Capturing Logarithmic Space and Polynomial Time on Chordal Claw-Free Graphs

Berit Grußien

Humboldt-Universität zu Berlin, Berlin, Germany
grussien@informatik.hu-berlin.de

Abstract

We show that the class of chordal claw-free graphs admits $\mathsf{LREC}_=$-definable canonization. $\mathsf{LREC}_=$ is a logic that extends first-order logic with counting by an operator that allows it to formalize a limited form of recursion. This operator can be evaluated in logarithmic space. It follows that there exists a logarithmic-space canonization algorithm for the class of chordal claw-free graphs, and that $\mathsf{LREC}_=$ captures logarithmic space on this graph class. Since $\mathsf{LREC}_=$ is contained in fixed-point logic with counting, we also obtain that fixed-point logic with counting captures polynomial time on the class of chordal claw-free graphs.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems, F.4.1 Mathematical Logics

Keywords and phrases Descriptive complexity, logarithmic space, polynomial time, chordal claw-free graphs

Digital Object Identifier 10.4230/LIPIcs.CSL.2017.26

1 Introduction

Descriptive complexity is a field of computational complexity theory that provides logical characterizations for the standard complexity classes. The starting point of descriptive complexity was a theorem of Fagin in 1974 [6], which states that existential second-order logic characterizes, or *captures*, the complexity class NP. Later, similar logical characterizations were found for further complexity classes. For example, Immerman proved that deterministic transitive closure logic DTC captures LOGSPACE [19], and independently of one another, Immerman [18] and Vardi [24] showed that fixed-point logic FP captures PTIME[1]. However, these two results have a draw-back: They only hold on ordered structures, that is, on structures with a distinguished binary relation which is a linear order on the universe of the structure. On structures that are not necessarily ordered, there have only been partial results towards capturing LOGSPACE or PTIME, so far.

A negative partial result towards capturing LOGSPACE follows from Etessami and Immerman's result that (directed) tree isomorphism is not definable in transitive closure logic with counting TC+C [5]. This implies that tree isomorphism is neither definable in deterministic nor in symmetric transitive closure logic with counting (DTC+C and STC+C), although it is decidable in LOGSPACE [22]. Hence, DTC+C and STC+C are not strong enough to capture LOGSPACE even on the class of trees. That is why, in 2011 a new logic with logarithmic-space data complexity was introduced [13, 14]. This logic, $\mathsf{LREC}_=$, is an extension of first-order logic with counting by an operator that allows a limited form of

[1] More precisely, Immerman and Vardi's theorem holds for least fixed-point logic (LFP) and the equally expressive inflationary fixed-point logic (IFP). Our indeterminate FP refers to either of these two logics.

recursion. $\mathsf{LREC}_=$ strictly contains $\mathsf{STC}+\mathsf{C}$ and $\mathsf{DTC}+\mathsf{C}$. In [13, 14], the authors proved that $\mathsf{LREC}_=$ captures $\mathsf{LOGSPACE}$ on the class of (directed) trees and on the class of interval graphs. In this paper we now show that $\mathsf{LREC}_=$ captures $\mathsf{LOGSPACE}$ also on the class of chordal claw-free graphs. More precisely, this paper's technical main contribution states that the class of chordal claw-free graphs admits $\mathsf{LREC}_=$-definable canonization. This does not only imply that $\mathsf{LREC}_=$ captures $\mathsf{LOGSPACE}$ on chordal claw-free graphs, but also that there exists a logarithmic-space canonization algorithm for the class of chordal claw-free graphs. Hence, the isomorphism and automorphism problem for this graph class is solvable in logarithmic space.

For polynomial time there also exist partial characterizations. Fixed-point logic with counting $\mathsf{FP}+\mathsf{C}$ captures PTIME, for example, on planar graphs [8], on all classes of graphs of bounded treewidth [15] and on K_5-minor free graphs [9]. Note that all these classes can be defined by a list of forbidden minors. In fact, Grohe showed in 2010 that $\mathsf{FP}+\mathsf{C}$ captures PTIME on all graph classes with excluded minors [11]. Instead of graph classes with excluded minors, one can also consider graph classes with excluded induced subgraphs, i.e. graph classes $\mathcal{C}$ that are closed under taking induced subgraphs. For some of these graph classes $\mathcal{C}$, e.g. chordal graphs [10], comparability graphs [21] and co-comparability graphs [21], capturing PTIME on $\mathcal{C}$ is as hard as capturing PTIME on the class of all graphs for any "reasonable" logic.[2] This gives us reason to consider subclasses of chordal graphs, comparability graphs and co-comparability graphs more closely. There are results showing that $\mathsf{FP}+\mathsf{C}$ captures PTIME on interval graphs (chordal co-comparability graphs) [20], on permutation graphs (comparability co-comparability graphs) [17] and on chordal comparability graphs [16]. Further, Grohe proved that $\mathsf{FP}+\mathsf{C}$ captures PTIME on chordal line graphs [10]. At the same time he conjectured that this is also the case for the class of chordal claw-free graphs, which is an extension of the class of chordal line graphs. Our main result implies that Grohe's conjecture is true: Since $\mathsf{LREC}_=$ is contained in $\mathsf{FP}+\mathsf{C}$, it yields that there exists an $\mathsf{FP}+\mathsf{C}$-canonization of the class of chordal claw-free graphs. Hence, $\mathsf{FP}+\mathsf{C}$ captures PTIME also on the class of chordal claw-free graphs.

Our main result is based on a study of chordal claw-free graphs. Chordal graphs are the intersection graphs of subtrees of a tree [2, 7, 25], and a clique tree of a chordal graph corresponds to a minimal representation of the graph as such an intersection graph. We prove that chordal claw-free graphs are (claw-free) intersection graphs of paths in a tree, and that for each connected chordal claw-free graph the clique tree is unique.

1.1 Structure

The preliminaries in Section 2 will be followed by a Section 3 where we analyze the structure of clique trees of chordal claw-free graphs, and, e.g., show that connected chordal claw-free graphs have a unique clique tree. In Section 4, we transform the clique tree of a connected chordal claw-free graph into a directed tree, and color each maximal clique with information about its intersection with other maximal cliques by using a special coloring with a linearly ordered set of colors. We obtain what we call the supplemented clique tree, and show that it is definable in $\mathsf{STC}+\mathsf{C}$ by means of a parameterized transduction. We know that there exists an $\mathsf{LREC}_=$-canonization of colored trees if the set of colors is linearly ordered [13, 14]. We apply this $\mathsf{LREC}_=$-canonization to the supplemented clique tree in Section 5 and obtain the

[2] Note that $\mathsf{FP}+\mathsf{C}$ does not capture PTIME on the class of all graphs [3]. Hence, it does not capture PTIME on the class of chordal graphs, comparability graphs or co-comparability graphs either.

canon of this colored directed tree. Due to the type of coloring, the information about the maximal cliques is also contained in the colors of the canon of the supplemented clique tree. This information and the linear order on the vertices of the canon of the supplemented clique tree allow us to define the maximal cliques of a canon of the connected chordal claw-free graph, from which we can easily construct the canon of the graph. By combining the canons of the connected components, we obtain a canon for each chordal claw-free graph.

2 Basic Definitions and Notation

We write $\mathbb{N}$ for the set of all non-negative integers. For all $n, n' \in \mathbb{N}$, we define $[n, n'] := \{m \in \mathbb{N} \mid n \leq m \leq n'\}$ and $[n] := [1, n]$. We often denote tuples $(a_1, \ldots, a_k)$ by $\bar{a}$. Given a tuple $\bar{a} = (a_1, \ldots, a_k)$, let $\tilde{a} := \{a_1, \ldots, a_k\}$. Let $n \geq 1$. Let $\bar{a}^i = (a^i_1, \ldots, a^i_{k_i})$ be a tuple of length k_i for each $i \in [n]$. We denote the tuple $(a^1_1, \ldots, a^1_{k_1}, \ldots, a^n_1, \ldots, a^n_{k_n})$ by $(\bar{a}^1, \ldots, \bar{a}^n)$. Mappings $f\colon A \to B$ are extended to tuples $\bar{a} = (a_1, \ldots, a_k)$ over A via $f(\bar{a}) := (f(a_1), \ldots, f(a_k))$. Let $\approx$ be an equivalence relation on a set S. Then $a/_\approx$ denotes the equivalence class of $a \in S$ with respect to $\approx$. For $\bar{a} = (a_1, \ldots, a_n) \in S^n$ and $R \subseteq S^n$, we let $\bar{a}/_\approx := (a_1/_\approx, \ldots, a_n/_\approx)$ and $R/_\approx := \{\bar{a}/_\approx \mid \bar{a} \in R\}$. A *partition* of a set S is a set $\mathcal{P}$ of disjoint non-empty subsets of S where $S = \bigcup_{A \in \mathcal{P}} A$. For a set S, we let $\binom{S}{2}$ be the set of all 2-element subsets of S.

2.1 Graphs and LO-colored Graphs

A *graph* is a pair (V, E) consisting of a non-empty finite set V of *vertices* and a set $E \subseteq \binom{V}{2}$ of *edges*. Let $G = (V, E)$ and $G' = (V', E')$ be graphs. The *union* $G \cup G'$ of G and G' is the graph $(V \cup V', E \cup E')$. For a subset $W \subseteq V$ of vertices, $G[W]$ denotes the *induced subgraph* of G with vertex set W. *Connectivity* and *connected components* are defined in the usual way. We denote the *neighbors* of a vertex $v \in V$ by $N(v)$. A set $B \subseteq V$ is a *clique* if $\binom{B}{2} \subseteq E$. A maximal clique, or *max clique*, is a clique that is not properly contained in any other clique.

A graph is *chordal* if all its cycles of length at least 4 have a chord, which is an edge that connects two non-consecutive vertices of the cycle. A *claw-free* graph is a graph that does not have a *claw*, i.e. a graph isomorphic to the complete bipartite graph $K_{1,3}$, as an induced subgraph. We denote the class of (connected) chordal claw-free graphs by (con-)CCF.

A subgraph P of G is a *path* of G if $P = (\{v_0, \ldots, v_k\}, \{\{v_0, v_1\}, \ldots, \{v_{k-1}, v_k\}\})$ for distinct vertices $v_0, \ldots, v_k$ of G. We also denote path P by the sequence $v_0, \ldots, v_k$ of vertices. We let v_0 and v_k be the *ends* of P. A connected acyclic graph is a *tree*. Let $T = (V, E)$ be a tree. A *subtree* of T is a connected subgraph of T. A vertex $v \in V$ of degree 1 is called a *leaf*.

A pair (V, E) is a *directed graph* if V is a non-empty finite set and $E \subseteq V^2$. A connected acyclic directed graph where the in-degree of each vertex is at most 1 is a *directed tree*. Let $T = (V, E)$ be a directed tree. The vertex of in-degree 0 is the *root* of T. If $(v, w) \in E$, then w is a *child* of v, and v the *parent* of w. Let w, w' be children of $v \in V$. Then w is a *sibling* of w' if $w \neq w'$. If there is a (directed) path from $v \in V$ to $w \in V$ in T, then v is an *ancestor* of w.

Let $G = (V, E)$ be a graph and $f\colon V \to C$ be a mapping from the vertices of G to a finite set C. Then f is a *coloring* of G, and the elements of C are called *colors*. In this paper we color the vertices of a graph with binary relations on a linearly ordered set. We call graphs with such a coloring LO-*colored graphs*. More precisely, an LO-colored graph is a tuple $G = (V, E, M, \trianglelefteq, L)$ with the following four properties:

1. The pair (V, E) is a graph. We call (V, E) the *underlying graph* of G.
2. The set of *basic color elements* M is a non-empty finite set with $M \cap V = \emptyset$.
3. The binary relation $\trianglelefteq \subseteq M^2$ is a linear order on M.
4. The relation $L \subseteq V \times M^2$ assigns to every $v \in V$ an LO-*color* $L_v := \{(d, d') \mid (v, d, d') \in L\}$.

Let $d_0, \dots, d_{|M|-1}$ be the enumeration of the basic color elements in M according to their linear order $\trianglelefteq$. We call $L_v^{\mathbb{N}} := \{(i,j) \in \mathbb{N}^2 \mid (d_i, d_j) \in L_v\}$ the NO-*color* of $v \in V$.

We can use the linear order $\trianglelefteq$ on M to obtain a linear order on the colors $\{L_v \mid v \in V\}$ of G. Thus, an LO-colored graph is a special kind of colored graph with a linear order on its colors.

2.2 Structures

A *vocabulary* is a finite set τ of relation symbols. Each relation symbol $R \in \tau$ has a fixed arity $\mathrm{ar}(R) \in \mathbb{N}$. A *$\tau$-structure* consists of a non-empty finite set $U(A)$, its *universe*, and for each relation symbol $R \in \tau$ of a relation $R(A) \subseteq U(A)^{\mathrm{ar}(R)}$.

An *isomorphism* between τ-structures A and B is a bijection $f\colon U(A) \to U(B)$ such that for all $R \in \tau$ and all $\bar{a} \in U(A)^{\mathrm{ar}(R)}$ we have $\bar{a} \in R(A)$ if and only if $f(\bar{a}) \in R(B)$. We write $A \cong B$ to indicate that A and B are *isomorphic*.

Let E be a binary relation symbol. Each graph corresponds to an $\{E\}$-structure $G = (V, E)$ where the universe V is the vertex set and E is an irreflexive and symmetric binary relation, the edge relation. To represent an LO-colored graph $G = (V, E, M, \trianglelefteq, L)$ as a logical structure we extend the 5-tuple by a set U to a 6-tuple $(U, V, E, M, \trianglelefteq, L)$, and we require that $U = V \dot\cup M$ in addition to the properties 1-4. The set U serves as the universe of the structure, and $V, E, M, \trianglelefteq, L$ are relations on U. We usually do not distinguish between (LO-colored) graphs and their representation as logical structures. It will be clear from the context which form we are referring to.

2.3 Logics

In this section we introduce symmetric transitive closure logic (with counting) and $\mathsf{LREC}_=$.

We assume basic knowledge in logic, in particular of *first-order logic* (FO). *First-order logic with counting* (FO+C) extends FO by a counting operator that allows for counting the cardinality of FO+C-definable relations. It lives in a two-sorted context, where structures A are equipped with a *number sort* $N(A) := [0, |U(A)|]$. FO+C has two types of variables: FO+C-variables are either *structure variables* that range over the universe $U(A)$ of a structure A, or *number variables* that range over the number sort $N(A)$. For each variable u, let $A^u := U(A)$ if u is a structure variable, and $A^u := N(A)$ if u is a number variable. Let $A^{(u_1,\dots,u_k)} := A^{u_1} \times \cdots \times A^{u_k}$. Tuples $(u_1, \dots, u_k)$ and $(v_1, \dots, v_\ell)$ of variables are *compatible* if $k = \ell$, and for every $i \in [k]$ the variables u_i and v_i have the same type. An *assignment in A* is a mapping α from the set of variables to $U(A) \cup N(A)$, where for each variable u we have $\alpha(u) \in A^u$. For tuples $\bar{u} = (u_1, \dots, u_k)$ of variables and $\bar{a} = (a_1, \dots, a_k) \in A^{\bar{u}}$, the assignment $\alpha[\bar{a}/\bar{u}]$ maps u_i to a_i for each $i \in [k]$, and each variable $v \notin \tilde{u}$ to $\alpha(v)$. By $\varphi(u_1, \dots, u_k)$ we denote a formula φ with $\mathrm{free}(\varphi) \subseteq \{u_1, \dots, u_k\}$, where $\mathrm{free}(\varphi)$ is the set of free variables in φ. Given a formula $\varphi(u_1, \dots, u_k)$, a structure A and $(a_1, \dots, a_k) \in A^{(u_1,\dots,u_k)}$, we write $A \models \varphi[a_1, \dots, a_k]$ if φ holds in A with u_i assigned to a_i for each $i \in [k]$. We write $\varphi[A, \alpha; \bar{u}]$ for the set of all tuples $\bar{a} \in A^{\bar{u}}$ with $(A, \alpha[\bar{a}/\bar{u}]) \models \varphi$. For a formula $\varphi(\bar{u})$ (with $\mathrm{free}(\varphi) \subseteq \tilde{u}$) we also denote $\varphi[A, \alpha; \bar{u}]$ by $\varphi[A; \bar{u}]$, and for a formula $\varphi(\bar{v}, \bar{u})$ and $\bar{a} \in A^{\bar{v}}$, we denote $\varphi[A, \alpha[\bar{a}/\bar{v}]; \bar{u}]$ also by $\varphi[A, \bar{a}; \bar{u}]$.

FO+C is obtained by extending FO with the following formula formation rules:

- $\phi := p \le q$ is a formula if p, q are number variables. We let $\mathrm{free}(\phi) := \{p, q\}$.
- $\phi' := \#\bar{u}\, \psi = \bar{p}$ is a formula if ψ is a formula, $\bar{u}$ is a tuple of variables and $\bar{p}$ a tuple of number variables. We let $\mathrm{free}(\phi') := (\mathrm{free}(\psi) \setminus \tilde{u}) \cup \tilde{p}$.

To define the semantics, let A be a structure and α be an assignment. We let

- $(A, \alpha) \models p \le q$ iff $\alpha(p) \le \alpha(q)$,
- $(A, \alpha) \models \#\bar{u}\, \psi = \bar{p}$ iff $|\psi[A, \alpha; \bar{u}]| = \langle \alpha(\bar{p}) \rangle_A$,

where for tuples $\bar{n} = (n_1, \dots, n_k) \in N(A)^k$ we let $\langle \bar{n} \rangle_A$ be the number

$$\langle \bar{n} \rangle_A \;:=\; \sum_{i=1}^{k} n_i \cdot (|U(A)| + 1)^{i-1}.$$

Symmetric transitive closure logic (with counting) $\mathsf{STC}(+\mathsf{C})$ is an extension of $\mathsf{FO}(+\mathsf{C})$ with stc-operators. The set of all $\mathsf{STC}(+\mathsf{C})$-formulas is obtained by extending the formula formation rules of $\mathsf{FO}(+\mathsf{C})$ by the following rule:

- $\phi := [\mathrm{stc}_{\bar{u},\bar{v}}\, \psi](\bar{u}', \bar{v}')$ is a formula if ψ is a formula and $\bar{u}, \bar{v}, \bar{u}', \bar{v}'$ are compatible tuples of structure (and number) variables. We let $\mathrm{free}(\phi) := \tilde{u}' \cup \tilde{v}' \cup \big(\mathrm{free}(\psi) \setminus (\tilde{u} \cup \tilde{v})\big)$.

Let A be a structure and α be an assignment. We let

- $(A, \alpha) \models [\mathrm{stc}_{\bar{u},\bar{v}}\, \psi](\bar{u}', \bar{v}')$ iff $(\alpha(\bar{u}'), \alpha(\bar{v}'))$ is contained in the symmetric transitive closure of $\psi[A, \alpha; \bar{u}, \bar{v}]$.

$\mathsf{LREC}_=$ is an extension of $\mathsf{FO}{+}\mathsf{C}$ with lrec-operators, which allow a limited form of recursion. We extend the formula formation rules of $\mathsf{FO}{+}\mathsf{C}$ by the following rule:

- $\phi := [\mathrm{lrec}_{\bar{u},\bar{v},\bar{p}}\ \varphi_=,\ \varphi_{\mathtt{E}},\ \varphi_{\mathtt{C}}](\bar{w}, \bar{r})$ is a formula if $\varphi_=$, $\varphi_{\mathtt{E}}$ and $\varphi_{\mathtt{C}}$ are formulas, $\bar{u}, \bar{v}, \bar{w}$ are compatible tuples of variables and $\bar{p}, \bar{r}$ are non-empty tuples of number variables.

  We let $\mathrm{free}(\phi) := \big(\mathrm{free}(\varphi_=) \setminus (\tilde{u} \cup \tilde{v})\big) \cup \big(\mathrm{free}(\varphi_{\mathtt{E}}) \setminus (\tilde{u} \cup \tilde{v})\big) \cup \big(\mathrm{free}(\varphi_{\mathtt{C}}) \setminus (\tilde{u} \cup \tilde{p})\big) \cup \tilde{w} \cup \tilde{r}$.

Let A be a structure and α be an assignment. We let

- $(A, \alpha) \models [\mathrm{lrec}_{\bar{u},\bar{v},\bar{p}}\ \varphi_=,\ \varphi_{\mathtt{E}},\ \varphi_{\mathtt{C}}](\bar{w}, \bar{r})$ iff $\big(\alpha(\bar{w})/_\sim, \langle \alpha(\bar{r}) \rangle_A\big) \in X$,

where X and $\sim$ are defined as follows: Let $\mathtt{V}_0 := A^{\bar{u}}$ and $\mathtt{E}_0 := \varphi_{\mathtt{E}}[A, \alpha; \bar{u}, \bar{v}]$. We define $\sim$ to be the reflexive, symmetric, transitive closure of the binary relation $\varphi_=[A, \alpha; \bar{u}, \bar{v}]$ over $\mathtt{V}_0$. Now consider the graph $\mathtt{G} = (\mathtt{V}, \mathtt{E})$ with $\mathtt{V} := \mathtt{V}_0/_\sim$ and $\mathtt{E} := \{(\bar{a}/_\sim, \bar{b}/_\sim) \in \mathtt{V}^2 \mid (\bar{a}, \bar{b}) \in \mathtt{E}_0\}$. To every $\bar{a}/_\sim \in \mathtt{V}$ we assign the set $\mathtt{C}(\bar{a}/_\sim) := \{\langle \bar{n} \rangle_A \mid \text{there is an } \bar{a}' \in \bar{a}/_\sim \text{ with } \bar{n} \in \varphi_{\mathtt{C}}[A, \alpha[\bar{a}'/\bar{u}]; \bar{p}]\}$ of numbers. Let $\bar{a}/_\sim \mathtt{E} := \{\bar{b}/_\sim \in \mathtt{V} \mid (\bar{a}/_\sim, \bar{b}/_\sim) \in \mathtt{E}\}$ and $\mathtt{E}\bar{b}/_\sim := \{\bar{a}/_\sim \in \mathtt{V} \mid (\bar{a}/_\sim, \bar{b}/_\sim) \in \mathtt{E}\}$. Then, for all $\bar{a}/_\sim \in \mathtt{V}$ and $\ell \in \mathbb{N}$,

$$(\bar{a}/_\sim, \ell) \in X \;:\Longleftrightarrow\; \ell > 0 \text{ and } \left|\left\{\bar{b}/_\sim \in \bar{a}/_\sim \mathtt{E} \;\middle|\; \left(\bar{b}/_\sim, \left\lfloor \frac{\ell - 1}{|\mathtt{E}\bar{b}/_\sim|} \right\rfloor\right) \in X\right\}\right| \in \mathtt{C}(\bar{a}/_\sim).$$

$\mathsf{LREC}_=$ semantically contains $\mathsf{STC}{+}\mathsf{C}$ [14]. Note that simple arithmetics like addition and multiplication are definable in $\mathsf{STC}{+}\mathsf{C}$, and therefore, in $\mathsf{LREC}_=$.

2.4 Transductions

Transductions (also known as *syntactical interpretations*) define certain structures within other structures. More on transductions can be found in [12, 16]. In the following we introduce parameterized transductions for $\mathsf{FO}(+\mathsf{C})$, $\mathsf{STC}(+\mathsf{C})$ and $\mathsf{LREC}_=$.

▶ **Definition 2.1** (Parameterized Transduction)**.** Let τ_1, τ_2 be vocabularies, and let L be a logic that extends FO.

1. A *parameterized* $\mathsf{L}[\tau_1, \tau_2]$*-transduction* is a tuple

$$\Theta(\bar{x}) \;=\; \Big(\theta_{\mathrm{dom}}(\bar{x}), \theta_U(\bar{x}, \bar{u}), \theta_{\approx}(\bar{x}, \bar{u}, \bar{u}'), \big(\theta_R(\bar{x}, \bar{u}_{R,1}, \dots, \bar{u}_{R,\mathrm{ar}(R)})\big)_{R \in \tau_2}\Big)$$

   of $\mathsf{L}[\tau_1]$-formulas, where $\bar{x}$ is a tuple of structure variables, and $\bar{u}, \bar{u}'$ and $\bar{u}_{R,i}$ for every $R \in \tau_2$ and $i \in [\mathrm{ar}(R)]$ are compatible tuples of variables.
2. The *domain* of $\Theta(\bar{x})$ is the class $\mathrm{Dom}(\Theta(\bar{x}))$ of all pairs $(A, \bar{p})$ such that $A \models \theta_{\mathrm{dom}}[\bar{p}]$, $\theta_U[A, \bar{p}; \bar{u}]$ is not empty and $\theta_{\approx}[A, \bar{p}; \bar{u}, \bar{u}']$ is an equivalence relation, where A is a τ_1-structure and $\bar{p} \in A^{\bar{x}}$. The elements in $\bar{p}$ are called *parameters*.

3. Let $(A, \bar{p})$ be in the domain of $\Theta(\bar{x})$, and let us denote $\theta_{\approx}[A, \bar{p}; \bar{u}, \bar{u}']$ by $\approx$. We define a τ_2-structure $\Theta[A, \bar{p}]$ as follows. We let

$$U(\Theta[A, \bar{p}]) := \theta_U[A, \bar{p}; \bar{u}]/_{\approx}$$

be the universe of $\Theta[A, \bar{p}]$. Further, for each $R \in \tau_2$, we let

$$R(\Theta[A, \bar{p}]) := \left(\theta_R[A, \bar{p}; \bar{u}_{R,1}, \ldots, \bar{u}_{R,\mathrm{ar}(R)}] \cap \theta_U[A, \bar{p}; \bar{u}]^{\mathrm{ar}(R)}\right)\Big/_{\approx}.$$

A parameterized $\mathsf{L}[\tau_1, \tau_2]$-transduction defines a parameterized mapping from τ_1-structures into τ_2-structures via $\mathsf{L}[\tau_1]$-formulas. A parameterized $\mathsf{L}[\tau_1, \tau_2]$-transduction $\Theta(\bar{x})$ is an $\mathsf{L}[\tau_1, \tau_2]$-*transduction* if $\bar{x}$ is the empty tuple. If $\theta_{\mathrm{dom}} := \top$ or $\theta_{\approx} := \bot$, we omit the respective formula in the presentation of the transduction.

An important property of $\mathsf{L}[\tau_1, \tau_2]$-transductions is that, for suitable logics L, they allow to *pull back* $\mathsf{L}[\tau_2]$-formulas, which means that for each $\mathsf{L}[\tau_2]$-formula there exists an $\mathsf{L}[\tau_1]$-formula that expresses essentially the same. Logic L is *closed under (parameterized)* L*-transductions* if for all vocabularies τ_1, τ_2 each (parameterized) $\mathsf{L}[\tau_1, \tau_2]$-transduction allows to pull back $\mathsf{L}[\tau_2]$-formulas. Each logic $\mathsf{L} \in \{\mathsf{FO}(+\mathsf{C}), \mathsf{STC}(+\mathsf{C}), \mathsf{LREC}_=\}$ is closed under (parameterized) $\mathsf{L}[\tau_1, \tau_2]$-transductions [4, 14, 16].

2.5 Canonization

In this section we introduce ordered structures, (definable) canonization and the capturing of the complexity class $\mathsf{LOGSPACE}$.

Let τ be a vocabulary with $\leq \notin \tau$. A $\tau \cup \{\leq\}$-structure A' is *ordered* if the relation symbol $\leq$ is interpreted as a linear order on the universe of A'. Let A be a τ-structure. A $\tau \cup \{\leq\}$-structure A' is an *ordered copy* of A if $A'|_\tau \cong A$. Let $\mathcal{C}$ be a class of τ-structures. A mapping f is a *canonization mapping* of $\mathcal{C}$ if it assigns every structure $A \in \mathcal{C}$ to an ordered copy $f(A) = (A_f, \leq_f)$ of A such that for all structures $A, B \in \mathcal{C}$ we have $f(A) \cong f(B)$ if $A \cong B$. We call the ordered structure $f(A)$ the *canon* of A.

Let L be a logic that extends FO. Let $\Theta(\bar{x})$ be a parameterized $\mathsf{L}[\tau, \tau \cup \{\leq\}]$-transduction, where $\bar{x}$ is a tuple of structure variables. We say $\Theta(\bar{x})$ *canonizes* a τ-structure A if there exists a tuple $\bar{p} \in A^{\bar{x}}$ such that $(A, \bar{p}) \in \mathrm{Dom}(\Theta(\bar{x}))$, and for all tuples $\bar{p} \in A^{\bar{x}}$ with $(A, \bar{p}) \in \mathrm{Dom}(\Theta(\bar{x}))$, the $\tau \cup \{\leq\}$-structure $\Theta[A, \bar{p}]$ is an ordered copy of A.[3] A *(parameterized)* L*-canonization* of a class $\mathcal{C}$ of τ-structures is a (parameterized) $\mathsf{L}[\tau, \tau \cup \{\leq\}]$-transduction that canonizes all $A \in \mathcal{C}$. A class $\mathcal{C}$ of τ-structures *admits* L*-definable canonization* if $\mathcal{C}$ has a (parameterized) L-canonization.

The following proposition and theorem are essential for proving that the class of chordal claw-free graphs admits $\mathsf{LREC}_=$-definable canonization in Section 5.

▶ **Proposition 2.2** ([12][4]). *Let $\mathcal{C}$ be a class of graphs, and $\mathcal{C}_{conn}$ be the class of all connected components of the graphs in $\mathcal{C}$. If $\mathcal{C}_{conn}$ admits $\mathsf{LREC}_=$-definable canonization, then $\mathcal{C}$ does as well.*

[3] Note that if the tuple $\bar{x}$ of parameter variables is the empty tuple, $\mathsf{L}[\tau, \tau \cup \{\leq\}]$-transduction Θ canonizes a τ-structure A if $A \in \mathrm{Dom}(\Theta)$ and the $\tau \cup \{\leq\}$-structure $\Theta[A]$ is an ordered copy of A.

[4] In [12, Corollary 3.3.21] Proposition 2.2 is only shown for $\mathsf{IFP}+\mathsf{C}$. The proof of Corollary 3.3.21 uses Lemma 3.3.18, the Transduction Lemma, and that connectivity and simple arithmetics are definable. As $\mathsf{LREC}_=$ is closed under parameterized $\mathsf{LREC}_=$-transductions, the Transduction Lemma also holds for $\mathsf{LREC}_=$ [14]. Connectivity and all arithmetics (e.g. addition, multiplication and Fact 3.3.14) that are necessary to show Lemma 3.3.18 and Corollary 3.3.21 can also be defined in $\mathsf{LREC}_=$. Further, Lemma 3.3.12 and 3.3.17, which are used to prove Lemma 3.3.18 can be shown by pulling back simple FO-formulas under $\mathsf{LREC}_=$-transductions. Hence, Corollary 3.3.21 also holds for $\mathsf{LREC}_=$.

▶ **Theorem 2.3** ([14, 16][5]). *The class of* LO*-colored directed trees admits* $\mathsf{LREC}_=$*-definable canonization.*

We can use definable canonization of a graph class to prove that LOGSPACE is captured on this graph class. Let L be a logic and $\mathcal{C}$ be a graph class. L *captures* LOGSPACE *on* $\mathcal{C}$ if for each class $\mathcal{D} \subseteq \mathcal{C}$, there exists an L-sentence defining $\mathcal{D}$ if and only if $\mathcal{D}$ is LOGSPACE-decidable.[6] A fundamental result was shown by Immerman:[7]

▶ **Theorem 2.4** ([19]). DTC *captures* LOGSPACE *on the class of all ordered graphs.*

Deterministic transitive closure logic DTC is a logic that is contained in $\mathsf{LREC}_=$ [14]. Therefore, we obtain the following corollary:

▶ **Corollary 2.5.** $\mathsf{LREC}_=$ *captures* LOGSPACE *on the class of all ordered graphs.*

Let us suppose there exists a parameterized $\mathsf{LREC}_=$-canonization of a graph class $\mathcal{C}$. Since $\mathsf{LREC}_=$ captures LOGSPACE on the class of all ordered graphs and we can pull back each $\mathsf{LREC}_=$-sentence that defines a logarithmic-space property on ordered graphs under this canonization, the capturing result transfers from ordered graphs to the class $\mathcal{C}$.

▶ **Proposition 2.6.** *Let* $\mathcal{C}$ *be a class of graphs. If* $\mathcal{C}$ *admits* $\mathsf{LREC}_=$*-definable canonization, then* $\mathsf{LREC}_=$ *captures* LOGSPACE *on* $\mathcal{C}$*.*

3 Structure of Clique Trees

Clique trees of connected chordal claw-free graphs play an important role in the subsequent canonization of chordal claw-free graphs. Thus, we analyze the structure of clique trees of connected chordal claw-free graphs in this section.

First we introduce clique trees of chordal graphs. Then we show that chordal claw-free graphs are intersection graphs of paths of a tree. We use this property to prove that each connected chordal claw-free graph has a unique clique tree. Finally, we introduce two different types of max cliques in a clique tree, star cliques and fork cliques, and show that each max clique of a connected chordal claw-free graph is of one of these types if its degree in the clique tree is at least 3.

Chordal graphs are precisely the intersection graphs of subtrees of a tree. A clique tree of a chordal graph G specifies a minimal representation of G as an intersection graph of subtrees of a tree. Clique trees were introduced independently by Buneman [2], Gavril [7] and Walter [25]. A detailed introduction of chordal graphs and their clique trees can be found in [1].

Let G be a chordal graph, and let $\mathcal{M}$ be the set of max cliques of G. Further, let $\mathcal{M}_v$ be the set of all max cliques in $\mathcal{M}$ that contain a vertex v of G. A *clique tree* of G is a tree $T = (\mathcal{M}, \mathcal{E})$ whose vertex set is the set $\mathcal{M}$ of all max cliques, where for all $v \in V$ the

[5] It is shown in [14, Remark 4.8], and in more detail in [16, Section 8.4] that the class of all colored directed trees that have a linear order on the colors admits LREC-definable canonization. This can easily be extended to LO-colored directed trees since an LO-colored graph is a special kind of colored graph that has a linear order on its colors. LREC is contained in $\mathsf{LREC}_=$ [14].

[6] A precise definition of what it means that a logic *(effectively strongly) captures* a complexity class can be found in [4, Chapter 11].

[7] Immerman proved this capturing result not only for the class of ordered graphs but for the class of ordered structures.

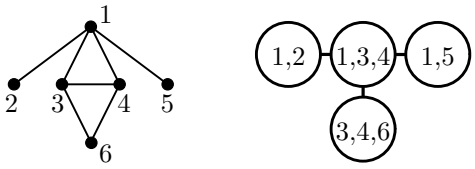


Figure 1 A chordal graph and a clique tree of the graph.

induced subgraph $T[\mathcal{M}_v]$ is connected. Hence, for each $v \in V$ the induced subgraph $T[\mathcal{M}_v]$ is a subtree of T. Then G is the intersection graph of the subtrees $T[\mathcal{M}_v]$ of T where $v \in V$. An example of a clique tree of a chordal graph is shown in Figure 1.

Let $T = (\mathcal{M}, \mathcal{E})$ be a clique tree of a chordal graph G. It is easy to see that clique tree T satisfies the *clique-intersection property*: Let $M_1, M_2, M_3 \in \mathcal{M}$ be vertices of the tree T. If M_2 is on the path from M_1 to M_3, then $M_1 \cap M_3 \subseteq M_2$.

In the following we consider the class CCF of chordal claw-free graphs. For each vertex v of a chordal claw-free graph, we prove that the set of max cliques $\mathcal{M}_v$ induces a path in each clique tree. Consequently, chordal claw-free graphs are intersection graphs of paths of a tree. Note that not all intersection graphs of paths of a tree are claw-free (see Figure 1).

▶ **Lemma 3.1.** *Let $T = (\mathcal{M}, \mathcal{E})$ be a clique tree of a chordal claw-free graph $G = (V, E)$. Then for all $v \in V$ the induced subtree $T[\mathcal{M}_v]$ is a path in T.*

Proof. Let $G = (V, E) \in$ CCF and let $T = (\mathcal{M}, \mathcal{E})$ be a clique tree of G. Let us assume there exists a vertex $v \in V$ such that the graph $T[\mathcal{M}_v]$ is not a path in T. As $T[\mathcal{M}_v]$ is a subtree of T, there exists a max clique $B \in \mathcal{M}_v$ such that B has degree at least 3. Let $A_1, A_2, A_3 \in \mathcal{M}_v$ be three distinct neighbors of B in $T[\mathcal{M}_v]$. Since A_i and B are distinct max cliques, there exists a vertex $a_i \in A_i \setminus B$, and for each $i \in [3]$, we have $A_i \in \mathcal{M}_{a_i}$, $B \notin \mathcal{M}_{a_i}$ and $T[\mathcal{M}_{a_i}]$ is connected. As T is a tree, A_1, A_2, and A_3 are all in different connected components of $T[\mathcal{M} \setminus \{B\}]$. Therefore, $\mathcal{M}_{a_i} \cap \mathcal{M}_{a_{i'}} = \emptyset$ for all $i, i' \in [3]$ with $i \neq i'$. Now, $\{v, a_1, a_2, a_3\}$ induces a claw in G, which contradicts G being claw-free: For all $i \in [3]$, there is an edge between v and a_i, because $v, a_i \in A_i$. To show that vertices a_i and $a_{i'}$ are not adjacent for $i \neq i'$, let us assume the opposite. If a_i and $a_{i'}$ are adjacent, then there exists a max clique M containing a_i and $a_{i'}$. Thus, $\mathcal{M}_{a_i} \cap \mathcal{M}_{a_{i'}} \neq \emptyset$, a contradiction. ◀

The following lemmas help us to show in Corollary 3.5 that the clique tree of a connected chordal claw-free graph is unique. This is a property that does not hold for unconnected chordal (claw-free) graphs in general.

▶ **Lemma 3.2.** *Let $T = (\mathcal{M}, \mathcal{E})$ be a clique tree of a chordal claw-free graph $G = (V, E)$. Further, let $v \in V$, and let A_1, A_2, A_3 be distinct max cliques in $\mathcal{M}_v$. Then A_2 lies between A_1 and A_3 on the path $T[\mathcal{M}_v]$ if and only if $A_2 \subseteq A_1 \cup A_3$.*

Proof. Let $G = (V, E) \in$ CCF and $T = (\mathcal{M}, \mathcal{E})$ be a clique tree of G. Further, let $v \in V$, and let $A_1, A_2, A_3 \in \mathcal{M}_v$ be distinct max cliques. First, let $A_2 \subseteq A_1 \cup A_3$, and let us assume that, w.l.o.g., A_1 lies between A_2 and A_3. Then $A_2 \cap A_3 \subseteq A_1$ according to the clique intersection property. Further, $A_2 \subseteq A_1 \cup A_3$ implies that $A_2 \setminus A_3 \subseteq A_1$. It follows that $A_2 \subseteq A_1$, which is a contradiction to A_1 and A_2 being distinct max cliques.

Now let max clique A_2 lie between A_1 and A_3 on the path $T[\mathcal{M}_v]$, and let us assume that there exists a vertex $a_2 \in A_2 \setminus (A_1 \cup A_3)$. Let $P = B_1, \ldots, B_l$ be the path $T[\mathcal{M}_v]$ (Lemma 3.1).

W.l.o.g., let $A_i = B_{j_i}$ for all $i \in [3]$ where $j_1, j_2, j_3 \in [l]$ with $j_1 < j_2 < j_3$. Further, let $A'_1 := B_{j_1+1}$ and $A'_3 := B_{j_3-1}$, and let $a_1 \in A_1 \setminus A'_1$ and $a_3 \in A_3 \setminus A'_3$. Similar to the proof of Lemma 3.1, we obtain that $\{v, a_1, a_2, a_3\}$ induces a claw in G, a contradiction. ◀

▶ **Lemma 3.3.** *Let $T_1 = (\mathcal{M}, \mathcal{E}_1)$ and $T_2 = (\mathcal{M}, \mathcal{E}_2)$ be clique trees of a chordal claw-free graph $G = (V, E)$. Then for every $v \in V$ we have $T_1[\mathcal{M}_v] = T_2[\mathcal{M}_v]$.*

Proof. Let $G = (V, E) \in \mathsf{CCF}$ and let $T_1 = (\mathcal{M}, \mathcal{E}_1)$ and $T_2 = (\mathcal{M}, \mathcal{E}_2)$ be clique trees of G. Let $v \in V$. According to Lemma 3.1, $T_1[\mathcal{M}_v]$ and $T_2[\mathcal{M}_v]$ are paths in T_1 and T_2, respectively. Let us assume there exist distinct max cliques $A, B \in \mathcal{M}_v$ such that, without loss of generality, A, B are adjacent in $T_1[\mathcal{M}_v]$ but not adjacent in $T_2[\mathcal{M}_v]$. As A and B are not adjacent in $T_2[\mathcal{M}_v]$, there exists a max clique $C \in \mathcal{M}_v$ that lies between A and B on the path $T_2[\mathcal{M}_v]$. Thus, $A \cap B \subseteq C$ according to the clique-intersection property. Since max cliques A and B are adjacent in $T_1[\mathcal{M}_v]$, either A lies between B and C, or B lies between A and C on the path $T_1[\mathcal{M}_v]$. W.l.o.g., let A lie between B and C on the path $T_1[\mathcal{M}_v]$. Then $A \subseteq B \cup C$ by Lemma 3.2. Thus, we have $A \setminus B \subseteq C$. Since $A \cap B \subseteq C$, this yields that $A \subseteq C$, which is a contradiction to A and C being distinct max cliques. ◀

▶ **Lemma 3.4.** *Let $T = (\mathcal{M}, \mathcal{E})$ be a clique tree of a connected chordal claw-free graph $G = (V, E)$. Then*

$$T = \bigcup_{v \in V} T[\mathcal{M}_v].$$

Proof. Let $G = (V, E)$ be a connected chordal claw-free graph and $T = (\mathcal{M}, \mathcal{E})$ be a clique tree of G. Clearly, the graphs T and $T' := \bigcup_{v \in V} T[\mathcal{M}_v]$ have the same vertex set, and T' is a subgraph of the tree T. In order to prove that $T = T'$, we show that T' is connected.

For all vertices $v \in V$, the graph $T'[\mathcal{M}_v]$ is connected because $T[\mathcal{M}_v]$ is connected. For each edge $\{u, v\} \in E$ of the graph G, there exists a max clique that contains u and v, and therefore, we have $\mathcal{M}_u \cap \mathcal{M}_v \neq \emptyset$. Hence, $T'[\mathcal{M}_u \cup \mathcal{M}_v]$ is connected for every edge $\{u, v\} \in E$. Since G is connected, it follows that $T'[\bigcup_{v \in V} \mathcal{M}_v]$ is connected. Clearly, $\bigcup_{v \in V} \mathcal{M}_v = \mathcal{M}$. Consequently, the graph T' is connected. ◀

As a direct consequence of Lemma 3.3 and Lemma 3.4 we obtain the following corollary. It follows that each connected chordal claw-free graph has a unique clique tree.

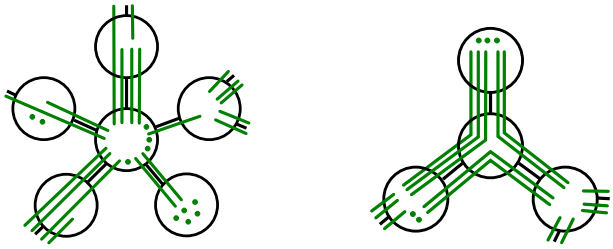
▶ **Corollary 3.5.** *Let T_1 and T_2 be clique trees of a connected chordal claw-free graph G. Then $T_1 = T_2$.*

In the following let $G = (V, E)$ be a connected chordal claw-free graph and let $T_G = (\mathcal{M}, \mathcal{E})$ be its clique tree.

Let B be a max clique of G. If for all $v \in B$ max clique B is an end of path $T_G[\mathcal{M}_v]$, we call B a *star clique*. Thus, B is a star clique if, and only if, every vertex in B is contained in at most one neighbor of B in T_G. A picture of a star clique can be found in Figure 2a. Clearly, every max clique of degree 1, i.e. every leaf, of clique tree T_G is a star clique.

A max clique B of degree 3 is called a *fork clique* if for every $v \in B$ there exist two neighbors A, A' of B with $A \neq A'$ such that $\mathcal{M}_v = \{B, A, A'\}$, and for all neighbors A, A' of B with $A \neq A'$ there exists a vertex $v \in B$ with $\mathcal{M}_v = \{B, A, A'\}$. Figure 2b shows a sketch of a fork clique. Note that two fork cliques cannot be adjacent.

The following lemma provides more information about the structure of the clique tree of a connected chordal claw-free graph. A proof can be found in Appendix A.



(a) A star clique **(b)** A fork clique

Figure 2 A star clique and a fork clique.

▶ **Lemma 3.6.** *Let $B \in \mathcal{M}$. If the degree of B in clique tree T_G is at least 3, then B is a star clique or a fork clique.*

▶ **Corollary 3.7.** *Let $B \in \mathcal{M}$ be a fork clique. Then every neighbor of B in clique tree T_G is a star clique.*

Proof. Let us assume max clique A is a neighbor of fork clique B, and A is not a star clique. Then the degree of A is at least 2. As A cannot be a fork clique, Lemma 3.6 implies that A has degree 2. Since B is a fork clique, there does not exist a vertex $v \in A$ that is contained in B and the other neighbor of A. Thus, A is a star clique, a contradiction. ◀

4 Defining the Supplemented Clique Tree in STC+C

In this section we define the supplemented clique tree of a connected chordal claw-free graph G. We obtain the supplemented clique tree by transferring the clique tree T_G into a directed tree and including some of the structural information about each max clique into the directed clique tree by means of an LO-coloring. We show that there exists a parameterized STC+C-transduction that defines for each connected chordal claw-free graph and every tuple of suitable parameters an isomorphic copy of such a supplemented clique tree. In order to do this, we first present (parameterized) transductions for the clique tree and the directed clique tree. Throughout this section we let $\bar{x}, \bar{y}$ and $\bar{y}'$ be triples of structure variables.

In a first step we present a transduction $\Theta = (\theta_U(\bar{y}), \theta_{\approx}(\bar{y}, \bar{y}'), \theta_E(\bar{y}, \bar{y}'))$ that defines for each connected chordal claw-free graph G a tree isomorphic to the clique tree of G.

In the following, let $G = (V, E)$ be a chordal claw-free graph, and let $\mathcal{M}$ be the set of max cliques of G. A triple $\bar{b} = (b_1, b_2, b_3) \in V^3$ *spans* a max clique $A \in \mathcal{M}$ if A is the only max clique that contains the vertices b_1, b_2 and b_3. Thus, $\bar{b}$ spans max clique $A \in \mathcal{M}$ if and only if $\mathcal{M}_{b_1} \cap \mathcal{M}_{b_2} \cap \mathcal{M}_{b_3} = \{A\}$. We call $\bar{b} \in V^3$ a *spanning triple* of G if $\bar{b}$ spans a max clique. We use spanning triples to represent max cliques. Note that this concept was already used in [20] and [14] to represent max cliques of interval graphs.

▶ **Lemma 4.1.** *Every max clique of a chordal claw-free graph is spanned by a triple of vertices.*

Proof. Let $T = (\mathcal{M}, \mathcal{E})$ be a clique tree of a chordal claw-free graph G. Let $B \in \mathcal{M}$ and let $v \in B$. By Lemma 3.1, the induced subgraph $T[\mathcal{M}_v]$ is a path $P = B_1, \ldots, B_l$. Let $B = B_i$. If $i > 1$, let u be a vertex in $B \setminus B_{i-1}$, and let w be a vertex in $B \setminus B_{i+1}$ if $i < l$. We let $u = v$ if $i = 1$, and we let $w = v$ if $i = l$. Then (u, v, w) spans max clique B: Clearly, $u, v, w \in B$. It remains to show, that there does not exist a max clique $A \in \mathcal{M}$ with $A \neq B$ and $u, v, w \in A$. Let us suppose such a max clique A exists. Since $v \in A$, max clique A is

a vertex on path P. W.l.o.g., let $A = B_j$ for $j < i$. According to the clique intersection property, we have $u \in A \cap B \subseteq B_{i-1}$, a contradiction. ◀

As a direct consequence of Lemma 4.1, there exists an at most cubic number of max cliques in a chordal claw-free graph.

The following observations contain properties that help us to define the transduction Θ. Proofs can be found in [16].

▶ **Observation 4.2.** *Let $G = (V, E)$ be a chordal claw-free graph. Let $\bar{v} = (v_1, v_2, v_3) \in V^3$. Then $\bar{v}$ is a spanning triple of G if, and only if, $\tilde{v}$ is a clique and $\{w_1, w_2\} \in E$ for all vertices $w_1, w_2 \in N(v_1) \cap N(v_2) \cap N(v_3)$ with $w_1 \neq w_2$.*

From the characterization of spanning triples in Observation 4.2, it follows that there exists an FO-formula $\theta_U(\bar{y})$ that is satisfied by a chordal claw-free graph $G = (V, E)$ and a triple $\bar{v} \in V^3$ if and only if $\bar{v}$ is a spanning triple of G.

▶ **Observation 4.3.** *Let $G = (V, E)$ be a chordal claw-free graph. Let A be a max clique of G, and let the triple $\bar{v} = (v_1, v_2, v_3) \in V^3$ span A. Then $w \in A$ if, and only if, $w \in \tilde{v}$ or $\{w, v_j\} \in E$ for all $j \in [3]$.*

Observation 4.3 yields that there further exists an FO-formula $\varphi_{\mathcal{M}}(\bar{y}, z)$ that is satisfied for $\bar{v} \in V^3$ and $w \in V$ in a chordal claw-free graph $G = (V, E)$ if, and only if, $\bar{v}$ spans a max clique A and $w \in A$. We can use this formula to obtain an FO-formula $\theta_{\approx}(\bar{y}, \bar{y}')$ such that for all chordal claw-free graphs $G = (V, E)$ and all triples $\bar{v}, \bar{v}' \in V^3$ we have $G \models \theta_{\approx}(\bar{v}, \bar{v}')$ if, and only if, $\bar{v}$ and $\bar{v}'$ span the same max clique.

In the following we consider connected chordal claw-free graphs G. The next observation is a direct consequence of Lemma 3.4 and Lemma 3.2.

▶ **Observation 4.4.** *Let $G = (V, E)$ be a connected chordal claw-free graph, and $T_G = (\mathcal{M}, \mathcal{E})$ be the clique tree of G. Let $A, B \in \mathcal{M}$. Max cliques A and B are adjacent in T_G if, and only if, there exists a vertex $v \in V$ such that $v \in A \cap B$ and for all $C \in \mathcal{M}$ with $v \in C$ we have $C \not\subseteq A \cup B$.*

It follows from Observation 4.4 that there exists an FO-formula $\theta_E(\bar{y}, \bar{y}')$ that is satisfied for triples $\bar{v}, \bar{v}' \in V^3$ in a connected chordal claw-free graph $G = (V, E)$ if, and only if, $\bar{v}$ and $\bar{v}'$ span adjacent max cliques.

It is not hard to see that $\Theta = (\theta_U, \theta_{\approx}, \theta_E)$ is an FO-transduction that defines for each connected chordal claw-free graph G a tree isomorphic to the clique tree of G.

▶ **Lemma 4.5.** *There exists an FO-transduction Θ such that $\Theta[G] \cong T_G$ for all $G \in$ con-CCF.*

Now we transfer the clique tree into a directed tree. Let R be a leaf of the clique tree T_G. We transform T_G into a directed tree by rooting T_G at max clique R. We denote the resulting directed clique tree by $T_G^R = (\mathcal{M}, \mathcal{E}_R)$. Since R is a leaf of T_G, the following corollary is an immediate consequence of Lemma 3.6.

▶ **Corollary 4.6.** *Let A be a max clique of a connected chordal claw-free graph G. Then A is a vertex with at least two children in T_G^R only if A is a star clique or a fork clique.*

In the following we show that there exists a parameterized STC-transduction $\Theta'(\bar{x})$ which defines an isomorphic copy of T_G^R for each connected chordal claw-free graph G and triple $\bar{r} \in V^3$ that spans a leaf R of T_G.

Clearly, we can define an FO-formula $\theta'_{\mathrm{dom}}(\bar{x})$ such that for all connected chordal claw-free graphs G and $\bar{r} \in V^3$ we have $G \models \theta'_{\mathrm{dom}}(\bar{r})$ if, and only if, $\bar{r} \in V^3$ spans a leaf of

T_G. Then θ'_{dom} defines the triples of parameters of transduction $\Theta'(\bar{x})$. Further, we let $\theta'_U(\bar{x},\bar{y}) := \theta_U(\bar{y})$ and $\theta'_{\approx}(\bar{x},\bar{y},\bar{y}') := \theta_{\approx}(\bar{y},\bar{y}')$. Finally, we let $\theta'_E(\bar{x},\bar{y},\bar{y}')$ be satisfied for triples $\bar{r},\bar{v},\bar{v}' \in V^3$ in a connected chordal claw-free graph $G=(V,E)$ if, and only if, $\bar{r}$, $\bar{v}$ and $\bar{v}'$ span max cliques R, A and A', respectively, and (A,A') is an edge in T_G^R. Note that (A,A') is an edge in T_G^R precisely if $\{A,A'\}$ is an edge in T_G and there exists a path between R and A in T_G after removing A'. Thus, formula θ'_E can be constructed in STC. We let $\Theta'(\bar{x}) := (\theta'_{\text{dom}}(\bar{x}),\theta'_U(\bar{x},\bar{y}),\theta'_{\approx}(\bar{x},\bar{y},\bar{y}'),\theta'_E(\bar{x},\bar{y},\bar{y}'))$, and conclude:

▶ **Lemma 4.7.** *There exists a parameterized* STC*-transduction* $\Theta'(\bar{x})$ *such that* $\text{Dom}(\Theta'(\bar{x}))$ *is the set of all pairs* $(G,\bar{r})$ *where* $G=(V,E) \in$ con-CCF *and* $\bar{r} \in V^3$ *spans a leaf* R *of* T_G*, and* $\Theta'[G,\bar{r}] \cong T_G^R$ *for all* $(G,\bar{r}) \in \text{Dom}(\Theta'(\bar{x}))$ *where* $\bar{r}$ *spans the max clique* R *of* G*.*

We now equip each max clique of the directed clique tree T_G^R with structural information. We do this by coloring the directed clique tree T_G^R with an LO-coloring. An LO-color is a binary relation on a linearly ordered set of basic color elements. Into each LO-color, we encode three numbers. Isomorphisms of LO-colored trees preserve the information that is encoded in the LO-colors. Thus, an LO-colored tree and its canon contain the same numbers encoded in their LO-colors. We call this LO-colored directed clique tree a supplemented clique tree. More precisely, let $G \in$ con-CCF and let R be a leaf of the clique tree T_G of G, then the *supplemented clique tree* S_G^R is the 5-tuple $(\mathcal{M},\mathcal{E}_R,[0,|V|],\leq_{[0,|V|]},L)$ where

- $(\mathcal{M},\mathcal{E}_R)$ is the directed clique tree T_G^R of G,
- $\leq_{[0,|V|]}$ is the natural linear order on the set of basic color elements $[0,|V|]$,
- $L \subseteq \mathcal{M} \times [0,|V|]^2$ is the ternary color relation where
  - $(A,0,n) \in L$ iff n is the number of vertices in A that are not in any child of A in T_G^R,
  - $(A,1,n) \in L$ iff n is the number of vertices that are contained in A and in the parent of A in T_G^R if $A \neq R$, and $n=0$ if $A=R$,
  - $(A,2,n) \in L$ iff n is the number of vertices in A that are in two children of A in T_G^R.[8]

In its structural representation the supplemented clique tree S_G^R corresponds to the 6-tuple $(\mathcal{M} \dot{\cup} [0,|V|],\mathcal{M},\mathcal{E}_R,[0,|V|],\leq_{[0,|V|]},L)$.

The properties encoded in the colors of the max cliques are easily expressible in STC+C. Therefore, we can extend the parameterized STC-transduction $\Theta'(\bar{x})$ to a parameterized STC+C-transduction $\Theta''(\bar{x})$ that defines an LO-colored graph isomorphic to S_G^R for every connected chordal claw-free graph G and triple $\bar{r} \in V^3$ that spans a leaf R of T_G. More details on how to construct $\Theta''(\bar{x})$ can be found in [16].

▶ **Lemma 4.8.** *There is a parameterized* STC+C*-transduction* $\Theta''(\bar{x})$ *such that* $\text{Dom}(\Theta''(\bar{x}))$ *is the set of all pairs* $(G,\bar{r})$ *where* $G=(V,E) \in$ con-CCF *and* $\bar{r} \in V^3$ *spans a leaf* R *of* T_G*, and* $\Theta''[G,\bar{r}] \cong S_G^R$ *for all* $(G,\bar{r}) \in \text{Dom}(\Theta''(\bar{x}))$ *where* $\bar{r}$ *spans the max clique* R *of* G*.*

5 Canonization

In this section we prove that there exists a parameterized $\text{LREC}_=$-canonization of the class con-CCF of connected chordal claw-free graphs. Then Proposition 2.2 implies that there also exists one for the class CCF of chordal claw-free graphs, and we obtain our main result:

▶ **Theorem 5.1.** *The class of chordal claw-free graphs admits* $\text{LREC}_=$*-definable canonization.*

[8] Let A be a max clique and n be the number of vertices in A that are in two children of A in T_G^R. Corollary 4.6 implies that A is a fork clique if and only if $n>0$.

As a consequence of Proposition 2.6 and the logarithmic-space data complexity of $\mathsf{LREC}_=$ we obtain the following corollaries.

▶ **Corollary 5.2.** $\mathsf{LREC}_=$ *captures* LOGSPACE *on the class of chordal claw-free graphs.*

▶ **Corollary 5.3.** *There exists a logarithmic-space canonization algorithm for the class of chordal claw-free graphs.*

Since $\mathsf{LREC}_=$ is contained in FP+C [14], Theorem 5.1 also implies that there exists an FP+C-canonization of the class of chordal claw-free graphs. We directly obtain (see e.g. [4]):

▶ **Corollary 5.4.** FP+C *captures* PTIME *on the class of chordal claw-free graphs.*

Now let us prove that there exists a parameterized $\mathsf{LREC}_=$-canonization of con-CCF. By Lemma 4.8 there exists a parameterized STC+C-, and therefore, $\mathsf{LREC}_=$-transduction $\Theta''(\bar{x})$ such that $\Theta''[G,\bar{r}]$ is isomorphic to the LO-colored tree S_G^R for all connected chordal claw-free graphs $G=(V,E)$ and all triples $\bar{r}\in V^3$ that span a leaf R of T_G. Further, there exists an $\mathsf{LREC}_=$-canonization Θ^{LO} of the class of LO-colored directed trees according to Theorem 2.3. In the following, we show that there also exists an $\mathsf{LREC}_=$-transduction Θ^K which defines for each canon $K(S_G^R)$ of a supplemented clique tree S_G^R of $G\in$ con-CCF the canon $K(G)$ of G. Then we can compose the (parameterized) $\mathsf{LREC}_=$-transductions $\Theta''(\bar{x})$, Θ^{LO} and Θ^K to obtain a parameterized $\mathsf{LREC}_=$-canonization of the class of connected chordal claw-free graphs (see [16]).

We let $\mathsf{LREC}_=[\{V,E,M,\trianglelefteq,L,\leq\},\{E,\leq\}]$-transduction $\Theta^K=(\theta_V(p),\theta_E(p,p'),\theta_\leq(p,p'))$ define for each canon $K(S_G^R)=(U_K,V_K,E_K,M_K,\trianglelefteq_K,L_K,\leq_K)$ of a supplemented clique tree of $G\in$ con-CCF an ordered isomorphic copy $K(G)=(V_K',E_K',\leq_K')$ of $G=(V,E)$. We let V_K' be the set $[|V|]$, and $\leq_K'$ be the natural linear order on $[|V|]$. As $[0,|V|]$ is the set of basic color elements of S_G^R, the set M_K of basic color elements of the canon $K(S_G^R)$ contains exactly $|V|+1$ elements. Hence, we can easily define the vertex set of $K(G)$ by counting the number of basic color elements of $K(S_G^R)$. We let $\varphi_V(p):=\exists q\left(p\leq q\wedge p\neq 0\wedge \#x\,M(x)=q\right)$. Further, we let $\theta_\leq(p,p'):=p\leq p'$. In order to show that there exists an $\mathsf{LREC}_=$-formula $\theta_E(p,p')$, which defines the edge relation of $K(G)$, we exploit the property that $\mathsf{LREC}_=$ captures LOGSPACE on ordered structures (Corollary 2.5), and show that there exists a logarithmic-space algorithm that computes the edge relation of $K(G)$, instead. In order to do this, we present a logarithmic-space algorithm that outputs the max cliques of $K(G)$. As every edge is a subset of some max clique and every two distinct vertices in a max clique are adjacent, such a logarithmic-space algorithm can easily be extended to a logarithmic-space algorithm that decides whether a pair of numbers is an edge of $K(G)$.

▶ **Lemma 5.5.** *There exists a logarithmic-space algorithm that, given the canon $K(S_G^R)$ of a supplemented clique tree of $G\in$* con-CCF*, computes the set of max cliques of the canon $K(G)$ of G.*

In the following, we sketch the idea of the algorithm. A detailed proof of Lemma 5.5 can be found in [16].

The algorithm performs a post-order tree traversal[9] on the underlying tree of the canon $K(S_G^R)$ of the supplemented clique tree S_G^R. Let $m_1,\ldots,m_{|\mathcal{M}|}$ be the respective post-order

[9] In a *tree traversal*, we visit every vertex of the tree exactly once. We obtain the *post-order traversal* (see e.g. [23]) of an ordered directed tree T with root r recursively as follows: Let d be the out-degree of r. For all $i\in\{1,\ldots,d\}$, in increasing order, perform a post-order traversal on the subtree rooted at child i of the root. Afterward, visit r. For example in [22], it is shown that there exists a logarithmic-space algorithm for depth-first tree traversal. By selecting all vertices that are not followed by the move **down** in the depth-first traversal in [22], we obtain the post-order traversal sequence in logarithmic space.

traversal sequence of the vertices. Each vertex $m_k \in V_K$ of the canon $K(S_G^R)$ corresponds to a vertex, i.e. a max clique $A_k \in \mathcal{M}$, in the supplemented clique tree S_G^R. We call $A_1, \ldots, A_{m_{|\mathcal{M}|}}$ the *transferred traversal sequence*. For all $k \in \{1, \ldots, |\mathcal{M}|\}$, starting with $k = 1$, the algorithm constructs for $m_k \in V_K$ a copy $B_{m_k} \subseteq [|V|]$ of A_k.

From the information encoded in the colors, we know the number of vertices in A_k that are not in any max clique that occurs before A_k in the transferred traversal sequence. For these vertices, we add the smallest numbers of $[|V|]$ to B_{m_k} that were not already used. Further, we need to know how many vertices of A_k are in a max clique A_i that occurs before A_k in the transferred traversal sequence, and what numbers these vertices were assigned to. These numbers have to be added to B_{m_k} as well.

Now, if A_k is a fork clique (the information of whether vertex m_k corresponds to a fork clique A_k is encoded in the color of m_k), then we know the vertices of A_k are all contained in at least one of its two children, which occur before A_k in the transferred traversal sequence; and we can use the information in the colors of m_k and its children to find out the number of vertices in A_k that are contained in the first child, the second child and both children of A_k, respectively.

If A_k is not a fork clique, then the vertices in A_k that are in max cliques that occur before A_k within the transferred traversal sequence are precisely the vertices in the pairwise intersection of A_k with its children, and the intersection of A_k with its sibling if A_k is the second child of a fork clique. Note that these intersections are disjoint sets of vertices because A_k is a star clique, or otherwise only has one child (Corollary 4.6) and cannot be the second child of a fork clique by Corollary 3.7. Again we can use the information within the colors to find out the number of vertices in A_k that are contained in the respective child of A_k, and the number of vertices in the intersection of A_k and its sibling if A_k is the second child of a fork clique.

Therefore, in both cases, all vertices of A_k that are in a max clique A_i that occurs before A_k in the transferred traversal sequence, are contained in the children of A_k, or the first child of a fork clique if A_k is the second one.

For the numbers in each max clique B_{m_j} (where m_j does not corresponds to the second child of a fork clique), we maintain the property that if a number $l \in B_{m_j}$ is contained in more ancestors of B_{m_j} than a number $l' \in B_{m_j}$, then $l > l'$. Thus, if B_{m_j} is a child of a max clique $B_{m_{j'}}$, then the intersection $B_{m_j} \cap B_{m_{j'}}$ contains precisely the $|B_{m_j} \cap B_{m_{j'}}|$ largest numbers of B_{m_j}. We can use this property to determine the numbers in B_{m_i} that have to be included into B_{m_k} if m_i is a child of m_k and also if m_i corresponds to the first child of a fork clique and m_k to the second one. Note that we do not have to remember the max cliques $B_{m_1}, \ldots, B_{m_{k-1}}$ as we can recompute all values that are necessary to obtain the numbers that have to be included into B_{m_k}. The recomputation can be done in logarithmic space. As a consequence, we obtain an algorithm that computes the max cliques of the canon $K(G)$ in logarithmic space.

6 Conclusion

Currently, there exist hardly any logical characterizations of LOGSPACE on non-trivial natural classes of unordered structures. The only ones previously presented are that $\mathsf{LREC}_=$ captures LOGSPACE on (directed) trees and interval graphs [13, 14]. By showing that $\mathsf{LREC}_=$ captures LOGSPACE also on the class of chordal claw-free graphs, we contribute a further characterization of LOGSPACE on an unordered graph class. It would be interesting to investigate further classes of unordered structures such as the class of planar graphs or classes of graphs of bounded treewidth. The author conjectures that $\mathsf{LREC}_=$ captures LOGSPACE on the class of all planar graphs that are equipped with an embedding.

Our main result, which states that the class of chordal claw-free graphs admits $\mathsf{LREC}_=$-definable canonization, does not only imply that $\mathsf{LREC}_=$ captures LOGSPACE on this graph class, but also that there exists a logarithmic-space canonization algorithm for the class of chordal claw-free graphs. Hence, the isomorphism and automorphism problem for this graph class is solvable in logarithmic space.

Further, we make a contribution to the investigation of PTIME's characteristics on restricted classes of graphs. It follows from our main result that there is an FP+C-canonization of the class of chordal claw-free graphs. As a consequence, FP+C captures PTIME on this graph class. Thus, we add the class of chordal claw-free graphs to the (so far) short list of graph classes that are not closed under taking minors and on which PTIME is captured.

Acknowledgements. The author wants to thank Nicole Schweikardt and the reviewers for helpful comments that contributed to improving the paper.

References

1. J. R. S. Blair and B. Peyton. An introduction to chordal graphs and clique trees. In A. George, J. R. Gilbert, and J. W. H. Liu, editors, *Graph Theory and Sparse Matrix Computation*, pages 1–29. Springer New York, 1993.
2. P. Buneman. A characterisation of rigid circuit graphs. *Discrete Math.*, 9:205–212, 1974.
3. J. Cai, M. Fürer, and N. Immerman. An optimal lower bound on the number of variables for graph identification. *Combinatorica*, 12:389–410, 1992.
4. H.-D. Ebbinghaus and J. Flum. *Finite Model Theory*. Springer-Verlag, 1999.
5. K. Etessami and N. Immerman. Tree canonization and transitive closure. *Information and Computation*, 157(1–2):2–24, 2000.
6. R. Fagin. Generalized first–order spectra and polynomial–time recognizable sets. In R.M. Karp, editor, *Complexity of Computation, SIAM-AMS Proceedings 7*, pages 43–73, 1974.
7. F. Gavril. The intersection graphs of subtrees in trees are exactly the chordal graphs. *Journal of Combinatorial Theory, Series B*, 16(1):47–56, 1974.
8. M. Grohe. Fixed-point logics on planar graphs. In *LICS*, pages 6–15, 1998.
9. M. Grohe. Definable tree decompositions. In *LICS*, pages 406–417, 2008.
10. M. Grohe. Fixed-point definability and polynomial time on chordal graphs and line graphs. In A. Blass, N. Dershowitz, and W. Reisig, editors, *Fields of Logic and Computation, Essays Dedicated to Yuri Gurevich on the Occasion of His 70th Birthday*, pages 328–353, 2010.
11. M. Grohe. Fixed-point definability and polynomial time on graphs with excluded minors. In *LICS*, 2010.
12. M. Grohe. Descriptive complexity, canonisation, and definable graph structure theory, 2013. [Online; accessed 2017-03-31]. URL: `http://lii.rwth-aachen.de/images/Mitarbeiter/pub/grohe/cap/all.pdf`.
13. M. Grohe, B. Grußien, A. Hernich, and B. Laubner. L-recursion and a new logic for logarithmic space. In *CSL*, pages 277–291, 2011.
14. M. Grohe, B. Grußien, A. Hernich, and B. Laubner. L-recursion and a new logic for logarithmic space. *Logical Methods in Computer Science*, 9(1), 2012.
15. M. Grohe and J. Mariño. Definability and descriptive complexity on databases of bounded tree-width. In *ICDT*, pages 70–82, 1999.
16. B. Grußien. *Capturing Polynomial Time and Logarithmic Space using Modular Decompositions and Limited Recursion*. PhD thesis, Humboldt-Universität zu Berlin, 2016.
17. B. Grußien. Capturing polynomial time using modular decomposition. In *LICS*, 2017.
18. N. Immerman. Relational queries computable in polynomial time. *Information and Control*, 68:86–104, 1986.

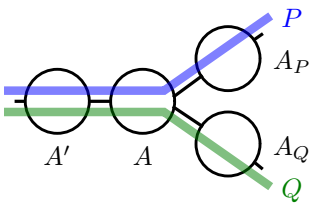


Figure 3 A fork of P and Q.

19 N. Immerman. Languages that capture complexity classes. *SIAM Journal on Computing*, 16:760–778, 1987.
20 B. Laubner. Capturing polynomial time on interval graphs. In *LICS*, pages 199–208, 2010.
21 B. Laubner. *The Structure of Graphs and New Logics for the Characterization of Polynomial Time*. PhD thesis, Humboldt-Universität zu Berlin, 2011.
22 S. Lindell. A logspace algorithm for tree canonization. In *STOC*, pages 400–404, 1992.
23 R. Sedgewick. *Algorithms in Java: Parts 1-4*. Addison-Wesley, 3rd edition, 2002.
24 M. Y. Vardi. The complexity of relational query languages. In *STOC*, pages 137–146, 1982.
25 J. R. Walter. *Representations of Rigid Cycle Graphs*. PhD thesis, Wayne State University, 1972.

A Proof of Lemma 3.6

In order to prove Lemma 3.6, we further analyze the structure of the clique tree. Throughout this section, we let $G = (V, E)$ be a connected chordal claw-free graph and $T_G = (\mathcal{M}, \mathcal{E})$ be its clique tree.

The following corollary follows directly from Lemma 3.2.

▶ **Corollary A.1.** *Let $v \in V$. Then for all $w \in V \setminus \{v\}$ the graph $T_G[\mathcal{M}_v \setminus \mathcal{M}_w]$ is connected.*[10]

Proof. Let $v \in V$ and let $w \in V \setminus \{v\}$. Let $P = A_1, \ldots, A_l$ be the path $T_G[\mathcal{M}_v]$, and let us assume $T_G[\mathcal{M}_v \setminus \mathcal{M}_w]$ is not connected. Then there exist $i, j, k \in [l]$ with $i < j < k$ such that $A_i, A_k \in \mathcal{M}_v \setminus \mathcal{M}_w$ and $A_j \in \mathcal{M}_w$. By Lemma 3.2 we have $A_j \subseteq A_i \cup A_k$. Thus, vertex $w \in A_j$ is also contained in A_i or A_k, a contradiction. ◀

Let P and Q be two paths in T_G. We call $(A', A, \{A_P, A_Q\}) \in V^2 \times \binom{V}{2}$ a *fork* of P and Q, if $P[\{A', A, A_P\}]$ and $Q[\{A', A, A_Q\}]$ are induced subpaths of length 3 of P and Q, respectively, and neither A_P occurs in Q nor A_Q occurs in P. Figure 3 shows a fork of paths P and Q. We say P and Q *fork (in B)* if there exists a fork $(A', A, \{A_P, A_Q\})$ of P and Q (with $A = B$).

▶ **Lemma A.2.** *Let $v, w \in V$. If the paths $T_G[\mathcal{M}_v]$ and $T_G[\mathcal{M}_w]$ fork, then $T_G[\mathcal{M}_v]$ and $T_G[\mathcal{M}_w]$ are paths of length 3.*

Proof. Let $v, w \in V$. Clearly, if $T_G[\mathcal{M}_v]$ and $T_G[\mathcal{M}_w]$ fork, then they must be paths of length at least 3. It remains to prove that their length is at most 3. For a contradiction, let us assume the length of $T_G[\mathcal{M}_v]$ is at least 4. Let $(A_1, B, \{A_2, A_2'\})$ be a fork of $T_G[\mathcal{M}_v]$ and $T_G[\mathcal{M}_w]$ where $A_2 \in \mathcal{M}_v \setminus \mathcal{M}_w$ and $A_2' \in \mathcal{M}_w \setminus \mathcal{M}_v$.

First let us assume there exists a max clique $A_0 \in \mathcal{M}_v$ such that $P = A_0, A_1, B, A_2$ is a subpath of $T_G[\mathcal{M}_v]$ of length 4. According to Corollary A.1, the graph $T_G[\mathcal{M}_v \setminus \mathcal{M}_w]$

[10] We define the empty graph as connected.

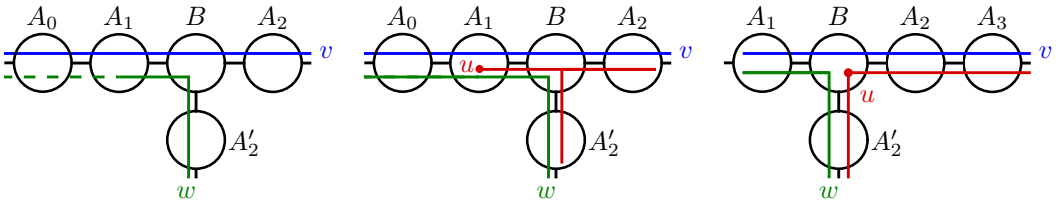


Figure 4 Illustrations for the proof of Lemma A.2.

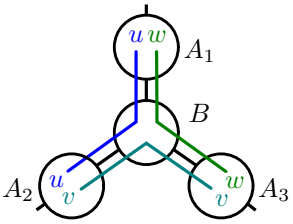


Figure 5 A fork triangle.

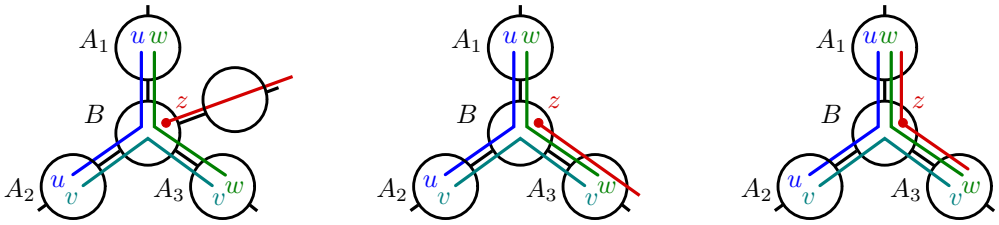
is connected. Thus, we have $A_0 \in \mathcal{M}_w$ (see Figure 4a). Now A_0 and A_1 are distinct max cliques. Therefore, there exists a vertex $u \in A_1 \setminus A_0$. As P is a subpath of $T_G[\mathcal{M}_v]$ and $P' = A_0, A_1, B, A'_2$ is a subpath of $T_G[\mathcal{M}_w]$, vertex u is not only contained in A_1 but also in B, A_2 and A'_2 by Lemma 3.2 (see Figure 4b). As a consequence, $T_G[\mathcal{M}_u]$ is not a path, a contradiction to Lemma 3.1.

Next, let us assume there exists a max clique $A_3 \in \mathcal{M}_v$ such that $P = A_1, B, A_2, A_3$ is a subpath of $T_G[\mathcal{M}_v]$ of length 4. Further, $P' = A_1, B, A'_2$ is a subpath of $T_G[\mathcal{M}_w]$. As A_1 and B are max cliques, there exists a vertex $u \in B \setminus A_1$. By Lemma 3.2, vertex u is also contained in A_2, A_3 and A'_2 as shown in Figure 4c. Now let us consider the paths $T_G[\mathcal{M}_v]$ and $T_G[\mathcal{M}_u]$. $Q = A_3, A_2, B, A_1$ is a subpath of $T_G[\mathcal{M}_v]$, and $Q' = A_3, A_2, B, A'_2$ is a subpath of $T_G[\mathcal{M}_u]$. Clearly, $(A_2, B, \{A_1, A'_2\})$ is a fork of $T_G[\mathcal{M}_v]$ and $T_G[\mathcal{M}_u]$. According to the previous part of this proof, we obtain a contradiction. ◀

The max cliques $A_1, A_2, A_3 \in \mathcal{M}$ form a *fork triangle* around a max clique $B \in \mathcal{M}$ if A_1, A_2 and A_3 are distinct max cliques in the neighborhood of B and there exist vertices $u, v, w \in V$ such that $\mathcal{M}_u = \{A_1, B, A_2\}$, $\mathcal{M}_v = \{A_2, B, A_3\}$ and $\mathcal{M}_w = \{A_3, B, A_1\}$. We say that max clique $B \in \mathcal{M}$ has a *fork triangle* if there exist max cliques $A_1, A_2, A_3 \in \mathcal{M}$ that form a fork triangle around B. Figure 5 depicts a fork triangle around a max clique B. Clearly, if a max clique B has a fork triangle, then B is a vertex of degree at least 3 in T_G.

▶ **Lemma A.3.** *Let $v, w \in V$, and let $B \in \mathcal{M}$ be a max clique. If $T_G[\mathcal{M}_u]$ and $T_G[\mathcal{M}_v]$ fork in B, then B has a fork triangle.*

Proof. Let $v, w \in V$, let $B \in \mathcal{M}$ be a max clique, and let $T_G[\mathcal{M}_u]$ and $T_G[\mathcal{M}_v]$ fork in B. Then $T_G[\mathcal{M}_u]$ and $T_G[\mathcal{M}_v]$ are paths of length 3 by Lemma A.2. Let $\mathcal{M}_u = \{A_2, B, A_1\}$ and $\mathcal{M}_v = \{A_2, B, A_3\}$ with $A_1 \neq A_3$. Since B and A_2 are max cliques, there exists a vertex $w \in B \setminus A_2$. Now, we can apply Lemma 3.2 to the paths $T_G[\mathcal{M}_u]$ and $T_G[\mathcal{M}_v]$, and obtain that $w \in A_1$ and $w \in A_3$. As $T_G[\mathcal{M}_w]$ and $T_G[\mathcal{M}_u]$ fork, the path $T_G[\mathcal{M}_w]$ must be of length 3 by Lemma A.2. Thus, $\mathcal{M}_w = \{A_3, B, A_1\}$. Hence, A_1, A_2, A_3 form a fork triangle around B. ◀



(a) $|\mathcal{M}_z \cap \mathcal{W}| = 0$ **(b)** $|\mathcal{M}_z \cap \mathcal{W}| = 1$ **(c)** $|\mathcal{M}_z \cap \mathcal{W}| = 2$

Figure 6 Illustrations for the proof of Lemma A.4.

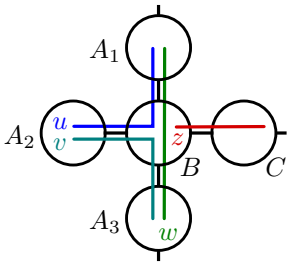


Figure 7 Illustration for the proof of Lemma A.5.

▶ **Lemma A.4.** *Let $z \in V$. If max clique $B \in \mathcal{M}_z$ has a fork triangle, then $|\mathcal{M}_z| = 3$ and B is in the middle of path $T_G[\mathcal{M}_z]$.*

Proof. Let $z \in V$, and let $B \in \mathcal{M}_z$ have a fork triangle. Then, there exist $u, v, w \in V$ and distinct neighbor max cliques A_1, A_2, A_3 of B such that $\mathcal{M}_u = \{A_1, B, A_2\}$, $\mathcal{M}_v = \{A_2, B, A_3\}$ and $\mathcal{M}_w = \{A_3, B, A_1\}$. Let $\mathcal{W}$ be the set $\{A_1, A_2, A_3\}$ of max cliques that form a fork triangle around B. Let us consider $|\mathcal{M}_z \cap \mathcal{W}|$. If $|\mathcal{M}_z \cap \mathcal{W}| \leq 1$, then $\mathcal{M}_z$ is a separating set of at least one of the paths $T_G[\mathcal{M}_u]$, $T_G[\mathcal{M}_v]$ or $T_G[\mathcal{M}_w]$ as shown in Figure 6a and 6b, and we have a contradiction to Corollary A.1. Clearly, we cannot have $|\mathcal{M}_z \cap \mathcal{W}| = 3$, since $T_G[\mathcal{M}_z]$ must be a path. It remains to consider $|\mathcal{M}_z \cap \mathcal{W}| = 2$, which is illustrated in Figure 6c. In this case, $T_G[\mathcal{M}_z]$ forks with one of the paths $T_G[\mathcal{M}_u]$, $T_G[\mathcal{M}_v]$ or $T_G[\mathcal{M}_w]$ in B, and must be of length 3 according to Lemma A.2. Obviously, B is in the middle of the path $T_G[\mathcal{M}_z]$. ◀

▶ **Lemma A.5.** *If max clique $B \in \mathcal{M}$ has a fork triangle, then the degree of B in T_G is 3.*

Proof. Let $B \in \mathcal{M}$ have a fork triangle. Thus, there exists vertices $u, v, w \in V$ and distinct neighbor max cliques A_1, A_2, A_3 of B such that $\mathcal{M}_u = \{A_1, B, A_2\}$, $\mathcal{M}_v = \{A_2, B, A_3\}$ and $\mathcal{M}_w = \{A_3, B, A_1\}$. Let us assume B is of degree at least 4. Let C be a neighbor of B in T_G that is distinct from A_1, A_2 and A_3. According to Lemma 3.4 there must be a vertex $z \in V$ such that $B, C \in \mathcal{M}_z$ (for an illustration see Figure 7). By Lemma A.4, we have $|\mathcal{M}_z| = 3$. W.l.o.g., let A_2 and A_3 be not contained in $\mathcal{M}_z$. Then $T_G[\mathcal{M}_v \setminus \mathcal{M}_z]$ is not connected, and we obtain a contradiction to Corollary A.1. ◀

▶ **Corollary A.6.** *If a max clique $B \in \mathcal{M}$ has a fork triangle, then B is a fork clique.*

Proof. Let B be a max clique that has a fork triangle. Then the degree of B is 3 by Lemma A.5. As B has a fork triangle, there exists a vertex $v \in B$ with $\mathcal{M}_v = \{B, A, A'\}$ for all neighbor max cliques A, A' of B with $A \neq A'$. Further, it follows from Lemma A.4

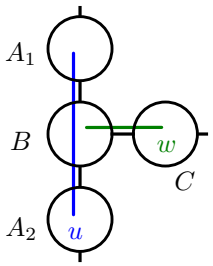


Figure 8 Illustration for the proof of Lemma 3.6.

that for every $v \in B$ there exist two neighbor max cliques A, A' of B with $A \neq A'$ such that $\mathcal{M}_v = \{B, A, A'\}$. ◀

▶ **Lemma 3.6** (restated). *Let $B \in \mathcal{M}$. If the degree of B in clique tree T_G is at least 3, then B is a star clique or a fork clique.*

Proof. Let B be a max clique of degree at least 3. Suppose B is not a star clique. Then there exists a vertex $u \in B$ and two neighbor max cliques A_1, A_2 of B in T_G that also contain vertex u. Let C be a neighbor of B with $C \neq A_1$ and $C \neq A_2$. Since $\{B, C\}$ is an edge of T_G, there must be a vertex $w \in V$ such that $B, C \in \mathcal{M}_w$ according to Lemma 3.4 (see Figure 8 for an illustration). By Corollary A.1, the graph $T_G[\mathcal{M}_u \setminus \mathcal{M}_w]$ must be connected. Thus, we have $A_1 \in \mathcal{M}_w$ or $A_2 \in \mathcal{M}_w$. Hence, $T_G[\mathcal{M}_u]$ and $T_G[\mathcal{M}_w]$ fork in B, and Lemma A.3 implies that B has a fork triangle. It follows from Corollary A.6 that B is a fork clique. ◀