $\mathcal{C}$ also has an attribute either "clique" or "independent" as follows: When $\ell(C) = C$, the attribute is the same as $\ell(C)$. Otherwise, the vertex set C is partitioned into two or more disjoint vertex sets $L_1, L_2, \ldots, L_k$ such that $\cup_i L_i = C$, $L_i \cap L_j = \emptyset$ with $i \neq j$, and there are no other set $L \in \mathcal{L}$ with $L_i \subset L \subset C$ for each i. (Note that $\ell(C) = L_i$ for some i.) In the case, when C is "independent," there are no edges between L_i and L_j for $i \neq j$. When C is "clique," every vertex in L_i is joined to all vertices in L_j for $i \neq j$. (vi) The graph G has no other edges. For simplicity, we define that the vertex set of size ≤ 1 is not an independent set but a clique. We denote the attribute of a label $\ell(C)$ or a node C by $a(\cdot)$.

It is easy to see that if all attributes are "clique," the graph is ptolemaic. This observation leads us to the following theorem and lemma:

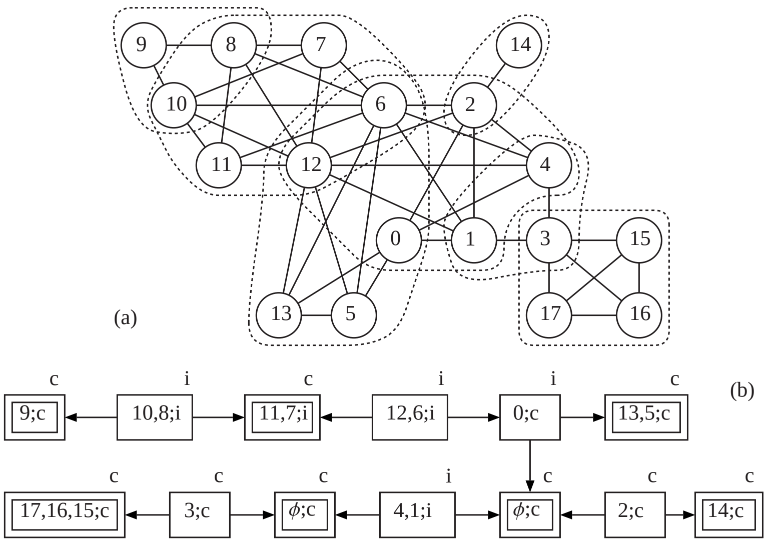

Figure 3: A distance hereditary graph and its CIL-tree.

Theorem 15 *Let $G = (V, E)$ be a graph that has the family $\mathcal{M}$ with the properties (i)~(vi). Let $\mathcal{F}$ be a family of the sets in $\mathcal{L}(G)$ such that $\cup_{L\in\mathcal{F}} L \subset M$ for some set $M \in \mathcal{M}(G)$. Then $\mathcal{F}$ is laminar.*

Lemma 16 *Let C_1, C_2 be any incomparable sets in $\mathcal{C}(G)$ for a graph $G = (V, E)$ with associated family $\mathcal{M}$ satisfying the properties (i)~(vi). Then $C_1 \cap C_2$ separates $C_1 \setminus C_2$ and $C_2 \setminus C_1$.*

We can also define a directed graph $\vec{T}(\mathcal{C}(G)) = (\mathcal{C}(G), A(G))$ and its underlying graph $T(\mathcal{C}(G))$ for a given graph $G = (V, E)$ and $\mathcal{M}$ in the same way as we introduced in Section 3.1, and obtain the following theorem.

Theorem 17 *If a given graph $G = (V, E)$ has a family $\mathcal{M}$ with the properties (i)~(vi), the graph $T(\mathcal{C}(G))$ is a tree.*

We say that a graph G has a *clique-independent laminar tree* (CIL-tree, for short) if G has a family $\mathcal{M}$ with the properties (i)~(vi). Figure 3(a) depicts an example of a distance hereditary graph, which is the same graph as in [1, Figure 10]. In the graph, we have $\mathcal{M} = \{\{0, 1, 2, 4, 6, 12\}, \{0, 5, 6, 12, 13\}, \{1, 3, 4\}, \{2, 14\}, \{3, 15, 16, 17\}, \{6, 7, 8, 10, 11, 12\}, \{8, 9, 10\}\}$, and $\mathcal{L} = \{\{0, 6, 12\}, \{6, 12\}, \{2\}, \{1, 4\}, \{3\}, \{8, 10\}\}$. Hence its CIL-tree is given as Figure 3(b). In the graph, for example, $a(\{0, 5, 6, 12, 13\})$ is "clique" and $a(\ell(\{0, 5, 6, 12, 13\}))$ is also "clique"; $a(\{6, 7, 8, 10, 11, 12\})$ is "clique" and $a(\ell(\{6, 7, 8, 10, 11, 12\}))$ is "independent." Although $a(\{0, 6, 12\})$ is "independent" and $a(\ell(\{0, 6, 12\}))$ is "clique," if the vertex 0 had weak siblings, $a(\ell(\{0, 6, 12\}))$ would be "independent" (hence we have all possible cases).

We note that $G = (V, E)$ with $\mathcal{M}$ is connected if and only if $a(M)$ is "clique" for every maximal set M in $\mathcal{M}$. Hereafter we assume that G is connected.

Lemma 18 *If a connected graph $G = (V, E)$ is distance hereditary, G has a CIL-tree.*

Proof. To show this, we use the characterization due to Bandelt and Mulder [1] in Theorem 1. To unify the notation, we assume that the graph G can be obtained from $\{v_n, v_{n-1}\}$ by splitting some vertex or attaching a pendant vertex according to the ordering $v_{n-2}, v_{n-3}, \dots, v_2, v_1$. For the ordering, we first set $C_M(v_n) = C_M(v_{n-1}) = \{v_n, v_{n-1}\}$. Then, according to the ordering, we incrementally grow the CIL-tree for each $i = n - 2, \dots, 2, 1$. We have three cases.
(α) The vertex v_i is added as a pendant vertex of $v_j \in N_{>i}(v_i)$. It is easy to see that $\ell(C(v_j))$ is the set of siblings of v_j. If $|\ell(C(v_j))| = 1$, that is, v_j has no sibling, we make $C_M(v_i) := \{v_i, v_j\}$ with $\ell(C_M(v_i)) := \{v_i\}$, and add an arc $(C(v_j), C_M(v_i))$. The attributes $a(C_M(v_i)) = a(\ell(C_M(v_i)))$:="clique." If $|C(v_j)| > 1$, we take v_j from $C := C(v_j)$ since v_j is shared by C and the set $\{v_i, v_j\}$. Then we make a new vertex set $C_L(v_j) := \{v_j\}$ with $\ell(C_L(v_j)) := \{v_j\}$, and $C_M(v_i) := \{v_i, v_j\}$ with $\ell(C_M(v_i)) := \{v_i\}$. We add arcs $(C_L(v_j), C)$ and $(C_L(v_j), C_M(v_i))$. The attributes $a(C_L(v_j)) = a(\ell(C_L(v_j))) = a(C_M(v_i)) = a(\ell(C_M(v_i)))$ ="clique."

(β) The vertex v_i is added as a strong sibling of some vertex v_j. If $a(\ell(C(v_j)))$ is "clique" (including the case $\left|\ell(C(v_j))\right| = 1$), we just add v_i into $\ell(C(v_j))$ by setting $\ell(C(v_j)) := \ell(C(v_j)) \cup \{v_i\}$ and $C(v_i) := C(v_j)$. When $a(\ell(C(v_j)))$ is "independent," we remove v_j from $C = C(v_j)$ (and $\ell(C(v_j))$), and make a new vertex set $C_M(v_i) = C_M(v_j) := (C \setminus \ell(C)) \cup \{v_i, v_j\}$ with an arc $(C', C(v_i))$ for each arc (C', C). We set $\ell(C(v_i)) := \{v_i, v_j\}$ and $a(C(v_i)) = a(\ell(C(v_i))) =$"clique".
(γ) The vertex v_i is added as a weak sibling of some vertex v_j. If $a(\ell(C(v_j)))$ is "independent," we just add v_i into $C(v_j)$ and $\ell(C(v_j))$. When $a(\ell(C(v_j)))$ is "clique," we have two subcases. If $\left|\ell(C(v_j))\right| = 1$, we add v_i into $\ell(C(v_j))$ and change $a(\ell(C(v_j)))$ from "clique" (of one vertex v_j) to "independent" (of two vertices v_i and v_j). Otherwise, we make a new vertex set $C(v_i) := C(v_j) \setminus \ell(C(v_j))$ with $\ell(C(v_i)) := \{v_i\}$. We add $(C', C(v_i))$ for each arc $(C', C(v_j))$ and set $a(\ell(C(v_i))) = a(C(v_i)) :=$"clique."

It is not difficult to see that each construction grows the tree as a CIL-tree for each $i = n-2, \ldots, 2, 1$. Thus we have the lemma. ∎

Lemma 19 *If a connected graph $G = (V, E)$ has a CIL-tree $\vec{T}(\mathcal{C}(G))$, G is distance hereditary.*

Proof. We show the lemma by induction of the number of vertices in V. For K_1 and K_2, we have the lemma immediately. We assume that $G = (V, E)$ has a corresponding CIL-tree, and $|V| \geq 3$. We will show that G contains at least one pendant vertex, or at least one pair of twins; then we can reduce the number of vertices.

If the CIL-tree has a node C such that $\ell(C)$ consists of two or more vertices, they are twins, and we have done. Thus we assume that every node in the CIL-tree T has label with at most one vertex, and hence T has no twins.

We remind that the CIL-tree $\vec{T}(\mathcal{C}(G))$ is a directed tree. Since it represents a laminar family joined by the vertex sets in $\mathcal{M}$, we can see that (1) each node of outdegree 0 in $\vec{T}(\mathcal{C}(G))$ implies a node in $\mathcal{M}$, (2) no two nodes in $\mathcal{M}$ are adjacent, (3) each internal node of $T(\mathcal{C}(G))$ in $\mathcal{M}$ has indegree at least two and no outdegree, and (4) each internal node of $T(\mathcal{C}(G))$ in $\mathcal{L}$ has outdegree at least two. The fourth property can be obtained from the fact that each vertex set in $\mathcal{L}$ is an intersection of at least two vertex sets.

We also observe the following claim. Let $M \in \mathcal{M}$ be a leaf in $T(\mathcal{C}(G))$ that has indegree 1 from a node C in $\vec{T}(\mathcal{C}(G))$. Then C is in $\mathcal{L}$ since any two maximal sets in $\mathcal{M}$ are incomparable. In this case, (5) if C has indegree 0, M contains a pendant vertex. The claim (5) is proved as follows. Suppose that C has indegree 0. Then, since C does not contain any other vertex set, and C does not contain any twins, we have $|\ell(C)| = |C| = 1$. Then, we have $|\ell(M)| = 1$ and $|M| = 2$, that is, $C = \{u\}$ and $M = \{u, v\}$ for some $u, v \in V$. Since G is connected, v is a pendant vertex.

We now regard $T(\mathcal{C}(G))$ as a tree rooted at any fixed node r. For the tree, we define the *depth* of each node; the root r has depth 0, denoted by $dep(r) = 0$, and the depth of each node p except root is defined by $dep(p) := dep(q) + 1$, where q is the parent of p.

Now, we pick up any node M of the maximum depth; clearly, M is a leaf and hence $M \in \mathcal{M}$. Let L be the parent of M. Then, by the claim (2), L is in $\mathcal{L}$. If L is the root of depth 0, L has at least two children M and M' since M has the maximum depth. Then, by the claim (1), M and M' are in $\mathcal{M}$. However, the vertices in $\ell(M)$ and $\ell(M')$ becomes twins in the case, which is a contradiction. Thus L is not the root. Hence, there is a parent C of L in the tree.

We first suppose that $\vec{T}(\mathcal{C}(G))$ contains an arc (C, L). In the case, by claim (4), L has at least two children M and M', which again implies a contradiction. Therefore, $\vec{T}(\mathcal{C}(G))$ contains an arc (L, C), and L has only one child M. However, in the case, L has indegree 0. Hence, by the claim (5), M contains a pendant vertex.

Therefore, when a connected graph G has a CIL-tree, G has at least one pendant vertex or a pair of twins. Hence, by Theorem 1, G is distance hereditary. ∎

By Lemmas 18 and 19, we have the main theorem in this section:

Theorem 20 *A graph $G = (V, E)$ is distance hereditary if and only if G has a CIL-tree.*

Hence we also denote by $\vec{T}(\mathcal{C}(G))$ and $T(\mathcal{C}(G))$ the CIL-tree and its underlying tree for a distance hereditary graph G, respectively. Using the CIL-tree model, we can characterize ptolemaic graphs and bipartite distance hereditary graphs.

Corollary 21 *(cf. [1, Chapter 6]) Let G be a distance hereditary graph, and $\vec{T}(\mathcal{C}(G))$ be its CIL-tree. If all attributes are "clique," the graph is ptolemaic. On the other hand, if all attributes of $\ell(C)$ are "independent" and all attributes of C are "clique," the graph is bipartite distance hereditary graph.*

In [23], Spinrad mentioned that there is no known intersection model of distance hereditary graphs. Theorems 7 and 20 give us a geometric model of distance hereditary graphs:

Corollary 22 *Let $\mathcal{T}$ be the set of directed subtrees $\vec{T_v}$ of a directed graph $\vec{T}$ that satisfies the properties stated in Corollary 9. Moreover, each node C of $\vec{T}$ has two attributes $a(C)$ and $a(\ell(C))$, and the values of the attributes are either "independent" or "clique." Let u and v be any vertices in V. Then $\{u, v\} \in E$ if and only if (1) $a(\ell(C)) =$"clique" if $\vec{T_u} = \vec{T_v}$ rooted at C, or (2) $a(C) =$"clique" if $T_u \neq T_v$ and the common subtree of $\vec{T_u}$ and $\vec{T_v}$ is rooted at C. Then a graph $G = (V, E)$ is distance hereditary if and only if the graph has the geometric model defined above.*

Theorem 23 *Let $G = (V, E)$ be a distance hereditary graph. Then its CIL-tree $\vec{T}(\mathcal{C}(G))$ can be constructed in linear time.*

Proof. For a given distance hereditary graph G, using the algorithm by Hammer and Maffray, we can compute the sequence of extensions in Theorem 1. Then the proof of Lemma 18 gives us the construction of a CIL-tree of G from the ordered vertices. It is easy to see that each step for a vertex v takes $O(\deg(v))$ time. ∎

5 Concluding Remarks

In Section 3, the partition of the vertex set is uniquely determined by its maximal cliques for a ptolemaic graph. Hence the clique laminar tree of a ptolemaic graph is uniquely determined up to isomorphism. However, for a CIL-tree of a distance hereditary graph, this is not the case. For example, for the graph given in Figure 3(a), we can let $\mathcal{M} = \{\{0, 1, 2, 4, 5, 13\}, \{1, 2, 4, 5, 6, 12, 13\}, \{1, 3, 4\}, \{2, 14\}, \{3, 15, 16, 17\}, \{6, 7, 8, 10, 11, 12\}, \{8, 9, 10\}\}$, and $\mathcal{L} = \{\{1, 2, 4, 5, 13\}, \{1, 4\}, \{2\}, \{3\}, \{6, 12\}, \{8, 10\}\}$. Then this family gives a different CIL-tree from the one in Figure 3(b). Intuitively, the reason is the following: Given a distance hereditary graph G, we first add suitable edges and make G into a ptolemaic graph G'. Then the clique laminar tree of G' gives a CIL-tree for G. However, we have several ways to make a distance hereditary graph into a ptolemaic graph. For example, Figure 3(b) is obtained when we regard the vertex sets $\{0, 5, 6, 12, 13\}$ and $\{0, 1, 2, 4, 6, 12\}$ as "maximal cliques," and the other CIL-tree is obtained when we regard the vertex sets $\{0, 1, 2, 4, 5, 13\}$ and $\{1, 2, 4, 5, 6, 12, 13\}$ as "maximal cliques." Therefore, we cannot solve the graph isomorphism problem for distance hereditary graphs by using CIL-trees immediately. We found a polynomial time algorithm that constructs a canonical CIL-tree for a given distance hereditary graph. However, efficient, especially linear time, algorithm that solves the graph isomorphism problem for distance hereditary graphs is remained open, which is mentioned by Spinrad in [23].

As noted in Introduction, distance hereditary graphs can be recognized in linear time by using the algorithms in [14, 9, 4]. However, they are still complicated. A simple linear algorithm for the recognition of distance hereditary graphs is still unknown. Especially, is there a simple algorithm based on LBFS that constructs a CIL-tree for any given distance hereditary graph (and rejects if it is not distance hereditary)?

In this paper, we present new tree representations (data structures) for ptolemaic graphs and distance hereditary graphs. Our results will enable us to use the dynamic programming technique to solve some basic problems on these graph classes. To develop such efficient algorithms based on the dynamic programming are future works. The authors are now preparing an efficient algorithm that finds a longest cycle in a given ptolemaic graph.

References

[1] H.-J. Bandelt and H.M. Mulder. Distance-Hereditary Graphs. *Journal of Combinatorial Theory, Series B*, 41:182–208, 1986.

[2] A. Brandstädt and F.F. Dragan. A Linear-Time Algorithm for Connected r-Domination and Steiner Tree on Distance-Hereditary Graphs. *Networks*, 31:177–182, 1998.

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

[4] A. Bretscher, D. Corneil, M. Habib, and C. Paul. A Simple Linear Time LexBFS Cograph Recognition Algorithm. In *Graph-Theoretic Concepts in Computer Science (WG 2003)*, pp. 119–130. Lecture Notes in Computer Science Vol. 2880, Springer-Verlag, 2003.

[5] H.J. Broersma, E. Dahlhaus, and T. Kloks. A Linear Time Algorithm for Minimum Fill-in and Treewidth for Distance Hereditary Graphs. *Discrete Applied Mathematics*, 99:367–400, 2000.

[6] M.-S. Chang, S.-Y. Hsieh, and G.-H. Chen. Dynamic Programming on Distance-Hereditary Graphs. In *Proceedings of 8th International Symposium on Algorithms and Computation (ISAAC '97)*, pp. 344–353. Lecture Notes in Computer Science Vol. 1350, Springer-Verlag, 1997.

[7] M.-S. Chang, S.-C. Wu, G.J. Chang, and H.-G. Yeh. Domination in Distance-Hereditary Graphs. *Discrete Applied Mathematics*, 116:103–113, 2002.

[8] D.G. Corneil. Lexicographic Breadth First Search — A Survey. In *Graph-Theoretic Concepts in Computer Science (WG 2004)*, pp. 1–19. Lecture Notes in Computer Science Vol. 3353, Springer-Verlag, 2004.

[9] G. Damiand, M. Habib, and C. Paul. A Simple Paradigm for Graph Recognition: Application to Cographs and Distance Hereditary Graphs. *Theoretical Computer Science*, 263:99–111, 2001.

[10] A. D'Atri and M. Moscarini. Distance-Hereditary Graphs, Steiner Trees, and Connected Domination. *SIAM Journal on Computing*, 17(3):521–538, 1988.

[11] M. Farber. Independent Domination in Chordal Graphs. *Operations Research Letters*, 1(4):134–138, 1982.

[12] F. Gavril. Algorithms for Minimum Coloring, Maximum Clique, Minimum Covering by Cliques, and Maximum Independent Set of a Chordal Graph. *SIAM Journal on Computing*, 1(2):180–187, 1972.

[13] M.C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Annals of Discrete Mathematics 57. Elsevier, 2nd edition, 2004.

[14] P.L. Hammer and F. Maffray. Completely Separable Graphs. *Discrete Applied Mathematics*, 27:85–99, 1990.

[15] E. Howorka. A Characterization of Distance-Hereditary Graphs. *Quart. J. Math. Oxford (2)*, 28:417–420, 1977.

[16] E. Howorka. A Characterization of Ptolemaic Graphs. *Journal of Graph Theory*, 5:323–331, 1981.

[17] S.-Y. Hsieh, C.-W. Ho, T.-S. Hsu, and M.-T. Ko. Efficient Algorithms for the Hamiltonian Problem on Distance-Hereditary Graphs. In *COCOON 2002*, pp. 77–86. Lecture Notes in Computer Science Vol. 2387, Springer-Verlag, 2002.

[18] P.N. Klein. Efficient Parallel Algorithms for Chordal Graphs. *SIAM Journal on Computing*, 25(4):797–827, 1996.

[19] J. Köbler, U. Schöning, and J. Torán. *The Graph Isomorphism Problem: Its Structural Complexity*. Birkhäuser, 1993.

[20] B. Korte and J. Vygen. *Combinatorial Optimization*, Vol. 21 of *Algorithms and Combinatorics*. Springer, 2000.

[21] F. Nicolai and T. Szymczak. Homogeneous Sets and Domination: A Linear Time Algorithm for Distance-Hereditary Graphs. *Networks*, 37(3):117–128, 2001.

[22] D.J. Rose, R.E. Tarjan, and G.S. Lueker. Algorithmic Aspects of Vertex Elimination on Graphs. *SIAM Journal on Computing*, 5(2):266–283, 1976.

[23] J.P. Spinrad. *Efficient Graph Representations*. American Mathematical Society, 2003.

[24] R.E. Tarjan and M. Yannakakis. Simple Linear-Time Algorithms to Test Chordality of Graphs, Test Acyclicity of Hypergraphs, and Selectively Reduce Acyclic Hypergraphs. *SIAM Journal on Computing*, 13(3):566–579, 1984.

[25] H.-G. Yeh and G.J. Chang. Centers and Medians of Distance-Hereditary Graphs. *Discrete Mathematics*, 265:297–310, 2003.