First-Order and Monadic Second-Order Model-Checking on Ordered Structures

Viktor Engelmann and Stephan Kreutzer and Sebastian Siebertz
School of Electrical Engineering and Computer Science,
Technical University Berlin, Berlin, Germany
{viktor.engelmann, stephan.kreutzer, sebastian.siebertz}@tu-berlin.de

***Abstract*—Model-checking for first- and monadic second-order logic in the context of graphs has received considerable attention in the literature. It is well-known that the problem of verifying whether a formula of these logics is true in a graph is computationally intractable but it does become tractable on interesting classes of graphs such as classes of bounded tree-width.**

In this paper we continue this line of research but study model-checking for first- and monadic second-order logic in the presence of an ordering on the input structure. We do so in two settings: the general ordered case, where the input structures are equipped with a fixed order or successor relation, and the *order invariant* case, where the formulas may resort to an ordering but their truth must be independent of the particular choice of order. In the first setting we show very strong intractability results for most interesting classes of graphs. In contrast, in order invariant case we obtain tractability results for order invariant monadic second-order logic on the same classes of graphs as in the unordered case. For first-order logic, we obtain tractability of successor-invariant FO on planar graphs.

I. Introduction

Faced with the seeming intractability of many common algorithmic problems, much work has been devoted to studying restricted classes of admissible inputs on which tractability results can be retained. A particularly rich source of structural properties which guarantee the existence of efficient algorithms for many problems on graphs comes from structural graph theory, especially graph minor theory. It has been found that most generally hard problems become tractable on graph classes of bounded tree-width and many remain tractable on planar graphs or graph classes excluding a fixed minor.

Besides many specific results giving algorithms for individual problems, of particular interest are results that establish tractability of a large class of problems on specific classes of instances. These results come in various flavours. Here we are mostly interested in results that take a descriptive approach, i.e., results that use a logic to describe algorithmic problems and then provide general tractability results for all problems definable in that logic on specific classes of inputs. Results of this form are usually referred to as *algorithmic meta-theorems*. The first explicit algorithmic meta-theorem was proved by Courcelle [3] establishing tractability of decision problems definable in monadic second-order logic (even with quantification over edge sets) on graph classes of bounded tree-width, followed by similar results for monadic second-order logic with only quantification over vertex sets on graph classes of bounded clique-width [4], for first-order logic on graph classes of bounded degree [26], on planar graphs and more generally graph classes of bounded local tree-width [14], on graph classes excluding a fixed minor [13], on graph classes locally excluding a minor [6] and graph classes of bounded local expansion [9].

So far, most of the work on algorithmic meta-theorems has focussed on unordered structures and as many results mentioned above rely on locality theorems for first-order logic such as Gaifman's locality theorem [15], the techniques used there do not readily extend to ordered structures. In this paper we study the complexity of first-order model-checking on structures where an ordering is available to be used in formulas. We do so in two different settings. The first is that the input structures are equipped with a fixed order or successor relation. We show that first-order logic on ordered structures as well as on structures with a successor relation is essentially intractable on nearly all interesting classes.

The other case we consider is order- or successor-invariant first-order or monadic second-order logic. In order-invariant first-order logic, we are allowed to use an order relation in the formulas but whether the formula is true in a given structure must not depend on the particular choice of order. Order-invariant logics have been studied in database- and finite model-theory in the past.

It is easily seen that the expressive power of order-invariant MSO is greater than that of plain MSO, as with an order we can formalise in MSO that a structure has an even number of elements, a property not definable without an order. In fact, the expressive power of order-invariant MSO is even greater than the expressive power of the extension of MSO with counting quantifiers CMSO [16]. Over restricted classes of structures, order-invariant MSO and CMSO have the same expressive power (see e.g. [5]). This holds true for successor-invariant MSO as well, as an order is definable from a successor relation via MSO. An unpublished result of Gurevich [17] states that the expressive power of order-invariant FO is stronger than that of plain FO. It is known that order-invariant FO collapses to FO on trees [1], [22], and that order-invariant FO is a subset of MSO on graphs of bounded degree and on graphs of bounded tree-width [1].

Weaker than order-invariance is successor-invariance, where the formulas are allowed to use a successor relation but must be invariant under the particular choice of successor relation. It was shown by Rossman [24] that successor-invariant FO is more expressive than FO without access to a successor relation.

In this paper we are interested in the complexity of model-checking order- or successor-invariant FO and MSO. As already plain FO is not tractable on the class of all graphs, order-invariant FO is not tractable on the class of all graphs either. We therefore follow the approach taken in the study of algorithmic meta-theorems and analyse the complexity of order- or successor-invariant FO and MSO on specific classes of structures or graphs.

For monadic second-order logic we are able to show that order-invariant MSO is tractable on essentially the same classes of graphs as plain MSO, i.e. we can increase the expressive power without restricting the tractable cases. To be precise, we show that the model-checking problem for order-invariant MSO on graphs of bounded clique-width is fixed-parameter tractable. Furthermore, combining the result of Courcelle [3] and a result in [2], [21] we get that model-checking for order-invariant MSO on graphs of bounded tree-width is fixed-parameter tractable.

For successor-invariant FO we are able to show that the model-checking problem is fixed-parameter tractable on planar graphs. Using the result of Seese [26] we get the same result for FO on any class of graphs of bounded degree equipped with an arbitrary fixed number of successor relations.

Organisation and Results. We formally define the setting of our work in Section II. In Section III we study the case of ordered structures, i.e. structures equipped with a fixed order or successor relation. The order-invariant case is considered in Section IV and the successor-invariant in Section V.

Acknowledgements. We thank Christoph Dittmann for many hours of fruitful discussions.

II. Model-Checking and Order-Invariance

We consider finite structures over finite signatures that contain only relation and constant symbols. Hence a signature $\tau = \{R_1, \dots, R_k, c_1, \dots, c_s\}$ is a finite set of relation symbols R_i and constant symbols c_i. To each relation symbol $R \in \tau$ we assign an *arity* $ar(R)$. A τ-structure $A = (V(A), R_1(A), \dots, R_k(A), c_1(A), \dots, c_s(A))$ consists of a set $V(A)$, the *universe* of A, for each $R_i \in \tau$ a relation $R_i(A) \subseteq V(A)^{ar(R_i)}$ and for each $c_i \in \tau$ a constant $c_i(A) \in V(A)$. If A is a τ-structure and R is a relation symbol not in τ with associated arity r and $R(A) \subseteq V(A)^r$ is an r-ary relation over A, we write $(A, R(A))$ for the $\tau \cup \{R\}$-structure obtained by extending A with the relation $R(A)$. The *order* $|A|$ of a τ-structure A is $|V(A)|$ and its *size* $\|A\|$ is $|\tau| + |V(A)| + \Sigma_{R\in\tau}|R(A)|$, which corresponds to the size of a representation of A in an appropriate model of computation. We call a structure G of signature $\{E\}$, where E is a binary relation symbol, a digraph and if $E(G)$ is symmetric and irreflexive, we call G a graph. We assume familiarity with the notion of tree-width of a graph, which we denote by $\mathrm{tw}(G)$ (see e.g. [7]). We write $\overline{a}$ for a finite sequence $(a_1, \dots, a_k)$ and usually leave it to the context to determine the length of a sequence. We write $b \in \overline{a}$ instead of $b \in \{a_1, \dots, a_k\}$ and $\overline{a} \subseteq A$ instead of $\overline{a} \in A^k$. The *Gaifman-graph* $G(A)$ of a τ-structure A is the graph with vertex set $V(A)$ and edge set $\{(u, v) : u \neq v$ and there is an $R \in \tau$ and a tuple $\overline{a} \in R(A)$ such that $u, v \in \overline{a}\}$.

We assume familiarity with first-order logic FO and monadic second-order logic MSO (see e.g. [10]) . We write FO(τ) and MSO(τ) for the set of all FO and MSO formulas over vocabulary τ, respectively. If φ is a formula of FO or MSO, we write $|\varphi|$ for the length (of an encoding) of φ. If φ is a sentence of FO(τ) or MSO(τ) and A a τ-structure, we write $A \models \varphi$ if φ is true in A. If $\varphi(\overline{x})$ has free variables $\overline{x}$ and $\overline{a} \subseteq A$ is a tuple of the same length as $\overline{x}$, we write $A \models \varphi(\overline{a})$ if φ is true in A where the free variables $\overline{x}$ are interpreted by the elements of $\overline{a}$ in the obvious way. We write $\varphi_R(A)$ for the relation $R := \{\overline{a} : A \models \varphi(\overline{a})\}$. We call a formula $\varphi(\overline{x})$ over vocabulary $\tau = \sigma \cup \{<\}$ order-invariant if for every σ-structure A, all tuples $\overline{a} \subseteq V(A)$ and all linear orders $<_1, <_2$ over $V(A)$ we have $(A, <_1) \models \varphi(\overline{a}) \Leftrightarrow (A, <_2) \models \varphi(\overline{a})$. Analogously, we call a formula $\varphi(\overline{x})$ over vocabulary $\tau = \sigma \cup \{S\}$ (where S is a binary relation symbol) successor-invariant if for every σ-structure A, all tuples $\overline{a} \subseteq V(A)$ and all successor relations S_1, S_2 over $V(A)$ we have $(A, S_1) \models \varphi(\overline{a}) \Leftrightarrow (A, S_2) \models \varphi(\overline{a})$. We write FO[$<$-*inv*] and MSO[$<$-*inv*] for the set of all order-invariant FO and MSO formulas, respectively, and FO[$+1$-*inv*] and MSO[$+1$-*inv*] for the set of all successor invariant FO and MSO formulas, respectively. We write FO[$<$] and MSO[$<$] for the set of all FO and MSO formulas, respectively, over a signature which contains at least the binary relation symbol $<$ and similarly for FO[$+1$] and MSO[$+1$].

Throughout the paper we study the complexity of order- and successor-invariant logics on restricted classes of structures. As usual in this type of research we focus on classes of graphs. More general structures can be reduced to this case using their Gaifman-graph. In our analysis we will use the framework of *parameterized complexity*, see [8], [11]. A *parameterized problem* is a pair (P, χ), where P is a decision problem and χ is a polynomial time computable function that associates with every instance w of P a positive integer, called the *parameter*.

Definition 2.1. *Let $\mathcal{C}$ be a class of graphs and $\mathcal{L}$ be one of first-order or monadic second-order logic. The* order-invariant model-checking problem MC($\mathcal{L}[<\text{-}inv], \mathcal{C}$) *of the logic $\mathcal{L}$ on the class $\mathcal{C}$ of graphs is defined as the problem*

MC($\mathcal{L}[<\text{-}inv], \mathcal{C}$)	
Input:	$G \in \mathcal{C}, \varphi \in \mathcal{L}(\{E, <\})$ where φ is order-invariant
Parameter:	$\lvert\varphi\rvert$
Problem:	$(G, <) \models \varphi$ for an arbitrary order $<$ on $V(G)$?

Analogously, we define the successor-invariant model-checking problem MC($\mathcal{L}[+1\text{-}inv], \mathcal{C}$) *for $\mathcal{L}$ on $\mathcal{C}$ where instead the formula φ is required to be successor-invariant.*

Finally, we define the ordered model-checking problem

MC($\mathcal{L}[<], \mathcal{C}$) *for $\mathcal{L}$ on $\mathcal{C}$ as*

MC($\mathcal{L}[<], \mathcal{C}$)	
Input:	$G \in \mathcal{C}$, $<$ a linear order of $V(G)$ and $\varphi \in \mathcal{L}(\{E, <\})$
Parameter:	$\lvert\varphi\rvert$
Problem:	$(G, <) \models \varphi$?

and likewise we define the model-checking problem MC($\mathcal{L}[+1], \mathcal{C}$) with successor *for $\mathcal{L}$ on $\mathcal{C}$ as*

MC($\mathcal{L}[+1], \mathcal{C}$)	
Input:	$G \in \mathcal{C}$, S a successor relation on $V(G)$ and $\varphi \in \mathcal{L}(\{E, S\})$
Parameter:	$\lvert\varphi\rvert$
Problem:	$(G, S) \models \varphi$?

The order- and successor-invariant model-checking problems are *fixed-parameter tractable*, or in the complexity class FPT, if there are algorithms that correctly decide on input (G, φ) whether $(G, <) \models \varphi$ for some linear order $<$ or $(G, S) \models \varphi$ for some successor relation S, respectively, in time $f(|\varphi|) \cdot \|G\|^{\mathcal{O}(1)}$, for some computable function $f : \mathbb{N} \to \mathbb{N}$. Similarly for MC($\mathcal{L}[<]$) and MC($\mathcal{L}[+1]$). The model-checking problem for first-order logic is complete for the parameterized complexity class AW[*], which is conjectured and widely believed to strictly contain the class FPT. Thus it is widely believed that model-checking for first-order logic (and thus also for monadic second-order logic) is not fixed-parameter tractable.

III. Model-Checking on Ordered Structures

A. Successor Structures

In this section we analyse the complexity of first-order logic on graphs with an additional successor relation. We show that model-checking becomes intractable even on very restricted classes of graphs. More precisely, we show that on the class of forests with a successor relation first-order model-checking becomes as hard as the general first-order model-checking problem. For this to work, however, we need trees of unbounded rank. And indeed, on graph classes of bounded degree with an additional successor relation, first-order logic can be shown to be fixed-parameter tractable.

We show the intractability result first.

Lemma 3.1. *Let $\mathcal{F}$ be the class of forests. Then* MC(FO$[+1], \mathcal{F}$) *is* AW$[*]$*-hard.*

Proof. We show how to construct for every graph G a forest G' equipped with a successor relation S such that $\|G'\| \in \mathcal{O}(p(\|G\|))$ for some polynomial p and for every sentence $\varphi \in$ FO($\{E\}$) a sentence $\varphi' \in$ FO($\{E, S\}$) with $\|\varphi'\| \in \mathcal{O}(q(\|\varphi\|))$ for some polynomial q such that $G \models \varphi \Leftrightarrow G' \models \varphi'$. The claim follows, as the model-checking problem for FO on graphs is complete for AW$[*]$ and AW$[*]$ is closed under polynomial-time reduction.

Let $G = (V, E)$, where $V = \{v_1, \ldots, v_n\}$. We construct a forest $G' = (V', E', S)$ equipped with a successor relation as follows. G' will consist of n trees $T_1, \ldots, T_n$. The root of tree T_i will represent the vertex v_i of G. Denote the root of tree T_i by v'_i. To ensure that the roots are uniquely definable via first-order logic, we will guarantee that each root has at least three children and no other vertices of degree three exist. We define the successor relation on the v'_i in the natural way, i.e. $S(v'_i, v'_{i+1})$ holds true for all $i < n$. We will then add a child l^i_j to a root v'_i if there is an edge $\{v_i, v_j\}$ in $E(G)$. Thus, for each edge $\{v_i, v_j\}$ we have two vertices l^i_j and l^j_i, l^i_j being the child of v'_i and l^j_i being the child of v'_j. To associate those vertices with an edge, we define l^i_j as a direct predecessor of l^j_i if $i < j$. As the linear order is not definable from the successor relation with first-order logic, we have to add more information to ensure that exactly the described pairs of direct successors are interpreted as edges in the original graph. As an intermediate step we color l^i_j blue if $i < j$. We complete the definition of the successor relation in an arbitrary way. Observe that the successor of every blue vertex is already defined and thus by completing the successor relation we only define successors of colorless vertices. In particular, vertex v_i is connected to v_j in G if v'_i has a child c_i and v'_j has a child c_j such that c_i and c_j are direct successors and the smaller one with respect to the successor relation is blue. As the final step, we remove the color of every blue vertex and attach a vertex instead. Then the vertices which were originally blue are the only ones of degree two and are thus uniquely definable. Again, we may complete the successor relation in an arbitrary way. We now give the formal details.

Let $R := \{v'_1, \ldots, v'_n\}$, $D_i := \{d^i_1, d^i_2, d^i_3\}$, $L_i := \{l^i_j : \{v_i, v_j\} \in E(G)\}$ and $B_i := \{b^i_j : \{v_i, v_j\} \in E(G), i < j\}$ for each $1 \le i \le n$. Let $V' := R \cup \bigcup_{1 \le i \le n} D_i \cup \bigcup_{1 \le i \le n} L_i \cup \bigcup_{1 \le i \le n} B_i$. Define $E' := \{\{v'_i, d^i_s\} : 1 \le i \le n, d^i_s \in D_i\} \cup \{\{v'_i, l^i_j\} : 1 \le i \le n, l^i_j \in L_i\} \cup \{\{l^i_j, b^i_j\} : 1 \le i \le n, l^i_j \in L_i, b^i_j \in B_i\}$. Define $S(v'_i, v'_{i+1})$ to hold if $i < n$ and $S(l^i_j, l^j_i)$ for all i, j with $i < j$. Complete the definition of the successor relation in an arbitrary way. By construction, for every edge $\{v_i, v_j\} \in E(G)$ there are exactly two vertices c_i and c_j with $\{v'_i, c_i\} \in E(G')$ and $\{v'_j, c_j\} \in E(G')$ such that c_i and c_j are direct neighbors in the successor relation and the smaller one of c_i and c_j has degree two and the larger one has degree one. Obviously, $\|G'\| \in O(p(\|G\|))$ for some polynomial p.

We now rewrite any sentence $\varphi \in$ FO($\{E\}$) to a sentence $\varphi' \in$ FO($\{E, S\}$). Let $\varphi \in$ FO($\{E\}$). Replace quantifiers by quantifiers restricted to vertices of degree at least three. Further, replace an atom Exy by a formula $\psi(x, y)$ which states that either x is connected to a vertex x' of degree two and y is connected to a vertex y' of degree one and $S(x', y')$ holds true or vice versa. Then $\|\varphi'\| \in O(q(\|\varphi\|))$ for some polynomial q and $G \models \varphi \Leftrightarrow G' \models \varphi'$ as required. $\square$

As a corollary of the previous lemma we get that MC(FO$[+1], \mathcal{C}$) is AW$[*]$-hard for the class $\mathcal{C}$ of planar graphs and the class of graphs of tree-width 1. However, for the proof to work it is essential that the trees are allowed to have unbounded degree. And indeed, on graph classes of bounded degree, successor-invariant FO model-checking is tractable.

Lemma 3.2. *For every $d \geq 0$ let $\mathcal{D}_d$ be the class of graphs of maximum degree at most d. Then for all $d \geq 0$,* $\mathrm{MC}(\mathrm{FO}[+1], \mathcal{D}_d)$ *is fixed-parameter tractable. In fact, we can allow any (fixed) number of successor relations on top of $\mathcal{D}_d$ and still have tractable first-order model-checking.*

Proof. By a result of Seese [26] the model-checking problem for FO on graphs of bounded degree and also on all structures with Gaifman-graph of bounded degree is fixed-parameter tractable. Adding a successor relation increases the degree of the Gaifman-graph of a structure by at most two. ☐

B. Order Relation

In the previous section we have shown that first-order logic with successor becomes intractable on very simple classes of graphs but is still tractable on classes of bounded degree. As the following lemma implies, first-order logic plus order is intractable even on degree two graphs.

Lemma 3.3. *Let $\mathcal{C}$ be the class of finite linear orders with one unary predicate. Then* $\mathrm{MC}(\mathrm{FO}[+1], \mathcal{C})$ *is* $\mathrm{AW}[*]$*-hard. Consequently, if $\mathcal{S}$ is the class of finite successor structures with one unary predicate, then* $\mathrm{MC}(\mathrm{FO}[<], \mathcal{S})$ *is* $\mathrm{AW}[*]$*-hard.*

Proof. We show how to construct for every graph $\mathcal{G}$ a structure $\mathfrak{A}$ of signature $\{<, S, P\}$, where P is a unary relation symbol such that $\|\mathfrak{A}\| \in \mathcal{O}(p(\|\mathcal{G}\|))$ for some polynomial p and for every sentence $\varphi \in \mathrm{FO}(\{E\})$ a sentence $\varphi' \in \mathrm{FO}(\{<, S, P\})$ with $\|\varphi'\| \in \mathcal{O}(q(\|\varphi\|))$ for some polynomial q such that $\mathcal{G} \models \varphi \Leftrightarrow \mathfrak{A} \models \varphi'$. The claim follows, as the model-checking problem for FO on graphs is complete for $\mathrm{AW}[*]$ and $\mathrm{AW}[*]$ is closed under polynomial-time reduction.

Let $\mathcal{G} = (V, E)$ be a graph, where $V = \{v_1, \dots, v_n\}$. We mimic the construction of Lemma 3.1. We construct a structure $\mathfrak{A}$ of signature $\{<, S, P\}$ as follows. For each $v_i \in V$ we will have an element v'_i. In the construction of Lemma 3.1 the elements v'_i were uniquely definable as the roots of trees. Here we make the v'_i uniquely definable by assigning them the predicate P. We order the v'_i in their natural order, i.e. we let $v'_i < v'_j$ if $i < j$. In the construction of Lemma 3.1 we had elements l^i_j and l^j_i for each edge $\{v_i, v_j\}$, where l^i_j was the child of v'_i and l^j_i was the child of v'_j. We have those elements again and if l^i_j was a child of v'_i in the construction of Lemma 3.1 we define l^i_j to lie between v'_i and v'_{i+1} (or simply after v'_i if v'_{i+1} does not exist). We furthermore order the l^i_j such that $l^i_j < l^i_k$ if $j < k$. Again, we encode an edge by defining l^i_j as the direct predecessor of l^j_i if there is an edge $\{v_i, v_j\} \in E(G)$ and $i < j$. We do not have to mark the smaller vertex by a color in this case because we can directly query whether $v'_i < v'_j$. We complete the definition of the successor relation such that no successors are added from any l^i_j to an l^k_l if $l^i_j < l^k_l$. This is done as follows. At this point of time, all l^i_j with $i < j$ have a successor and all l^j_i with $i < j$ have a predecessor. We define $S(l^j_i, l^i_k)$ to hold true if $i < j$ for the smallest $k > j$ such that l^i_k exists or $S(l^j_i, v'_{i+1})$ to hold true if no such l^i_k exists. After this step, all l^i_j have a successor and for each i only the smallest l^i_j with $i < j$ does not have a predecessor. All v'_i do not have a successor yet and v'_i does not have a predecessor if and only if $i = 1$ or $i > 1$ and between v'_{i-1} and v'_i there is no element with a successor larger than v'_i. We can thus complete the definition of the successor relation by defining $S(v'_i, l^i_j)$ to hold true if j is minimal with $i < j$ such that l^i_j exists and $S(v'_i, v'_{i+1})$ to hold true if no such l^i_j exists.

Then there is an edge $\{v_i, v_j\} \in E$ for $i < j$ if and only if there are elements a, b with $v'_i < a < v'_{i+1}$ if $i < n$ and $v'_j < b$ and $b < v'_{j+1}$ if $j < n$ that are direct successors with respect to S. We now give the formal details.

Let $R := \{v'_i, \dots, v'_n\}$ and $L_i := \{l^i_j : \{v_i, v_j\} \in E(G)\}$. Let $A := R \cup \bigcup_{1 \leq i \leq n} L_i$. Define $P := R$. Define $v'_i < v'_j$ if $i < j$ and $v'_i < l^i_j < l^i_k$ for all i and for all $j < k$. Define $l^i_j < v'_{i+1}$ if $i < n$. Define $S(v'_i, l^i_j)$ to hold true for the smallest possible j such that l^i_j with $i < j$ exists and $S(v'_i, v'_{i+1})$ to hold true if no such l^i_j exists. Define $S(l^i_j, l^j_i)$ to hold true if $i < j$ and $S(l^j_i, l^i_k)$ to hold true if $i < j$ for the smallest $k > j$ such that l^i_k exists or $S(l^j_i, v'_{i+1})$ if no such l^i_k exists. This completes the definition of $\mathfrak{A}$. Obviously, $\|\mathfrak{A}\| \in \mathcal{O}(p(\|G\|))$ for some polynomial p.

We now rewrite any sentence $\varphi \in FO(\{E\})$ to a sentence $\varphi' \in \mathrm{FO}(\{<, S, P\})$ with the required properties. Let $\varphi \in \mathrm{FO}(\{E\})$. Replace quantifiers by quantifiers restricted to elements of P. Further, replace an atom Exy by a formula $\psi(x, y)$ which states that either $x < y$ and there are a and b such that $x < a < x'$ for the unique $x' > x$ with Px' and $y < b < y'$ for the unique $y' > y$ with Py' or no such y' exists and such that b is a direct successor of a, or vice versa. Then $\|\varphi'\| \in \mathcal{O}(q(\|\varphi\|))$ for some polynomial q and $G \models \varphi \Leftrightarrow \mathfrak{A} \models \varphi'$ as required. ☐

IV. ORDER-INVARIANT MSO

In this section we consider order-invariant logics. The most expressive logic studied in the context of algorithmic meta-theorems is monadic second-order logic, the extension of first-order logic by quantification over sets of elements. With respect to graphs, there are two variants of MSO usually considered, one, called MSO_1, where we can quantify over sets of vertices and the other, called MSO_2, where we can quantify over sets of edges as well as sets of vertices. It was shown by Courcelle [3], that MSO_2 is fixed-parameter tractable on every class of graphs of bounded tree-width. Later, Courcelle et al. [4] showed that MSO_1 is fixed-parameter tractable on every class of graphs of bounded clique-width, a concept more general than bounded tree-width. In this section, we show that for both logics we can allow order-invariance without increase in complexity.

We first consider the case of MSO_2. As stated in [21] (see also the exposition in [2]), for every graph G of tree-width k there is a successor relation S on $V(G)$ such that the graph obtained from G by adding the edges in S has tree-width at most $k+5$. From the proof one can derive an algorithm running in time $f(k) \cdot p(G)$, where f is an exponential function and p a fixed polynomial, which, given a graph G of tree-width k as input, computes this successor relation. In combination with Courcelle's theorem, this implies the following result.

Theorem 4.1. $\mathrm{MC}(\mathrm{MSO}[<\text{-}inv], \mathcal{C})$ *is fixed-parameter tractable on any class* $\mathcal{C}$ *of bounded tree-width.*

In fact, $\mathrm{MC}(\mathrm{MSO}[<\text{-}inv])$ is fixed-parameter tractable with parameter $|\varphi| + \mathrm{tw}(G)$, where $\mathrm{tw}(G)$ is the tree-width of a graph G. We prove next that also for MSO_1 we can allow order-invariance without loss of tractability.

Theorem 4.2. $\mathrm{MC}(\mathrm{MSO}[<\text{-}inv], \mathcal{C})$ *is fixed-parameter tractable on every class* $\mathcal{C}$ *of graphs of bounded clique-width.*

We first review the definition of clique-width. For the rest of this section we fix a relational signature σ in which every relation symbol has arity at most 2.

Definition 4.3 (**$\boldsymbol{\sigma}$-clique-expression of width $\boldsymbol{k}$**)**.** *Let $k \in \mathbb{N}$ be fixed. A σ-clique-expression of width k is a pair (T, λ), where T is a directed tree and $\lambda : V(T) \to \{\mathbf{1}, \dots, \mathbf{k}, \oplus, \mathrm{edge}_{R,i \to j}, \mathrm{rename}_{i \to j} : 1 \leq i, j \leq k, R \in \sigma\}$, such that for all $t \in V(T)$: if $\lambda(t) \in \{\mathbf{1}, \dots, \mathbf{k}\}$ then t is a leaf of T, if $\lambda(t) = \oplus$ then t has exactly two successors and in all other cases t has exactly one successor.*

Definition 4.4. *Let (T, λ) be a σ-clique-expression of width k. With every $t \in V(T)$ we associate a σ-structure $G(t)$ in which vertices are coloured by colours $\mathbf{1}, \dots, \mathbf{k}$ as follows.*

- *If t is a leaf, then $G(t)$ consists of one element coloured by $\lambda(t)$.*
- *If $\lambda(t) = \oplus$ and t has successors t_1, t_2 then $G(t)$ is the disjoint union $G(t_1) \dot\cup G(t_2)$.*
- *If $\lambda(t) = \mathrm{edge}_{R,i \to j}$ and t_1 is the successor of t, then $G(t)$ is the structure obtained from $G(t_1)$ by adding to the relation $R(G(t))$ all pairs (u, v) such that u has colour i and v has colour j.*
- *If $\lambda(t) = \mathrm{rename}_{i \to j}$ and t_1 is the successor of t then $G(t)$ is the structure obtained from $G(t_1)$ by changing the colour of all vertices v which have colour i in $G(t_1)$ to colour j in $G(t)$.*

*The σ-*structure generated by *(T, λ) is the structure $G(r)$, where r is the root of T, from which we remove all colours $\{\mathbf{1}, \dots, \mathbf{k}\}$. Finally, the* clique-width *of a σ-structure G is the minimal width of a clique-expression generating G.*

Combining results from [18] and [23] yields the following well-known result. In the following, we view graphs as $\{E\}$-structures in the obvious way.

Theorem 4.5. *There are computable functions $f, g : \mathbb{N} \to \mathbb{N}$ and an algorithm which, given a graph G of clique-width at most k as input, computes a clique-expression of order $\leq g(k)$ in time $f(k) \cdot |G|^3$.*

Here, the function $g(k)$ can be taken as $g(k) = 2^{k+1}$. The following result is due to Courcelle et al. [4].

Theorem 4.6. $\mathrm{MC}(\mathrm{MSO}_1, \mathcal{C})$ *is fixed-parameter tractable on any class* $\mathcal{C}$ *of graphs of bounded clique-width.*

In fact, the result applies to any σ-structure of bounded clique-width provided that the clique-expression generating the structure is given. The next lemma is the main technical ingredient for the theorem above.

Lemma 4.7. *There is an algorithm which, on input a graph G of clique-width at most k, computes a linear order $<$ on $V(G)$ and a clique-expression of width at most $2g(k)$ generating the structure $(G, <)$, where g is the function defined in Lemma 4.5.*

Proof. Let G and k be given. Using Theorem 4.5, we first compute an $\{E\}$-clique-expression (T, λ) of width at most $g(k)$ generating G. Let r be the root of T. For every node $t \in V(T)$ we fix an ordering of its successors. Let $\prec$ be the partial order on $V(T)$ induced by this.

Let $t \in V(T)$ be a node and let $s \neq t$ be the first node on the path P from t to r with $\lambda(s) = \oplus$, if it exists. Let t_1, t_2 be the successors of s with $t_1 \prec t_2$. We call t a *left node* if $t_1 \in V(P)$ and a *right node* otherwise. If there is no node labelled $\oplus$ strictly above t then we call t a left node as well.

For every $t \in V(T)$ let T_t be the subtree of T with root t and let $\lambda_{|T_t}$ be the restriction of λ to the subtree T_t. We recursively define a transformation $\rho(T_t, \lambda_{|T_t})$ on the subtrees of T defined as follows. Intuitively, we will produce a new clique-expression (T', λ') over the signature $\{E, <\}$ using colours $\{(i, l), (i, r) : 1 \leq i \leq k\}$. Essentially, the new clique-expression will generate the same graph as (T, λ) but so that if t is a node in T and T_t generates the graph G_t, then T' contains a node t' generating an ordered version $G'_t := (G_t, <)$ of G_t so that if $v \in V(G_t)$ has colour i then, in G'_t, v has colour (i, l) if t is a left node and (i, r) if t is a right node. Hence, whenever in T we take the disjoint union of G_t and G_s and $t \prec s$ then we can define the ordering on $G'_t \dot\cup G'_s$ by adding all edges from nodes in G'_t to G'_s, i.e. all edges from vertices coloured (i, l) to (j, r) for all pairs i, j. Formally, the transformation is defined as follows.

- If $t \in V(T)$ is a leaf, then $\rho(t) := (T', \lambda')$, where T' consists only of t and $\lambda'(t) := (\lambda(t), l)$ if t is a left node and $\lambda'(t) := (\lambda(t), r)$ if t is a right node.
- Suppose $\lambda(t) = \mathrm{rename}_{i \to j}$ and let s be the successor of t. Then $\rho(T_t, \lambda_{|T_t}) := (T', \lambda')$, where T' is a tree defined as follows. Let $(T'', \lambda'') := \rho(T_s, \lambda_{|T_s})$ and let r'' be the root of T''. Then T' is obtained from T'' by adding a new root r' with successor r''. We define $\lambda'(r') := \mathrm{rename}_{(i,l) \to (j,l)}$ and $\lambda'(r'') := \mathrm{rename}_{(i,r) \to (j,r)}$ and $\lambda'(u) = \lambda''(u)$ for all other $u \in V(T'')$.
- Suppose $\lambda(t) = \mathrm{edge}_{E,i \to j}$ and let s be the successor of t. Then $\rho(T_t, \lambda_{|T_t}) := (T', \lambda')$ where T' is a tree defined as follows. Let $(T'', \lambda'') := \rho(T_s, \lambda_{|T_s})$ and let r'' be the root of T''. Then T' is obtained from T'' by adding a path (v_1, v_2, v_3) of length 2 and making r'' a successor of v_3. We define $\lambda'(v_1) := \mathrm{edge}_{E,(i,l) \to (j,l)}$, $\lambda'(v_2) := \mathrm{edge}_{E,(i,r) \to (j,l)}$ $\lambda'(v_3) := \mathrm{edge}_{E,(i,l) \to (j,r)}$ $\lambda'(r'') := \mathrm{edge}_{E,(i,r) \to (j,r)}$ and $\lambda'(u) = \lambda''(u)$ for all

other $u \in V(T'')$.

- Finally, suppose $\lambda(t) = \oplus$ and let t_1, t_2 be the successors of t such that $t_1 \prec t_2$. Then $\rho(T_t, \lambda_{|T_t}) := (T', \lambda')$ where T' is a tree defined as follows. For $i = 1, 2$ let $(T_i, \lambda_i) = \rho(T_{t_i}, \lambda_{|T_{t_i}})$ and let r_i be the root of T_i. T' consists of the union of T_1, T_2 and additional vertices $v_1, \dots, v_k, v_e, v_o$, edges (v_i, v_{i+1}) for all $1 \leq i < k$, (v_k, v_e), (v_e, v_o) and (v_o, r_i), for $i = 1, 2$. For every node $s \in V(T_i)$ we define $\lambda'(s) := \lambda_i(s)$, $i = 1, 2$. Furthermore, we define $\lambda(v_o) := \oplus$ and $\lambda'(v_e) := \text{edge}_{<, l \to r}$. Finally, if t is a left node then we define $\lambda(v_i) := \text{rename}_{(i,r) \to (i,l)}$, for all $i \leq k$, and if t is a right node then we define $\lambda(v_i) := \text{rename}_{(i,l) \to (i,r)}$.

Now, it is easily seen that (T', λ') generates an $\{E, <\}$-structure $(V, E, <)$ where (V, E) is the graph generated by (T, λ) and $<$ is a linear order on V. The width of (T', λ') is twice the width of (T, λ) and hence at most $2g(k)$. □

We are now ready to prove Theorem 4.2.

Proof of Theorem 4.2. Let $\mathcal{C}$ be a class of graphs of tree-width at most k. On input $G \in \mathcal{C}$ and $\varphi \in \text{MSO}_1(\{E, <\})$, we apply Lemma 4.7 to obtain a clique-expression (T, λ) of width $2g(k)$ generating an ordered copy $(G, <)$ of G, where g is the function from Theorem 4.5. We can now apply Theorem 4.6 to decide whether $(G, <) \models \varphi$ in time $f(2g(k)) \cdot p(|G|)$, where f is a computable function and p a polynomial. As φ is order-invariant, if $(G, <) \models \varphi$ then $(G, <') \models \varphi$ for any linear order $<'$ on G. Hence, if $(G, <) \models \varphi$ we can return "yes" and otherwise reject the input. This concludes the proof. □

It is worth pointing out the following feature of the model-checking algorithms established in Theorem 4.1 and Theorem 4.2: instead of designing new model-checking algorithms, we reduce the verification of order-invariant MSO on classes of small tree- or clique-width to the standard model-checking algorithms for MSO on classes of (slightly larger) tree- and clique-width, respectively. The advantage of this approach is that we can reuse existing results on MSO on such classes of graphs. For instance, in [20] the authors report on a practical implementation of Courcelle's theorem, i.e. on the implementation of a model-checker for MSO_2 on graph classes of bounded tree-width, and obtain astonishing performance results in practical tests. Our technique allows us to reuse this implementation so that with minimal effort it is possible to implement our algorithm on top of the work in [20].

Furthermore, in [12] it is shown that on graph classes $\mathcal{C}$ of bounded tree-width, the set of all satisfying assignments of a given MSO formula $\varphi(X)$ with free variables in a graph $G \in \mathcal{C}$ can be computed in time linear in the size of the output and the size of G. Again we can use the same algorithm to obtain the same result for order-invariant MSO.

V. Successor-Invariant First-Order Logic

In this section we study successor-invariant first-order logic FO$[+1\text{-}inv]$. Recall that successor-invariant FO means that we are given a graph G as input and a formula φ which may use a successor relation but its truth in G must be invariant under the exact choice of the successor relation. Hence, to verify whether φ is true in G it suffices to compute one particular successor relation S and decide $(G, S) \models \varphi$.

In graph theoretical terms, a successor relation corresponds to a Hamiltonian path in a graph G. Hence, if $\sigma := \{E, F\}$ is a signature with two binary relation symbols and $\mathcal{C}$ is a class of graphs such that

1) first-order model-checking is fixed-parameter tractable on the class of all σ-structures whose Gaifman-graphs are in $\mathcal{C}$, and
2) all graphs in $G \in \mathcal{C}$ are Hamiltonian and, furthermore, a Hamiltonian path in G can be computed in polynomial time,

then we can conclude that MC(FO$[+1\text{-}inv]$, $\mathcal{C}$) is fixed-parameter tractable as follows: on input $G \in \mathcal{C}$ we first compute a Hamiltonian path P and colour its edges by a special colour F to obtain a σ-structure with Gaifman-graph G. We can then use the model-checking algorithm on $\mathcal{C}$ to decide whether $(G, F) \models \varphi$.

One example of this is the class of 4-connected planar graphs. Tutte and later Thomassen (see [28] and references therein) showed that any 4-connected planar graph contains a Hamiltonian path which can be computed in polynomial time. Together with the result in [14] that MC(FO, Planar) is fixed-parameter tractable, where Planar is the class of planar graphs, we immediately get the following result.

Theorem 5.1. MC(FO$[+1\text{-}inv]$, $\mathcal{C}$) *is fixed-parameter tractable on the class of* 4*-connected planar graphs.*

There are some other classes of graphs with similar properties, but not every planar graph is Hamiltonian and it is not always possible to obtain a Hamiltonian planar graph from an arbitrary planar graph by adding edges or vertices in a way that would be useful for our purposes. See e.g. [19] and references therein. Therefore, to show that successor-invariant FO is fixed-parameter tractable on planar graphs, the main result of this section, we have to use different techniques.

Theorem 5.2. MC(FO$[+1\text{-}inv]$, Planar) *is fixed-parameter tractable.*

The approach we take to prove the theorem is summarised in the following lemma.

Lemma 5.3. *Let $\mathcal{C}$ be a class of graphs, σ a signature and $\mathcal{D}$ a class of σ-structures with the following properties.*

1) *There is a polynomial-time algorithm which, on input $G \in \mathcal{C}$, computes a σ-structure $H \in \mathcal{D}$ and formulas $\varphi_V(x), \varphi_E(x, y), \varphi_S(x, y) \in$ FO(σ) such that $G' := (\varphi_V(H), \varphi_E(H))$ is isomorphic to G and $\varphi_S(H)$ defines the edge set of a successor relation on $V(G')$.*
2) MC(FO, $\mathcal{D}$) *is fixed-parameter tractable.*

Then MC(FO$[+1\text{-}inv]$, $\mathcal{C}$) *is fixed-parameter tractable.*

Proof. Given $G \in \mathcal{C}$ and a successor-invariant sentence $\varphi \in$ FO(E, S), we can decide $G \models \varphi$ as follows. We apply our algorithm to obtain the structure H and the formulas $\varphi_V, \varphi_E, \varphi_S$ as above. Using standard techniques of first-order interpretations, we can transform the formula φ into a formula $\varphi' \in \mathrm{FO}(\sigma)$ such that $H \models \varphi'$ if and only if for some, and hence every, successor relation S of G, $(G, S) \models \varphi$. Essentially, we relativise all quantifiers in φ to φ_V, replace atoms $E(x, y)$ by $\varphi_E(x, y)$ and atoms $S(x, y)$ by $\varphi_S(x, y)$. We can then apply the model-checking algorithm for first-order logic on $\mathcal{D}$ to decide whether $H \models \varphi'$. □

In the rest of this section we show how this lemma can be used to prove Theorem 5.2. The class $\mathcal{D}$ we aim for will be a class of σ-structures of bounded local tree-width, for a suitable signature σ. Essentially, we will add to a planar graph G some copies of G with slight modifications, called *circular extensions* below, and show how to define Hamiltonian paths in these copies. This path will then be projected to G by first-order formulas.

A. Preliminaries

Let G be a graph. For $v \in V(G)$ we write $N^G(v)$ for the set of neighbours of v. For a set $S \subseteq V(G)$ we write $N^G(S) := \bigcup_{v \in S} N^G(v)$. We define $N_r^G(v)$ as the set of vertices of distance at most r from v and analogously $N_r^G(S)$. Note that $N_1^G(v) := N(v) \cup \{v\}$. We usually drop the index ^{G} when G is clear from the context. If $U \subseteq V(G)$, we write $G[U]$ for the subgraph of G induced by U.

Definition 5.4. *Let G and H be graphs. The* lexicographic product *$G \bullet H$ of G and H is the graph with vertex set $V := V(G) \times V(H)$ and edge set $E := \{\{(v, v'), (u, u')\} : \{v, u\} \in E(G)$, $v', u' \in V(H)$ or $v = u$ and $\{v', u'\} \in E(H)\}$. If $\mathcal{C}$ is a class of graphs and H is a graph, we define $\mathcal{C} \bullet H := \{G \bullet H : G \in \mathcal{C}\}$.*

B. Cops and Robber Games

The *Cops and Robber game* on a graph is a game-theoretical characterisation of the tree-width. k cops try to catch a robber who may run along paths in the graph. While the robber is confined to moving along paths in the graph, the cops may move to any vertex at any time.

Definition 5.5 (Cops and Robber Game)**.** *Given a graph G, the* k-cops and robber *game on G is played between two players, the* cop *and the* robber *player, as follows:*

At the beginning, the cop player chooses $X_0 \subseteq V(G)$ with $|X_0| \leq k$, and the robber player chooses a vertex r_0 of $V \setminus X_0$, giving position (X_0, r_0). From position (X_i, r_i), the cop player chooses $X_{i+1} \subseteq V(G)$ with $|X_{i+1}| \leq k$, and the robber player chooses a vertex r_{i+1} of $V \setminus X_{i+1}$ such that there is a path from r_i to r_{i+1} which does not pass through a vertex in $X_i \cap X_{i+1}$. If no such vertex exists, then the robber player loses.

A play *in the game is a (finite or infinite) sequence $\pi := (X_0, r_0)(X_1, r_1) \ldots$ of positions such that the transition from (X_i, r_i) to (X_{i+1}, r_{r+1}) is a valid move by the rules above and such that the play is finite if and only if $r_n \in X_n$ for the final position (X_n, r_n). A play is winning for the robber player if and only if it is infinite.*

The minimal number of cops required to catch the robber is called the game width of a graph and it was shown by Seymour and Thomas [27] that the game width of a graph equals its tree-width plus 1.

Theorem 5.6. *The tree-width of a graph G is equal to the mininum number of cops required to capture a robber on G minus* 1.

C. Local Tree-Width

Definition 5.7. *The* local tree-width *of a graph G is the function $\mathrm{ltw}_G : \mathbb{N} \to \mathbb{N}$ with $\mathrm{ltw}_G(r) := \max\{\mathrm{tw}(G[N_r^G(v)]) : v \in V(G)\}$. A class $\mathcal{C}$ of graphs has* bounded local tree-width *if there is a computable non-decreasing function $f : \mathbb{N} \to \mathbb{N}$ such that $\mathrm{ltw}_G(r) \leq f(r)$ for all $G \in \mathcal{C}$ and $r \geq 0$.*

Lemma 5.8. *Let G be a graph, $k \geq 0$ and $H := G \bullet K_k$. Then $\mathrm{ltw}_H(r) \leq k \cdot (\mathrm{ltw}_G(r) + 1) - 1$ for all $r > 0$.*

Proof. Let $r > 0$ and let $N_H := N_r^H(v)$ be the r-neighbourhood of a vertex $v := (u, x) \in V(H)$. Let $N_G := N_r^G(u)$. Then $N_H = \{(x, y) : x \in N_G\}$. Let $w := \mathrm{tw}(G[N_G]) \leq \mathrm{ltw}_G(r)$. By Theorem 5.6, $w + 1$ cops have a winning strategy against the robber on $G[N_G]$. But then $k(w + 1)$ cops have a winning strategy against the robber in $H[N_H]$ as follows. Whenever in the play on G the cop player places a cop on a vertex $v \in V(G)$, the cop player places k cops on all $(v, x) \in V(H)$. It is easy to see that this is indeed a winning strategy. Hence $\mathrm{tw}(H[N_H]) \leq k(w + 1) - 1$. □

Corollary 5.9. *Let $\mathcal{C}$ be a class of graphs of bounded local tree-width and $k \geq 0$. Then $\mathcal{C} \bullet K_k$ has bounded local tree-width.*

We also need the following result by Frick and Grohe [14].

Theorem 5.10. $\mathrm{MC}(\mathrm{FO}, \mathcal{C})$ *is fixed-parameter tractable on any class $\mathcal{C}$ of bounded local tree-width.*

D. The Circular Extension of a Planar Graph

For the following exposition we fix a planar graph G and a plane embedding Γ of G. For a vertex $v \in V(G)$, a *cyclic ordering of v respecting* Γ is an ordering (bijection) $\rho : \{0, \ldots, l-1\} \to N(v)$, where $l := |N(v)|$, of the neighbours of v obtained by listing $N(v)$ in clockwise order starting with some $v_0 \in N(v)$. For each $v \in V(G)$ we fix a cyclic ordering ρ_v of v respecting Γ.

We now define a graph $\mathfrak{C}(G)$ obtained from G by the following operations. We first subdivide every edge twice. We then connect the new neighbours of a vertex v in the order specified by ρ_v to form a cycle around v respecting the plane embedding of G.

Definition 5.11. *The* circular extension *$\mathfrak{C}(G)$ of G is the graph with vertex set $V(G) \cup \{u_{v,e} : e = \{v, v'\} \in E(G)\}$ and edge set $\big\{\{v, u_{v,e}\} : e = \{v, v'\} \in E(G)\big\} \cup \big\{\{u_{v,e}, u_{v',e}\} : e = \{v, v'\} \in E(G)\big\} \cup E'$ where E' is defined as follows. Let $v \in V(G)$ with $l := |N(v)| >$ and let $e_0 = \{v, \rho_v(0)\}, \ldots, e_{l-1} = \{v, \rho_v(l-1)\}$ be the edges of G incident to v. Then E' contains*

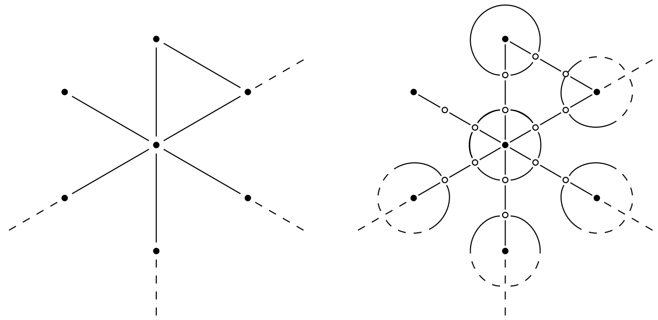

Figure 1. A planar graph G and its circular extension $\mathfrak{C}(G)$. The original vertices are black and the circular vertices are white.

all edges $\{u_{v,e_i}, u_{v,e_{i+1 \bmod l}}\}$. *We call the vertices of the set* $V(G)$ original vertices *and the vertices of the set* $\{u_{v,e} : e = \{v, v'\} \in E(G)\}$ circular vertices.

The concept of circular extensions is illustrated in Figure 1. The figure shows a (part of a) planar graph and its circular extension, where the original vertices are black and the circular vertices are white. It is easily seen that if G is planar, then so is $\mathfrak{C}(G)$. By the following result of Robertson and Seymour [25], the circular extension of any planar graph has bounded local tree-width.

Lemma 5.12. *The class of planar graphs has bounded local tree-width. More precisely, every planar graph of radius* r *has tree-width* $\leq 3r + 1$.

Corollary 5.13. *The class* $\mathcal{C} := \{\mathfrak{C}(G) : G$ *is a planar graph* $\}$ *has bounded local tree-width.*

E. Definable Hamiltonian Cycles

Let G be a connected planar graph and let T be a spanning tree of G with root t and consider the graph $\mathfrak{C}(T) \bullet K_2$. Recall that $\mathfrak{C}(T) \bullet K_2$ consists of two copies of $\mathfrak{C}(T)$ and some edges between them. We will refer to these copies as $\mathfrak{C}(T)$ and $\mathfrak{C}(T)'$ respectively and call $\mathfrak{C}(T)$ the *principle copy* of $\mathfrak{C}(T) \bullet K_2$. For a vertex $v \in V(\mathfrak{C}(T))$ we denote by v' the corresponding vertex of $\mathfrak{C}(T)'$. The aim of this section is to show that $\mathfrak{C}(T) \bullet K_2$ contains a Hamiltonian path H such that the original vertices of the principal copy $\mathfrak{C}(T)$ have short distance along H. This will allow us to define a successor relation on G in first-order logic. To find such a Hamiltonian path, we follow a depth-first, left-to-right traversal of T, walking alternatingly on vertices of $\mathfrak{C}(T)$ and $\mathfrak{C}(T)'$. Instead of using a vertex v in T multiple times, we use the circular edges between the successors of v, which exist in $\mathfrak{C}(T)$ (this is why we use a left-to-right traversal of T – the additional edges exist only between neighbouring successors).

Lemma 5.14. *Let* T *be a tree with root* t. *Then* $\mathfrak{C}(T) \bullet K_2$ *contains a Hamiltonian path* $H := (t = h_1, h_2, \dots, h_n = t')$ *such that every subpath of* H *of length at least* 7 *contains an original vertex of the principle copy* $\mathfrak{C}(T)$. *Furthermore, such a path can be computed in polynomial time.*

Proof. We show the following stronger property by induction on the height h of T. $\mathfrak{C}(T) \bullet K_2$ contains a Hamiltonian path $H := (t = h_1, h_2, \dots, h_n = t')$ such that $\{h_i : \max\{1, n-3\} \leq i \leq n\}$ contains an original vertex of the principle copy $\mathfrak{C}(T)$ and every subpath of H of length at least 7 contains an original vertex of the principle copy $\mathfrak{C}(T)$.

For $h = 0$, T contains only an isolated vertex t and $H = (t, t')$ is a Hamiltonian path in $\mathfrak{C}(T) \bullet K_2 = (\{t, t'\}, \{\{t, t'\}\})$ with the above properties.

For $h > 0$, let $t_1, \dots, t_d$ be the successors of t and let $T_1, \dots, T_d$ be the subtrees of T rooted at $t_1, \dots, t_d$. Each T_i has height $< h$ and by induction hypothesis, $\mathfrak{C}(T_i) \bullet K_2$ has a Hamiltonian path $H_i = (t_i = h_{i_1}, \dots, h_{i_{n_i}} = t'_i)$ with the desired properties. For a path P write P^{rev} for the path P in reverse direction.

Then

$$\begin{aligned} H := (t',\, & u_{t,\{t,t_1\}}, (u_{t_1,\{t_1,t\}})', H_1, u_{t_1,\{t_1,t\}}, (u_{t,\{t,t_1\}})', \\ & u_{t,\{t,t_2\}}, (u_{t_2,\{t_2,t\}})', H_2, u_{t_2,\{t_2,t\}}, (u_{t,\{t,t_2\}})', \\ & \dots, \\ & u_{t,\{t,t_d\}}, (u_{t_d,\{t_d,t\}})', H_d, u_{t_d,\{t_d,t\}}, (u_{t,\{t,t_d\}})', \\ & t)^{\mathsf{rev}} \end{aligned}$$

is a Hamiltonian path in $\mathfrak{C}(T) \bullet K_2$ with the desired properties. It is clear that H can be computed in polynomial time. $\square$

We are now ready to show that for any planar graph G we can construct a σ-structure H which satisfies the requirements of Lemma 5.3.

Definition 5.15. *Let* G *be a connected graph. Let* $\sigma := \{E, F, R, B\}$, *where* E, F *are binary and* R, B *are unary relation symbols. The* Hamiltonian extension $\mathfrak{H}(G)$ *of* G *is a* σ-*structure* $H := \mathfrak{H}(G)$ *defined as follows. Let* T *be a spanning tree of* G. *Then* H *consists of the disjoint union of* G *and* $T' := \mathfrak{C}(T) \bullet K_2$. *Furthermore, if* C *is the principle copy of* $\mathfrak{C}(T)$ *in* T', *then we add an edge between every* $v \in V(G)$ *and its copy in* C. *All edges defined so far together define the relation* $E(H)$. *Let* P *be the Hamiltonian path in* T' *defined in the proof of Lemma 5.14. We define* $F(H)$ *as the directed edges of an orientation* P *so that* F *is a directed Hamiltonian path. Finally, we colour the components of* H *by defining* $R(H) := V(G)$ *and* $B(H) := V(C)$.

Note that $\mathfrak{H}(G)$ depends on T. However, our results below do not depend on the particular choice of T and we therefore write $\mathfrak{H}(G)$ for any choice of T.

We now show how to find the successor of a vertex $v \in V(G)$. Let $v' \in V(C)$ be the copy of v in C. Let u' be the next vertex on the directed Hamiltonian path $F(\mathfrak{H}(G))$ in $V(C)$ such that u' is connected to a vertex u in $V(G)$, i.e. u' corresponds to an original vertex of G. We then define $u \in V(G)$ as the successor of v. As F is a Hamiltonian path of T', this construction defines indeed a successor relation, on

$V(G)$. By construction, u' has distance at most 7 from v' in F and thus it is easily seen that S is first-order definable by a formula $\varphi_S(x, y) \in \mathrm{FO}(\sigma)$, as the colors R, B can be used to distinguish between the copies of $V(G)$.

Lemma 5.16. *The class $\{\mathfrak{H}(G) : G$ is a planar connected graph$\}$ has bounded local tree-width.*

Proof. We claim that every r-neighbourhood in the Gaifman-graph of $\mathfrak{H}(G)$ can be found as a minor of a $3r + 2$-neighbourhood of the Gaifman-graph of $\mathfrak{C}(G) \bullet K_3$, which immediately implies the statement of the lemma.

We identify vertices of the original copy of G in $\mathfrak{H}(G)$ with their corresponding original center vertices of the first copy of $\mathfrak{C}(G)$ in $\mathfrak{C}(G) \bullet K_3$ and all other vertices of $\mathfrak{H}(G)$ with their corresponding vertices in $\mathfrak{C}(G) \bullet K_3$ in the obvious way. We first show that for every path of length r in $\mathfrak{H}(G)$ between vertices $u, v \in V(\mathfrak{H}(G))$ we can find a corresponding walk of length at most $3r$ in $\mathfrak{C}(G) \bullet K_3$ between u and v. Let P be a path in $\mathfrak{H}(G)$ and let e be an edge of P. If e is an edge of the original copy of G in $\mathfrak{H}(G)$, e can be replaced by a path of length 3 in the first copy of $\mathfrak{C}(G)$ in $\mathfrak{C}(G) \bullet K_3$, using subdivided edges. If e is an edge between different copies in $\mathfrak{H}(G)$ or a non-circular edge of $\mathfrak{C}(G)$ then e is also present in $\mathfrak{C}(G) \bullet K_3$ and thus does not have to be replaced. If e is a circular edge of $\mathfrak{C}(T)$ which is not present in the respective copy $\mathfrak{C}(G)$ of $\mathfrak{C}(G) \bullet K_3$, this edge can be replaced by a path of length 2 in $\mathfrak{C}(G)$ by walking along the original center to which e belongs.

We now show how to find the r-neighbourhood of a vertex u as a minor of the $3r + 2$-neighbourhood of u in $\mathfrak{C}(G) \bullet\ K_3$. From the above considerations it follows that all required vertices are present in the $3r$-neighbourhood of u in $\mathfrak{C}(G) \bullet\ K_3$. What remains to be shown is how to find the circular edges of $\mathfrak{C}(T)$ in $\mathfrak{C}(G) \bullet K_3$. Let $v \in V(G)$ be a vertex and let $N_v := (v_0, \ldots, v_l) \subseteq N^T(v)$ be the set of neighbours of v in T listed in the order given by the cyclic ordering ρ_v fixed in the construction of $\mathfrak{C}(G)$. If for some $i \leq l$, v_i and $v_{i+1 \bmod (l+1)}$ are not also neighbours in the ordering ρ_v, then in $\mathfrak{C}(G)$ there is a path between any v_i and $v_{i+1 \bmod l+1}$ consisting of vertices used to subdivide original edges and their connecting new edges. This is illustrated in Figure 2. The figure shows a spanning tree of the graph displayed in Figure 1 and the corresponding circular extension. Note that the vertices b and c are neighbours of a in the graph G but they are no longer neighbours of a in the spanning tree T. The edge in the circular extension marked by a thick line in Figure 2 therefore corresponds to a path between neighbours v_i and $v_{i+1 \bmod l+1}$ of a in the circular extension of the graph G.

If both v_i and $v_{i+1 \bmod l+1}$ lie in the r-neighbourhood of some vertex u in $\mathfrak{H}(G)$ then every vertex on the path between v_i and $v_{i+1 \bmod l+1}$ lies in the $3r + 2$-neighbourhood of u in $\mathfrak{C}(G) \bullet K_3$. This again follows from the fact that every circular edge of $\mathfrak{C}(T)$ can be replaced by a path length 2 in $\mathfrak{C}(G)$. We can thus contract the path between v_i and $v_{i+1 \bmod l+1}$ and delete all edges to neighbours in $\mathfrak{C}(G)$ which are not

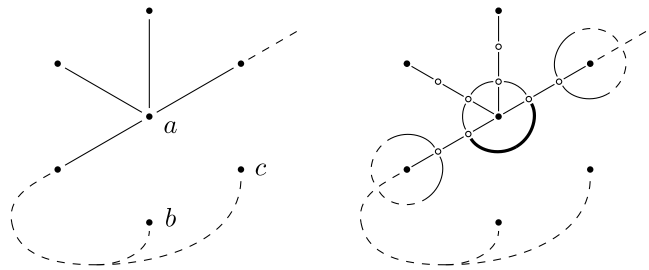

Figure 2. A spanning tree T of the graph G of Figure 1 and its circular extension $\mathfrak{C}(T)$. Vertices b and c are not connected to T via a and thus $\mathfrak{C}(T)$ contains edges that are not edges of $\mathfrak{C}(G)$.

neighbours in $\mathfrak{C}(T)$. This concludes the claim. □

We now show that Lemma 5.16 does not only hold for connected planar graphs but for planar graphs in general.

Lemma 5.17. *Let $\sigma := \{E, F, R, B\}$, where E, F are binary and R, B are unary relation symbols. There is a polynomial time algorithm which, given a planar graph G as input, computes a σ-structure $\mathfrak{H}(G)$ and a formula $\varphi_S(x, y) \in \mathrm{FO}(\sigma)$ such that the class $\{\mathfrak{H}(G) : G$ a planar graph$\}$ has bounded local tree-width and $\varphi_S(\mathfrak{H}(G))$ defines a successor relation on G.*

Proof. Construct the Hamiltonian extension $\mathfrak{H}(C)$ for every component C of G and connect the resulting Hamiltonian paths in some order. The local tree-width of the resulting structure increases by at most 1. The formula defining the successor relation is modified accordingly. □

Combining Lemma 5.17 and Theorem 5.10 we can now apply the method established in Lemma 5.3 to obtain the main theorem of the section, Theorem 5.2.

VI. CONCLUSION

We have shown that order-invariant MSO and MSO are tractable on essentially the same classes of structures, i.e. there is no price on tractability we have to pay for the increase in expressive power order-invariance yields. For FO we are not yet able to match the best tractability bounds of plain FO but we obtain tractability for successor-invariant FO on the class of planar graphs. Our construction does not extend beyond planar graphs. In particular, it is not difficult to construct (non-planar) simple circular graphs of order k but unbounded tree-width. We expect that the methods developed here can be extended to graphs of bounded genus, though. It is conceivable that the result can be extended to classes of graphs of bounded expansion. However, different techniques are required for this case and we leave this for future research.

REFERENCES

[1] M. Benedikt and L. Segoufin. Towards a characterization of order-invariant queries over tame structures. In *CSL*, pages 276–291, 2005.
[2] Y. Chen and J. Flum. On the Ordered Conjecture. In *Logic in Computer Science (LICS)*, 2012.

[3] B. Courcelle. Graph rewriting: An algebraic and logic approach. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume 2, pages 194 – 242. Elsevier, 1990.
[4] B. Courcelle, J. Makowsky, and U. Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory of Computing Systems*, 33(2):125–150, 2000.
[5] B. Courcelle. The monadic second-order logic of graphs X: Linear orderings. *Theor. Comput. Sci.*, 160(1&2):87–143, 1996.
[6] A. Dawar, M. Grohe, and S. Kreutzer. Locally excluding a minor. In *Logic in Computer Science (LICS)*, pages 270–279, 2007.
[7] R. Diestel. *Graph Theory*. Springer-Verlag, 3rd edition, 2005.
[8] R. Downey and M. Fellows. *Parameterized Complexity*. Springer, 1998.
[9] Z. Dvořák, D. Král, and R. Thomas. Deciding first-order properties for sparse graphs. In *Foundations of Computer Science (FOCS)*, 2010.
[10] H.-D. Ebbinghaus, J. Flum, and W. Thomas. *Mathematical Logic*. Springer, 2nd edition, 1994.
[11] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006. ISBN 3-54-029952-1.
[12] J. Flum, M. Frick and M. Grohe. Query evaluation via tree-decompositions. *Journal of the ACM*, 49(6):716–752, 2002.
[13] J. Flum and M. Grohe. Fixed-parameter tractability, definability, and model checking. *SIAM Journal on Computing*, 31:113 – 145, 2001.
[14] M. Frick and M. Grohe. Deciding first-order properties of locally tree-decomposable structures. *Journal of the ACM*, 48:1148 – 1206, 2001.
[15] H. Gaifman. On local and non-local properties. In J. Stern, editor, *Herbrand Symposium, Logic Colloquium '81*, pages 105 – 135. 1982.
[16] T. Ganzow and S. Rubin. Order-invariant MSO is stronger than counting MSO in the finite. In *STACS*, pages 313–324, 2008.
[17] Y. Gurevich. *Unpublished result*.
[18] P. Hlinený and S.-I. Oum. Finding branch-decompositions and rank-decompositions. *SIAM J. Comput.*, 38(3):1012–1032, 2008.
[19] Ken-ichi Kawarabayashi and Kenta Ozeki. Spanning closed walks and TSP in 3-connected planar graphs. ACM/SIAM Symp. on Discrete Algorithms (SODA), 671–682, 2012.
[20] J. Kneis, A. Langer, and P. Rossmanith. Courcelle's Theorem - a game-theoretic approach. Discrete Optimization, 8(4):568–594, 2011.
[21] J.A. Makowsky. Coloured tutte polynomials and kauffman bounded tree width brackets for graphs of bounded tree width. *Discrete Applied Mathematics*, 145(2):276–290, January 2005.
[22] H. Niemistö. On locality and uniform reduction. In *LICS*, pages 41–50. IEEE Computer Society, 2005.
[23] S.-I. Oum and P. Seymour. Approximating clique-width and branch-width. *Journal of Combinatorial Theory, Series B*, 96:514 – 528, 2006.
[24] B. Rossman. Successor-invariant first-order logic on finite structures. *J. Symb. Log.*, 72(2):601–618, 2007.
[25] N. Robertson and P. Seymour. Graph minors. III. Planar tree-width *J. of Comb. Th., Series B*, 36:49–64, 1984
[26] D. Seese. Linear time computable problems and first-order descriptions. *Mathematical Structures in Computer Science*, 5:505–526, 1996.
[27] P.D. Seymour and R. Thomas. Graph searching and a min-max theorem for tree-width. *J. of Comb. Th., Series B*, 58(1):22–33, 1993.
[28] C. Thomassen. A theorem on paths in planar graphs. *Journal of Graph Theory*, 7:169–176, 1983.