Definability Equals Recognizability for k-Outerplanar Graphs and l-Chordal Partial k-Trees[*]

Hans L. Bodlaender[†1], Pinar Heggernes[2], Lars Jaffke[‡3], and Jan Arne Telle[2]

[1]Utrecht University, The Netherlands
Eindhoven University of Technology, The Netherlands
[2]University of Bergen, Norway
[3]CWI Amsterdam, The Netherlands

Abstract

One of the most famous algorithmic meta-theorems states that every graph property which can be defined in counting monadic second order logic (CMSOL) can be checked in linear time on graphs of bounded treewidth, which is known as Courcelle's Theorem [12]. These algorithms are constructed as finite state tree automata and hence every CMSOL-definable graph property is recognizable. Courcelle also conjectured that the converse holds, i.e. every recognizable graph property is definable in CMSOL for graphs of bounded treewidth. In this paper we prove two special cases of this conjecture, first for the class of k-outerplanar graphs, which are known to have treewidth at most $3k - 1$ [6], and for graphs of bounded treewidth without chordless cycles of length at least some constant ℓ.

We furthermore show that for a proof of Courcelle's Conjecture it is sufficient to show that all members of a graph class admit constant width tree decompositions whose bags and edges can be identified

[*]This paper is based on extended abstracts appearing in the proceedings of IPEC 2015 [23] and EUROCOMB 2015 [7].

[†]The research of this author was partially funded by the Networks programme, funded by the Dutch Ministry of Education, Culture and Science through the Netherlands Organisation for Scientific Research.

[‡]The research was done while this author was a student at Utrecht University.

with MSOL-predicates. For graph classes that admit MSOL-definable constant width tree decompositions that have bounded degree or allow for a linear ordering of all nodes with the same parent we even give a stronger result: In that case, the counting predicates of CMSOL are not needed.

1 Introduction

A seminal result from 1990 by Courcelle states that for every graph property P that can be formulated in a language called counting monadic second order logic (CMSOL), and each fixed k, there is a linear time algorithm that decides P for a graph given a tree decomposition of width at most k [12] (while similar results were discovered by Arnborg et al. [2] and Borie et al. [10]). Counting monadic second order logic generalizes monadic second order logic (MSOL) with a collection of predicates testing the size of sets modulo constants. Courcelle showed that this makes the logic strictly more powerful [12], which can be seen in the following example.

Example 1.1 ([12]). Let P_{even} denote the property that a graph has a vertex set of even size. Then, P_{even} is trivially definable in CMSOL, but it is not in MSOL.

The algorithms constructed in Courcelle's proof have the shape of a finite state tree automaton and hence we can say that CMSOL-definable graph properties are recognizable (or, equivalently, regular or finite-state). Courcelle's Theorem generalizes one direction of a classic result in automata theory by Büchi, which states that a language is recognizable, if and only if it is MSOL-definable [11]. Courcelle conjectured in 1990 that the other direction of Büchi's result can also be generalized for graphs of bounded treewidth in CMSOL, i.e. that each recognizable graph property is CMSOL-definable.

This conjecture is still regarded to be open. Its claimed resolution by Lapoire [27] is not considered to be valid by several experts. In the course of time proofs were given for the classes of trees and forests [12], partial 2-trees [13], partial 3-trees and k-connected partial k-trees [25]. A sketch of a proof for graphs of pathwidth at most k appeared at ICALP 1997 [24].

We add to this list the class of k-outerplanar graphs and graphs of bounded treewidth without chordless cycles of length at least ℓ.

In our proofs, we use another classic result rooted in automata theory, the Myhill-Nerode Theory, which was discovered independently by Myhill [28] in 1957 and Nerode [29] in 1958. It states that a language L of words over an alphabet is recognizable if and only if there exists an equivalence relation $\sim_L$, characterizing L, with a finite number of equivalence classes (i.e. $\sim_L$ has *finite index*). Abrahamson and Fellows [1] noted that the Myhill-Nerode Theorem can be generalized to graphs of bounded treewidth (see also [21, Theorem 12.7.2]): Each graph property P is recognizable if and only if there exists an equivalence relation $\sim_P$ of finite index, characterizing[1] P, defined over bounded treewidth terminal graphs with a bounded number of terminal vertices. This result was recently generalized to hypergraphs [4].

The rest of the paper is organized as follows. In Section 2, we give the basic notions used at various places throughout the text and in Section 3 we present our Myhill-Nerode type proof framework of Courcelle's Conjecture. What is then left to show is how to construct tree decompositions, which can be encoded by MSOL-predicates for a particular graph class. We describe these constructions for k-outerplanar graphs in Section 4 and for ℓ-chordal partial k-trees in Section 5. We give concluding remarks in Section 6.

2 Preliminaries

In this section we define the basic concepts used throughout this paper. We begin by giving some notational conventions.

Throughout the text, $k \in \mathbb{N}$ denotes a constant. Let $a, b \in \mathbb{N}$ with $a < b$. Then, $\mathbb{N}_{|a,b}$ denotes the set $\{a, a+1, \ldots, b\}$ and for $a \in \mathbb{N}$, we let $\mathbb{N}_{|a} = \mathbb{N}_{|1,a}$.

Let X be a set. Then $\mathcal{P}(X)$ denotes the powerset of X, $\mathcal{P}_c(X)$ the set of all subsets of X of size c (and $\mathcal{P}_{\leq c}(X)$ the set of all subsets up to size c) and by $\mathcal{P}_c^{\mathcal{M}}(X)$ we denote the set of all multisets over X up to size c (and define $\mathcal{P}_{\leq c}^{\mathcal{M}}(X)$ accordingly).

2.1 Graphs and Tree Decompositions

The reader is assumed to be familiar with the basic notions of graph theory, and is referred to [20] for an overview of the necessary background.

A graph $G = (V, E)$ with vertex set V and edge set E is always assumed to be undirected, connected and simple (unless stated otherwise). Let H be a graph. We denote the *subgraph* relation by $G \sqsubseteq H$, and the *proper subgraph* relation by $G \sqsubset H$. For a set $W \subseteq V$, $G[W]$ denotes the *induced subgraph* over $W \subseteq V$ in G, i.e.

[1]See Section 3 for a formal definition of what the term 'characterize' means in this context.

$G[W] = (W, E \cap (W \times W))$. As a notational convention, we denote the edge set of an induced subgraph $G[W]$ by $E_G[W]$.

We call a set $C \subset V$ a *cut* of G, if $G[V \setminus C]$ is disconnected. An *ℓ-cut* of G is a cut of size ℓ. A set $S \subseteq V$ is said to be *incident* to an ℓ-cut C, if $C \subset S$. We call a graph *ℓ-connected*, if it does not contain a cut of size at most $\ell - 1$. We now turn to the notion of tree decompositions and some related concepts.

Definition 2.1 (Tree Decomposition, Treewidth, Partial k-Tree)**.** A *tree decomposition* of a graph $G = (V, E)$ is a pair (T, X) of a tree $T = (N, F)$ and an indexed family of vertex sets $(X_t)_{t \in N}$ (called *bags*), such that the following properties hold.

(i) Each vertex $v \in V$ is contained in at least one bag.

(ii) For each edge $e \in E$ there exists a bag containing both endpoints.

(iii) For each vertex $v \in V$, the bags in the tree decomposition that contain v form a subtree of T.

The *width* of a tree decomposition is the size of the largest bag minus 1 and the *treewidth* of a graph (sometimes denoted by $tw(G)$) is the minimum width of all its tree decompositions. We sometimes refer to a graph of treewidth at most k as a *partial k-tree.*[2]

To avoid confusion, in the following we will refer to elements of N as *nodes* and elements of V as *vertices.* Sometimes, to shorten the notation, we might not differ between the terms *node* and *bag* in a tree decomposition.

Definition 2.2 (Node Types)**.** We distinguish three types of nodes in a tree decomposition (T, X), listed below.

(i) The nodes corresponding to leaves in T are called *leaf* nodes.

(ii) If a node has exactly one child it is called an *intermediate* node.

(iii) If a node has more than one child it is called a *branch* node.

As we will typically speak of some direction between nodes in tree decompositions, such as a parent-child relation, we define the following.

Definition 2.3 (Rooted and Ordered Tree Decomposition)**.** Let $(T = (N, F), X)$ be a tree decomposition. We call (T, X) *rooted,* if there is one distinguished node $r \in N$, called the *root* of T, inducing a parent-child relation on all edges in F. If there exists a fixed ordering on all nodes sharing the same parent node, then (T, X) is called *ordered.*

[2]For a discussion of different characterizations related to the treewidth of a graph, see [6, Section 2].

We sometimes refer to an edge between two nodes $t, t' \in N$ in a rooted tree decomposition as $(t, t') \in F$ (instead of $\{t, t'\} \in F$) to emphasize that t is the parent node of t'.

We use the following notation. If P denotes a graph property (e.g. a graph contains a Hamiltonian cycle), then by '$P(G)$' we express that a graph G has property P.

2.2 Tree Automata for Graphs of Treewidth k

We briefly review the concept of tree automata and recognizability of graph properties for graphs of bounded treewidth. For an introduction to the topic we refer to [21, Chapter 12]. For the formal details of the following notions, the reader is referred to [25].

A tree automaton $\mathcal{A}$ is a finite state machine accepting as an input a rooted tree structure over an alphabet Σ as opposed to words in classical word automata. Formally, $\mathcal{A}$ is a triple $(\mathcal{Q}, \mathcal{Q}_{Acc}, f)$ of a set of states $\mathcal{Q}$, a set of accepting states $\mathcal{Q}_{Acc} \subseteq \mathcal{Q}$ and a transition function f, deriving the state of a node in the input tree $\mathcal{T}$ from the states of its children and its own symbol $s \in \Sigma$. $\mathcal{T}$ is *accepted* by $\mathcal{A}$, if the state of the root node of $\mathcal{T}$ is an element of the accepting states $\mathcal{Q}_{Acc}$ after a run of $\mathcal{A}$ with $\mathcal{T}$ as an input.

To recognize a graph property on graphs of treewidth at most k, one encodes a rooted width-k tree decompositions as a labeled tree over a special type of alphabet, in the following denoted by Σ_k (see Definition 3.5, Proposition 3.6 in [25]). We say that a tree automaton over such an alphabet *processes* width-k tree decompositions.

Definition 2.4 (Recognizable Graph Properties)**.** Let P denote a graph property. We call P *recognizable* (for graphs of treewidth k), if there exists a tree automaton $\mathcal{A}_P$ processing width-k tree decompositions, such that the following are equivalent.

(i) (T, X) is a width-k tree decomposition of a graph G with $P(G)$.

(ii) $\mathcal{A}_P$ accepts (the rooted labeled tree over Σ_k corresponding to) (T, X).

2.3 Monadic Second Order Logic of Graphs

We now define counting monadic second order logic of graphs $G = (V, E)$, using terminology from [10] and [25]. Variables in this predicate logic are either single vertices/edges or vertex/edge sets. We form predicates by joining *atomic predicates* (vertex equality $v = w$, vertex membership $v \in V$, edge membership $e \in E$ and vertex-edge incidence $\mathrm{Inc}(v, e)$) via negation $\neg$, conjunction $\wedge$, disjunction $\vee$, implication $\rightarrow$

and equivalence $\leftrightarrow$ together with existential quantification $\exists$ and universal quantification $\forall$ over variables in our domain $V \cup E$. To extend this monadic second order logic (MSOL) to *counting* monadic second order logic (CMSOL), one additionally allows the use of predicates $\text{mod}_{p,q}(S)$ for sets S, which are true, if and only if $|S| \bmod q = p$, for constants p and q (with $p < q$).

Let ϕ denote a predicate without unquantified (so-called *free*) variables constructed as explained above and G be a graph. We call ϕ a *sentence* and denote by $G \models \phi$ that ϕ yields a truth assignment when evaluated with the graph G.

Definition 2.5 (Definable Graph Properties)**.** Let P denote a graph property. We say that P is *(C)MSOL-definable*, if there exists a (C)MSOL-sentence ϕ_P such that $P(G)$ if and only if $G \models \phi_P$.

A predicate ϕ can have two types of free variables. The first one is a number of *arguments* $x_1, \dots, x_a$ and we denote our predicate by $\phi(x_1, \dots, x_a)$. Predicates with arguments are used to define relations in (C)MSOL and typically appear as sub-predicates in more complex statements defining a graph property. Secondly, a predicate can have a number of *parameters*, which can be seen as auxiliary variables to define a graph property in (C)MSOL and do not appear in the notation of a predicate.

Example 2.6. Let P denote the property that a graph has a k-coloring and $\phi_{col}(v, w)$ a predicate, which is true, if and only if a vertex v has a lower numbered color than w in a given coloring. Then ϕ_{col} has two arguments, vertices v and w, and k parameters, the k color classes. Clearly, the choice of the parameters influences the evaluation of ϕ_{col}, but in most applications of parameters for predicates, it is sufficient to show that one can guess *some* variables of the evaluation graph to define a property (or a relation).

We introduce another term of definability based on predicates that have arguments and/or parameters.

Definition 2.7 (Existential Definability)**.** Let $R(x_1, \dots, x_r)$ denote a relation with arguments $x_1, \dots, x_r$. We say that R is *(C)MSOL-definable*, if there exists a parameter-free predicate $\phi_R(x_1, \dots, x_r)$, encoding the relation R. Furthermore we call R *existentially (C)MSOL-definable*, if there exists a predicate $\phi_R(x_1, \dots, x_r)$ with parameters $x_1, \dots, x_p$, which, after substituting the parameters by fixed values of an evaluation graph, encodes the relation R. For a graph property P and an argument-free predicate ϕ_P with parameters $x_1, \dots, x_p$, we define the term existential (C)MSOL-definability analogously.[3]

[3]In the more informal parts of this paper, we might not always differ between the terms

A central concept used in this paper is an implicit representation of tree decompositions in monadic second order logic, as we cannot refer to its bags and edges as variables in MSOL directly. We have to define predicates, which encode the construction of a tree decomposition of each member of a given graph class. We require two types of predicates. The **Bag**-predicates will allow us to verify whether a vertex is contained in some bag and whether any vertex set in the graph constitutes a bag in its tree decomposition. Each bag will be associated with either a vertex or an edge in the underlying graph (its *witness*) together with some *type*, whose definition depends on the graph class under consideration. The **Parent**-predicate allows for identifying edges in the tree decomposition, i.e. for any two vertex sets S_p and S_c, this predicate will be true if and only if both S_p and S_c are bags in the tree decomposition and S_p is the bag corresponding to the parent node of S_c.

Definition 2.8 (MSOL-definable tree decomposition)**.** A tree decomposition $(T = (N, F), X)$ of a graph $G = (V, E)$ is called *existentially MSOL-definable*, if the following are existentially MSOL-definable with $\mathcal{O}(1)$ parameters.

(i) Each bag $X_p, p \in N$ in the tree decomposition is associated with either a vertex $v \in V$ or an edge $e \in E$ (called its *witness*) and can be identified by one of the following predicates (where $S \subseteq V$ and s and t are constants).

  (a) $\mathbf{Bag}_{\tau_1}(v, S), \ldots, \mathbf{Bag}_{\tau_t}(v, S)$: The vertex set S forms a bag in the tree decomposition of G, i.e. $S = X_p$ for some $p \in N$, it is of type τ_i $(1 \leq i \leq t)$ and its witness is v.

  (b) $\mathbf{Bag}_{\sigma_1}(e, S), \ldots, \mathbf{Bag}_{\sigma_s}(e, S)$: The vertex set S forms a bag in the tree decomposition of G, i.e. $S = X_p$ for some $p \in N$, it is of type σ_i $(1 \leq i \leq s)$ and its witness is e.

(ii) Each edge in F can be identified with a predicate $\mathbf{Parent}(S_p, S_c)$, where $S_p, S_c \subseteq V$: The vertex sets S_p and S_c form bags in (T, X), i.e. $S_p = X_p$ and $S_c = X_c$ for some $p, c \in N$, and p is the parent node of c in T.

Let (T, X) be ordered. (T, X) is called *ordered existentially MSOL-definable*, if (i), (ii) and the following predicate are existentially MSOL-definable with $\mathcal{O}(1)$ parameters.

(iii) $\mathrm{nb}_<(S_l, S_r)$: There are nodes $t, t_l, t_r \in N$ with $(t, t_l) \in F$, $(t, t_r) \in F$, $S_l = X_{t_l}$ and $S_r = X_{t_r}$, such that t_l is a left sibling of t_r.

'definability' and 'existential definability'.

We can now show that if we have an existentially MSOL-definable tree decomposition of width k for a graph class $\mathcal{C}$, one can write a parameter-free predicate in monadic second order logic, which encodes a width-k tree decomposition of an evaluation graph $G \in \mathcal{C}$, after replacing the parameters with fixed values of G.

Lemma 2.9. *Let (T, X) be an existentially MSOL-definable tree decomposition with parameters $x_1, \dots, x_p$. There exists a predicate ϕ with zero parameters and p arguments, which is true if and only if the predicates $\boldsymbol{Bag}_{\tau_1}, \dots, \boldsymbol{Bag}_{\tau_t}$, $\boldsymbol{Bag}_{\sigma_1}, \dots, \boldsymbol{Bag}_{\sigma_s}$ and **Parent** describe a width-k rooted tree decomposition of an evaluation graph G.*

Proof. The proof can be done analogously to the proof of Lemma 4.7 in [25]. □

A fundamental result about definable graph properties, which we use extensively throughout our proofs, states that one can define any edge orientation of partial k-trees in MSOL. For an in-depth study of MSOL-definable edge orientations on graphs, see [15].

Lemma 2.10 (cf. page 120 in [15], Lemma 4.8 in [25])**.** *Any direction over a subset of the edges of an undirected graph of treewidth at most k is existentially MSOL-definable with $k + 2$ parameters.*

The idea of the proof of Lemma 2.10 is to guess a $(k + 1)$-coloring $\gamma : V \to \mathbb{N}_{k+1}$, represented in MSOL by $k + 1$ vertex sets, and a subset F of the edges of the graph with the following interpretation. Let $e = \{v, w\} \in E$. Then, e is oriented from v to w if

(i) $\gamma(v) < \gamma(w)$ and $e \in F$ or

(ii) $\gamma(v) > \gamma(w)$ and $e \notin F$

and from w to v otherwise. Hence, given a $(k + 1)$-coloring of G, any orientation which uses predicates to indicate whether a vertex is the head or tail vertex of an incident edge (in the following denoted by head(e, v) and tail(e, v), respectively), can be encoded by guessing the corresponding edge set F.

2.4 Courcelle's Conjecture

We are now ready to state Courcelle's Conjecture formally.[4]

[4]Throughout this text, we will abbreviate Conjecture 2.11 to: 'Recognizability implies definability for...'

Conjecture 2.11 (Courcelle, 1990). Let P denote a graph property. For any fixed $k \in \mathbb{N}$, the following holds. If P is recognized by a finite state tree automaton $\mathcal{A}$ for graphs of treewidth k, then there exists a (C)MSOL-sentence Φ, such that $G \models \Phi$ if and only if $\mathcal{A}$ accepts (T, X), a width-k tree decomposition of G.

The first step in proving the above mentioned special cases of this conjecture is to show that it holds for each graph class that admits existentially MSOL-definable width-k tree decompositions, i.e. we prove the following lemma. Note that by Lemma 2.9 existential definability is sufficient to ensure that there are parameter-free predicates encoding the corresponding tree decomposition.

Lemma 2.12. *Let $\mathcal{C}$ denote a graph class which admits existentially MSOL-definable width-k tree decompositions. Let ζ denote the set of all such tree decompositions. Recognizability implies*

(i) MSOL-definability for $\mathcal{C}$, if all $(T, X) \in \zeta$ have bounded degree.

(ii) MSOL-definability for $\mathcal{C}$, if all $(T, X) \in \zeta$ are ordered.

(iii) CMSOL-definability for $\mathcal{C}$.

Note that if we can find ordered or bounded degree existentially MSOL-definable width-k tree decompositions for a graph class, we obtain an even stronger result, as we do not require the counting predicates of CMSOL.

In the following section, we will give a proof of Lemma 2.12 using the generalized Myhill-Nerode Theory for graphs of bounded treewidth.

We would like to note that one can prove Lemma 2.12 in the tree automata perspective as well, see e.g. Lemma 5.4 in [25], and one can obtain the same separation between MSOL- and CMSOL-definability, cf. the discussion on page 575 in [19].

3 A Myhill-Nerode Type Argument for Courcelle's Conjecture

In this section, we present the proof of Lemma 2.12, i.e. we show that if a graph class $\mathcal{C}$ admits existentially MSOL-definable tree decompositions for all its members, then each recognizable graph property is definable in CMSOL for $\mathcal{C}$.

To do so, we use another classic result from automata theory which has been generalized to graphs of bounded treewidth. This result states that each recognizable graph property has an equivalence relation of finite index, which we will now define formally.

Definition 3.1 (Terminal Graph)**.** A *terminal graph* $G = (V, E, X)$ is a graph with vertex set V, edge set E and an ordered terminal set $X \subseteq V$. If $|X| = t$, we call G a *t-terminal graph.*

Definition 3.2 (Gluing via $\oplus$)**.** Let $G = (V_G, E_G, X_G)$ and $H = (V_H, E_H, X_H)$ be two terminal graphs with $|X_G| = |X_H|$. The graph $G \oplus H$ is obtained by taking the disjoint union of G and H and for each i, $1 \leq i \leq |X_G|$, identifying the i-th vertex in X_G with the i-th vertex in X_H.

We drop parallel edges, if they occur.

Throughout this paper, we will only consider equivalence relations over terminal graphs, whose terminal set has bounded size and whose underlying graph has bounded treewidth. We would like to note that there are alternative definitions, for which some of the results mentioned in this section do not hold, see [1].

Definition 3.3 (Equivalence Relation over Terminal Graphs)**.** Let P denote a graph property and let all (terminal) graphs below be of treewidth k. We denote by $\sim_{P_t}$ the equivalence relation over t-terminal graphs of treewidth k, where $1 \leq t \leq k+1$, defined as follows. Let G, H and K be t-terminal graphs. Then we have:

$$G \sim_{P_t} H \Leftrightarrow \Big(\forall K : P(G \oplus K) \Leftrightarrow P(H \oplus K)\Big)$$

For terminal graphs of treewidth k with *at most* $k+1$ terminals, we furthermore define the equivalence relation $\sim_{P_{\leq k}}$. Let $1 \leq t \leq k+1$.

$$G \sim_{P_{\leq k}} H \Leftrightarrow |X_G| = |X_H| = t \wedge G \sim_{P_t} H$$

This yields notions of *equivalence classes* and *finite index* (for both $\sim_{P_t}$ and $\sim_{P_{\leq k}}$) in the ordinary way. We might drop the index of $\sim_{P_{\leq k}}$ and refer to it as $\sim_P$ or simply $\sim$, if it is clear from the context.

We illustrate Definition 3.3 with an example.

Example 3.4. Let P denote the property that a graph has a Hamiltonian cycle. Let G and H be two terminal graphs (of bounded treewidth) with terminal sets X_G and X_H, respectively (where $|X_G| = |X_H| = t$). We say that G and H are equivalent w.r.t. $\sim_{P_t}$, if for all terminal graphs K (with terminal set X_K, $|X_K| = t$), the graph $G \oplus K$ contains a Hamiltonian cycle if and only if $H \oplus K$ contains a Hamiltonian cycle. A simple case when this hols is when both G and H contain a Hamiltonian path such that their terminal sets consist of the two endpoints of the path. For an illustration see Figure 1.

We now relate the concept of a terminal graph to bags in tree decompositions.

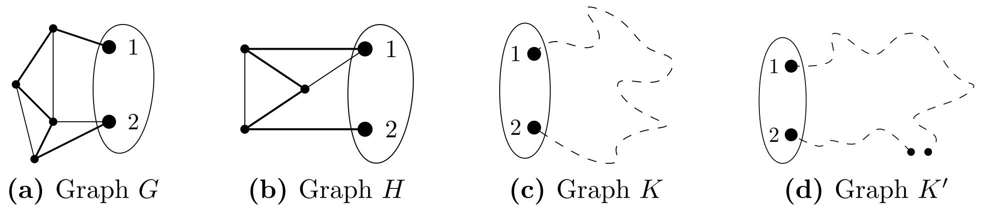


Figure 1: Two 2-terminal graphs of treewidth 2, G and H, with $G \sim_P H$, where P denotes the property 'contains a Hamiltonian cycle'. Note that since G, H and K (indicated by the dotted line) contain a Hamiltonian path between their two terminal vertices, and K' does not, we have that $G \oplus K$ and $H \oplus K$ are Hamiltonian, while $G \oplus K'$ and $H \oplus K'$ are not.

Definition 3.5 (Terminal Subgraph)**.** Let $(T = (N, F), X)$ be a rooted tree decomposition of a graph $G = (V, E)$ and $X_t, t \in N$ a bag. The *terminal subgraph G_t of t* is the terminal graph constructed in the following way. Let $N_t^+ \subseteq N$ denote the set of nodes, which are descendants of t in T (including t itself) and $V_t = \cup_{t' \in N_t^+} X_{t'}$. Then,

$$G_t = (V_t, E_G[V_t], X_t),$$

with X_t in an arbitrary but fixed ordering.

Note that in the proofs of the following results, an ordering on the terminal sets of terminal subgraphs of bags in tree decompositions has to be fixed. Since we do not require this ordering to have any specific properties, one can use an ordering based on finding a $(k+1)$-vertex coloring of the graph.

Definition 3.6 (k-Coloring Order)**.** Let $G = (V, E)$ be a graph and (T, X) a tree decomposition of G of width $(k-1)$. Suppose we have a k-coloring $\gamma : V \to \{1, \ldots, k\}$ of G such that in each bag X_t, $t \in N$, each vertex has a different color. Then, the order of the vertices in X_t for each $t \in N$, induced by the coloring γ, is called the *k-coloring order of G for (T, X)*.

One can easily prove the following.

Proposition 3.7. *Let $G = (V, E)$ be a graph and (T, X) a width-k tree decomposition of G. Then, there exists a $(k+1)$-coloring order of G for (T, X).*

Remark 3.8. If a graph class admits existentially definable tree decompositions of width at most k, then clearly the ordering stated in Proposition 3.7 is as well existentially MSOL-definable with at most $k+1$ additional parameters, the vertex sets representing the color classes.

To decide whether the underlying graphs of terminal graphs in a certain equivalence class have property P, we introduce the notion of *P-equivalence classes.* Intuitively speaking this is the analogous concept to accepting states in the automata theory perspective.

Definition 3.9 (P-Equivalence Class)**.** Let P denote a graph property and C an equivalence class of $\sim_{P_{\leq k}}$. We call C a *P-equivalence class*, if the following holds.

$$\forall G \in C : P(G \oplus (X_G, \emptyset, X_G))$$

We are now ready to state the Myhill-Nerode analog for graphs of bounded treewidth formally.

Theorem 3.10 (Myhill-Nerode Analog for Graphs of Treewidth at most k (cf. Theorem 12.7.2 in [21]))**.** *Let P denote a graph property. Then the following are equivalent for any fixed k.*

(i) P is recognizable for graphs of treewidth at most k.

(ii) $\sim_{P_{\leq k}}$ has finite index.

Remark 3.11. According to the proof of Theorem 3.10, see [21, Section 12.7], we know that the indices of the P-equivalence classes of $\sim_P$ are known, as they correspond to the accepting states in the automaton.

By Theorem 3.10, we can reformulate Conjecture 2.11 in the following way.[5]

Conjecture 3.12. Let P denote a graph property. For any fixed $k \in \mathbb{N}$, the following holds. If $\sim_{P_{\leq k}}$ has finite index, then there is a (C)MSOL-sentence Φ, such that $G \models \Phi$, if and only if the terminal subgraph of the root of a width-k tree decomposition (T, X) of G is contained in a P-equivalence class of $\sim_{P_{\leq k}}$.

In the remainder of this section we will show that we can encode the equivalence class membership of terminal subgraphs in existentially definable bounded width tree decompositions by a constant-length predicate in (C)MSOL, hence proving Lemma 2.12.

The rest of this section is organized as follows. In Section 3.1 we give algebraic proof regarding the existence of functions which describe the relations between equivalence classes of terminal subgraphs in tree decompositions. These functions we will then use in the proofs of Sections 3.2 and 3.3, where we prove Lemma 2.12(i) and 2.12(ii), and Lemma 2.12(iii), respectively (in the light of Theorem 3.10).

[5]Throughout this text, we will abbreviate Conjecture 3.12 to 'Finite index implies definability for…'.

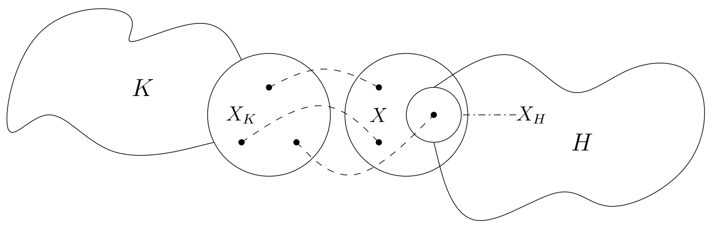


Figure 2: Terminal graphs H and K, and a terminal set X. The dashed lines indicate, which vertices are being identified in the corresponding $\oplus$-operation.

3.1 Algebraic Preliminaries

In this section we show that we can derive the equivalence class membership of terminal subgraphs from the equivalence classes their children/siblings are contained in. In particular, we prove a functional relation between them which in turn will us allow to define the equivalence class membership of terminal subgraphs in existentially MSOL-definable tree decompositions of bounded width.

Throughout the rest of this section, we use the following notation. We denote by $C_1, \ldots, C_r$ the equivalence classes of any equivalence relation $\sim_{P_{\leq k+1}}$ of finite index, defined for at most $(k+1)$-terminal graphs (and subsequently for terminal subgraphs of tree decompositions of width at most k). We refer to the indices of the equivalence classes of $\sim_P$ by $\mathbb{N}_{|r} = \{1, \ldots, r\}$.

3.1.1 Intermediate Nodes

We now show how to derive equivalence class membership of a terminal subgraph of an intermediate node from its child node.

Definition 3.13 (Gluing via $\oplus_T$)**.** Let $G = (V, E, \cdot)$ be a (terminal) graph and X an ordered set of vertices. The operation $\oplus_T$ is defined as follows.

$$G \oplus_T X = (V \cup X, E, X)$$

Note that $\oplus_T$ can either be used to make a graph a terminal graph, or to equip a terminal graph with a new terminal set.

Lemma 3.14. *Let $H = (V_H, E_H, X_H)$ and $H' = (V_{H'}, E_{H'}, X_H)$ be terminal graphs and X an ordered set of vertices. If $H \sim H'$, then $H \oplus_T X \sim H' \oplus_T X$.*

Proof. (For an illustration of the proof, see Figure 2.) Let $K = (V_K, E_K, X_K)$ be any terminal graph with $|X_K| = |X|$. Furthermore we denote by K_X the graph obtained by gluing X to K, i.e.

$K_X = K \oplus (X, \emptyset, X)$. Then,

$$K \oplus (H \oplus_T X) = H \oplus (K_X \oplus_T X_H).$$

In the left-hand side, we first extend the terminal graph H to have terminal set X and then glue the resulting graph to K. Thus the i-th vertex in X_K is identified with the i-th vertex in X, $i = 1, \dots, |X_K|$. The same vertices are being identified in the first step in computing the right-hand side, which is constructing the graph K_X (see above). We then extend this graph to have terminal set X_H and glue it to the graph H. Since in both of the constructions the same vertices get identified and both graphs have equal vertex and edge sets, we see that our claim holds. We use this argument to conclude our proof as follows.

$$\begin{aligned}\forall K:\ & P(K \oplus (H \oplus_T X)) \Leftrightarrow P(H \oplus (K_X \oplus_T X_H)) \\ \Leftrightarrow & P(H' \oplus (K_X \oplus_T X_H)) \Leftrightarrow P(K \oplus (H' \oplus_T X))\end{aligned}$$

□

Corollary 3.15. *There is a function $f_I : \mathcal{P}_{\leq k+1}(V) \times \mathbb{N}_{|r} \to \mathbb{N}_{|r}$, such that for any intermediate node $t \in N$ with child $t' \in N$ the following holds. If $G_{t'} \in C_i$, then $G_t \in C_{f_I(i, X_t)}$.*

Proof. Apply Lemma 3.14 with $H = G_t$. □

3.1.2 Ordered Branch Nodes

Next, we consider branch nodes of rooted and ordered tree decompositions.

Definition 3.16 (Partial Terminal Subgraph)**.** Let $(T = (N, F), X)$ be a rooted and ordered tree decomposition of a graph $G = (V, E)$ and $X_t, t \in N$ a branch bag with child bag $X_{t'}, t' \in N$. The *partial terminal subgraph $G_{t|t'}$ of t w.r.t. t'* is the terminal graph constructed as follows. Let $N^\ell_{t'} \subset N$ denote the left siblings of t' with terminal graphs $G_{t_\ell} = (V_{t_\ell}, E_{t_\ell}, X_{t_\ell})$ for $t_\ell \in N^\ell_{t'}$ and let $V_{t|t'} = X_t \cup \bigcup_{t_\ell \in N^\ell_{t'}} V_{t_\ell}$. Then,

$$G_{t|t'} = (V_{t|t'}, E_G[V_{t|t'}], X_t)$$

with X_t in an arbitrary but fixed ordering.

Definition 3.17 (Gluing via $\oplus_\triangleright$)**.** Let $G = (V_G, E_G, X_G)$ and $H = (V_H, E_H, X_H)$ be terminal graphs with $V_G \cap V_H = X_G \cap X_H$. The operation $\oplus_\triangleright$ is defined as:

$$G \oplus_\triangleright H = (V_G \cup V_H, E_G \cup E_H, X_G)$$

We drop parallel edges, if they occur.

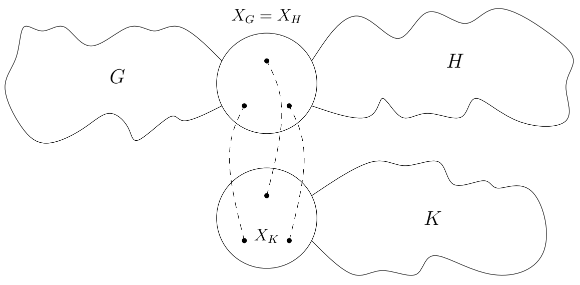


Figure 3: Terminal graphs G, H and K as in the proof of Proposition 3.20. The dashed lines indicate, which vertices are being identified in the corresponding $\oplus$-operation.

Lemma 3.18. *Let $G = (V_G, E_G, X_G)$ and $H = (V_H, E_H, X_H)$ be terminal graphs with $V_G \cap V_H = X_G \cap X_H$. Furthermore, let $G' = (V_{G'}, E_{G'}, X_G)$ and $H = (V_{H'}, E_{H'}, X_H)$ be terminal graphs with $V_{G'} \cap V_{H'} = X_G \cap X_H$, $G \sim G'$ and $H \sim H'$. Then, $(G \oplus_{\triangleright} H) \sim (G' \oplus_{\triangleright} H')$.*

Proof. First, we rewrite the operator $\oplus_{\triangleright}$ in terms of $\oplus$ and $\oplus_T$.

Proposition 3.19. *Let $G = (V_G, E_G, X_G)$ and $H = (V_H, E_H, X_H)$ be two terminal graphs with $V_G \cap V_H = X_G \cap X_H$. Then,*

$$G \oplus_{\triangleright} H = \overbrace{(G \oplus \underbrace{(H \oplus_T X_G)}_{(b)}) \oplus_T X_G}^{(a)}. \tag{1}$$

We prove the lemma in two steps. First, we observe that its statement for part (b) of Equation 1 holds by Lemma 3.14, i.e. using the notation of Proposition 3.19 we know that $(H \oplus_T X_G) \sim (H' \oplus_T X_G)$. Then, what remains to show is the following.

Proposition 3.20. *Let $G = (V_G, E_G, X_G)$ and $H = (V_H, E_H, X_H)$ be two terminal graphs with $X_G = X_H$. Let $G' = (V_{G'}, E_{G'}, X_{G'})$ and $H' = (V_{H'}, E_{H'}, X_{H'})$ be two terminal graphs with $X_{G'} = X_G$, $X_{H'} = X_H$, $G \sim G'$ and $H = H'$. Then,*[6]

$$(G \oplus H) \oplus_T X_G \sim (G' \oplus H') \oplus_T X_{G'}.$$

Proof. Let $K = (V_K, E_K, X_K)$ be any terminal graph with $|X_K| = |X_G|$. By Figure 3 we can observe the following.

$$K \oplus ((G \oplus H) \oplus_T X_G) = G \oplus ((K \oplus H) \oplus_T X_G)$$

[6]Note that the purpose of the operation '$\oplus_T X_G$' is simply to make the resulting graph a terminal graph, since (non-terminal) graphs are not members of an equivalence relation.

Regardless of the order in which we apply the operators, both graphs will have the same vertex and edge sets. As for the identifying step (using the $\oplus$-operator), one can see that for all $i = 1, \ldots, |X_K|$ we have that the i-th vertex in X_K is identified with the i-th vertex in X_G in the left-hand side of the equation and with the i-th vertex in X_H in the right-hand side. The equality still holds, since $X_G = X_H$. We use this argument (and the fact that $X_{G'} = X_G = X_H = X_{H'}$) to show the following.

$$\begin{aligned}\forall K:\ & P(K \oplus ((G \oplus H) \oplus_T X_G)) \Leftrightarrow P(G \oplus ((K \oplus H) \oplus_T X_G)) \\ & \Leftrightarrow P(G' \oplus ((K \oplus H) \oplus_T X_{G'})) \Leftrightarrow P(H \oplus ((K \oplus G') \oplus_T X_H)) \\ & \Leftrightarrow P(H' \oplus ((K \oplus G') \oplus_T X_{H'})) \Leftrightarrow P(K \oplus ((G' \oplus H') \oplus_T X_{G'}))\end{aligned}$$

Hence, our claim follows. □

This concludes the proof of Lemma 3.18. □

Corollary 3.21. *There is a function $f_B^{<} : \mathbb{N}_{|r} \times \mathbb{N}_{|r} \to \mathbb{N}_{|r}$, such that for any branch node $t \in N$ with child $t' \in N$ the following holds. Suppose $G_{t|t'} \in C_i$ and $G_{t'} \in C_j$.*

(i) If t' is the rightmost child of t, then $G_t \in C_{f_B^{<}(i,j)}$.

(ii) Otherwise, $G_{t|r(t')} \in C_{f_B^{<}(i,j)}$, where $r(t')$ denotes the leftmost right sibling of t'.

Proof. Apply Lemma 3.18 with $G = G_{t|t'}$ and $H = G_t$. □

3.1.3 Bounded Degree Branch Nodes

We now show that we can apply the results given in the previous sections to deal with the case of bounded degree branch nodes as well.

Lemma 3.22. *Let $t \in N$ be a branch bag with child nodes $t_1, \ldots, t_c$ (for some constant c). For $1 \le i \le c$, let $G_{t_i} = (V_{t_i}, E_{t_i}, X_{t_i})$ be the terminal subgraphs of $t_1, \ldots, t_c$ and $H_i = (V_{H_i}, E_{H_i}, X_{t_i})$ be terminal graphs.*

If $G_{t_1} \sim H_1, \ldots, G_{t_c} \sim H_c$, then

$$(G_{t_1} \oplus_T X_t) \oplus_\triangleright \cdots \oplus_\triangleright (G_{t_c} \oplus_T X_t) \sim (H_1 \oplus_T X_t) \oplus_\triangleright \cdots \oplus_\triangleright (H_c \oplus_T X_t).$$

Proof. Let $G_{t_1}^{X_t}$ and $G_{t-t_1}^{X_t}$ be two terminal graphs as indicated below.

$$\underbrace{(G_{t_1} \oplus_T X_t)}_{G_{t_1}^{X_t}} \oplus_\triangleright \underbrace{(G_{t_2} \oplus_T X_t) \oplus_\triangleright \cdots \oplus_\triangleright (G_{t_c} \oplus_T X_t)}_{G_{t-t_1}^{X_t}}$$

Since $G_{t_1} \sim H_1$, we know by Lemma 3.14 that $(G_{t_1} \oplus_T X_t) \sim (H_1 \oplus_T X_t)$. Let $H_1^{X_t} = (H_1 \oplus_T X_t)$, then we have that $G_{t_1}^{X_t} \sim H_1^{X_t}$. Now, by Lemma 3.18, we know that $(G_{t_1}^{X_t} \oplus_{\triangleright} G_{t-t_1}^{X_t}) \sim (H_1^{X_t} \oplus_{\triangleright} G_{t-t_1}^{X_t})$ and hence:

$$G_{t_1}^{X_t} \oplus_{\triangleright} G_{t-t_1}^{X_t} \sim (H_1 \oplus_T X_t) \oplus_{\triangleright} G_{t-t_1}^{X_t}$$

We can apply this argument repeatedly and our claim follows. Note that the child nodes $t_1, \ldots, t_c$ do not need a specific ordering, as in this context the operation $\oplus_{\triangleright}$ is commutative (all graphs, which it is applied to, have terminal set X_t). $\square$

Corollary 3.23. *For any pair of constants c and k there exists a function $f_B^{\deg c} : \mathcal{P}_{\leq k+1}(V) \times \mathcal{P}_c^{\mathcal{M}}(\mathbb{N}_{|r}) \to \mathbb{N}_{|r}$, such that the following holds. Let $t \in N$ be a branch node with c child nodes $t_1, \ldots, t_c \in N$. If $G_{t_1} \in C_{i_1}, \ldots, G_{t_c} \in C_{i_c}$, then $G_t \in C_{f_B^{\deg c}(X_t, i_1, \ldots, i_c)}$.*

3.2 Results Regarding MSOL-Definability

In this section we show that finite index implies definability in monadic second order logic (i.e. we do not require counting predicates) for all graph classes that admit either ordered or bounded degree existentially MSOL-definable width-k tree decompositions (i.e. we prove Lemma 2.12(i) and 2.12(ii)). We use the observations and corresponding functions presented in the previous section.

Lemma 3.24 (cf. Lemma 2.12(i))**.** *Let $\mathcal{C}$ be a graph class which admits MSOL-definable width-k tree decompositions of maximum degree c, for constants c and k. Then, finite index implies MSOL-definability of all members in $\mathcal{C}$.*

Proof. In the following, let $G = (V, E) \in \mathcal{C}$ and $(T = (N, F), X)$ an MSOL-definable width-k tree decomposition of G. Furthermore we denote by $C_1, \ldots, C_r$ the equivalence classes of $\sim_{P_{\leq k+1}}$ for any P, such that $\sim_{P_{\leq k}}$ has finite index. We mimic the proof of Büchi's famous result in terms of word automata [11], as shown in [32, Theorem 3.1]. That is, we define witness sets for each equivalence class in the following way. For each pair of an equivalence class C_i, $1 \leq i \leq r$ and a vertex (edge) bag type $\tau \in \{\tau_1, \ldots, \tau_t\}$ ($\sigma \in \{\sigma_1, \ldots, \sigma_s\}$), we define sets $W_{i,\tau}^V$ ($W_{i,\sigma}^E$) with the following interpretation. For $v \in V$ ($e \in E$) let $\tau(v) \in N$ ($\sigma(e) \in T$) denote the bag of type τ (σ), whose witness is v (e). Then,

$$v \in W_{i,\tau}^V \Leftrightarrow G_{\tau(v)} \in C_i \quad \text{and}$$
$$e \in W_{i,\sigma}^E \Leftrightarrow G_{\sigma(e)} \in C_i.$$

Before we go into the details of the proof, we give a high level outline of the predicates we construct. The main predicate Φ_{EQC} consists of three parts. The first part, ϕ_{Leaf} fills the witness sets of the leaf nodes in T. In the second part, ϕ_{TSG} we derive the equivalence class membership of non-leaf nodes via a case analysis on which equivalence classes their children are contained in. For this step we use the functions f_I and $f_B^{\deg c}$ defined in Corollaries 3.15 (for intermediate nodes) and 3.23 (for bounded degree branch nodes), respectively. Eventually, to verify whether G has property P we check whether any witness set corresponding to a P-equivalence class contains the vertex or edge, which is the witness of the root bag. We denote this part of the predicate by ϕ_{Root}. To summarize, we have

$$\Phi_{EQC} = (\mathcal{Q})\ \phi_{Leaf} \wedge \phi_{TSG} \wedge \phi_{Root}, \tag{2}$$

where the term $(\mathcal{Q})$ denotes the quantifications over the required set variables. One easily sees that by this construction we have that $G \models \Phi_{EQC}$ if and only if $P(G)$. We now turn to defining the details of the above mentioned predicates.

The first part is to guess the sets $W_{\cdot,\cdot}^{\cdot}$, i.e.:

$$\Phi_{EQC} = \Big((\exists W_{i,\tau}^V \subseteq V)(\exists W_{i,\sigma}^E \subset E)\Big)_{\substack{i=1,\dots,r\\ \tau\in\{\tau_1,\dots,\tau_t\}\\ \sigma\in\{\sigma_1,\dots,\sigma_s\}}} \phi_{Root} \wedge \phi_{TSG} \wedge \phi_{Leaf}$$

Remark 3.25. In the following, we assume for simplicity that all bags in (T, X) have vertex witnesses. Note that our predicates can easily be modified for tree decomposition with both vertex and edge witnesses by the obvious additions/replacements.

We now give the details for ϕ_{Leaf}. We modify the tree decomposition (T, X) to ensure that all leaf bags have size one, such that it is still existentially MSOL-definable. Suppose $t \in N$ is a leaf node with $|X_t| > 1$ of type τ^* for a vertex v. We introduce another vertex bag type τ_{Leaf}, such that for each bag X_t as described above, it contains the vertex $w \in X_t$ with lowest numbered color according to a $(k+1)$-coloring order of (T, X) (see Definition 3.6, Proposition 3.7). Let t' be this newly introduced node in N. We make v the witness of $X_{t'}$ and add the edge (t, t') to F. We introduce another edge bag type σ_{Leaf} with the same interpretation. It is easy to see that there are MSOL-predicates encoding the newly introduced bags and edges in (T, X). Since now, all leaf bags have size one, we know that there is a unique equivalence class, say C_L, which contains the terminal subgraphs of all leaf bags of (T, X). We encode ϕ_{Leaf} as follows.

$$\phi_{Leaf} = \forall v \forall S\ \Big(\text{Leaf}(S) \to \Big(\bigwedge_{\tau} \mathbf{Bag}_\tau^V(v, S) \to v \in W_{L,\tau}^V\Big)\Big)$$

Note that a predicate Leaf(S), which is true if and only if a vertex set $S \subseteq V$ is a leaf bag of an MSOL-definable tree decomposition can be easily encoded using the **Parent**-relation.

For the predicate ϕ_{Root}, we observe the following. Let $C_{P_1}, \ldots, C_{P_a}$ denote the P-equivalence classes of $\sim_{P_{\le k+1}}$ (and recall that by Remark 3.11, their indices are known). One simply checks, whether the witness v_r (e_r) of the root bag, say of type τ^* (σ^*), is contained in a set $W^V_{P_i,\tau^*}$ ($W^E_{P_i,\sigma^*}$) for $1 \le i \le a$.

$$\phi_{Root} = \forall v \forall S \Big(\mathrm{Root}(S) \to \Big(\bigwedge_{\tau} \mathbf{Bag}^V_\tau(v,S) \to \bigvee_{P_i} v \in W^V_{P_i,\tau}\Big)\Big)$$

On a high level, encoding the predicate ϕ_{TSG} requires the following. Let $t \in N$ be a node of (T, X).

(i) Check whether t is a branch or an intermediate node.

(ii) If t is an intermediate node, identify its child node t', the type and witness of $X_{t'}$. Derive the equivalence class of $G_{t'}$ by checking membership of the witness of $X_{t'}$ in one of the sets $W^{\cdot}_{\cdot,\cdot}$, say $G_{t'} \in C_{i^*}$. Then encode the fact that $G_t \in C_{f_I(X_t,i^*)}$.

(iii) If t is a branch node, identify its children $t_1, \ldots, t_d$ (where $d \le c$) and identify the equivalence classes of G_{t_i}, where $1 \le i \le d$ in the same way as in (ii). Suppose $G_{t_1} \in C_{j_1}, \ldots, G_{t_d} \in C_{j_d}$. Encode the fact that $G_t \in C_{f_B^{\deg d}(j_1,\ldots,j_d,X_t)}$.

We first encode (i), i.e. we check whether a bag is an intermediate or a branch node. Note that the predicates Int(S) and Branch$_d$(S), which are true if and only if the set S is an intermediate or degree d branch bag, respectively, can again be encoded easily.

$$\begin{aligned}\phi_{TSG} \Leftrightarrow \forall S \Big(&\mathrm{Int}(S) \to \bigwedge_{\tau,\tau'} \phi^{\tau,\tau'}_{Int}(S) \\ &\wedge \bigwedge_{d=1,\ldots,c} \mathrm{Branch}_d(S) \to \bigwedge_{\substack{\tau \\ \tau_1,\ldots,\tau_d}} \phi^{\tau,\tau_1,\ldots,\tau_d}_{Branch_d}(S)\Big)\end{aligned}$$

Note that the predicates $\phi^{\tau,\tau'}_{Int}$ and $\phi^{\tau,\tau_1,\ldots,\tau_d}_{Branch_d}$ are encoded for any combination of bag types. As mentioned above (see Remark 3.25), we restrict ourselves to vertex bag types but we would like to note that including edge bag types can be done in a straightforward way. We now encode (ii).

$$\begin{aligned}\phi^{\tau,\tau'}_{Int}(S) \Leftrightarrow \forall v \forall v' \forall S' \Big(&(\mathbf{Bag}_\tau(v,S) \wedge \mathbf{Bag}_{\tau'}(v',S') \wedge \mathbf{Parent}(S,S')) \\ &\to \bigwedge_{i=1,\ldots,r} v' \in W^V_{i,\tau} \to v \in W^V_{f_I(S,i)}\Big)\end{aligned}$$

For (iii), we define the following.

$$\begin{aligned}\phi^{\tau,\tau_1,\dots,\tau_d}_{Branch_d}(S) \Leftrightarrow \forall v \forall v_1 \cdots \forall v_d \forall S_1 \cdots \forall S_d \Big(& (\mathbf{Bag}_{\tau}(v,S) \wedge \mathbf{Bag}_{\tau_1}(v_1,S_1) \\ & \wedge \cdots \wedge \mathbf{Bag}_{\tau_d}(v_d,S_d) \wedge \mathbf{Parent}(S,S_1) \wedge \cdots \wedge \mathbf{Parent}(S,S_d)) \\ \rightarrow \bigwedge_{i_1,\dots,i_d} \Big(& (v_1 \in W^V_{i_1,\tau_1} \wedge \cdots \wedge v_d \in W^V_{i_d,\tau_d}) \rightarrow v \in W^V_{f_B^{\deg d}(i_1,\dots,i_d,S),\tau} \Big)\Big)\end{aligned}$$

This completes the proof of Lemma 3.24. $\square$

We now turn to the case of ordered MSOL-definable tree decompositions, i.e. we additionally have a predicate $\text{nb}_<(S,S')$, which is true if and only if $S = X_t$ and $S' = X_{t'}$ for some $t,t' \in N$ and t' is a right sibling of t in T.

Lemma 3.26 (cf. Lemma 2.12(ii))**.** *Let $\mathcal{C}$ denote a graph class, whose members admit existentially MSOL-definable width-k ordered tree decompositions. Then, finite index implies MSOL-definability for all members in $\mathcal{C}$.*

Proof. By the proof of Lemma 3.24, what remains to show is how to define the equivalence class membership of partial terminal subgraphs w.r.t. branch bags in a tree decomposition. We use the same notation as in its proof and when showing details of the predicates assume again in the following that all witnesses of the bags in (T,X) are vertices (see Remark 3.25). Let $t,t' \in N$ be two nodes in (T,X), such that t is a branch node and the parent of t', i.e. $(t,t') \in F$. Let X_t be of type τ with witness $v \in V$ and $X_{t'}$ of type τ' with witness $v' \in V$ (for $\tau,\tau' \in \{\tau_1,\dots,\tau_t\}$). We guess two sets $W^{V|P}_{i,\tau}$ and $W^{V|C}_{i,\tau'}$ (the parent and child set) and let $v \in W^{V|P}_{i,\tau}$ and $v' \in V^{V|C}_{i,\tau'}$ if and only if $G_{t|t'} \in C_i$. We guess edge set equivalents with the same meaning. Hence, the quantification part $(\mathcal{Q})$ of Φ_{EQC} as in Equation 2 becomes the following.

$$(\mathcal{Q}) = \Big(\exists W^V_{i,\tau} \exists W^E_{i,\sigma} \exists W^{V|P}_{i,\tau} \exists W^{V|C}_{i,\tau} \exists W^{E|P}_{i,\sigma} W^{E|C}_{i,\sigma}\Big)_{\substack{i=1,\dots,r \\ \tau \in \{\tau_1,\dots,\tau_t\} \\ \sigma \in \{\sigma_1,\dots,\sigma_s\}}}$$

Let $t \in N$ be a branch node with child node $t' \in N$. Since we determine the equivalence class membership of G_t by the equivalence classes of the partial terminal subgraphs w.r.t. the child nodes of t from left to right (according to the order $\text{nb}_<$), we have to distinguish three cases and use the functions f_I and $f_B^<$, whose existence we proved in Corollaries 3.15 and 3.21, respectively.[7]

[7]Let $t,t' \in N$ with $(t,t') \in F$. We denote by $r(t)$ the leftmost right sibling of t'.

(i) If t' is the leftmost child of t and $G_{t'} \in C_i$, we determine the equivalence class of $G_{t|r(t')}$ by $f_I(X_t, i)$.[8]

(ii) If t' is neither the leftmost nor the rightmost child node of t, $G_{t|t'} \in C_i$ and $G_{t'} \in C_j$, then we determine the equivalence class of $G_{t|r(t')}$ by $f_B^{<}(i,j)$.

(iii) If t' is the rightmost child of t, and $G_{t|t'} \in C_i$ and $G_{t'} \in C_j$, then we determine the equivalence class of G_t by $f_B^{<}(i,j)$.

The details of the predicate ϕ_{Branch} are as follows. First, we distinguish the three different cases.[9]

$$\begin{aligned}\phi_{Branch}(S) \Leftrightarrow \forall v \forall v' \forall v'' \forall S' \forall S'' \bigwedge_{\tau,\tau',\tau''} \Big(\Big(&\mathbf{Parent}(S,S') \wedge r_{S'}(S'') \\ &\wedge \mathbf{Bag}_{\tau}(v,S) \wedge \mathbf{Bag}_{\tau'}(v',S') \wedge \mathbf{Bag}_{\tau''}(v'',S'') \Big) \\ &\to \Big(\mathrm{Left}(S') \to \phi_{Branch}^{L|\tau,\tau',\tau''}(v,v',v'') \\ &\quad \wedge \mathrm{Middle}(S') \to \phi_{Branch}^{M|\tau,\tau',\tau''}(v,v',v'') \\ &\quad \wedge \mathrm{Right}(S'') \to \phi_{Branch}^{R|\tau,\tau',\tau''}(v,v',v'') \Big) \Big)\end{aligned}$$

Note that the predicates Left(S), Middle(S) and Right(S) can easily be encoded using the ordering $\mathrm{nb}_<$.

We now encode Case (i).

$$\begin{aligned}\phi_{Branch}^{L|\tau,\tau',\tau''}(v,v',v'') \Leftrightarrow \bigwedge_{i=1,\dots,r} \Big(v' \in W_{i,\tau'}^{V} \to \Big(&v \in W_{f_I(S,i),\tau}^{V|P} \\ &\wedge v'' \in W_{f_I(S,i),\tau''}^{V|C} \Big) \Big)\end{aligned}$$

We now turn to Case (ii), i.e. the child node of t is neither the leftmost nor the rightmost one, and encode the corresponding predicate as follows.

$$\begin{aligned}\phi_{Branch}^{M|\tau,\tau',\tau''}(v,v',v'') \Leftrightarrow \bigwedge_{i,j} \Big(\Big(&v \in W_{i,\tau}^{V|P} \wedge v' \in W_{i,\tau'}^{V|C} \wedge v' \in W_{j,\tau}^{V} \Big) \\ &\to \Big(v \in W_{f_B^{<}(i,j),\tau}^{V|P} \wedge v'' \in W_{f_B^{<}(i,j),\tau''}^{V|C} \Big) \Big)\end{aligned}$$

Lastly, we show how to derive the equivalence class of G_t using information about the equivalence class of the partial terminal subgraph

[8]It is easy to see that $G_{t|r(t')}$ would be equal to the terminal subgraph of t, if t' was the only child node of t.

[9]Note that here, $r_S(S')$ is a predicate which is true if and only if S' is the leftmost right sibling of S, which can be encoded using **Parent** and $\mathrm{nb}_<$ in a straightforward way.

of the rightmost child of t, which is described in Case (iii).

$$\begin{aligned}\phi^{R|\tau,\tau',\tau''}_{Branch}(v,v',v'') \Leftrightarrow \bigwedge_{i,j} \Big(\Big(& v \in W^{V|P}_{i,\tau} \wedge v'' \in W^{V|C}_{i,\tau''} \wedge v'' \in W^{V}_{j,\tau''} \Big) \\ & \rightarrow v \in W^{V}_{f^{<}_{B}(i,j),\tau} \Big)\end{aligned}$$

This completes the proof of Lemma 3.26. □

3.3 Results Regarding CMSOL-Definability

In the previous section we saw that finite index implies definability in MSOL for graph classes that admit bounded degree or ordered existentially MSOL-definable width-k tree decompositions. In our proofs, we did not have to make use of the counting predicate of CMSOL. We will now prove that finite index implies CMSOL-definability for graph classes that admit (unordered and unbounded degree) existentially MSOL-definable width-k tree decomposition. We do so by showing how to use the counting trick to encode a constant-length predicate which defines the equivalence class membership of an unordered branch node of unbounded degree.

Before we give the details of the proof, we give a high-level overview of its idea. Let $t \in N$ be a branch node in a tree decomposition $(T = (N, F), X)$ of a graph $G = (V, E)$. We group all child nodes of t according to which equivalence class their terminal subgraphs are contained in. We then show that it is sufficient to count the size of each such group modulo $r!$ to determine the equivalence class of the terminal graph formed by taking the union of all terminal graphs in one group with terminal set X_t (the *partial terminal group subgraph*, see below). Once we know the equivalence classes these terminal graphs are contained in, we again can use a function with r arguments (one for each group) to determine the equivalence class of G_t. Hence, the number of variables and subsequently the length of our predicates stays constant.

We begin by defining terminology related to groups of a branch node.

Definition 3.27 (Group of an Equivalence Class, Partial Terminal Group Subgraph)**.** Let $X_t, t \in N$ be a branch bag with an unbounded number of children.

(i) A set of nodes $N_{t|\mathcal{G}_i} \subset N$ is called the *group i w.r.t. t*, if it contains all child nodes of t, whose terminal subgraph is contained in equivalence class i, i.e.

$$t' \in N_{t|\mathcal{G}_i} \Leftrightarrow (t, t') \in F \wedge G_{t'} \in C_i.$$

(ii) The *partial terminal group subgraph of t w.r.t. C_i*, denoted by $G_{t|\mathcal{G}_i}$ is the terminal graph constructed as follows. Let $N_{t|\mathcal{G}_i} \subset N$ denote the group i w.r.t. t and for $t' \in N_{t|\mathcal{G}_i}$, denote by $G_{t'} = (V_{t'}, E_{t'}, X_{t'})$ the terminal subgraph of t'. Furthermore, let $V_{t|\mathcal{G}_i} = X_t \cup \bigcup_{t' \in N_{t|\mathcal{G}_i}} V_{t'}$. Then,

$$G_{t|\mathcal{G}_i} = (V_{t|\mathcal{G}_i}, E_G[V_{t|\mathcal{G}_i}], X_t)$$

with X_t in an arbitrary but fixed ordering.

Proposition 3.28. *There is a recursive function $f^c_\mathcal{G} : \mathcal{P}_{\leq k+1}(V) \times \mathbb{N} \to \{\epsilon\} \cup \mathbb{N}$ of depth $c \leq r!$,*[10] *such that for any $t \in N$ and group i: $G_{t|\mathcal{G}_i} \in C_{f_\mathcal{G}(X_t,i)}$.*

Proof. Let $t_1, t_2, \ldots, t_m \in N_{t|\mathcal{G}_i}$ denote the nodes contained in group i w.r.t. t. Let $i \in \mathbb{N}_m$. By Corollary 3.23, we know that there is a function $g : \mathcal{P}_{\leq k+1}(V) \times \mathbb{N}_{|r} \times \mathbb{N}_{|r} \to \mathbb{N}_{|r}$, such that for any $i, j \in \mathbb{N}_m$ we have that $((G_{t_i} \oplus_T X_t) \oplus_\triangleright (G_{t_j} \oplus_T X_t)) \in C_{g(X_t,i,j)}$. Furthermore, by Corollary 3.15 we know that there is a function to compute j, such that $(G_{t_i} \oplus_T X_t) \in C_j$. We define a recursive function $f^n_\mathcal{G}$ as follows:

$$\begin{aligned}
&f^0_\mathcal{G}(X_t, j) = \epsilon,\\
&f^1_\mathcal{G}(X_t, j) = g(X_t, j, j) \quad \text{and}\\
&f^n_\mathcal{G}(X_t, j) = g(X_t, f^{n-1}_\mathcal{G}(X_t, j), j) \quad \forall n \in \mathbb{N}, n > 1.
\end{aligned}$$

It is easy to see that the function $f^n_\mathcal{G}$ becomes periodic. There are only r elements in the image of g and the second argument, g is invoked with, is always j. The length of this period depends on $\sim_P$ and the equivalence class i but we can always bound it by $r!$ (as the product of all period lengths over r elements). Hence, we know that for all $n \in \mathbb{N}$ we have

$$f^n_\mathcal{G}(X_t, j) = f^{n \bmod r!}_\mathcal{G}(X_t, j).$$

We let $f^c_\mathcal{G}(X_t, i) = f^{|N_{t|\mathcal{G}_i}| \bmod r!}_\mathcal{G}(X_t, j)$ and our claim follows from the observation that $G_{t|\mathcal{G}_i} \in f^m_\mathcal{G}(X_t, j)$. □

Corollary 3.29. *There is a function $f_B : \mathcal{P}_{\leq k+1}(V) \times (\{\epsilon\} \cup \mathbb{N}_{|r})^r \to \mathbb{N}$, such that for any branch node $t \in N$, $G_t \in f_B(X_t, i_1, \ldots, i_r)$, if for all $a = 1, \ldots, r$, either $i_a = \epsilon$ or we know that $G_{t|\mathcal{G}_{i_a}} \in C_{i_a}$.*

Proof. We view each partial terminal group subgraph $G_{t|\mathcal{G}_{i_a}}$ as a terminal subgraph of a single child node t_{i_a} of t with $X_{t_{i_a}} = X_t$ and apply the same argument as in Corollary 3.23. □

[10]Note that the ϵ in the image of $f^c_\mathcal{G}$ is meant to represent the case when a group i is empty, see the proof of Lemma 3.30.

By Proposition 3.28 and Corollary 3.29 we know that we can derive the equivalence class membership of the terminal subgraph of an unordered branch node of unbounded degree by functions which use the counting predicate. The main step in the proof of the following lemma is to define the use of this function in CMSOL.

Lemma 3.30 (cf. Lemma 2.12(iii)). *Let $\mathcal{C}$ denote a graph class which admits existentially MSOL-definable width-k tree decompositions. Then, finite index implies CMSOL-definability for all members in $\mathcal{C}$.*

Proof. By the proofs of Lemmas 3.24 and 3.26, the only thing which is left to show is how to define the equivalence class membership of a terminal subgraph w.r.t. and unordered branch node $t \in N$ of unbounded degree in CMSOL.

To represent the equivalence class membership of a partial terminal group subgraph $G_{t|\mathcal{G}_i}$ we guess sets for each type $\tau \in \{\tau_1, \dots, \tau_t\}$ ($\sigma \in \{\sigma_1, \dots, \sigma_s\}$) and a pair of indices $i = 1, \dots, r$ and $j = \epsilon, 1, \dots, r$ (where ϵ represents the case that group i is empty) in the following way. We let $v \in W_{j,\tau}^{V|\mathcal{G}_i}$ ($e \in W_{j,\sigma}^{E|\mathcal{G}_i}$) if and only if $G_{t|\mathcal{G}_i} \in C_j$, where v (e) is the witness of the bag X_t of type τ (σ). Hence, we add the following sets to the quantification part ($\mathcal{Q}$) of Φ_{EQC} as in Equation 2, where i, j, τ and σ take the previously discussed values.

$$\left(\exists W_{j,\tau}^{V|\mathcal{G}_i} \exists W_{j,\sigma}^{E|\mathcal{G}_i}\right)_{i,j,\tau,\sigma}$$

We now define a predicate which checks whether a group i w.r.t. t has size $c \bmod r!$. Since we do not know whether the witnesses of the child bags are vertices or edges in G and since t might have more than one child, whose witness is a vertex v (but the bag is of a different type), it is not sufficient to count the number of witnesses. Instead, we define $r(t + s)$ sets which contain the witnesses of a group i and a type τ (σ).

$$\begin{aligned} V' = Y_{i,\tau}^{V}(S) \Leftrightarrow v \in V' \leftrightarrow \exists S'(&\mathbf{Parent}(S, S') \wedge \mathbf{Bag}_\tau(v, S') \\ &\wedge v \in W_{i,\tau}^{V}) \\ E' = Y_{i,\sigma}^{E}(S) \Leftrightarrow e \in E' \leftrightarrow \exists S'(&\mathbf{Parent}(S, S') \wedge \mathbf{Bag}_\sigma(e, S') \\ &\wedge e \in W_{i,\sigma}^{E}) \end{aligned}$$

Our predicate '$|N_{t|\mathcal{G}_i}| \bmod r! = c$' verifies whether the sum of the sizes of all these sets modulo $r!$ is equal to c. Let $S = X_t$ for some branch node $t \in N$ and in the following we denote $Y^{\cdot}_{\cdot,\cdot}(S)$ by $Y^{\cdot}_{\cdot,\cdot}$.

$$|N_{t|\mathcal{G}_i}| \bmod r! = c \Leftrightarrow \left(\exists Y^V_{i,\tau} \exists Y^E_{i,\sigma}\right)_{\substack{\tau \in \{\tau_1,\dots,\tau_t\} \\ \sigma \in \{\sigma_1,\dots,\sigma_s\}}} \bigvee_{\substack{(\sum_\tau c_\tau + \sum_\sigma c_\sigma) \bmod r! = c \\ \sum_\tau c_\tau + \sum_\sigma c_\sigma \le r!(r+s)}} \Big(|Y^V_{i,\tau_1}| \bmod r! = c_{\tau_1} \wedge \cdots \wedge |Y^V_{i,\tau_t}| \bmod r! = c_{\tau_t} \wedge |Y^E_{i,\sigma_1}| \bmod r! = c_{\sigma_1} \wedge \cdots \wedge |Y^E_{i,\sigma_s}| \bmod r! = c_{\sigma_s}\Big)$$

We now turn to defining the equivalence class membership of partial terminal group subgraphs using the function $f^c_{\mathcal{G}}$ (see Proposition 3.28). Again, we assume that the witness of our branch node $X_t = S$ is a vertex v (and the bag is of type τ) and note that the edge witness cases can be defined accordingly (cf. Remark 3.25).

$$\bigwedge_{c=0,\dots,r!-1} |N_{t|\mathcal{G}_i}| \bmod r! = c \rightarrow v \in W^{V|\mathcal{G}_i}_{f^c_{\mathcal{G}}(S, f_I(S,i)),\tau}$$

To derive the equivalence class of G_t, we use the function f_B, defined in Corollary 3.29. The rest is a straightforward case analysis.

$$\bigwedge_{\substack{i_1 = \epsilon,1,\dots,r \\ \cdots \\ i_r = \epsilon,1,\dots,r}} \left(\left(v \in W^{V|\mathcal{G}_1}_{i_1,\tau} \wedge \cdots \wedge v \in W^{V|\mathcal{G}_r}_{i_r,\tau}\right) \rightarrow v \in W^V_{f_B(S,i_1,\dots,i_r),\tau}\right)$$

This completes the proof of Lemma 3.30. □

By the results presented in this section we have now proved Lemma 2.12.

4 k-outerplanar Graphs

Definition 4.1 ((Planar) Embedding)**.** A drawing of a graph in the plane is called an *embedding*. If no pair of edges in this drawing crosses, then it is called *planar*.

Definition 4.2 (k-outerplanar Graph)**.** Let $G = (V, E)$ be a graph. G is called a *planar graph*, if there exists a planar embedding of G. An embedding of a graph G is 1-*outerplanar*, if it is planar, and all vertices lie on the exterior face. For $k \geq 2$, an embedding of a graph G is k-*outerplanar*, if it is planar, and when all vertices on the outer face are deleted, then one obtains a $(k-1)$-outerplanar embedding of the resulting graph. If G admits a k-outerplanar embedding, then it is called a k-*outerplanar graph*.

In this section we give the proof of our first main result, which states that every recognizable graph property for k-outerplanar graphs is definable in counting monadic second order logic. By Lemma 2.12 (page 9), what is left to show is how to construct an existentially MSOL-definable bounded width tree decomposition for each k-outerplanar graph. Hence, we prove the following.

Lemma 4.3. *(i) Each 3-connected k-outerplanar graph admits an existentially MSOL-definable width-$3k$ tree decomposition of degree at most 3 with $\mathcal{O}(1)$ parameters.*

(ii) Each k-outerplanar graph admits an existentially MSOL-definable tree decomposition of width at most $3k+3$ with $\mathcal{O}(1)$ parameters.

Note that by Lemma 4.3(i) we have an even stronger result for 3-connected k-outerplanar graphs, since by Lemma 2.12(i), we do not require the counting predicates for this graph class.

We now give basic definitions and review the well-known upper bound on the treewidth of k-outerplanar graphs being $3k-1$ [6], which we will use in the proofs throughout this section.

Definition 4.4 (Fundamental Cycle)**.** Let $G = (V, E)$ be a graph with maximal spanning forest $T = (V, F)$. Given an edge $e = \{v, w\}$, $e \in E \setminus F$, its *fundamental cycle* is the cycle which is formed by the unique path from v to w in F together with the edge e.

Proposition 4.5. *There are predicates to verify whether the fundamental cycle of an edge $e \in E \setminus F$ contains a vertex $v \in V$, denoted by FundCyc(e, v), or an edge $f \in F$, denoted by FundCyc(e, f), with one parameter, the edge set F.*

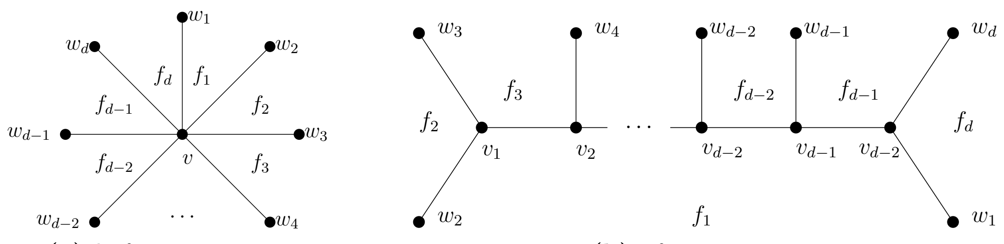
Proof. This can easily be done using the Cycle(V, E)-predicate as shown in [10, Theorem 4]. □

Definition 4.6 (Vertex and Edge Remember Number)**.** Let $G = (V, E)$ be a graph with maximal spanning forest $T = (V, F)$. The *vertex remember number* of G (with respect to T), denoted by $vr(G, T)$, is the maximum number over all vertices $v \in V$ of fundamental cycles (in G given T) that use v. Analogously, we define the *edge remember number*, denoted by $er(G, T)$.

Bodlaender showed that one can construct a bounded width tree decomposition of a graph based on a maximal spanning forest with bounded vertex and edge remember number.

Theorem 4.7 (Theorem 71 in [6])**.** *Let G be a graph and $\mathcal{T}_G$ all its maximal spanning forests. Then,*

$$tw(G) \leq \min_{T \in \mathcal{T}_G} \max\{vr(G, T), er(G, T) + 1\}.$$



(a) before expansion **(b)** after expansion

Figure 4: Expanding a vertex v, where f_1 is a layer with lowest layer number.

The idea of the proof of Theorem 4.7 is to create a bag for each vertex and edge in the spanning tree, containing the vertex itself (or the two endpoints of the edge, respectively) and one endpoint of each edge, whose fundamental cycle uses the corresponding vertex/edge. The tree structure of the decomposition is inherited by the structure of the spanning tree.

In a k-outerplanar graph G one can split the vertices of degree $d > 3$ into a path of $d - 2$ vertices of degree three without increasing the outerplanarity index of G (the so-called *vertex expansion step*, see Figure 4). In this expanded graph G' there exists a spanning tree of vertex remember number at most $3k - 1$ and edge remember number at most $2k$ [6, Lemmas 81 and 82]. Applying Theorem 4.7, this yields a tree decomposition of width at most $3k - 1$ for G' and by simple replacements one finds a tree decomposition for G of the same width. A constructive version of finding such a spanning tree was given by Katsikarelis [26]. The expansion step is the major challenge in defining a tree decomposition of a k-outerplanar graph in monadic second order logic, since we cannot use these newly created vertices as variables. We find an implicit representation of this step in Section 4.1. We show how to construct an existentially MSOL-definable tree decomposition of a 3-connected k-outerplanar graph in Section 4.2 and for the general case of k-outerplanar graphs in Section 4.3.

4.1 An Implicit Representation of the Vertex Expansion Step

As outlined before, the central step in constructing a width-$(3k - 1)$ tree decomposition of a k-outerplanar graph G is splitting the vertices of degree $d > 3$ into a path of $d - 2$ vertices of degree 3 without increasing the outerplanarity index of the graph G (see above). Since we cannot mimic this expansion step in MSOL directly, we have to find another characterization of this method, the first step of which is to partition the vertices of a k-outerplanar graph into its *stripping* *layers.*

Definition 4.8 (Stripping Layer of a k-outerplanar Graph)**.** Let G be a k-outerplanar graph. Removing the vertices on the outer face of an embedding of G is called a *stripping step.* When applied repeatedly, the set of vertices being removed in the i-th stripping step is called the i-th *stripping layer* of G, where $1 \leq i \leq k$.

Lemma 4.9. *Let $G = (V, E)$ be a k-outerplanar graph. The partition of V into the stripping layers of G is existentially MSOL-definable with k parameters.*

Proof. We first introduce another characterization of stripping layers of k-outerplanar graphs, which we can use later to define our predicates.

Proposition 4.10. *Let $G = (V, E)$ be a k-outerplanar graph. A partition $V_1, \dots, V_k$ of V represents its stripping layers, if and only if:*

(i) *$G[V_i]$ is an outerplanar graph for all $i = 1, \dots, k$.*

(ii) *For each vertex $v \in V_i$, all its adjacent vertices are contained in either V_{i-1}, V_i or V_{i+1}.*

Proof. ($\Rightarrow$) Since in each step we remove the vertices on the outer face of the graph, it is easy to see that (i) holds. For (ii), suppose not. Wlog. assume that $v \in V_i$ has a neighbor w in V_{i+2}. Before stripping step i, v lies on the outer face. Now, for w to not lie on the outer face after stripping step i, there needs to be a cycle crossing the edge $\{v, w\}$, hence the embedding of G is not planar and we have a contradiction.

($\Leftarrow$) We use induction on k. The case $k = 1$ is trivial. Now assume that $G = (V, E)$ is an ℓ-outerplanar graph with a partition of V into $V_1, \dots, V_\ell$ such that our claim holds. Let $V_{\ell+1}$ be a set of vertices with neighbors only in $V_{\ell+1}$ and V_ℓ. We denote the corresponding edge set by $E_{\ell+1}$. Clearly, placing the vertices in V_ℓ on the outer face results in an $(\ell + 1)$-outerplanar embedding of the graph $G' = (V \cup V_{\ell+1}, E \cup E_{\ell+1})$. However, some vertices in V_ℓ might still lie on the outer face. Denote this vertex set by V_ℓ^O. We let $V'_{\ell+1} = V_{\ell+1} \cup V_\ell^O$ and $V'_\ell = V_\ell \setminus V_\ell^O$. Then, the partition $V_1, \dots, V_{\ell-1}, V'_\ell, V'_{\ell+1}$ satisfies our claim and the result follows (reversing the indices of the sets in the partition). $\square$

It is well known that a graph is outerplanar if it does not contain K_4, the clique of four vertices, and $K_{2,3}$, the complete bipartite graph on two and three vertices, as a minor (cf. [20, p. 112], [31]). Borie et al. showed that the fixed minor relation is MSOL-definable [10, Theorem 4], so in our definition we use the predicates Minor_{K_4} and $\mathrm{Minor}_{K_{2,3}}$ for stating the respective minor containment, i.e.:

$$\mathrm{Outerpl}(V', E') \Leftrightarrow \neg(\mathrm{Minor}_{K_4}(V', E') \vee \mathrm{Minor}_{K_{2,3}}(V', E'))$$

We can now encode the partition of V according to Proposition 4.10.[11]

$$\begin{aligned}\exists V_1 \cdots V_k \Big(&\mathrm{Part}_V(V, V_1, \ldots, V_k) \wedge \mathrm{Outerpl}(V_1, \mathrm{IncE}(V_1)) \\ &\wedge \cdots \wedge \mathrm{Outerpl}(V_k, \mathrm{IncE}(V_k)) \\ &\wedge \forall v \Big(\bigwedge_{i=1,\ldots,k} v \in V_i \to \forall e \forall w (\mathrm{Edge}(e, v, w) \\ &\to (w \in V_{i-1} \vee w \in V_i \vee w \in V_{i+1}))\Big)\Big)\end{aligned}$$

This completes the proof of Lemma 4.9. □

Definition 4.11 (Layer Number)**.** Let $G = (V, E)$ be a planar graph. The *layer number* of a face is defined in the following way. The outer face gets layer number 0. Then, for each other face, we let the layer number be one higher than the minimum layer number of all its adjacent faces.[12]

The expansion step does not preserve facial adjacency, so in order to not increase the outerplanarity index of the graph, one makes sure that all faces are adjacent to a face with lowest layer number. (We illustrate the expansion step of a vertex in Figure 4.) Following the ideas of the proofs given in [6, Section 13], we define another type of remember number to implicitly represent the expansion step for creating a tree decomposition of a k-outerplanar graph.

Definition 4.12 (Face Remember Number)**.** Let $G = (V, E)$ be a planar graph with a given embedding $\mathcal{E}$ and $T = (V, F)$ a maximal spanning forest of G. The *face remember number* of G w.r.t. T, denoted by $fr(G, T)$ is the maximum number of fundamental cycles C of G given T, such that $bd_E(f) \cap E(C) \neq \emptyset$, where $bd_E(f)$ denotes the boundary edges of a face f, over all faces f in $\mathcal{E}$, excluding the outer face.

For an illustration of face remember numbers, see Figure 5. Now, consider the vertex v_1 in Figure 4b and let e be an edge whose fundamental cycle C_e uses v_1 in some spanning tree of G'. We observe that

[11]For an encoding of $\mathrm{Part}_V(V, V_1, \ldots, V_k)$, which is true if and only if $V_1, \ldots, V_k$ is a partition of the vertex set V, see e.g. [10, Theorem 4].

[12]Unless stated otherwise, we call to faces adjacent, if they share an incident vertex.

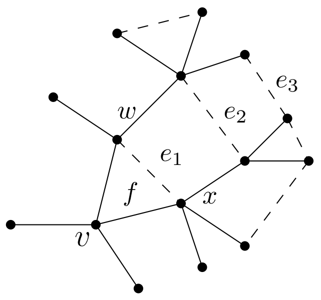


Figure 5: A spanning tree of a planar graph with some additional edges (dashed lines). The remember number of the face f, bounded by $bd(f) = \{v, w, x\}$, is 3 in this graph, since the fundamental cycles of the edges e_1, e_2 and e_3 intersect with $bd_E(f)$.

C_e intersects with one of the face boundaries of f_1, f_2 or f_3. Since v_1 is a vertex in the expanded graph, we know that in each tree decomposition based on a spanning tree of G' there will be a bag containing one endpoint of each edge, whose fundamental cycle intersects with the face boundary of f_1, f_2 or f_3. Using this observation, we can also show that one can find a tree decomposition of a planar graph, whose width is bounded by the face remember number of a maximal spanning forest, without explicitly expanding vertices.

Lemma 4.13. *Let $G = (V, E)$ be a planar graph with maximal spanning forest $T = (V, F)$. The treewidth of G is at most* $\max\{er(G, T) + 1, 3 \cdot fr(G, T)\}$.

Proof. Recall the vertex expansion step and see Figure 4 for an illustration. In the following, we will construct a tree decomposition (T, X) of the unexpanded graph G, imitating the ideas of the expansion step. That is, for each vertex $v \in V$ we create a path in (T, X) in the following way. First, we add v to each of these bags. Let f_1 denote a face with lowest layer number of all faces incident to v and let all face indices be as depicted in Figure 4a.[13] Let $C(f_i)$ denote the set, containing one endpoint of each edge $e \in E \setminus F$, whose fundamental cycle C_e intersects with the edge set of the boundary of the face f_i, i.e. $bd_E(f_i) \cap E(C_e) \neq \emptyset$. Let $\deg(v) = d$. We create bags containing the vertices in $C(f_1) \cup C(f_i) \cup C(f_{i+1})$, where $i = 2, \dots, d-1$. (For an edge e_i incident to v, f_i and f_{i+1} are its incident faces.) We make two bags adjacent, if they share two sets $C(f_i)$ and $C(f_j)$ and belong to the same vertex. Note that this way we precisely imitate the construction of bags for the artificially created vertices during the expansion step.

[13]Note that by by Proposition 4.22, this number will be either i or $i - 1$, if $v \in V_i$.

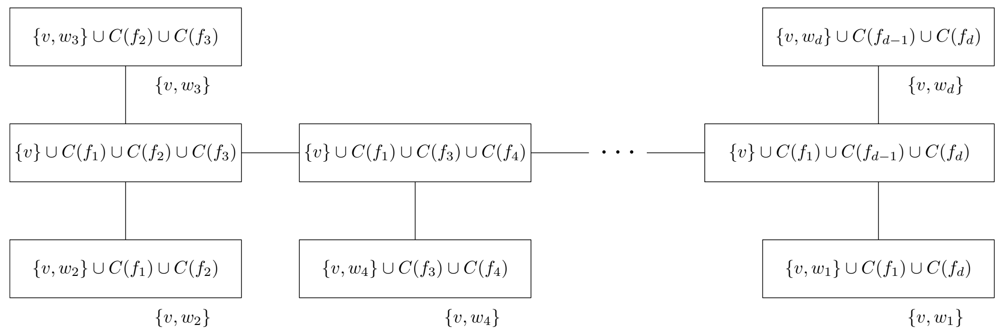

$\{v, w_3\} \cup C(f_2) \cup C(f_3)$
$\{v, w_3\}$
$\{v\} \cup C(f_1) \cup C(f_2) \cup C(f_3)$
$\{v\} \cup C(f_1) \cup C(f_3) \cup C(f_4)$
$\cdots$
$\{v, w_d\} \cup C(f_{d-1}) \cup C(f_d)$
$\{v, w_d\}$
$\{v\} \cup C(f_1) \cup C(f_{d-1}) \cup C(f_d)$
$\{v, w_2\} \cup C(f_1) \cup C(f_2)$
$\{v, w_4\} \cup C(f_3) \cup C(f_4)$
$\{v, w_1\} \cup C(f_1) \cup C(f_d)$
$\{v, w_2\}$
$\{v, w_4\}$
$\{v, w_1\}$

Figure 6: A part of a tree decomposition corresponding to a vertex, as used in the proof of Lemma 4.13 (assuming, for explanatory purposes, that all incident edges of v are contained in the maximal spanning forest of the graph).

Furthermore, for each edge $e_i \in F$, we create a bag containing both its endpoints and one endpoint of each edge $e_{fc} \in E \setminus F$, whose fundamental cycle uses e. We observe that the set $C(f_i) \cup C(f_j)$ contains precisely one vertex for each such edge e_{fc}, where f_i and f_j are the two faces incident to e_i. We then make this bag adjacent to each bag created in the step before, which corresponds to both $C(f_i)$ and $C(f_j)$ and one more set $C(f')$. For each incident vertex there will always be precisely one such bag and hence, each edge bag will have two neighbors in the tree decomposition (one for each endpoint). For an illustration of the constructed part of the tree decomposition, see Figure 6.

One can verify that this construction yields a tree decomposition of G, and since we know that by definition $|C(f)| \leq fr(G,T)$ for all faces f (except the outer face) we know that its width is bounded by $\max\{er(G,T) + 1, 3 \cdot fr(G,T)\}$. □

To apply this result to a k-outerplanar graph G, we show that we can find a maximal spanning forest of G of bounded edge and face remember number.

Lemma 4.14. *Let $G = (V,E)$ be a k-outerplanar graph. There exists a maximal spanning forest $T = (V,F)$ of G with $er(G,T) \leq 2k$ and $fr(G,T) \leq k$.*

Proof. The proof can be done analogously to the proof of Lemma 81 in [6]. □

4.2 3-Connected k-outerplanar Graphs

We now show that the construction of the tree decomposition given in the proofs of Lemmas 4.13 and 4.14 is existentially MSOL-definable for 3-connected k-outerplanar graphs. In Particular we will make use of the fact that the face boundaries of a 3-connected planar graph can be defined by a predicate in monadic second order logic. We will then define an ordering of all incident edges of a vertex to create a path in the tree decomposition as described in the proof of Lemma 4.13.

A classic result by Whitney states that every 3-connected planar graph has a unique embedding [36] (up to the choice of the outer face). Reconstructing this proof, Diestel has shown that the face boundaries of this embedding can be characterized in strictly combinatorial terms.

Proposition 4.15 (Proposition 4.2.7 in [20]). *The face boundaries in a 3-connected planar graph are precisely its non-separating induced cycles.*

We immediately have the following.

Proposition 4.16. *The face boundaries of a 3-connected planar graph are MSOL-definable.*

Proof. We use Proposition 4.15 and define a predicate, which is true if and only if a vertex set V' is the face boundary of a 3-connected planar graph in the following straightforward way.[14]

$$\mathrm{FaceBd}_3(V') \Leftrightarrow \mathrm{Cycle}(V', \mathrm{IncE}(V')) \wedge \mathrm{Conn}(V \setminus V', E \setminus \mathrm{IncE}(V'))$$

We can use this predicate to define this notion in terms of edge sets as well.

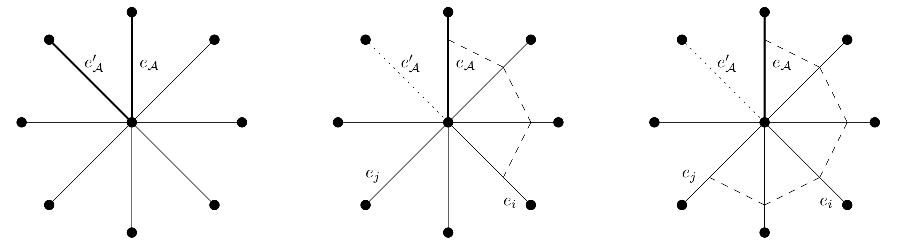
$$\mathrm{FaceBd}_3(E') \Leftrightarrow \mathrm{FaceBd}_3(\mathrm{IncV}(E'))$$

□

Using these observations, we can define predicates encoding an ordering on the incident edges of each vertex, which we will later use to induce orientations on the edges of the resulting MSOL-definable tree decomposition. We first need another definition.

Definition 4.17 (Face-Adjacency of Edges). Let $G = (V, E)$ be a planar graph and $v \in V$. We call two incident edges $e, f \in E$ of v *face-adjacent*, if there is a face-boundary containing both e and f.

Lemma 4.18. *Let $G = (V, E)$ be a 3-connected k-outerplanar graph, $v \in V$ with* $\deg(v) > 3$ *and $e_{\mathcal{A}}$ an incident edge of v, called its* anchor*. There exists an ordering $nb_<(e, f)$, which mimics a clockwise (or counter-clockwise) traversal (in the unique embedding of G) on all incident edges of v, starting at $e_{\mathcal{A}}$, which is existentially MSOL-definable with two parameters $e_{\mathcal{A}}$ and $e'_{\mathcal{A}}$.*



(a) A vertex v with the anchor edge $e_{\mathcal{A}}$ and edge $e'_{\mathcal{A}}$.

(b) The path from $e_{\mathcal{A}}$ to e_i, according to face adjacency.

(c) The path from $e_{\mathcal{A}}$ to e_j, according to face adjacency.

Figure 7: A vertex v with two edges e_i and e_j, such that $\text{nb}_<(e_i, e_j)$ as described in the proof of Lemma 4.18, defining a clockwise ordering on the incident edges of v. Note that paths in the other direction starting at $e_{\mathcal{A}}$ do not exists, since $e'_{\mathcal{A}}$ cannot be included in such a path.

Proof. We first observe an important property of 2-connected planar graphs, which we will use to define the ordering later in the proof.

Proposition 4.19. *Let $G = (V, E)$ be a 2-connected planar graph and $v \in V$. Then, all faces incident to v are pairwise different.*

Proof. Suppose not. Then $\{v\}$ is a separator of G. □

Let $e'_{\mathcal{A}}$ be another incident edge of v, which is also face-adjacent to $e_{\mathcal{A}}$. (Note that there are exactly two such edges in G, the choice of which decides whether the ordering is clockwise or counter-clockwise.) For any pair of incident edges of v, e_i and e_j, we let $\text{nb}_<(e_i, e_j)$, if and only if we can find sets of edges E_i and E_j with the following properties. Let $\text{Inc}(v)$ denote the set of incident edges of v.

(i) For $\ell = i, j$, the set E_ℓ consists of the edge e_ℓ, $e_{\mathcal{A}}$ and a subset of $\text{Inc}(v) \setminus \{e'_{\mathcal{A}}\}$ and contains precisely all pairs of face-adjacent edges that, according to face-adjacency, form a path from $e_{\mathcal{A}}$ to e_ℓ.

(ii) $E_i \subset E_j$.

For an illustration of the meaning of these edge sets see Figure 7. We now turn to defining this ordering in MSOL. By Proposition 4.19, we know that all faces adjacent to v are pairwise different and hence, we can use Proposition 4.16 to define paths in terms of face-adjacency in the unique embedding of G between two incident edges of v. We

[14]Note that the predicate $\text{Conn}(V', E')$ which is true if and only if $G' = (V', E')$ is connected can easily be encoded, see e.g. [10, Theorem 4].

encode our predicates as follows. For face-adjacency of two edges, we use Proposition 4.16.

$$\begin{aligned}\mathrm{Adj}_F(e,f) \Leftrightarrow\ & \exists v(\mathrm{Inc}(v,e) \wedge \mathrm{Inc}(v,f)) \\ & \wedge (\exists E' \subseteq E)(\mathrm{FaceBd}_3(E') \wedge e \in E' \wedge f \in E')\end{aligned}$$

Next, we define a predicate to check whether a given edge set E' is a face-adjacency path between to incident edges of a vertex. Intuitively speaking, it encodes the following.

(i) $e \in E'$ and $f \in E'$.

(ii) E' is a subset of the incident edges of v and $e'_{\mathcal{A}} \notin E'$ (see above).

(iii) For each $e' \in E'$, one of the following holds.

 (a) $e' = e$ or $e' = f$ and e' has precisely one neighbor in E'.

 (b) Otherwise, e' has precisely two neighbors in E'.

$$\begin{aligned}\mathrm{Path}_F(E',e,f) \Leftrightarrow\ & (\exists E'' \subseteq (\mathrm{IncE}(v) \setminus e'_{\mathcal{A}}))(E' = E'' \cup \{e,f\}) \\ & \wedge e_1 \in E' \leftrightarrow \Big(\Big((e_1 = e \vee e_1 = f) \wedge (\exists e_2 \in E')(\mathrm{Adj}_F(e_1,e_2) \\ & \qquad \wedge (\forall e_3 \in E')((\neg e_2 = e_3) \rightarrow \neg \mathrm{Adj}_F(e_1,e_3)))\Big) \\ & \qquad \vee \Big(\neg\ (e_1 = e \vee e_1 = f) \wedge (\exists e_2 \in E')(\exists e_3 \in E') \\ & \qquad \Big(\mathrm{Adj}_F(e_1,e_2) \wedge \mathrm{Adj}_F(e_1,e_3) \wedge (\forall e_4 \in E') \\ & \qquad ((\neg(e_4 = e_2 \vee e_4 = e_3)) \rightarrow \neg \mathrm{Adj}_F(e_1,e_4))\Big)\Big)\Big)\end{aligned}$$

We are now ready to define the ordering $\mathrm{nb}_<(e,f)$.

$$\mathrm{nb}_<(e,f) \Leftrightarrow \exists E_e \exists E_f(\mathrm{Path}_F(E_e,e_{\mathcal{A}},e) \wedge \mathrm{Path}_F(E_f,e_{\mathcal{A}},f) \wedge E_e \subset E_f)$$

This completes the proof of Lemma 4.18. □

Note that one can lead an alternative proof of Lemma 4.18, using the notion of *rotation systems*, introduced in [18]. Furthermore one can see that the relation $\mathrm{nb}_<(e,f)$ is existentially MSOL-definable for a graph G (as opposed to a single vertex, as stated in the Lemma) by replacing the parameters in the formulation of Lemma 4.18 with the corresponding edge set equivalents.

Defining the Tree Decomposition

Lemma 4.20 (cf. Lemma 4.3(i)). *Let $G = (V,E)$ be a 3-connected k-outerplanar graph. G admits an existentially MSOL-definable tree decomposition of width at most $3k$ and maximum degree 3 with $4k+5$ parameters.*

Proof. We mimic the construction given in the proof of Lemma 4.13 and use the same notation. We first prove the definability of the spanning tree, upon which the construction of our tree decomposition is based.

Proposition 4.21. *Let $G = (V, E)$ be a 3-connected k-outerplanar graph. There exists a spanning tree $T = (V, F)$ of G with $er \leq 2k$ and $fr(G, T) \leq k$, which is existentially MSOL-definable with one parameter, the edge set F of T.*

Proof. By Lemma 4.14 we know that such a spanning tree T exists. We encode two predicates, which given the edge set F check whether $er(G, T) \leq \mu$ and $fr(G, T) \leq \lambda$ for any pair of constants μ, λ.[15]

$$er(V, E, F) \leq \mu \Leftrightarrow (\forall f \in F)(\forall e_1 \in E \setminus F) \cdots (\forall e_{\mu+1} \in E \setminus F)$$
$$\Big(\Big(\bigwedge_{i=1,\ldots,\mu+1} \text{FundCyc}(e_i, f)\Big) \to \bigvee_{1 \leq i,j \leq \mu+1} e_i = e_j\Big)$$

Similarly, we define $fr(G, T) \leq \lambda$. This predicate checks for each combination of a vertex and an incident face boundary FB, whether the number of fundamental cycles that intersect FB is bounded by λ.

$$fr(V, E, F) \leq \lambda \Leftrightarrow \forall v (\forall E_{FB} \subseteq E)\Big((\text{FaceBd}_3(E_{FB}) \to (\forall e_1 \in E \setminus F)$$
$$\cdots (\forall e_{\lambda+1} \in E \setminus F))\Big(\Big(\bigwedge_{1 \leq i \leq \lambda+1} (\exists E_C \subseteq E)(\text{FundCyc}(e_i, E_C)$$
$$\wedge \ \text{Inc}(v, E_C) \wedge \neg(E_C \cap E_{FB} = \emptyset))\Big) \to \bigvee_{1 \leq i < j \leq \lambda+1} e_i = e_j\Big)\Big)$$

Let $\mu = 2k$ and $\lambda = k$ and our claim follows. □

We direct the spanning tree T of Proposition 4.21 as shown in Lemma 2.10 (see the discussion after its statement) to be a rooted tree, using a $3k$-coloring Γ_G of G. Note that two colors would already suffice, but we will later use these color sets to impose an (arbitrary) orientation on the edges in $E \setminus F$ as well.

We now choose the set of anchor and co-anchor edges $E_{\mathcal{A}}$ and $E'_{\mathcal{A}}$, respectively, to fix an ordering on the incident edges of a vertex as shown in Lemma 4.18. For a vertex v, let e_{ℓ_1} and e_{ℓ_2} denote the edges bounding a face f_ℓ with lowest layer number. (If there is more than one face with lowest layer number, we choose the one whose boundary has a shortest face-adjacency path from the unique incoming edge in

[15]Note that the we assume the existence of the predicate FundCyc(e, f), which is true if and only if the fundamental cycle of $e \in E \setminus F$ uses the edge $f \in F$, see Proposition 4.5.

the spanning tree T.) We then add e_{ℓ_1} to $E_{\mathcal{A}}$ and e_{ℓ_2} to $E'_{\mathcal{A}}$. Hence, we have that $\mathrm{nb}_<(e_{\ell_1}, e)$, for all incident edges e of v. We now prove some auxiliary results to show that the above observations are MSOL-definable.

Proposition 4.22. *Let $G = (V, E)$ be a k-outerplanar graph with stripping layers $V_1, \dots, V_k$.*

(i) Let f denote a face with layer number i. Then, the face boundary of f contains a vertex v with $v \in V_i$.

(ii) Let $v \in V_i$. Each face f incident to v has either layer number i or $i-1$. Furthermore, f has layer number $i-1$, if the boundary of f contains a vertex w with $w \in V_{i-1}$.

(iii) The layer numbers of the faces of a 3-connected k-outerplanar graph are MSOL-definable.

(iv) Let $v \in V_i$ and G 3-connected. There is an MSOL-predicate identifying a unique face incident to v with lowest layer number.

Proof. (i) and (ii). We observe that removing all vertices on the outer face makes a face of layer number i become a face of layer number $i-1$ and both claims follow.

(iii). We use (i) and encode a predicate $\mathrm{Layer}_i(E')$, which is true if and only if the edge set E' is the edge set of a face boundary with layer number i.

$$\mathrm{Layer}_i(E') \Leftrightarrow \mathrm{FaceBd}_3(E') \wedge \exists v(\mathrm{Inc}(v, E') \wedge v \in V_i)$$

(iv). According to the discussion given before the statement of the proposition and (ii), we have to analyze two cases: In the first case, there exists an incident face with layer number $i-1$ and in the second case there is no such face. In either of the two, we have to make sure that the face being identified has minimum distance from the unique incoming edge (to v) in the spanning tree according to face adjacency paths (see the proof of Lemma 4.18).[16] Let e^* denote this edge.

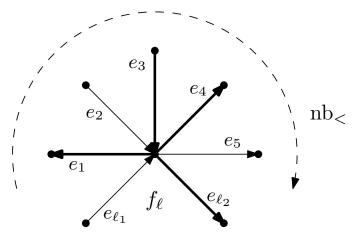
$$E' = E_{f_\ell}(v) \;\Leftrightarrow (\exists e' \in E')\Big(\mathrm{Inc}(v, e') \wedge \bigwedge_{i=1,\dots,k} v \in V_i$$

$$\to \Big(\Big(\mathrm{Layer}_{i-1}(E') \wedge \neg(\exists e'' \exists E''(\mathrm{Inc}(v, e'') \wedge e'' \in E'' \tag{3}$$

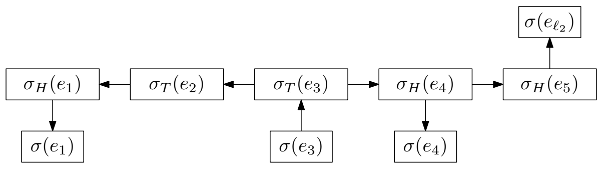
$$\wedge\, \mathrm{Path}_F(e^*, e'') \subset \mathrm{Path}_F(e^*, e')))\Big) \tag{4}$$

$$\vee \Big(\mathrm{Layer}_i(E') \wedge \neg(\exists E''(\mathrm{Inc}(v, E'') \wedge \mathrm{Layer}_{i-1}(E'') \wedge (*)))\Big)\Big)\Big) \tag{5}$$

[16]Note that the predicate $\mathrm{Path}_F(e, e')$, which identifies the set of a face adjacency path between two incident edges of v can be encoded in the same way as in the proof of Lemma 4.18. We pick one face-adjacent edge e' of e^* and remove it from the set of possible paths to fix a direction on all paths starting at e^*.



(a) A vertex with incident (directed) edges. Bold edges are in the spanning tree.



(b) The corresponding part of the tree decomposition, where the edge-orientation describes the **Parent**-relation.

Figure 8: A component of a definable tree decomposition as described in the proof of Lemma 4.20, corresponding to a vertex v with a clockwise ordering on its edges, anchored at e_{ℓ_1}, where f_ℓ is a face with lowest layer number of all incident faces of v.

We replace $(*)$ in line 5 by a repetition of lines 3 and 4, where instead of 'Layer$_{i-1}$' in line 3, we write 'Layer$_i$'. $\square$

Proposition 4.22 enables us to encode the anchor and co-anchor set $E_{\mathcal{A}}$ and $E'_{\mathcal{A}}$, respectively, to define $\mathrm{nb}_<(e, e')$ as discussed above.

We define three types of bag predicates, all associated with edges. The first type, σ, contains the endpoints of an edge $e \in F$ in the spanning tree of G and one endpoint of each edge, whose fundamental cycle uses e. A predicate $\mathbf{Bag}_\sigma(e, S)$ can easily be encoded using the methods given in the proof of Proposition 4.21.

We fix an arbitrary orientation on all edges in $E \setminus F$ using the coloring Γ_G together with the empty edge set (see Lemma 2.10). Then we define two more types of bags, σ_H and σ_T for each edge $e_i \in \mathrm{Inc}(v) \setminus \{e_{\ell_1}, e_{\ell_2}\}$ for all $v \in V$. Let $e_i = \{v, w\}$ with orientation from v to w, where f_i and f_{i-1} denote the incident faces of e_i. Then, we create a bag of type σ_H, containing v and one endpoint of each edge in $C(v, f_\ell) \cup C(v, f_{i-1}) \cup C(v, f_i)$,[17] meaning that σ_H is a type associated with the head vertex of an edge. We similarly define a type associated with the tail vertex of an edge, σ_T, which is created in the same way as σ_H, except that it contains the tail vertex instead of the head vertex of e_i (in this case: w). We observe that we can encode predicates identifying sets $C(\mathrm{head}(e), f_i)$ $(C(\mathrm{tail}(v), f_i))$, for some incident face f_i of $\mathrm{head}(e)$ $(\mathrm{tail}(e))$ applying Proposition 4.22 and hence we can encode the predicates $\mathbf{Bag}_{\sigma_H}(e, S)$ and $\mathbf{Bag}_{\sigma_T}(e, S)$ accordingly.

We now turn to defining the $\mathbf{Parent}(S_p, S_c)$-predicate. For an

[17]As opposed to the notation in the proof of Lemma 4.13, we use the vertex v as an argument for sets C as well to clarify that the faces we are considering in this step are incident faces of v.

illustration of any of the below mentioned cases, we refer the reader to Figure 8, which gives an example of a part of a tree decomposition constructed for a vertex.

First we consider bags of type σ. Let $e = \{v, w\} \in F$ such that v is its tail vertex and denote the corresponding σ-bag by X. Then, we make X the parent of the bag Y of type σ_T for the edge e. If v is the head vertex of e, then we make the bag Y of type σ_H for the edge e the parent of the bag X. As mentioned above, we do not create bags of type σ_H and σ_T for the two edges bounding the fixed face with lowest layer number f_ℓ (for details see the proof of Lemma 4.13). Let $e_\ell \in \{e_{\ell_1}, e_{\ell_2}\}$. Then, we make the bag X of type σ corresponding to e_ℓ the parent of a bag Y of type σ_T corresponding to an edge e, if e and e_ℓ bound a face together, which is adjacent (in this case, sharing an edge) to the face f_ℓ. Analogously, we make Y the parent of X, if X is of type σ_H for such an edge e_ℓ.

Furthermore, we need to add edges between bags of types σ_T and σ_H as well. Note that by now, the only bag, which already has a parent is the bag of type σ_T for the unique incoming edge $e^* \in F$ in the spanning tree of G. We use the ordering $\text{nb}_<(e, f)$ of the incident edges of a vertex v to make sure that the resulting tree decomposition is rooted. Let $\text{nb}_\prec(e, f)$ express that two incident edges e, f of v are direct neighbors in the ordering $\text{nb}_<(e, f)$. Suppose that X^* is the σ_T-bag for the edge e^* and Y is either a σ_H- or σ_T-bag for an edge f with either $\text{nb}_\prec(e^*, f)$ or $\text{nb}_\prec(f, e^*)$. In all of these cases, we make X^* the parent of Y, since X^* already has a parent bag. We observe that we have to direct the remaining edges in such a way that they point away from the bag X^*. Let $e, f \in \text{Inc}(v) \setminus \{e^*, e_{\ell_1}, e_{\ell_2}\}$ with $\text{nb}_\prec(e, f)$, X the σ_H/σ_T-bag of e and Y the σ_H/σ_T-bag of f. We have to analyze two cases. Note that always precisely one of the two holds.

(i) If $\text{nb}_<(e^*, e)$, then make X the parent of Y.

(ii) If $\text{nb}_<(f, e^*)$, then make Y the parent of X.

This completes existentially defining the tree decomposition as constructed in the proof of Lemma 4.13 in monadic second order logic for a 3-connected k-outerplanar graph.

We now count the parameters used in this proof. To find a face with lowest layer number for each vertex, we need the partition into its stripping layers as shown in Lemma 4.9. For this step we need k parameters. Furthermore we use an edge set for finding a face with lowest layer number and smallest distance from the unique incoming edge in the spanning tree for each vertex, see the proof of Proposition 4.22(iv). For directing the edges of G we use $3k$ color sets (G has treewidth at most $3k-1$ [6]) and one edge set (see Lemma 2.10). We fix edge sets for the spanning tree and the anchors $E_{\mathcal{A}}$ and co-anchors $E'_{\mathcal{A}}$ of the edge ordering $\mathrm{nb}_{<}(e, f)$. Hence, total number of parameters is $4k + 5$. □

Before we continue generalizing these results to arbitrary connected k-outerplanar graphs, we would like to note that by Lemma 2.12(i) (page 9) we have an even stronger result for 3-connected k-outerplanar graphs, i.e. we do not require the counting predicates of CMSOL for this graph class.

Theorem 4.23. *Recognizability implies MSOL-definability for 3-connected k-outerplanar graphs.*

Proof. Combine Lemmas 2.12(i) and 4.20. □

4.3 Hierarchical Graph Decompositions

A *block decomposition* of a connected graph G is a tree decompositions, whose bags contain either the endpoints of a single edge or maximal 2-connected subgraphs[18] of G (called the *blocks* of G) or a cut-vertex of G (called the *cuts*) by making a block-bag adjacent to a cut-bag $\{v\}$ if the block bag contains v (see e.g. Section 2.1 in [20]).

Analogously, Tutte showed that given a 2-connected graph (or a block of a connected graph) one can find a *3-block decomposition* into its *2-cuts* and *3-blocks*, the latter of which are either 3-connected graphs or cycles (but not necessarily subgraphs of G, see below), which can be joined in a tree structure in the same way [33, Chapter 11] [34, Section IV.3]. Courcelle showed that both of these decompositions of a graph are MSOL-definable [17] and also proved that one can find an MSOL-definable tree decomposition of width 2, if all 3-blocks of a graph are cycles [17, Corollary 4.11]. In this section, we will use these methods to prove Courcelle's Conjecture for k-outerplanar graphs by showing that the results of the previous section can be applied to define tree decompositions of 3-connected 3-blocks of a k-outerplanar graph.

As many of our proofs make explicit use of the structure of Tutte's decomposition of a 2-connected graph into its 3-connected components, we will now review this concept more closely.

Definition 4.24 (3-Block)**.** Let $G = (V, E)$ be a 2-connected graph, $\mathcal{S}$ a set of 2-cuts of G and $W \subseteq V$. A graph $H = (W, F)$ is called a *3-block*, if it can be obtained by taking the induced subgraph of W in

[18]Let $G = (V, E)$ be a graph and $W \subseteq V$. $H = G[W]$ is called a *maximal 2-connected subgraph* of G, if $G[W]$ is 2-connected and for all $W' \supset W$, $G[W']$ is not 2-connected.

G and for each incident 2-cut $S = \{x, y\} \in \mathcal{S}$, adding the edge $\{x, y\}$ to F (if not already present), plus one of the following holds.

(i) H is a cycle of at least three vertices (referred to as a *cycle 3-block*).

(ii) H is a 3-connected graph (referred to as a *3-connected 3-block*).

Definition 4.25 (Tutte Decomposition)**.** Let $G = (V, E)$ be a 2-connected graph. A tree decomposition $(T = (N, F), X)$ is called a *Tutte decomposition* of G, if the following hold. Let $\mathcal{S}$ denote a set of 2-cuts of G.

(i) For each $t \in N$, X_t is either a 2-cut $S \in \mathcal{S}$ (called the *cut bags*) or the vertex set of a 3-block (called the *block bags*).

(ii) Each edge $f \in F$ is incident to precisely one cut bag.

(iii) Each cut bag is adjacent to precisely two block bags.

(iv) Let $t \in T$ denote a cut node with vertex set X_t. Then, t is adjacent to each block node t' with $X_t \subset X_{t'}$.

Tutte has shown that additional restrictions can be formulated on the choice of the set of 2-cuts, such that the resulting decomposition is unique for each graph (for details see the above mentioned literature). In the following, when we refer to *the* Tutte decomposition of a graph, we always mean the one that is unique in this sense, which is also the one that Courcelle defined in his work [17]. Similarly, by a 3-connected 3-block (cycle 3-block, 2-cut etc.) of a graph G we mean a 3-connected 3-block in the Tutte decomposition of a block of G.

We will now state a property of Tutte decompositions, which will be useful in later proofs.

Definition 4.26 (Adhesion)**.** Let $(T = (N, F), X)$ be a tree decomposition. The *adhesion* of (T, X) is the maximum over all pairs of adjacent nodes $t, t' \in N$ of $|X_t \cap X_{t'}|$.

Proposition 4.27. *Each Tutte decomposition has adhesion 2.*

Proof. The claim follows directly from Definition 4.25 (ii) and (iv). □

For the proof of the next lemma, we need the notion of *W-paths*.

Definition 4.28 (W-Path)**.** Let $G = (V, E)$ be a graph, $W \subseteq V$ and $x, y \in V$. Then, a path $P_{xy} = (V_P, E_P)$ between x and y is called a *W-path*, if $x, y \in W$ and $V_P \cap W = \{x, y\}$, i.e. P_{xy} avoids all vertices in W except its endpoints.

Lemma 4.29. *Let $G = (V, E)$ be a 2-connected graph with Tutte decomposition $(T = (N, F), X)$. If G is k-outerplanar, then all 3-connected 3-blocks $C = (W, F)$ of (T, X) are at most k-outerplanar.*

Proof. We know that $W = X_t$ for some $t \in N$. Let $S = \{x, y\}$ denote a 2-cut of G, which is incident to W. If $\{x, y\} \in E$, we do not have to consider S any further, so in the following, if we refer to a 2-cut S, we always assume that $\{x, y\} \notin E$. Since each such pair $\{x, y\}$ appears in precisely two 3-blocks (Definition 4.25 (iii)), we know that there is always at least one W-path between x and y in G.

Proposition 4.30 ([9])**.** *Let $(T = (N, F), X)$ be a tree decomposition of adhesion 2 and $t \in T$. Let P_1 and P_2 denote two X_t-paths. If P_1 and P_2 share an internal vertex, then P_1 and P_2 have the same endpoints.*

Proof. Let $t \in N$. Then, all internal vertices of an X_t-path P are contained in a set of bags of a unique component T_t of $T[N \setminus \{t\}]$. Let $t' \in T_t$ be a neighbor of t. Then, the endpoints of P_1 and P_2 are contained in $X_t \cap X_{t'}$. Since (T, X) has adhesion 2, both paths have to have the same endpoints. □

Let $G' = G[W]$ denote the induced subgraph of G over the vertex set W. For each 2-cut S incident to W we add one W-path from G to G', connecting the two vertices in S. Since G is planar and G' is a subgraph of G, we know that G' is planar. Since (T, X) has adhesion 2 (Proposition 4.27), we know by Proposition 4.30 that there is no pair of W-paths corresponding to two different incident 2-cuts, sharing an internal vertex. Hence, we can contract each of these paths to a single edge such that the embedding of G' stays planar. Clearly, G' is isomorphic to C after contraction and the outerplanarity index of G' is less than or equal to k. □

For an illustration of the proof of Lemma 4.29, see Figure 9. The ideas in this proof can be applied to more general graph classes as well and we have the following consequence. For the proof of statement (ii), we need the following definition.

Definition 4.31 (Safe Separator [8])**.** Let $G = (V, E)$ be a connected graph with separator $S \subset V$. S is called a *safe separator*, if the treewidth of G is at most the maximum of the treewidth of all connected components W of $G[V \setminus S]$, by making S a clique in $G[W]$.

Corollary 4.32. *Let G be a 2-connected graph with Tutte decomposition (T, X).*

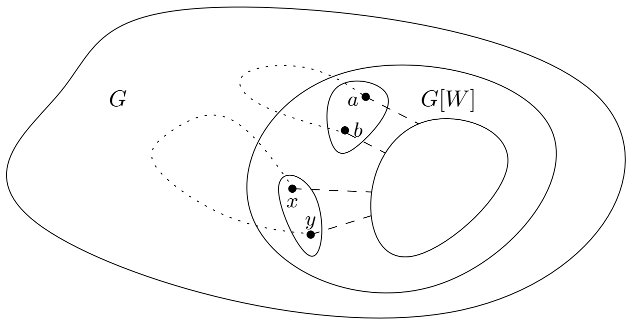


Figure 9: A 2-connected graph G with induced subgraph $G[W]$ over the vertex set of a 3-connected 3-block of G with incident 2-cuts $\{a, b\}$ and $\{x, y\}$. The dashed lines indicate that there might be several edges between a vertex and the depicted set and dotted lines represent (W-)paths in G.

(i) If G is planar, then the 3-connected 3-blocks of (T, X) are planar.

(ii) If G is a partial k-tree, then the 3-connected 3-blocks of (T, X) are partial k-trees (for $k \geq 2$).

(iii) If G is $\mathcal{H}$-minor free, then the 3-connected 3-blocks of (T, X) are $\mathcal{H}$-minor free, where $\mathcal{H}$ is a set of fixed graphs.

Proof. (i) and (iii) follow from the same argumentation (and, clearly, (i) is a consequence of (iii) by Wagner's Theorem [35]). For (ii), we observe the following. By [17, Corollary 4.12] we know that each cut bag $S = \{x, y\}$ is a safe separator of G and hence, there is a width-k tree decomposition of G which has a bag X_{xy} containing both x and y. Subsequently, adding the edge between x and y does not increase the treewidth of a 3-connected 3-block B_3. (One simply performs a short case analysis of whether X_{xy} is contained in the tree decomposition of B_3 or not.) $\square$

Replacing Edge Quantification by Vertex Quantification

As discussed above, a 3-block is in general not a subgraph of a graph G, as we add edges between the 2-cuts of the Tutte decomposition to turn the 3-blocks into cycles or 3-connected graphs. Since these absent edges cannot be used as variables in MSOL-predicates (which would make our logic non-monadic), we need to find another way to quantify over them.

In [14], Courcelle discusses several structures over which one can define monadic second order logic of graphs, which we will now review.

Definition 4.33 (cf. Definition 1.7 in [14])**.** Let $G = (V, E)$ be a graph. We associate with G two relational structures, denoted by $|G|_1 = \langle V, \text{edg}\rangle$ and $|G|_2 = \langle V \cup E, \text{edg}'\rangle$.

(i) All (C)MSOL-sentences and -predicates over $|G|_1$ only use vertices or vertex sets as variables and we have that $\text{edg}(x, y)$ is true for $x, y \in V$, if and only if there is some edge $\{x, y\} \in E$. (C)MSOL-sentences and -predicates over $|G|_2$ use both vertices and edges and vertex and edge sets as variables. Furthermore, $\text{edg}'(e, x, y)$ is true if and only if $e = \{x, y\}$ and $e \in E$.

(ii) If we can express a graph property in the structure $|G|_1$, we call it *1-definable* and if we can express a graph property in the structure $|G|_2$, we call it *2-definable.*

Clearly, the monadic second order logic we are using throughout this paper is the one represented by the structure $|G|_2$. We use both vertex and edge quantification and one simply rewrites '$\text{Inc}(v, e)$' to '$\exists w \ \text{edg}'(e, v, w)$'. Since every 1-definable property is trivially also 2-definable, we can conclude that both 1-definability and 2-definability imply (C)MSOL-definability in our sense. Some of the main results of [14] can be summarized as follows.

Theorem 4.34 ([14])**.** *1-Definability equals 2-definability for*

(i) planar graphs.

(ii) partial k-trees.

(iii) $\mathcal{H}$-minor free graphs, where $\mathcal{H}$ is a set of fixed graphs.

Hence, by Theorem 4.34 we know that we can rewrite each formula using vertex and edge quantification to one only using vertex quantification, if a graph is a member of one of these classes. We will now show that this result can be used to implicitly quantify over virtual edges of a graph, if these virtual edges can be expressed by an (existentially) MSOL-definable relation. (For a similar application of this result, see [17, Problem 4.10].)

Lemma 4.35. *Let $G = (V, E)$ be a graph which is a member of a graph class $\mathcal{C}$ as stated in Theorem 4.34 and let P denote a graph property, which is 2-definable by a predicate ϕ_P. Let $E' \subseteq V \times V$ denote a set of virtual edges, such that there is a predicate $edg_{Virt}(v, w)$, which is true if and only if $\{v, w\} \in E'$. Then, P is 1-definable for the graph $G' = (V, E \cup E')$, if G' is a member of $\mathcal{C}$.*

Proof. By Theorem 4.34, P is 1-definable for the graph G. Let $\phi_{P|1}$ denote the predicate expressing P in $|G|_1$. We replace each occurrence of '$\text{edg}(x, y)$' in $\phi_{P|1}$ by '$\text{edg}(x, y) \vee \text{edg}_{Virt}(x, y)$' and denote the resulting predicate by $\phi'_{P|1}$, which expresses the property P for the graph G' in $|G'|_1$. Since $G' \in \mathcal{C}$, one can replace quantification over sets of virtual edges (or mixed sets of edges and virtual edges) by vertex set quantification in the same way as for G. $\square$

For the specific case of k-outerplanar graphs, we can now derive the following.

Corollary 4.36. *Let $G = (V, E)$ be a (not necessarily 3-connected) k-outerplanar graph and P a graph property which is (C)MSOL-definable for 3-connected k-outerplanar graphs. Let B_3 denote a 3-block of G, including the virtual edges between all incident 2-cuts of B_3. Then, P is (C)MSOL-definable for B_3.*

Proof. By [17, Section 3] we know that there is a predicate $\phi_{\mathcal{C}_2}(x, y)$, which is true, if and only if $\{x, y\}$ is a 2-cut in the Tutte decomposition of (a block of) G. We know that B_3 (including the virtual edges) is still k-outerplanar (Lemma 4.29). Hence let $\text{edg}_{Virt}(x, y) = \phi_{\mathcal{C}_2}(x, y)$ and apply Lemma 4.35. $\square$

Note that the statements of Lemma 4.35 and Corollary 4.36 also hold for existential definability.

Defining the Tree Decomposition

By Corollary 4.36 we now know that every graph property, which can be defined for a 3-connected k-outerplanar graph, can also be defined for a 3-block of any k-outerplanar graph G (including its virtual edges).

To apply these results to any k-outerplanar graph G, we first show how to construct an existentially definable tree decomposition of G, assuming that there exist predicates existentially defining bounded width tree decompositions for the 3-connected 3-blocks of (the Tutte decompositions of the 2-blocks of) G. For an illustration of the proof idea of the following Lemma, see Figure 10, which illustrates how to fix a parent-child ordering of the hierarchical graph decomposition of G. After replacing the 3-blocks of G by their corresponding tree decompositions (taking into account the direction of the edges in the hierarchical decomposition), one can see that we have a bounded width tree decomposition of the graph G.

Remark 4.37. Note that in the proofs of the following results, one fixes a root vertex $r \in V$ of a k-outerplanar graph $G = (V, E)$, which will be used to induce a parent-relation on the bags of the hierarchical decomposition of G (see Figure 10). In a later proof, one guesses a

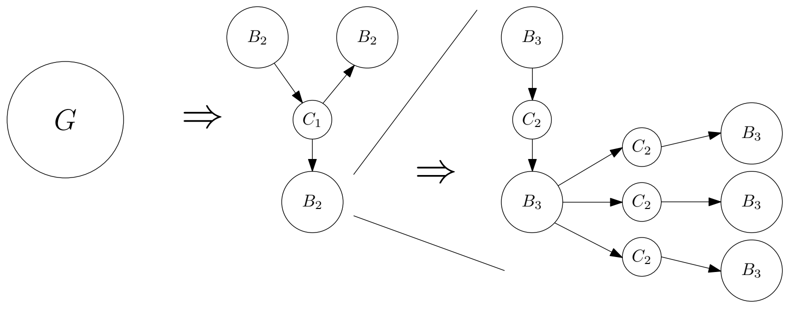


Figure 10: An example hierarchical decomposition of a graph G. A bag labeled C_1 contains a cut-vertex of G, C_2 a 2-cut of G. Bags labeled B_2 contain a 2-block (a single edge or a maximal 2-connected component). If a 2-block contains a maximal 2-connected component of G, it is decomposed further into its 2-cuts and 3-blocks, labeled by B_3, which contain either a cycle or a 3-connected 3-block.

rooted spanning tree of G, from which one derives a set of edges that contains a spanning tree of each 3-connected 3-block of G (see Lemma 4.43). The root of this spanning tree will be precisely this vertex r, hence ensuring that we have a conflict-free parent-child relation in the resulting tree decomposition of G.

Lemma 4.38. *Let $G = (V, E)$ be a k-outerplanar graph with Tutte decompositions (T, X) of its 2-connected blocks. Then, G admits an existentially MSOL-definable tree decomposition of width at most $3k + 3$ with a constant number of parameters, if there exist predicates existentially defining width-$3k$ tree decompositions for the 3-connected 3-blocks of G with $\mathcal{O}(1)$ parameters.*

Proof. Recall the decomposition of a graph into its 3-connected components described in the beginning of Section 4.3 and see Figure 10 for an illustration. We will first show how to construct a rooted tree decomposition $(\mathcal{T} = (\mathcal{N}, \mathcal{F}), \mathcal{X})$ of G width at most $3k+3$ and then prove that $(\mathcal{T}, \mathcal{X})$ is indeed MSOL-definable. Naturally, the description of the tree decomposition is already aimed at providing straightforward methods to define its predicates in MSOL.

I. Constructing the tree decomposition. We use the following notation. $\mathcal{C}_1$ denotes the set of singletons containing a cut-vertex of G and $\mathcal{C}_2$ denotes the set of 2-cuts in all Tutte decompositions of the 2-connected blocks of G. Furthermore, $\mathcal{B}_2$ denotes the set of blocks of G, $\mathcal{B}_2^E$ the set of blocks that are single edges and $\mathcal{B}_3$ denotes the set of 3-blocks of (T, X). Let $\Theta_{\mathcal{B}_3} = \{\Theta_1, \dots, \Theta_r\}$ denote the set of tree decompositions of all elements in $\mathcal{B}_3$. Then, we create a bag in $(\mathcal{T}, \mathcal{X})$ for all elements in $\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{B}_2^E$ and all bags of each Θ_i in $\Theta_{\mathcal{B}_3}$, where $1 \leq i \leq r$. Note that if a 3-block $B_3 \in \mathcal{B}_3$ is a cycle, one can find a tree decomposition of B_3 of width 2 directly. We will later study how to find an MSOL-definable tree decomposition of such a cycle in a more detailed way.

(In the following, keep Remark 4.37 in mind.) We add an edge to $\mathcal{F}$ between all pairs of adjacent bags originating from a tree decomposition Θ_i with the same orientation. To make $\mathcal{T}$ a directed tree, we add edges to $\mathcal{F}$ between the above mentioned components in the following way. First, we fix an arbitrary root $r \in V$ of the graph, which is not a member of a cut or a 2-cut of G. For each vertex $x \in V$, we let P_x denote the paths from r to x in S (and sometimes, slightly abusing notation, we might denote it as if it was one path, if the meaning of the corresponding statement is clear from the context).

Let $B_2 \in \mathcal{B}_2 \setminus \mathcal{B}_2^E$ with Tutte decomposition $(T = (N, F), X)$. We know that the bags of (T, X) either contain a 2-cut $C_2 \in \mathcal{C}_2$ or a 3-block $B_3 \in \mathcal{B}_3$ with tree decomposition $\Theta_i \in \Theta_{\mathcal{B}_3}$ for some i with $1 \leq i \leq r$. We now show which edges we need to add to $\mathcal{F}$ and how to direct them to obtain a rooted tree decomposition of B_2 of width at most $3k + 2$. We know that each edge in F is incident to one cut bag and one block bag (Definition 4.25(ii), cf. Figure 10). Let C_2, B_3 and Θ_i be as above and additionally $C_2 \subset B_3$. By Definition 4.25(iv) we know that there has to be an edge in $\mathcal{F}$ between C_2 and one bag in Θ_i, as there is an edge in F between C_2 and B_3 in the Tutte decomposition (T, X). We use the following (MSOL-definable) properties to create a rooted tree decomposition of a 2-block of G.

Proposition 4.39. *Let $C_2 = \{x, y\} \in \mathcal{C}_2$ and denote by $\mathcal{B}_3(C_2)$ its (two) neighbors in the corresponding Tutte decomposition and $r \in V$ an arbitrarily chosen but fixed root vertex, which is not a member of a 2-cut. Then, for each of the following two statements, there is* precisely one *3-block B_3 which satisfies it.*

(i) For all $v \in B_3$: $P_x \sqsubset P_v$ or $P_y \sqsubset P_v$.

(ii) There exists at least one $v \in B_3$, such that $P_v \sqsubset P_x$ or $P_v \sqsubset P_y$.

Proof. Observe that C_2 separates G into two components, say G^+ and G^-, where $r \in V(G^+)$. Then it immediately follows that (i) holds for the component $B_3^- \in \mathcal{B}_3(C_2)$ with $B_3^- \subseteq V(G^-)$. Now, let $B_3^+ \in \mathcal{B}_3(C_2) \setminus \{B_3^-\}$. Clearly, $B_3^+ \subseteq V(G^+)$. Denote by $\mathcal{C}_2(B_3^+)$ the neighbors of B_3^+. Then, there is a 2-cut $C_2' \in \mathcal{C}_2(B_3^+)$, such that (i) holds for C_2' w.r.t. B_3^+. By definition, we know that there is a vertex $z \in C_2 \nabla C_2'$ (where '∇' denotes the symmetric difference) and z is also contained in B_3^+ (again, by definition). Hence, B_3^+ satisfies (ii) (with $v = z$). $\square$

In case (i), we let $C_2 = \{x, y\}$ be the parent bag of B_3. Recall that Θ_i denotes a tree decomposition of B_3. We add both x and y to all bags in Θ_i and make C_2 the parent bag of the root of Θ_i.

In case (ii), we let B_3 be the parent of C_2. Note that while a cut bag is always the parent of precisely one block bag, a block bag can be the parent of any number of cut bags (cf. Figure 10). Hence, adding all vertices of these 2-cuts to the tree decomposition Θ_i could increase the width of Θ_i to a non-constant number. Instead, we observe the following. Since there is a (virtual or non-virtual) edge between x and y in B_3, we know that there is at least one bag containing both x and y. Denote the set of such bags by $\mathcal{X}_{xy}$. Since we have to choose precisely one bag in this set to make it a parent of C_2, we observe the following. Either, there is a bag $\mathcal{X}_t^* \in \mathcal{X}_{xy}$ for some $t \in \mathcal{N}$, whose parent does *not* contain both x and y or both x and y are contained in the root bag of Θ_i. In the latter case, we let $\mathcal{X}_t^*$ be the root of Θ_i. We then make $\mathcal{X}_t^*$ the parent of C_2.

One can verify that this yields a rooted tree decomposition of width at most $3k + 2$ for any $B_2 \in \mathcal{B}_2 \setminus \mathcal{B}_2^E$.

To finish the construction of the rooted tree decomposition $(\mathcal{T}, \mathcal{X})$, we need to show, which edges to add to $\mathcal{F}$ between bags in $\mathcal{C}_1$ and (tree decompositions of elements in) $\mathcal{B}_2$. We use the same idea as before, based on a fixed root vertex r in G. In the following let $C_1 = \{x\} \in \mathcal{C}_1$ and $B_2 \in \mathcal{B}_2$ with $C_1 \subset B_2$. Since C_1 is a separator of G, one of the following holds for all $v \in B_2$, $v \neq x$.

(i) $P_x \sqsubset P_v$.

(ii) $P_v \sqsubset P_x$.

Again, in case (i), we make C_1 the parent bag of B_2. We add x to all bags in the tree decomposition of B_2 and make $\mathcal{X}_t$ the parent of a bag $\mathcal{X}_{t'}$, where $\mathcal{X}_{t'}$ is a bag with $\mathcal{X}_{t'} = B_2$ in case $B_2 \in \mathcal{B}_2^E$ and if $B_2 \in \mathcal{B}_2 \setminus \mathcal{B}_2^E$, $X_{t'}$ is the root bag of the tree decomposition of B_2, constructed as described above. In case (ii) we make B_2 the parent bag of C_1. If $B_2 \in \mathcal{B}_2^E$, we simply let the bag $\mathcal{X}_t$ with $\mathcal{X}_t = B_2$ be the parent of the bag $\mathcal{X}_{t'}$ with $\mathcal{X}_{t'} = C_1$. If $B_2 \in \mathcal{B}_2 \setminus \mathcal{B}_2^E$, we observe the following. Since x is a cut vertex of G, no 2-cut of a block of G can contain x. Hence we know that there exists one unique 3-block $B_3^* \in \mathcal{B}_3$ with $x \in B_3^*$. We denote its tree decomposition by Θ_i^*. Again, we find a bag $\mathcal{X}_t$ in Θ_i^*, such that its parent does not contain x. If no such bag exists, we let $\mathcal{X}_t$ be the root of Θ_i^*. We again let $\mathcal{X}_{t'}$ be the bag with $\mathcal{X}_{t'} = C_1$ and make $\mathcal{X}_t$ the parent of $\mathcal{X}_{t'}$.

One can verify that now $(\mathcal{T}, \mathcal{X})$ is a rooted tree decomposition and since in the last stage we introduced at most one vertex to each bag of a tree decomposition of an element in $\mathcal{B}_2$, its width is at most $3k + 3$.

II. Definability. For defining all necessary predicates for the tree decomposition $(\mathcal{T}, \mathcal{X})$, we will refer to G as the graph after adding all virtual edges of its Tutte decomposition. We might write down predicates quantifying over virtual edges or having virtual edges as free variables, and by Corollary 4.36 we know that all these predicates can be defined only using vertex quantification as well.

By some trivial definitions, the statement of the lemma, and the results of [17] we know that the predicates listed below exist.

Proposition 4.40 (cf. [17]). *Let $G = (V, E)$ be a k-outerplanar graph, for whose 2-blocks all Tutte decompositions are known. Let $G' = (V, E \cup E')$ denote the graph obtained by adding all corresponding virtual edges E' to G and $\gamma : V \to \mathbb{N}_{|3k+1}$ a coloring of V in G'. The following predicates are MSOL-definable.*

(I) $\boldsymbol{Bag}_{\mathcal{C}_1}(v, S)$*: $S \in \mathcal{C}_1$ and $S = \{v\}$.*

(II) $\boldsymbol{Bag}_{\mathcal{B}_2^E}(e, S)$*: $S \in \mathcal{B}_2^E$ and $S = \{v, w\}$, where $e = \{v, w\}$.*

(III) $2\text{-}Conn_{\mathcal{B}_2 \setminus \mathcal{B}_2^E}(S)$*: S is the vertex set of a 2-connected 2-block of G.*

(IV) $\boldsymbol{Bag}_{\mathcal{C}_2}(v, S)$*: $S \in \mathcal{C}_2$, $v \in S$ and for $w \in S$, $v \neq w$, we have $\gamma(v) < \gamma(w)$.*

(V) $3\text{-}Conn_{\mathcal{B}_3}(S)$*: S is the vertex set of a 3-connected 3-block of G.*

(VI) $Cycle_{\mathcal{B}_3}(S)$*: S is a set of vertices forming a cycle block in a 2-block of G.*

(VII) $\boldsymbol{Bag}^{\mathcal{B}_3}_{\tau_1}(v, S), \ldots, \boldsymbol{Bag}^{\mathcal{B}_3}_{\tau_t}(v, S)$ *and* $\boldsymbol{Bag}^{\mathcal{B}_3}_{\sigma_1}(e, S), \ldots, \boldsymbol{Bag}^{\mathcal{B}_3}_{\sigma_s}(e, S)$*: The **Bag**-predicates of the tree decompositions of the 3-connected 3-blocks of G.*

(VIII) $\boldsymbol{Parent}_{\mathcal{B}_3}(S_p, S_c)$*: The **Parent**-predicate of the tree decompositions of the 3-connected 3-blocks of G.*

Proof. (I) and (II) follow from [17, Lemma 2.1], (III) from [17, Section 2] and (IV) from [17, Section 3] and Corollary 4.36. (V) is shown in [17, Corollary 4.8] and a proof of (VI) can done with the same argument. Finally, (VII) and (VIII) are part of the statement of the lemma. □

By the previous proposition, we know that there is a predicate $Block(S)$, which is true if and only if S is the a (3-)block of the hierarchical decomposition of G. We use the ideas explained above (cf. Proposition 4.39) to define a predicate $\mathbf{Parent}_{Block}(S_p, S_c)$, which checks whether a vertex set S_p is the parent bag of a vertex set S_c in the hierarchical decomposition of G.[19]

$$\begin{aligned}\mathbf{Parent}_{Block}(S_p, S_c) \Leftrightarrow\ & Block(S_p) \wedge Block(S_c) \wedge \neg(S_p = S_c) \\ & \wedge (S_p \cap S_c = S_p \vee S_p \cap S_c = S_c) \\ \wedge\ S_p \subseteq S_c \rightarrow\ & (\forall v \in S_c)(\exists x \in S_p)(\mathrm{Path}_{\rightarrow}(r, x) \subset \mathrm{Path}_{\rightarrow}(r, v)) \\ \wedge\ S_c \subseteq S_p \rightarrow\ & (\exists v \in S_p)(\exists x \in S_c)(\mathrm{Path}_{\rightarrow}(r, v) \subset \mathrm{Path}_{\rightarrow}(r, x))\end{aligned}$$

We now turn to defining tree decompositions for the cycle 3-blocks of a graph, after which we only need to show that gluing together all components of our construction explained above is MSOL-definable.

Proposition 4.41. *Let $G = (V, E)$ be a graph and $C = (W, F)$ a cycle 3-block of G (including virtual edges). There is an existentially definable predicate $\boldsymbol{Bag}_{Cyc}(e, S)$, which is true if and only if S is a bag of a tree decomposition of C associated with a (possibly virtual) edge e and an existentially definable predicate $\boldsymbol{Parent}_{Cyc}(S_p, S_c)$ encoding a parent-relation of a tree decomposition of C.*

Proof. Recall that for orienting the edges of our tree decomposition, we first fix a root vertex r in the graph G and note that by Proposition 4.40(V), W is MSOL-definable. To create a definable tree decomposition of C, we now find a root $r_C \in W$ of C. If $r \in W$, we let $r_C = r$, otherwise we know that there is one incident parent cut $C_P \in \mathcal{C}_1 \cup \mathcal{C}_2$ of C in G. C_P can be identified by checking for all 1- and 2-cuts C_C, which are incident to W, if all paths in S from r to the vertices $w \in W$ pass through (at least one of the vertices in) C_C. This can be defined in a straightforward way and one can see that there is always precisely one such cut. If $C_P = \{x\} \in \mathcal{C}_1$, then we let $r_C = x$ and if $C_P = \{x, y\} \in \mathcal{C}_2$, then we let $r_C = x$, if $\gamma(x) < \gamma(y)$ in a fixed coloring γ of C. We create a bag X for each edge $f = \{v, w\} \in F$, which is not incident to r_C and let $X = \{r_C, v, w\}$. Hence, the predicate $\mathbf{Bag}_{Cyc}(e, S)$ is also definable in a straightforward way.

We then orient the edges in F in such a way that C is a directed cycle. Note that one can find a conflict-free ordering for all cycle blocks in the graph G. (Otherwise, we might violate the cardinality constraint of MSOL.) The predicate $\mathbf{Parent}_{Cyc}(S_p, S_c)$ is true, if and only if the following hold.

(i) There are two edges $e, f \in F$, such that $\mathbf{Bag}_{Cyc}(e, S_p)$ and $\mathbf{Bag}_{Cyc}(f, S_c)$ (and e and f are contained in the same cycle).

(ii) The directed path from r_C to tail(e) in C is a strict subpath of the path from r_C to tail(f).

[19] We denote by $\mathrm{Path}_{\rightarrow}(v, w)$ the edge set of the unique directed path from v to w in the spanning tree of G. This path always exists in the cases where we use this notation.

(iii) $|S_p \cap S_c| = 2$.

Note that we only need one additional parameter, the edge set defining the edge orientation of F, since we already have a coloring for the graph G (see Proposition 4.40).

We now show how to encode the predicates. We first give a predicate to identify the root vertex of a cycle.

$$\begin{aligned} v = r_C \Leftrightarrow (r \in W \wedge v = r) \vee \Big(&(\exists C_P \subset V)(\mathbf{Parent}_{Block}(C_P, C) \\ &\wedge (\mathbf{Bag}_{\mathcal{C}_1}(v, C_P) \vee (\mathbf{Bag}_{\mathcal{C}_2}(v, C_P) \\ &\wedge (\forall w \in C_P)(\neg(v = w) \rightarrow \mathrm{col}_<(v, w))))) \Big) \end{aligned}$$

The predicates $\mathbf{Bag}_{Cyc}(e, S)$ and $\mathbf{Parent}(S_p, S_c)$ can be defined in a straightforward way according to the above discussion.

$$\begin{aligned} \mathbf{Bag}_{Cyc}(e, S) &\Leftrightarrow \neg\mathrm{Inc}(e, r_C) \wedge (v \in X \leftrightarrow (\mathrm{Inc}(v, e) \vee v = r_C)) \\ \mathbf{Parent}_{Cyc}(S_p, S_c) &\Leftrightarrow \exists e \exists f \Big(\mathbf{Bag}_{Cyc}(e, S_p) \wedge \mathbf{Bag}_{Cyc}(f, S_c) \\ &\wedge |S_P \cap S_c| = 2 \ \wedge (\exists C \subseteq V)\Big(\mathrm{Cycle}_{\mathcal{B}_3}(C) \wedge \mathrm{Inc}(e, C) \\ &\wedge \mathrm{Inc}(f, C) \wedge (\exists P_e \subseteq \mathrm{IncE}(C))(\exists P_f \subseteq \mathrm{IncE}(C)) \\ &(\mathrm{Path}_{\rightarrow}(r_C, \mathrm{tail}(e), P_e) \wedge \mathrm{Path}_{\rightarrow}(r_C, f, P_f) \wedge P_e \subset P_f)\Big)\Big) \end{aligned}$$

□

To unify the parent-relations for all tree decompositions of 3-blocks, we can write

$$\mathbf{Parent}'_{\mathcal{B}_3}(S_p, S_c) \Leftrightarrow \mathbf{Parent}_{\mathcal{B}_3}(S_p, S_c) \vee \mathbf{Parent}_{Cyc}(S_p, S_c).$$

As described above, to create the according parent-relation between blocks of the hierarchical decomposition of G, we need to add a number of vertices to some of the bags of the tree decomposition $(\mathcal{T}, \mathcal{X})$. Let $\mathbf{Bag}_*(\cdot, S)$ denote any bag type of the MSOL-definable tree decompositions for 3-connected k-outerplanar graphs. We define a predicate $\mathbf{Bag}'_*(\cdot, S)$ which includes the above mentioned modifications.

$$\begin{aligned} \mathbf{Bag}'_*(\cdot, S') \Leftrightarrow (\exists S \subseteq S')\Big(&\mathbf{Bag}_*(\cdot, S) \wedge v \in S' \setminus S \leftrightarrow \exists S_p \exists S_c \Big(S \subseteq S_c \\ &\wedge (2\text{-Conn}_{\mathcal{B}_2}(S_c) \vee 3\text{-Conn}_{\mathcal{B}_3}(S_c) \vee \mathrm{Cycle}_{\mathcal{B}_3}(S_c)) \\ &\wedge \mathbf{Parent}_{Block}(S_p, S_c) \wedge v \in S_p \Big)\Big) \end{aligned}$$

We can define the **Parent**-predicate for $(\mathcal{T}, \mathcal{X})$ using the ideas explained above to add edges between blocks and cut-bags. We denote this predicate by $\mathbf{Parent}_{\mathcal{BC}}(S_p, S_c)$. We consider the case of adding edges between the tree decomposition of a 3-block and a cut bag and note that for 2-blocks the definition works analogously. In the following, $\mathbf{Bag}'_{\mathcal{B}_3}(S)$ denotes any bag type of an MSOL-definable tree decomposition of a 3-connected k-outerplanar graph.

$$\begin{aligned}
\mathbf{Parent}_{\mathcal{CB}_3}(S_p, S_c) \Leftrightarrow\ & (\mathbf{Bag}_{\mathcal{C}_1}(S_p) \vee \mathbf{Bag}_{\mathcal{C}_2}(S_p)) \wedge \mathbf{Bag}'_{\mathcal{B}_3}(S_c) \\
& \wedge \mathrm{Root}'_{\mathcal{B}_3}(S_c) \wedge \exists S'(S_c \subseteq S' \wedge \mathbf{Parent}_{Block}(S_p, S')) \\
\mathbf{Parent}_{\mathcal{B}_3\mathcal{C}}(S_p, S_c) \Leftrightarrow\ & \mathbf{Bag}'_{\mathcal{B}_3}(S_p) \wedge (\mathbf{Bag}_{\mathcal{C}_2}(S_c) \vee \mathbf{Bag}_{\mathcal{C}_1}(S_c)) \\
& \wedge \exists S'(S_p \subseteq S' \wedge \mathbf{Parent}_{Block}(S_p, S')) \\
& \wedge \neg(\exists S'(\mathbf{Parent}'_{\mathcal{B}_3}(S', S) \wedge S' \subseteq S_c))
\end{aligned}$$

Let $\mathbf{Parent}_{\mathcal{BC}}(S_p, S_c)$ denote the predicate, which encodes all necessary cases. Then, the **Parent**-predicate of $(\mathcal{T}, \mathcal{X})$ can be defined as:

$$\mathbf{Parent}(S_p, S_c) \Leftrightarrow \mathbf{Parent}'_{\mathcal{B}_3}(S_p, S_c) \vee \mathbf{Parent}_{\mathcal{BC}}(S_p, S_c)$$

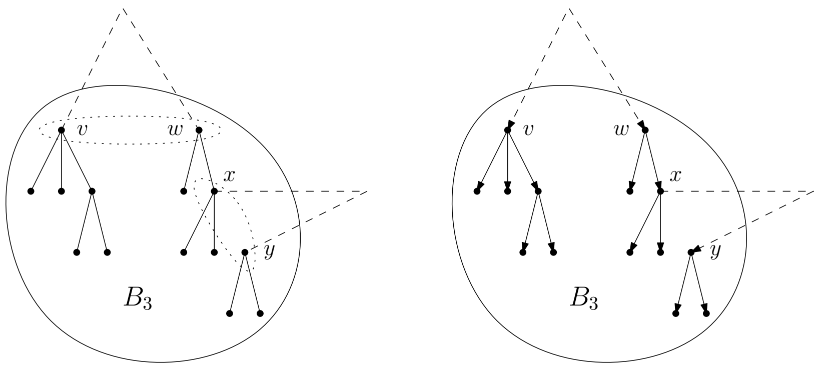
To show that the number of parameters that we need to define the above mentioned predicates is constant, we note that we only use constructions of previous results with constant numbers of parameters. (For the exact number see the corresponding result.) Note that for the cycle components one additional parameter is as well enough (see the proof of Proposition 4.41) to turn all cycles into directed cycles, since they are connected in a tree structure in the Tutte decomposition of G. Hence, fixing the direction of one cycle will always yield the possibility to direct adjacent (i.e. sharing a 2-cut) cycles in a conflict-free manner.

This completes the proof of Lemma 4.38. □

As mentioned in the previous proof, another obstacle in applying Lemma 4.20 to define a tree decomposition for G using its (definable) hierarchical graph decomposition is the cardinality constraint of MSOL. We illustrate this problem with an example.

Example 4.42 ([37]). Let G be a k-outerplanar graph, whose Tutte decompositions have $\mathcal{O}(n/\log n)$ 3-connected 3-blocks of size $\mathcal{O}(\log n)$. Let P denote a graph property, which is definable for 3-connected k-outerplanar graphs by a predicate ϕ_P. Suppose that ϕ_P uses $\mathcal{O}(1)$ parameters. When applying ϕ_P to all 3-connected 3-blocks of G, this might result in a predicate using $\mathcal{O}(n/\log n)$ parameters and hence, P not definable in this straightforward way for G.

However, for the case of encoding a tree decomposition of a k-outerplanar graph, we can avoid this problem. When defining a tree decomposition for a 3-connected k-outerplanar graph in MSOL, one first guesses a rooted spanning tree of G. To avoid guessing a non-constant number of spanning trees, we will find a set of edges $\mathcal{S}_E$,



(a) T_{B_3} without edge direction. **(b)** T_{B_3} with edge direction.

Figure 11: A forest T_{B_3} of a 3-connected 3-block of an example graph. The dashed lines indicate the paths in T between two endpoints of a incident cut of B_3. Here, $\{v, w\}$ is the root cut of B_3 and $\{x, y\}$ a child cut. Note that by Propositions 4.46 and 4.47, this small example is already somewhat general.

which contains a spanning tree with bounded edge and face remember number for each 3-connected 3-block of G. Furthermore we guess one set $\mathcal{R}_V$, containing one unique vertex for each 3-connected 3-block of G, which we will use as the root of its spanning tree. We need to make some observations about such candidate sets $\mathcal{S}_E$ and $\mathcal{R}_V$. We first prove the existence of these sets and then their MSOL-definability.

Lemma 4.43. *Let $G = (V, E)$ be a planar graph and $G = (V, E \cup E')$ the graph obtained by adding the virtual edges E' of the Tutte decompositions of the 2-connected blocks of G to G. Let $T = (V, F)$ be a spanning tree of G with $er(G, T) \leq \lambda$ and $fr(G, T) \leq \mu$. Let $B_3 = (V_{B_3}, E_{B_3}) \in \mathcal{B}_3$ be a 3-connected 3-block of G' (including virtual edges) and $T_{B_3} = T[V_{B_3}]$. One can construct from T_{B_3} a spanning tree $T^*_{B_3}$ of B_3 with $er(B_3, T^*_{B_3}) \leq \lambda$ and $fr(B_3, T^*_{B_3}) \leq \mu$ by adding edges from $E \cup E'$ to T_{B_3}.*

Proof. Clearly, $T_{B_3} = (V_{B_3}, F_{B_3})$ is a forest in B_3 and in the following we denote its tree components by $F_1 = (V_{F_1}, E_{F_1}), \ldots, F_c = (V_{F_c}, E_{F_c})$. We will now show how to connect these components to a tree. Let $\mathcal{C}'_2 \subseteq \mathcal{C}_2$ denote the set of incident 2-cuts of B_3.

Proposition 4.44. *Let T'_{B_3} denote the graph obtained by adding an edge between all 2-cuts $\{x, y\} \in \mathcal{C}'_2$ in T_{B_3} (if not already present). Then, T'_{B_3} is connected.*

Proof. Let $(T_T = (N_T, F_T), X)$ denote the Tutte decomposition containing B_3 and let $B_3 = X_t$ with $t \in N_T$. Let $v, w \in V_{B_3}$ and consider the unique path P_{vw} between v and w in T. There are two cases: (I) The path P_{vw} is completely contained in B_3 and v and w belong to the same connected component. (II) Suppose that they do not and let F_i denote the component with $v \in V_{F_i}$ and F_j the component with $w \in V_{F_j}$. Let x and y be the vertices on the path P_{vw} with $x, y \in V_{B_3}$ (and $x \neq y$), such that x has a neighbor $x' \notin V_{B_3}$ and y has a neighbor $y' \notin V_{B_3}$ (both in P_{vw}). Denote this subpath by P_{xy}. Then, P_{xy} is a V_{B_3}-path in G. Hence, there is a unique component in $T'_T = T_T[N_T \setminus \{t\}]$ containing all internal vertices of P_{xy}. Since the neighbor of t in T'_T is a cut-bag, we know that it has to contain both x and y and hence $\{x, y\} \in \mathcal{C}'_2$. □

By Proposition 4.44 we know that we can find a subset of incident 2-cuts of each 3-connected 3-block to turn T_{B_3} into a tree. We now prove that adding these edges does not increase the edge and face remember number. Consider a 2-cut $C_2 = \{x, y\} \in \mathcal{C}'_2$, such that $\{x, y\} \notin F$. Since T is a spanning tree of G, we know that there is one unique path P_{xy} between x and y in T. Let $T'_{B_3} = (V'_{B_3}, F'_{B_3})$ denote the tree obtained by adding the above described paths between the components of T_{B_3}. Then, T'_{B_3} is a spanning tree of the graph $G'_{B_3} = (V'_{B_3}, E_{B_3} \cup F'_{B_3})$ with $er(G'_{B_3}, T'_{B_3}) \leq \lambda$ and $fr(G'_{B_3}, T'_{B_3}) \leq \mu$, since $G'_{B_3} \sqsubseteq G$ and no edges, which are not members of T'_{B_3}, are introduced in G'_{B_3}. Subsequently, replacing each path P_{xy} by a single edge in T'_{B_3} does not increase the edge and face remember number as well and after these replacements, we have that $T'_{B_3} = T^*_{B_3}$ and our claim follows. For an illustration of this proof see Figure 11a. □

Lemma 4.45. *The statement of Lemma 4.43 also holds, if one replaces the term* spanning tree *by* rooted spanning tree. *Furthermore there is a set $\mathcal{R}_V \subseteq V$, which contains precisely one vertex acting as a root for a spanning tree for each 3-connected 3-block of G.*

Proof. We use the same notation as in the proof of Lemma 4.43. Since $T = (V, F)$ is a rooted spanning tree, we know that its components $F_1, \ldots, F_c$ in B_3 are rooted trees as well, see Figure 11b for an illustration. Since the direction between block and cut bags of a Tutte decomposition of a block of G are based on the root of the spanning tree T (see Remark 4.37 on page 44 and the proof of Lemma 4.38 on pages 45 ff.), we observe the following. Let $C_2 = \{x, y\} \in \mathcal{C}_2$ denote an incident 2-cut of B_3 with $\{x, y\} \notin F$. There are two cases we have to consider. Either, C_2 is the parent cut of B_3 or it is a child cut.

Proposition 4.46. *Let C_2 be a child cut of B_3. Wlog. x is a vertex in a tree F_i and y is the root of a tree F_j.*

Proof. Suppose not. We know that there is a path P_{xy} between x and y in T. If y is a non-root vertex in F_j, then we cannot direct the edges of P_{xy} in T such that every vertex has precisely one parent. Hence, T is not a directed tree and we have a contradiction. □

Proposition 4.47. *Let C_2 be the parent cut of B_3. Then, x and y are roots of two trees F_i and F_j.*

Proof. For any vertex $v \in V_{B_3}$ we know by definition (see the proof of Lemma 4.38) that for every vertex $v \in V_{B_3}$, the directed path from the root r of T to v in T is either a subpath of the directed path from r to x or from r to y. Hence, neither x nor y can have a parent in T_{B_3}. □

We can direct the additional edges using Propositions 4.46 and 4.47. In the case that C_2 is a child cut, we can always direct the edge $\{x, y\}$ from x to y (using the notation of Proposition 4.46). If C_2 is the parent cut, we know by Proposition 4.47 that we can orient $\{x, y\}$ arbitrarily. There are two cases we need to analyze to make sure we do not create a conflicting orientation of $\mathcal{S}_E$. In the first case, the edge $\{x, y\}$ has been added to $\mathcal{S}_E$ by the parent block of C_2. We then use the same orientation. In the second case, if $\{x, y\} \notin \mathcal{S}_E$, we can choose the direction arbitrarily.

We now turn to finding the set of roots $\mathcal{R}_V$. If B_3 is the root block according to the spanning tree of G with root r_G, then we add r_G to $\mathcal{R}_V$ as the root of B_3. Otherwise, we find its parent cut $C_2 = \{x, y\}$. Assume wlog. that the edge $\{x, y\}$ is directed from x to y according to the construction explained above. Then we add x to $\mathcal{R}_V$. Since each cut-bag has precisely one child block bag (Definition 4.25(ii)), we know that this vertex is unique for each 3-block B_3. □

Lemma 4.48. *The sets $\mathcal{S}_E$ and $\mathcal{R}_V$ of Lemmas 4.43 and 4.45 are existentially MSOL-definable with $3k + 2$ parameters.*

Proof. Let $G = (V, E)$ denote a k-outerplanar graph, such that the virtual edges introduced by the Tutte decompositions of its 2-connected blocks are already included in E. On a high level, for defining $\mathcal{R}_V$ and $\mathcal{S}_E$, we need to encode is the following:

(i) There are sets $\mathcal{R}_V \subseteq V$, $F \subseteq E$ and $F' \subseteq E$ with $\mathcal{S}_E = F \cup F'$.

(ii) Guess a root $r_T \in V$, such that F is the edge set of a rooted spanning tree in G.

(iii) An edge $e = \{x, y\}$ is possibly (but not necessarily) a member of F', if $\{x, y\} \in \mathcal{C}_2$ and $e \notin F$.

(iv) For all $B_3 \in \mathcal{B}_3$, the graph $T^*_{B_3} = (B_3, \mathcal{S}_E \cap (B_3 \times B_3))$ is a spanning tree of the graph $G_{B_3} = G[B_3]$ with $er(G_{B_3}, T^*_{B_3}) \leq 2k$ and $fr(G_{B_3}, T^*_{B_3}) \leq k$.

(v) A vertex $v \in V$ is possibly (but not necessarily) a member of $\mathcal{R}_V$, if it is a member of a 2-cut $\{v, w\} \in \mathcal{C}_2$.

(vi) For each 3-connected 3-block $B_3 \in \mathcal{B}_3$, there is a vertex $r_{B_3} \in \mathcal{R}_V$, such that $T^*_{B_3}$ can be rooted at r_{B_3} (without altering the edge direction of any other edge in $\mathcal{S}_E$).

The existence of such sets $\mathcal{R}_V$ and $\mathcal{S}_E$ is shown in Lemmas 4.43 and 4.45, so we do not need to encode all details mentioned in the corresponding proofs explicitly. Property (iv) is MSOL-definable by Proposition 4.21, since G_{B_3} is 3-connected. The details are as follows. To shorten our notation, we will use the symbol $E_{B_3,\mathcal{S}_E}$ instead of the term 'IncE$(B_3) \cap \mathcal{S}_E$'.

(i) $(\exists \mathcal{R}_V \subseteq V)(\exists F \subseteq E)(\exists F' \subseteq E)(\exists \mathcal{S}_E \subseteq E)(\mathcal{S}_E = F \cup F')$

(ii) $(\exists r_T \in V)(\text{Tree}_{\rightarrow}(r_T, F))$

(iii) $e \in F' \rightarrow \exists x \exists y(\neg x = y \wedge \text{Inc}(x, e) \wedge \text{Inc}(y, e) \wedge \neg e \in F$
$\wedge \exists X(\mathbf{Bag}_{\mathcal{C}_2}(X) \wedge x \in X \wedge y \in X))$

(iv) $(\forall B_3 \subseteq V)\Big(\text{3-Conn}_{\mathcal{B}_3}(B_3) \rightarrow \Big(er(B_3, \text{IncE}(B_3), E_{B_3,\mathcal{S}_E}) \leq 2k$
$\wedge fr(B_3, \text{IncE}(B_3), E_{B_3,\mathcal{S}_E}) \leq k\Big)\Big)$

(v) $v \in \mathcal{R}_V \rightarrow \exists X(\mathbf{Bag}_{\mathcal{C}_2}(X) \wedge v \in X)$

(vi) $(\forall B_3 \subseteq V)\Big(\text{3-Conn}_{\mathcal{B}_3}(B_3) \rightarrow (\exists r_{B_3} \in \mathcal{R}_V)$
$\Big(\text{Tree}_{\rightarrow}(r_{B_3}, B_3, E_{B_3,\mathcal{S}_E})\Big)\Big)$

As parameters we have the edge set of the spanning tree and again a $3k$-coloring and one edge set to fix the orientation of the edges in $\mathcal{S}_E$. □

We can now use the above results to conclude that we can find predicates defining tree decompositions of 3-connected 3-blocks of k-outerplanar graphs.

Corollary 4.49. *Let $G = (V, E)$ be a k-outerplanar graph. Then, there exist predicates existentially defining tree decompositions of width at most $3k$ for each 3-connected 3-block of G with $\mathcal{O}(1)$ parameters.*

Proof. By Lemma 4.20 we know that a 3-connected k-outerplanar graph admits an MSOL-definable tree decomposition of width $3k$, based on a rooted spanning tree of the graph. By Corollary 4.36

we can define such a tree decomposition in a structure, which also includes the virtual edges of a 3-block in G (and by Lemma 4.29 we know that this graph is still k-outerplanar). Finally, by Lemmas 4.43, 4.45 and 4.48 we know that we can find definable edge and vertex sets which contain the edges of spanning trees for each 3-connected 3-block with the required bound on their vertex and edge remember numbers without violating the cardinality constraint of monadic second order logic. Similarly, we can find sets containing anchor and co-anchor edges for all 3-connected 3-blocks in a straightforward way. Hence, also for defining the ordering of all incident edges of all vertices in a 3-connected 3-block, two sets are sufficient.

Eventually we observe that number of parameters involved is bounded by a constant in each step, for the exact bounds see the corresponding result. □

Combining Lemma 4.38 and Corollary 4.49 yields that k-outerplanar graphs admit existentially MSOL-definable tree decompositions of width at most $3k + 3$ with $\mathcal{O}(1)$ parameters, hence we proved Lemma 4.3(ii). By Lemma 2.12(iii) (page 9) and in the light of Courcelle's Theorem [12] we have the first main result of this paper.

Theorem 4.50. *CMSOL-definability equals recognizability for k-outerplanar graphs.*

5 ℓ-Chordal Partial k-Trees

In this section we prove the second main result of this paper, which is that recognizability implies CMSOL-definability for all partial k-trees that do not contain a cycle of length at least some constant ℓ as an induced subgraph (in the following referred to as *ℓ-chordal partial k-trees*). Our proof strategy is the same as in the previous section, i.e. we show that all ℓ-chordal partial k-trees admit existentially MSOL-definable bounded width tree decompositions which by Lemma 2.12(iii) (page 9) yields the desired result.

Definition 5.1 (Chordality, Triangulation, Fill Edges)**.** Let $G = (V, E)$ be a graph.

(i) The *chordality* of G is the length of a longest induced cycle.

(ii) G is called *chordal*, if it has chordality 3 (and ℓ-chordal if it has chordality ℓ).

(iii) A *triangulation* of G is a graph $H = (V, E \cup F) \sqsupseteq G$ which is chordal. The edges in F are called the *fill edges*.

(iv) Let $H = (V, E \cup F)$ be a triangulation of G. H is called *minimal*, if the graph $H' = (V, E \cup F')$ is not chordal for any $F' \subset F$.

The following result is a well-known connection between graphs of bounded treewidth and minimal triangulations.

Proposition 5.2. *Let G be a graph. G has treewidth at most k if and only if it has a triangulation with maximum clique size at most $k+1$.*

In particular, we will first show how to construct an existentially MSOL-definable tree decomposition for each chordal graph of treewidth k in Section 5.1 and in Section 5.2 we extend this construction to all graphs of bounded chordality. The main idea in the latter proof, as indicated by Proposition 5.2, is to show that the construction for chordal partial k-trees can be applied to a triangulation of an ℓ-chordal partial k-tree as well. We will show that we can define in monadic second order logic a predicate to identify the fill edges of minimal triangulations of such graphs. This in turn allows for another application of Lemma 4.35 (page 43), which states that every definable graph property for a subclass of partial k-trees is also definable for graphs equipped with an additional set of virtual edges, if there exists a predicate identifying these edges.

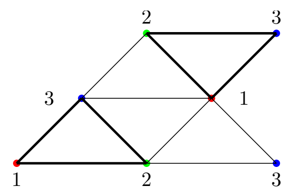
5.1 Chordal Partial k-Trees

We will now show how to construct existentially MSOL-definable width-k tree decompositions for chordal partial k-trees.

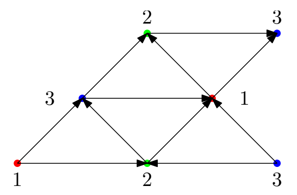
Definition 5.3 (Perfect Elimination Order (peo))**.** Let $G = (V, E)$ be a graph and $n = |V|$. A linear order $\pi : V \to \mathbb{N}_{|n}$ is called *perfect elimination order*, if for every $v \in V$, we have that $G[\{w \mid \{v, w\} \in E \wedge \pi(w) > \pi(v)\}]$, i.e. the graph induced by all higher numbered neighbors of v in π, is a clique.

To (implicitly) encode perfect elimination orders in MSOL, we now define a property of edge orientations of graphs.

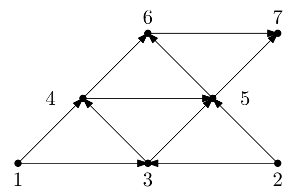
Definition 5.4 (Adjacent Out-Neighbors (aon) Property)**.** Let $G = (V, E)$ be a graph and ϕ an orientation on the edges in E. For an edge $e = \{v, w\} \in E$ we denote by $(v, w)_\phi$ that e is oriented from v to w according to ϕ. We say that ϕ has the *adjacent out-neighbors* property, if for any pair of edges $\{u, v\}, \{u, w\} \in E$, if $(u, v)_\phi$ and $(u, w)_\phi$, then $\{v, w\} \in E$. We call an acyclic orientation with the aon-property an *aon-order*. We will sometimes denote the directed graph obtained from G by orienting all edges according to ϕ by G_ϕ.



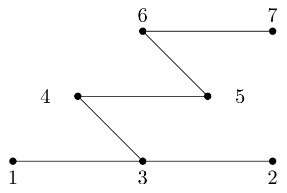
(a) A 3-coloring $\gamma = \{1,2,3\}$ and a chosen subset F of the edges in bold.



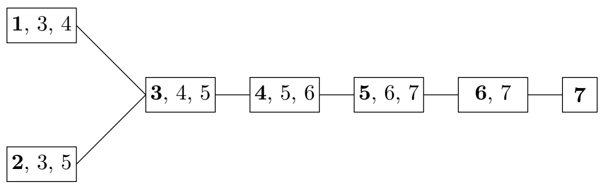
(b) The edge orientation induced by γ and F.



(c) A perfect elimination order π defined by γ and F.



(d) The aon-defined spanning tree of π in G.



(e) The tree decomposition defined by γ and F. Vertices $v \in V$ are identified with their peo-number $\pi(v)$, where the bold vertex in each bag is its witness.

Figure 12: A chordal graph $G = (V, E)$ of treewidth 2 and its tree decomposition.

Theorem 5.5. *Let $G = (V, E)$ be a graph. The following are equivalent.*

(i) G is chordal.

(ii) G has a perfect elimination order.

(iii) G has a tree decomposition of minimum width, where each bag induces a clique in G.

(iv) There is an acyclic orientation ϕ on the edges of G that has the adjacent out-neighbors property.

Proof. (i) $\Leftrightarrow$ (ii) $\Leftrightarrow$ (iii) is well-known, see e.g. [22]. (ii) $\Rightarrow$ (iv). Let π denote the peo of G. We construct ϕ as follows. For each edge $\{v, w\} \in E$ we let $(v, w)_\phi$ if and only if $\pi(v) < \pi(w)$. (iv) $\Rightarrow$ (ii). Take an arbitrary topological order of the (directed) graph G_ϕ. One easily verifies that this is a peo. $\square$

Proposition 5.6. *Let $G = (V, E)$ be a chordal graph of treewidth k. There is an aon-order ϕ_{aon} on E which is existentially MSOL-definable with $k + 2$ parameters.*

Proof. The existence of ϕ_{aon} is stated in Theorem 5.5, so in the following we show its existential MSOL-definability. By Lemma 2.10 (page 8), every orientation on the edge set E is existentially MSOL-definable with $k + 2$ parameters. For an edge $e = \{v, w\} \in E$, denote by $(v, w)_{\phi_{aon}}$ that e is oriented from v to w according to ϕ_{aon} (and note that a predicate to verify this can be encoded by Lemma 2.10). Furthermore, using methods similar to [10, Theorem 4], one can encode a predicate $\mathrm{Cycle}^{\phi_{aon}}_{\rightarrow}(V, E)$, which is true if and only if (V, E) is a directed cycle according to ϕ_{aon}. We can now encode ϕ_{aon} straightforwardly.

$$\begin{aligned}&\neg((\exists E' \subseteq E)(\mathrm{Cycle}^{\phi_{aon}}_{\rightarrow}(\mathrm{IncV}(E'), E')))\\ &\wedge \forall u \forall v \forall w((((u, v)_{\phi_{aon}} \wedge (u, w)_{\phi_{aon}})) \rightarrow \{v, w\} \in E)\end{aligned}$$

□

We now have all the ingredients to construct existentially MSOL-definable width-k tree decompositions for chordal partial k-trees. For an illustration of the ideas presented above and the construction of the tree decomposition in the following proof, see Figure 12.

Lemma 5.7. *Chordal partial k-trees admit existentially MSOL-definable width-k tree decompositions with $k + 3$ parameters.*

Proof. Let $G = (V, E)$ be a chordal partial k-tree. Suppose we have an MSOL-definable aon-order ϕ_{aon} on E as stated in Proposition 5.6. We use ϕ_{aon} to construct an MSOL-definable spanning tree of G, based on which we will construct a width-k tree decomposition of G, which is existentially MSOL-definable. Our construction will be based on a spanning tree T of G which can be derived from ϕ_{aon}.

Definition 5.8. Let $G = (V, E)$ be a chordal partial k-tree, ϕ_{aon} as above and π the perfect elimination order corresponding to ϕ_{aon}. A spanning tree $T = (V, F)$ of G is called *aon-defined spanning tree*, if the following holds.

(i) For each vertex $v \in V$ with out-degree (according to ϕ_{aon}) at least one and out-neighbor $w \in V$, $\{v, w\} \in F$ iff[20]

$$w = \underset{\substack{x:\{v,x\}\in E\\ \pi(x)>\pi(v)}}{\operatorname{argmin}} \pi^{-1}(x).$$

(ii) The root of T is the unique vertex with outdegree zero in ϕ_{aon}.

One can verify that there is precisely one vertex satisfying condition (ii) of the previous definition.

[20] I.e. the vertex w is the neighbor of v that appears first in π.

Proposition 5.9. *Let G, π and ϕ_{aon} be as above. The aon-defined spanning tree $T = (V, F)$ of G is existentially MSOL-definable with one additional parameter, its edge set F.*

Proof. We guess an edge set F and encode predicates verifying the two conditions stated in Definition 5.8. We first note that by Theorem 5.5 and its proof, for an edge $\{v, w\} \in E$ the statements $\pi(v) < \pi(w)$ and $(v, w)_{\phi_{aon}}$ are equivalent. Let $\deg_{\rightarrow}^{\phi_{aon}}(v)$ denote the out-degree of a vertex $v \in V$. Note that one can encode predicates comparing the out-degree of vertices to constants in a similar way as in [10, Theorem 4], then we encode property (i) of T as follows. Let for now $N_{\rightarrow}^{\phi_{aon}}(v)$ denote the set of out-neighbors of v.

$$\forall v\Big(\deg_{\rightarrow}^{\phi_{aon}}(v) > 1 \rightarrow (\forall w \in N_{\rightarrow}^{\phi_{aon}}(v))\Big(\{v, w\} \in F$$
$$\leftrightarrow (\forall x \in N_{\rightarrow}^{\phi_{aon}}(v) \setminus \{w\})((w, x)_{\phi_{aon}})\Big)\Big)$$

We can encode a predicate identifying the root of T trivially.

$$\text{Root}(v) \Leftrightarrow \deg_{\rightarrow}^{\phi_{aon}}(v) = 0$$

□

In the following, let $T = (V, F)$ denote the aon-defined spanning tree of G. We construct from T a width-k tree decomposition $(\mathcal{T} = (\mathcal{N}, \mathcal{F}), \mathcal{X})$ as follows. For each vertex $v \in V$ there is one node $t \in \mathcal{N}$, such that $\mathcal{X}_t = \{v\} \cup N_{\rightarrow}^{\phi_{aon}}(v)$, i.e. $\mathcal{X}_t$ contains v and all its out-neighbors according to ϕ_{aon}. For any pair of nodes $t_v, t_w \in \mathcal{N}$, where t_v is associated with $v \in V$ and t_w is associated with $w \in V$, we add the edge $\{t_v, t_w\}$ to $\mathcal{F}$, if and only if $\{v, w\} \in F$, i.e. there is an edge between v and w in the aon-defined spanning tree T. To make $(\mathcal{T}, \mathcal{X})$ a rooted tree decomposition, we give $\{t_v, t_w\}$ the opposite orientation of $\{v, w\}$ according to ϕ_{aon}. (Recall that the vertex with out-degree 0 is the root of the tree.)

We now turn to encoding the **Bag**- and **Parent**-predicates. We only require one vertex bag type τ and define $\mathbf{Bag}_\tau(v, S)$ according to the above explained construction.

$$\mathbf{Bag}_\tau(v, S) \Leftrightarrow \forall v'(v' \in S \leftrightarrow (v = v' \vee (v, v')_{\phi_{aon}}))$$

We define $\mathbf{Parent}(S_p, S_c)$ accordingly.

$$\mathbf{Parent}(S_p, S_c) \Leftrightarrow \exists v_p \exists v_c(\mathbf{Bag}_\tau(v_p, S_p) \wedge \mathbf{Bag}_\tau(v_c, S_c)$$
$$\wedge \{v_p, v_c\} \in F \wedge (v_c, v_p)_{\phi_{aon}})$$

The number of parameters is $k + 2$ for defining ϕ_{aon} and the edge set F is the parameter for the aon-defined spanning tree.

This completes the proof of Lemma 5.7. □

5.2 Extension to ℓ-Chordal Partial k-Trees

We will now prove that the construction given in the previous section can be applied to ℓ-chordal partial k-trees. We first introduce the notion of (non-perfect) elimination orders.

Definition 5.10 (Elimination Order)**.** Let $G = (V, E)$ be a graph and $n = |V|$. An *elimination order* $\pi : V \to \mathbb{N}_n$ is a linear order on the vertices of G. The fill edges, again denoted by Fill, introduced by π are the edges obtained by removing the vertices of the graph in the order given by π and making all its non-adjacent neighbors adjacent via a fill edge. The graph $G_\pi = (V, E \cup \mathit{Fill})$ thus obtained is called the *filled graph.*

Proposition 5.11. *Let G be a graph and π an elimination order of G. Then, π is a perfect elimination order of G_π.*

A central step in the following proof is that each ℓ-chordal partial k-tree has an elimination order π such that G_π is a chordal partial k-tree. Furthermore we will see that the fill edges Fill can be partitioned into sets $F_1, \dots, F_q$, with $q \leq \ell - 3$, according to when they were introduced (and we let $F_0 = E$). That is, for an edge $f \in \mathit{Fill}$, how many fill edges had to be added to make the two non-adjacent endpoints of f neighbors. Suppose v has two non-adjacent neighbors w and x with $\{v, w\} \in F_i$ and $\{v, x\} \in F_j$. Then, $\{w, x\} \in F_{1+\max\{i,j\}}$. Since this gives us a partition into a constant number of sets, we will be able to guess a constant number of witness sets to identify these edges.

For the proof of our next lemma, we use the following famous characterization of minimal triangulations by Parra and Scheffler [30].

Theorem 5.12 (Theorem 4.7 in [30])**.** *A triangulation of G is minimal if and only if it is obtained by completing into cliques a maximal set of non-crossing minimal separators of G.*

Lemma 5.13. *If G has treewidth k and chordality $\ell \geq 3$ then G has an elimination order π giving $\mathit{Fill} = F_1 \cup F_2 \cup ... \cup F_q$ with $q \leq \ell - 3$ and with the filled graph G_π being chordal and having treewidth k.*

Proof. It is well-known that G has treewidth k if and only if G has a minimal triangulation where the largest clique is of size $k + 1$ (Proposition 5.2). Consider such a minimal triangulation G' of G, and let π be a perfect elimination order of G' such that $G' = G_\pi$. Let $\mathit{Fill} = F_1 \cup F_2 \cup \cdots \cup F_q$ be the fill given by π on G. Let $F_0 = E$. For $i = 0, 1, 2, 3, \dots, q$, let G_i be the graph $G_i = (V, F_0 \cup F_1 \cup \cdots \cup F_i)$. Hence $G_0 = G$ and $G_q = G_\pi$.

We will show by induction on decreasing i starting from $i = q$ that G_i contains an induced cycle of size at least $q - i + 3$. This implies that in $G = G_0$ there is an induced cycle of length at least $q+3$. Since G has chordality ℓ, it will follow that $q \leq \ell - 3$, completing the proof of the lemma.

The induction hypothesis is the following statement: G_i has an induced cycle of size at least $q-i+3$ that contains an edge belonging to F_i.

For the base case $i = q$, in G_q, let $\{u, v\}$ be a fill edge in F_q and let x be the vertex whose elimination introduced the edge $\{u, v\}$. Clearly $\{u, v\}$ is an edge of the 3-cycle $\{u, v, x\}$.

Assume that the induction hypothesis is true for i, and let us prove it for $i-1$. In G_i, let C be an induced cycle of size at least $q-i+3$ that contains an edge $\{u, v\} \in F_i$. Let x be the vertex whose elimination resulted in the edge $\{u, v\}$. If $i < q$ then clearly C does not contain x since C has no chord. In case $i = q$, since u and v are in a minimal separator of G_q, they separate x from a vertex y. Since G_q is chordal, u and v are adjacent to y, and x is not adjacent to y. In this case, we take the cycle $\{u, v, y\}$ as C. Observe that neither $\{x, u\}$ nor $\{x, v\}$ belongs to F_i, and at least one of them belongs to F_{i-1}. Since by Theorem 5.12 every fill edge appears inside a minimal separator of G and no fill edges are ever added across such a minimal separator, we know that x is not adjacent in G_q to any vertex of C other than u and v. Hence when we remove edge $\{u, v\}$, we get a cycle C' of length $q - i + 4$ that contains one less edge from F_i, and at least one edge of F_{i-1}. If there are edges of F_i remaining on C', we can repeatedly use the same argument to remove each of them and get a new cycle of length one more and containing one less edge of F_i, until all edges of F_i are removed from the current cycle. Consequently, in G_{i-1} there is an induced cycle of length at least $q - i + 4 = q - (i - 1) + 3$ that contains at least one edge from F_{i-1}. □

Our strategy to complete the proof of Courcelle's Conjecture for graphs of bounded treewidth and bounded chordality is now as follows. We show that there are predicates identifying edges of each fill-level and then apply Lemma 4.35 (page 43). Recall that this result states that every definable graph property is also definable for a graph with some set of *virtual* edges (in this case: the set $Fill$), if we can encode a predicate $\phi_{Fill}(v, w)$ identifying edges $\{v, w\} \in Fill$. This will in turn allow for an application of Lemma 5.7 for the filled graph, taking into account the additional parameters which are necessary for defining $\phi_{Fill}(v, w)$. Hence, we will now prove the following.

Lemma 5.14. *Let $G = (V, E = F_0)$ be a partial k-tree of chordality ℓ. There is an existentially MSOL-definable predicate $\phi_{Fill}(v, w)$, which is true if and only if $\{v, w\} \in Fill = F_1 \cup F_2 \cup \cdots \cup F_q$, where $q \leq \ell - 3$, with $\mathcal{O}(1)$ parameters.*

Proof. Suppose without loss of generality that we are given a $(k+1)$-coloring of V, and in the following we denote its color classes by $V_1, \ldots, V_{k+1}$. We use this coloring throughout the proof to induce an orientation on the edges of G (and filled versions of G). We denote this orientation by ψ and let $(v, w)_\psi$ indicate that there is an edge $\{v, w\}$ in G, which is oriented from v to w according to ψ. We furthermore assume that we are given an elimination order π as stated in Lemma 5.13.

To guess the fill edges F_i of level i, we guess $k(k+1)/2$ vertex sets $Z_{1,2}, Z_{1,3}, \ldots, Z_{k,k+1}$ with the following interpretation. Suppose v and w are two non-adjacent vertices that share a common neighbor x, such that $\{x, v\} \in F_j$ and $\{x, w\} \in F_{j'}$ with $j, j' < i$ and either $j = i - 1$ or $j' = i - 1$. (Note that the latter is a necessary condition for the edge $\{v, w\}$ to be a member of the fill level F_i.) Suppose that $v \in V_{c_1}$ and $w \in V_{c_2}$ with $c_1 < c_2$, $(x, v)_\psi$ and $(x, w)_\psi$. Then, to represent the edge $\{v, w\} \in F_i$, we add x as a witness for this edge to Z_{c_1,c_2}. We are now ready to encode $\phi_{Fill_i}(v, w)$, which identifies such edges. First, we guess the witness sets of the edges:

$$(\exists Z^i_{1,2} \subseteq V)(\exists Z^i_{1,3} \subseteq V) \cdots (\exists Z^i_{k,k+1} \subseteq V)$$

The details for $\phi_{Fill_i}(v, w)$ are as follows, where $\phi_{Fill_0}(v, w) \Leftrightarrow \{v, w\} \in E$. Note that this construction is recursive, i.e. when defining ϕ_{Fill_i}, we assume that $\phi_{Fill_{i-1}}$ is already defined and the aon-ordering ψ has been updated in each step, see the next paragraph of the proof.

$$\begin{aligned}
\phi_{Fill_i}(v, w) \Leftrightarrow \exists x \Bigg(& \bigvee_{\substack{j,j'=0,\ldots,i-1 \\ j=i-1 \vee j'=i-1}} \Big(\phi_{Fill_j}(x, v) \wedge \phi_{Fill_{j'}}(x, w)\Big) \\
& \wedge (x, v)_\psi \wedge (x, w)_\psi \\
& \wedge \bigvee_{c_1 < c_2 = 1,\ldots,k+1} \Big(v \in V_{c_1} \wedge w \in V_{c_2} \wedge x \in Z^i_{c_1,c_2}\Big)\Bigg)
\end{aligned}$$

We are now ready to define $\phi_{Fill}(v, w)$.

$$\phi_{Fill}(v, w) \Leftrightarrow \bigvee_{i=1,\ldots,q} \phi_{Fill_i}(v, w)$$

To complete the proof, we need to ensure that $\phi_{Fill}(v, w)$ indeed identifies the edges of the filled graph G_π, i.e. we have to encode the following.

(i) G_π is a chordal partial k-tree.

(ii) $V_1, \ldots, V_{k+1}$ is a valid coloring in G_π.

(iii) The edge orientation ψ has the aon-property.

We note that every definable graph property for chordal graphs is definable for G_π as well by an application of Lemma 4.35 (page 43) with $\phi_{Virt}(v, w) = \phi_{Fill}(v, w)$. Hence the definability of (ii) follows directly, (iii) follows from a combination with Proposition 5.6 and for (i) we observe the following. It is not hard to encode the chordality constraint using the predicate $\text{Cycle}(V', \text{IncE}(V'))$ (due to [10, Theorem 4]) on sets $V' \subseteq V$ and another way of stating that G_π is a partial k-tree is to say that the out-neighborhood of each vertex v in G_ψ (the graph obtained by orienting the edges of G according to ψ) is a clique of at most k vertices, see Theorem 5.5. The latter condition can as well be encoded in a straightforward way using a predicate $\text{Clique}_k(N_\rightarrow(v), \text{IncE}(N_\rightarrow(v)))$ (again due to [10, Theorem 4]).

This completes the proof of Lemma 5.14. □

We now have the following consequence.

Corollary 5.15. *Partial k-trees of chordality at most ℓ admit existentially MSOL-definable width-k tree decompositions with $\mathcal{O}(1)$ parameters.*

Proof. Combine Lemmas 4.35 (page 43), 5.7 and 5.14. □

By Lemma 2.12(iii) (page 9), Corollary 5.15 and Courcelle's Theorem [12] we have the main result of this section.

Theorem 5.16. *Recognizability equals CMSOL-definability for ℓ-chordal partial k-trees.*

6 Conclusion

In this paper, we investigated a conjecture by Courcelle from 1990, which states that every recognizable graph property is definable in counting monadic second order logic [12]. We introduced a new proof technique, based on the Myhill-Nerode theory for graphs of bounded treewidth (see [21, Section 12.7] and Theorem 3.10 on page 12) to give self-contained proofs of two special cases of this conjecture — for the class of k-outerplanar graphs and for ℓ-chordal graphs of bounded treewidth.

In particular, we proved that defining a bounded width tree decomposition for a graph class $\mathcal{C}$ in monadic second order logic suffices to resolve Courcelle's Conjecture for $\mathcal{C}$ (Lemma 2.12, page 9) and in some cases we even obtain a stronger result. That is, for all graph classes that admit bounded degree or ordered MSOL-definable tree decompositions, the counting predicates of CMSOL are not needed. We showed that this holds for 3-connected k-outerplanar graphs (Theorem 4.23, page 39). However, Courcelle proved that CMSOL-definability equals MSOL-definability for each graph class whose members have an MSOL-definable linear order on their vertices [16] (see also [5]). This includes every graph class, whose members have spanning trees of bounded degree and subsequently the statement holds for each 3-connected planar graph, since a well-known result by Barnette states that every 3-connected planar graph has a spanning tree of degree at most 3 [3]. Hence, we would like to know whether MSOL-definable linear orders are indeed a necessary condition for avoiding the use of the counting predicate in a proof of (a special case of) Courcelle's Conjecture.

Open Question 6.1. Is it possible to construct ordered or bounded degree MSOL-definable tree decompositions for a graph class whose members do not admit MSOL-definable linear orders?

In both our proofs, a result by Courcelle which states that for some graph classes, including partial k-trees, all predicates which use quantification over vertex and edge sets can be translated into predicates only using vertex quantification [14], has been an essential tool. This way we were able to construct MSOL-definable tree decompositions for more restricted graph classes which had favorable properties in terms of definability in MSOL. We then showed that these results generalize to the whole graph class, once we added a set of MSOL-definable virtual edges to each member. In particular, for k-outerplanar graphs the main step of our construction was to find MSOL-definable tree decompositions for 3-connected graphs and for partial k-trees of bounded chordality, constructing an MSOL-definable tree decompositions for chordal graphs was the main step. However, these results did not generalize trivially, see Sections 4.3 and 5.2.

It seems though that the general idea of this approach might be the right step towards resolving Courcelle's Conjecture in general, which can be formulated as the following question.

Open Question 6.2. Is there a set of fill edges, which can be identified with an MSOL-predicate for any partial k-tree, such that the filled graph admits a width-k MSOL-definable tree decomposition?

Acknowledgements

The first author thanks Bruno Courcelle and Mike Fellows for inspiring discussions. The third author thanks an anonymous referee of IPEC 2015 for very useful comments on a proof in Section 4, Paul Bonsma for his clear presentation of Tutte decompositions which contained the proof of Proposition 4.30 [9, Lecture V.(9)] and Tom van der Zanden for Example 4.42 [37].

References

[1] Karl R. Abrahamson and Michael R. Fellows. Finite automata, bounded treewidth, and well-quasi-ordering for bounded treewidth. In *Proceedings of the AMS Summer Workshop on Graph Minors and Graph Structure Theory*, volume 147 of *Contemporary Mathematics*, pages 539–564. AMS, 1993.

[2] Stefan Arnborg, Jens Lagergren, and Detlef Seese. Easy problems for tree-decomposable graphs. *Journal of Algorithms*, 12(2):308–340, 1991.

[3] David Barnette. Trees in polyhedral graphs. *Canadian Journal of Mathematics*, 18(58):731–736, 1966.

[4] René van Bevern, Rodney G. Downey, Michael R. Fellows, Serge Gaspers, and Frances A. Rosamond. Myhill–Nerode methods for hypergraphs. *Algorithmica*, 73(4):696–729, 2015.

[5] Achim Blumensath and Bruno Courcelle. Monadic second-order definable graph orderings. *Logical Methods in Computer Science*, 10(2), 2014.

[6] Hans L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science*, 209(1-2):1–45, 1998.

[7] Hans L. Bodlaender, Pinar Heggernes, and Jan Arne Telle. Recognizability equals definability for graphs of bounded treewidth and bounded chordality. In *Proceedings EUROCOMB 2015*, Electronic Notes in Discrete Mathematics. Elsevier, 2015.

[8] Hans L. Bodlaender and Arie M.C.A. Koster. Safe separators for treewidth. *Discrete Mathematics*, 306(3):337 – 350, 2006.

[9] Paul Bonsma. Lecture material for algorithmic graph theory 2011/2012 at Humboldt University, Berlin. Retrieved 03/05/2015 via http://www2.informatik.hu-berlin.de/logik/lehre/WS11-12/Para/index.html.

[10] Richard B. Borie, R. Gary Parker, and Craig A. Tovey. Automatic generation of linear-time algorithms from predicate calculus descriptions of problems on recursively constructed graph families. *Algorithmica*, 7(1-6):555–581, 1992.

[11] J. Richard Büchi. Weak second-order arithmetic and finite automata. *Mathematical Logic Quarterly*, 6(1-6):66–92, 1960.

[12] Bruno Courcelle. The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. *Information and Computation*, 85(1):12–75, 1990.

[13] Bruno Courcelle. The monadic second-order logic of graphs V: On closing the gap between definability and recognizability. *Theoretical Computer Science*, 80(2):153–202, 1991.

[14] Bruno Courcelle. The monadic second order logic of graphs VI: On several representations of graphs by relational structures. *Discrete Applied Mathematics*, 54(2-3):117–149, 1994.

[15] Bruno Courcelle. The monadic second-order logic of graphs VIII: Orientations. *Annals of Pure and Applied Logic*, 72(2):103–143, 1995.

[16] Bruno Courcelle. The monadic second-order logic of graphs X: Linear orderings. *Theoretical Computer Science*, 160(12):87–143, 1996.

[17] Bruno Courcelle. The monadic second-order logic of graphs XI: Hierarchical decompositions of connected graphs. *Theoretical Computer Science*, 224(1-2):38–53, 1999.

[18] Bruno Courcelle. The monadic second-order logic of graphs XII: Planar graphs and planar maps. *Theoretical Computer Science*, 237(1-2):1–32, 2000.

[19] Bruno Courcelle and Joost Engelfriet. *Graph Structure and Monadic Second-Order Logic — A Language-Theoretic Approach*, volume 138 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 2012.

[20] Reinhard Diestel. *Graph Theory.* Number 173 in Graduate Texts in Mathematics. Springer, 4th edition, 2012. Corrected reprint.

[21] Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity.* Texts in Computer Science. Springer, 2013.

[22] Martin Charles Golumbic. *Algorithmic Graph Theory and Perfect Graphs.* Number 57 in Annals of Discrete Mathematics. Elsevier, 2nd edition, 2004.

[23] Lars Jaffke and Hans L. Bodlaender. Definability equals recognizability for k-outerplanar graphs. In *Proceedings IPEC 2015*, Leibniz International Proceedings in Informatics, pages 175–186. Dagstuhl Publishing, 2015.

[24] Valentine Kabanets. Recognizability equals definability for partial k-paths. In *Proceedings ICALP 1997*, volume 1256 of *LNCS*, pages 805–815. Springer, 1997.

[25] Damon Kaller. Definability equals recognizability of partial 3-trees and k-connected partial k-trees. *Algorithmica*, 27(3-4):348–381, 2000.

[26] Ioannis Katsikarelis. Computing bounded-width tree and branch decompositions of k-outerplanar graphs. *ArXiv e-prints*, 2013. http://arxiv.org/abs/1301.5896.

[27] Denis Lapoire. Recognizability equals monadic second-order definability for sets of graphs of bounded tree-width. In *Proceedings STACS 1998*, volume 1373 of *LNCS*, pages 618–628. Springer, 1998.

[28] John R. Myhill. Finite automata and the representation of events. Technical Report WADC TR-57-624, Wright-Paterson Air Force Base, 1957.

[29] Anil Nerode. Linear automaton transformations. *Proceedings of the American Mathematical Society*, 9(4):541–544, 1958.

[30] Andreas Parra and Petra Scheffler. Characterizations and algorithmic applications of chordal graph embeddings. *Discrete Applied Mathematics*, 79(1-3):171–188, 1997.

[31] Maciej M. Sysło. Characterizations of outerplanar graphs. *Discrete Mathematics*, 26(1):47 – 53, 1979.

[32] Wolfgang Thomas. Languages, automata, and logic. In *Handbook of Formal Languages. Beyond Words*, volume 3, pages 389–455. Springer, 1996.

[33] William T. Tutte. *Connectivity in Graphs.* University of Toronto Press, 1966.

[34] William T. Tutte. *Graph Theory*, volume 21 of *Encyclopedia of Mathematics and its Applications.* Addison-Wesley, 1984.

[35] Klaus Wagner. Über eine Eigenschaft der ebenen Komplexe. *Mathematische Annalen*, 114(1):570–590, 1937.

[36] Hassler Whitney. Congruent graphs and the connectivity of graphs. *American Journal of Mathematics*, 54:150–168, 1932.

[37] Tom van der Zanden. Personal communication, 2015.