ON THE PARAMETERIZED INTRACTABILITY OF MONADIC SECOND-ORDER LOGIC

STEPHAN KREUTZER

School for Electrical Engineering and Computer Science, Technical University Berlin, Sekr. TEL 7-1, Ernst-Reuter-Platz 7, 10587 Berlin, Germany
e-mail address: stephan.kreutzer@tu-berlin.de

ABSTRACT. One of Courcelle's celebrated results states that if $\mathcal{C}$ is a class of graphs of bounded tree-width, then model-checking for monadic second order logic (MSO_2) is fixed-parameter tractable (fpt) on $\mathcal{C}$ by linear time parameterized algorithms, where the parameter is the tree-width plus the size of the formula. An immediate question is whether this is best possible or whether the result can be extended to classes of unbounded tree-width.

In this paper we show that in terms of tree-width, the theorem cannot be extended much further. More specifically, we show that if $\mathcal{C}$ is a class of graphs which is closed under colourings and satisfies certain constructibility conditions and is such that the tree-width of $\mathcal{C}$ is not bounded by $\log^{84} n$ then MSO_2-model checking is not fpt unless SAT can be solved in sub-exponential time. If the tree-width of $\mathcal{C}$ is not poly-logarithmically bounded, then MSO_2-model checking is not fpt unless all problems in the polynomial-time hierarchy can be solved in sub-exponential time.

1. INTRODUCTION

Classical logics such as first-order or fragments of second-order logic have played a crucial role in the development and analysis of query or specification languages in database theory, formal language theory and many other areas. In these application areas, computational logic problems such as satisfiability and model checking occur frequently and much effort has gone into analysing the complexity of these computational tasks.

In this paper we are mostly concerned with model checking for monadic second-order logic (MSO_2), the extension of first-order logic by quantification over sets of elements (i.e. vertices and edges). The model-checking problem for MSO_2 is the problem to decide for a given structure and a formula whether the formula is true in the structure. A reduction from the PSPACE-complete *quantified boolean formula*-problem (QBF) immediately shows that the model-checking problem for first-order and monadic second-order logic

is Pspace-hard. In fact the problems are Pspace-complete [32]. The problem even remains Pspace-complete on a fixed structure with only two elements, showing that the high complexity is already generated by the formula alone.

However, especially in a database context, where a formula specifies a query and the structure is the database, it can usually be assumed that the formula is reasonably small whereas the database is very large. Vardi [32] therefore proposed the concept of *data complexity* which is the complexity of model-checking against a fixed formula. For first-order logic, it can be shown that the data complexity is always polynomial time whereas for monadic second-order logic the model checking problem can already be NP-hard for a fixed formula. See e.g. Section 3 for an example defining the NP-complete 3-colourability problem. However, even for first-order logic, where the model-checking problem has polynomial time data complexity, the algorithms witnessing this usually run in time $|\mathfrak{A}|^{\mathcal{O}(|\varphi|)}$ and hence in time exponential in the formula. As the database $\mathfrak{A}$ was assumed to be huge, this is unacceptable even for relatively small formulas φ.

A more refined analysis of the model-checking complexity separating the complexity with respect to the formula from the complexity in terms of the database is offered by the framework of *parameterized complexity* [8, 10]. In this framework, the input to a model-checking problem again consists of a pair $(\mathfrak{A}, \varphi)$, where $\mathfrak{A}$ is a finite structure and φ is a formula, but now we declare $|\varphi|$ as the *parameter*. We call the problem *fixed parameter tractable* (fpt), if it can be solved in time $f(|\varphi|) \cdot |\mathfrak{A}|^c$, where f is a computable function and c a constant. Hence, we allow arbitrary amount of time with respect to the size of the formula but only fixed polynomial time in the size of the structure. The problem is in the parameterized complexity class XP if it can be solved in time $|\mathfrak{A}|^{f(|\varphi|)}$. The class FPT of all fixed-parameter tractable problems is the parameterized analogoue of polynomial time in classical complexity as model of tractable computation. The class XP takes over the role of exponential time in classical complexity.

Model-checking problems have received particular attention in the context of parameterized complexity. See e.g. [24] for a discussion on query complexity in databases theory with respect to the framework of parameterized complexity.

As the example of an MSO-formula defining 3-colourability shows, on general graphs model-checking for monadic second-order logic is not fixed-parameter tractable unless Ptime = NP. However, fixed-parameter tractability can be retained by restricting the class of admissible structures, for instance to words or trees. Studying properties and complexity results for monadic second-order logic on restricted classes of structures has a very long tradition in computer science, going back to by now classical results by Büchi, Rabin, Doner, Thatcher and Wright that on words and trees any formula of monadic second-order logic is equivalent to a word- or tree-automaton and hence, in terms of parameterized complexity, the model checking problem on such structures becomes fixed-parameter tractable as follows: given a tree T and a monadic second-order logic formula φ, we first convert φ into an equivalent tree-automaton, which is costly but only depends on $|\varphi|$, and then let the automaton run on the tree T to verify $T \models \varphi$. The latter runs in linear time in the size of T, hence the whole model checking algorithm runs in time $f(|\varphi|) \cdot |T|$ and is therefore fixed-parameter linear.

The observation that even such a powerful logic as monadic second-order logic becomes fixed-parameter linear on trees has been used in numerous contexts and applications. In database theory in particular, it has influenced the development of query languages for XML databases, a database model designed for data integration on the web. XML databases are

tree-like, in the sense that their skeleton is a tree (but there may be additional *references* creating edges violating the tree-property). The tree-structure and unbounded depth of XML databases necessitates new query languages such as XPath and others which allow to navigate in the tree, especially along paths from a node to its direct or indirect successors. In this context, monadic second-order logic has played the role of a yardstick as MSO-queries can be evaluated in linear time yet prove to be very expressive.

To be able to fully explore the potential of logics such as monadic second-order logic or first-order logic for future applications in databases and elsewhere, a thorough understanding of the structural properties of models that allow for tractable model-checking would prove most useful.

Ideally, for common logics $\mathcal{L}$ such as FO or variants of MSO, we aim at identifying a property P such that the parameterized model-checking problem for $\mathcal{L}$ becomes tractable on a class $\mathcal{C}$ of databases (or logical structures) if, and only if, $\mathcal{C}$ has the structural property P (under reasonable complexity theoretical assumptions).

There may not always exist such a property that precisely captures tractability for a logic, and sometimes we may have to compromise and impose further restrictions on the class $\mathcal{C}$, such as closure under sub-structures. But any reasonably precise characterisation would have great potential for future use of these logics in query and specification languages.

In this paper we establish a first characterisation in this sense of monadic second-order logic (MSO$_2$), or more generally *guarded second-order logic*.

In 1990, Courcelle proved a fundamental result stating that every property of graphs definable in monadic second-order logic (MSO$_2$) can be decided in linear time on any class $\mathcal{C}$ of structures of bounded tree-width (see below for a definition of tree-width). Besides the applications to logic outlined above, Courcelle's theorem has had significant impact on the theory of parameterized problems on graphs. In the design of efficient algorithms on graphs, it can often be used as a simple way of establishing that a property can be solved in linear time on graph classes of bounded tree-width. Furthermore, results such as Courcelle's theorem, usually called *algorithmic meta-theorems*, lead to a better understanding how far certain algorithmic techniques range and establish general upper bounds for the parameterized complexity of a wide range of problems. See [14, 19, 15] for recent surveys on algorithmic meta-theorems.

From a logical perspective, Courcelle's theorem establishes a sufficient condition for tractability of MSO$_2$ formula evaluation on classes $\mathcal{C}$ of structures: whatever the class $\mathcal{C}$ may look like, if it has bounded tree-width, then MSO$_2$-model checking is tractable on $\mathcal{C}$. An obvious question is how tight Courcelle's theorem is, i.e. whether it can be extended to classes of unbounded tree-width and if so, how "unbounded" the tree-width of graphs in the class can be in general. This question is the main motivation for the work reported here.

In this paper we establish an intractability result by showing that in its full generality, Courcelle's theorem can not be extended much further to classes of unbounded tree-width. Throughout the paper we consider structures over a binary signature $\sigma := \{R_1, \ldots, R_k, P_1, \ldots, P_l, c_1, \ldots, c_k\}$, where the R_i are binary relation symbols, the P_i are unary and the c_i are constants. We require that σ contains at least two binary and two unary relation symbols. See Section 3 for details. To give an example, an XML database over a fixed schema can naturally be modelled by a structure over a binary signature where each axis label yields a binary relation in the obvious way. Another intuitive way of looking

at binary structures is to view them as coloured graphs, where the binary relations correspond to edge colours and the unary relations to vertex colours. As it helps simplifying the presentation, we will adapt this way of looking at binary structures.

To state our main result, we first need some notation.

Definition 1.1. Fix a binary signature σ as before. The Gaifman-graph $\mathcal{G}(\mathfrak{A})$ of a σ-structure $\mathfrak{A}$ is the graph with the same universe as $\mathfrak{A}$ and an edge $\{a, b\}$ if there is a binary $R_i \in \sigma$ such that $(a, b) \in R_i^{\mathfrak{A}}$ or $(b, a) \in R_i^{\mathfrak{A}}$. A class $\mathcal{C}$ of σ-structures is said to be closed under colourings, if whenever $\mathfrak{A} \in \mathcal{C}$ and $\mathcal{G}(\mathfrak{A}) \cong \mathcal{G}(\mathfrak{B})$ then $\mathfrak{B} \in \mathcal{C}$.

Informally, whenever two σ-structures only differ in the colours of edges and vertices, then they both belong to $\mathcal{C}$ or both do not.

Definition 1.2. Let σ be a binary signature. Let $f : \mathbb{N} \to \mathbb{N}$ be a function and $p(n)$ be a polynomial.

The tree-width of a class $\mathcal{C}$ of σ-structures is (f, p)-*unbounded*, if for all $n \geq 0$

(1) there is a graph $G_n \in \mathcal{C}$ of tree-width $\text{tw}(G_n)$ between n and $p(n)$ such that $\text{tw}(G_n) > f(|G|)$ and

(2) given n, G_n can be constructed in time 2^{n^ε}, for some $\varepsilon < 1$.

The tree-width of $\mathcal{C}$ is *poly-logarithmically unbounded* if there are polynomials $p_i(n)$, $i \geq 0$, so that $\mathcal{C}$ is $(\log^i, p_i)$-unbounded for all i.

See Section 2 for a definition of tree-width and related concepts. Essentially, the first condition ensures that there are not too big gaps between the tree-width of graphs witnessing that the tree-width of $\mathcal{C}$ is not bounded by $f(n)$. The second condition ensures that we can compute such witnesses efficiently, i.e. in time polynomial in their size. We will see below why these conditions are needed. The following is the main result of the paper.

Theorem 1.3. *Let σ be a binary signature with at least two binary and two unary relation symbols. Let $\mathcal{C}$ be a class of σ-structures closed under colourings.*

(1) *If the tree-width of $\mathcal{C}$ is poly-logarithmically unbounded then $\text{MC}(\text{MSO}_2, \mathcal{C})$ is not in XP and hence not fixed-parameter tractable unless all problems in NP (in fact, all problems in the polynomial-time hierarchy) can be solved in sub-exponential time.*

(2) *If the tree-width of $\mathcal{C}$ is $(\log^c, p)$-unbounded, for some $c > d \cdot 84$ and polynomial p of degree d, then $\text{MC}(\text{MSO}_2, \mathcal{C})$ is not in XP and hence not fixed-parameter tractable unless SAT can be solved in sub-exponential time.*

See Section 3 for a precise definition of MSO_2 over structures and Section 4 for a definition of FPT and XP.

Essentially, as far as classes closed under colourings are concerned, if the tree-width of a class of graphs is not logarithmically bounded, then it has intractable MSO_2 model-checking. In this sense the theorem shows that tractability results as general as Courcelle's are not possible for classes of more than logarithmic tree-width. The restriction to classes closed under colourings is obviously a real restriction and it is possible that there are very special classes of σ-structures of tree-width not bounded by $\log^{84} n$ but with tractable model-checking. However, the usefulness of monadic second-order logic and tractability results such as Courcelle's theorem lie in their general applicability as specification and query languages. After all, we want a query language to be tractable on all databases of a certain structure, and not just if they have the right labels on their axes. And our result shows that beyond logarithmic tree-width, MSO no longer fulfills this promise.

Compared to Courcelle's theorem, there is a gap between constant tree-width to which Courcelle's theorem applies and tree-width not bounded by $\log^{84} n$ to which our theorem applies. The bound $c > d \cdot 84$ can be improved to $c > d \cdot 48$, see Section 9, and conceivably can be improved further. However, Makowsky and Mariño [23] exhibit a class of graphs whose tree-width is only bounded by $\log n$, i.e. it is $(\log^c n, n)$-unbounded for all $c < 1$, but where MSO_2 model-checking is tractable. It is easily seen that the closure of this class under colourings still admits tractable MSO_2 model-checking. Hence, there is no hope to extend our result to classes of tree-width less than logarithmic.

Let us give some applications of the theorem. For $c > 0$ let $\mathcal{C}_c$ be the class of all graphs G of tree-width at most $\log^c |G|$. Then the closure under colourings has intractable MSO_2 model-checking, if $c > 84$. Similarly, intractability follows for the class of planar graphs of tree-width at most $\log^c n$, as colours in this class can easily be encoded. All these examples show that Courcelle's theorem can not be extended to classes of graphs with only poly-logarithmic or a $\log^c n$ bound on the tree-width, for $c > 84$.

Following Courcelle's theorem, a series of algorithmic meta-theorems for first-order logic on planar graphs [12], (locally) H-minor-free graphs [11, 6] and various other classes have been obtained. Again, no deep lower bounds, i.e. intractability conditions, are known (see [19] for some bounds and [14, 19, 15] for recent surveys of the topic). The aim of this paper is to initiate a thorough study of sufficient conditions for intractability in terms of structural properties of input instances.

Related work.   Lower bounds for the complexity of monadic second-order logic for specific classes of graphs have been considered in the literature before. In [23], Makowsky and Mariño show that if a class of graphs has unbounded tree-width and is closed under topological minors then model-checking for MSO_2 is not fixed-parameter tractable unless $P = NP$.

In [4], Courcelle et al. show that unless $EXPTIME = NEXPTIME$, model-checking for MSO_2 is not fixed-parameter tractable on the class of complete graphs.

More closely related to the result reported here, Grohe [14, Conjecture 8.3] conjectures that MSO-model checking is not fixed-parameter tractable on any class $\mathcal{C}$ of graphs which is closed under taking subgraphs and whose tree-width is not poly-logarithmically bounded, i.e. there are no constants c, d such that $tw(G) \leq d \cdot \log^c |G|$ for all $G \in \mathcal{C}$.

Grohe's conjecture was affirmed in [21, 20] with respect to certain technical conditions similar to the notion of (f, p)-unboundedness defined above. It was proved that if $\mathcal{C}$ is closed under sub-graphs and its tree-width is $(\log^c, p)$-unbounded for some small constant c, then MSO_2 model-checking is not fpt on $\mathcal{C}$ unless SAT can be solved in sub-exponential time. The proof of this result is considerably more complex and much more technical than the proof reported here, especially in its combinatorial core.

It is worth noting that the two results are somewhat incomparable. In particular, closure under sub-structures in this context is a stronger requirement than it might seem at first sight: while tree-width is preserved by taking sub-graphs, logarithmic or poly-logarithmic tree-width is not. I.e., a sub-graph of a graph of tree-width at most k also has tree-width at most k, but if G has tree-width at most logarithmic in its order, this may not be the case for sub-graphs. Hence, the results in [20] are much more restrictive in this sense than our result here. On the other hand, they do not require closure under colourings and are therefore much more general in this aspect.

Organisation.   We fix our notation and review the graph theoretical notions we need in Section 2. Monadic second-order logic is defined in Section 3 and its complexity is reviewed

in Section 4. We give an informal and intuitive presentation of the main proof idea in Section 5. The proof is presented in full detail in Sections 6 to 8. We conclude in Section 9.

2. Preliminaries

In this section we fix our notation and review concepts from graph theory needed below.

2.1. **General Notation.** If M is a set we write $\mathcal{P}(M)$ for the set of all subsets of M. If M, N are two sets, we define $M \dot\cup N$ as the *disjoint union* of M and N, obtained by taking the union of M and a copy N' of N disjoint from M. We also apply this notation to graphs and other structures for which a union operation is defined.

We write $\mathbb{Z}$ for the set of integers and $\mathbb{N}$ for the set of non-negative integers.

2.2. **Graphs and Colourings.** We will use standard notation from graph theory and refer to [7] for background on graphs and details on the graph theoretical concepts introduced in this section.

All graphs in this paper are finite, undirected and simple, i.e. without multiple edges or loops. We write $V(G)$ for the set of vertices and $E(G)$ for the set of edges in a graph G. We will always assume that $V(G) \cap E(G) = \varnothing$.

The *order* $|G|$ of a graph is defined as $|V(G)|$ and its *size* $||G||$ as the number of edges. For $l \geq 1$ we denote the *l-clique*, the complete graph on l vertices, by K_l.

A graph H is a *sub-division* of G (a 1-*subdivision*) if H is obtained from G by replacing edges in G by paths of arbitrary length (of length 2, resp.). H is a *topological minor* of G if a subgraph $G' \subseteq G$ is isomorphic to a sub-division of H.

A graph H is a *minor* of G if it can be obtained from a sub-graph $G' \subseteq G$ by contracting edges. An equivalent, sometimes more intuitive, characterisation of the minor relation can be obtained using the concept of *images*. H is a minor of G if there is a map μ mapping each $v \in V(H)$ to a tree $\mu(v) \subseteq G$ and each edge $e \in E(H)$ to an edge $\mu(e) \in E(G)$ such that if $u \neq v \in V(H)$ then $\mu(v) \cap \mu(u) = \varnothing$ and if $\{u, v\} \in E(H)$ then $\mu(\{u, v\}) = \{u', v'\}$ for some $u' \in V(T_u)$ and $v' \in V(T_v)$. μ is called the *image map* and $\bigcup_{v \in V(H)} \mu(v) \cup \bigcup_{e \in E(H)} \mu(e) \subseteq G$ is called the *image* of H in G. It is not difficult to see that $H \preceq G$ if, and only if, there is an image of H in G.

Let G be a graph and $A, B \subseteq V(G)$. A set $S \subseteq V(G)$ is an *A-B-separator* if there is no path in $G \setminus S$ from a vertex in A to a vertex in B. An *A-B-path* $P \subseteq G$ is a path in G with one endpoint in A and the other in B.

Theorem 2.1 (Menger)**.** *Let G be a graph and $A, B \subseteq V(G)$. The minimal cardinality $|S|$ of an A-B-separator $S \subseteq V(G)$ is equal to he maximum number of vertex disjoint A-B-paths in G.*

Finally, we will be using the concept of intersection graphs.

Definition 2.2. Let G be a graph and $\mathcal{P}, \mathcal{Q}$ be two sets of pairwise disjoint paths in G. The *intersection graph* $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ is defined as the graph with vertex set $\mathcal{P} \dot\cup \mathcal{Q}$ where P, Q are adjacent if, and only if, $P \cap Q \neq \varnothing$.

2.3. **Tree-Width and Obstructions.** Tree-width is a measure of similarity of graphs to being a tree that was introduced by Robertson and Seymour in their graph minor project ([28]), even though equivalent concepts have been studied under different names before [17, 29].

Definition 2.3. A *tree-decomposition* of a graph G is a pair $(T, (B_t)_{t \in V(T)})$ where T is a tree and $B_t \subseteq V(G)$ such that

(1) for all $v \in V(G)$, the set $\{t \in V(T) : v \in B_t\}$ is non-empty and connected in T and
(2) for every edge $e := \{u, v\} \in E(G)$ there is a $t \in V(T)$ such that $u, v \in B_t$.

The *width* of a tree-decomposition is $\max_{t \in V(T)} |B_t| - 1$ and the *tree-width* $\mathrm{tw}(G)$ of a graph G is the minimal width of any of its tree-decompositions.

A class $\mathcal{C}$ of graphs has *bounded tree-width* if there is a constant $c \in \mathbb{N}$ such that $\mathrm{tw}(G) \leq c$ for all $G \in \mathcal{C}$.

Many natural classes of graphs are found to have bounded tree-width, for instance series-parallel graphs or control-flow graphs of goto-free C programs [31], and many generally NP-hard problems can be solved efficiently on graph classes of small tree-width. This is witnessed in particular by Courcelle's Theorem 4.2 below.

In this paper, we will mostly be concerned with graphs of large tree-width and structural information we can gain about a graph once we know that its tree-width is large. This leads to the concept of *obstructions*, i.e. structures we can find in any graph of large enough tree-width. In this paper, we will use two such obstructions, *brambles* and *grids*.

Definition 2.4. Let G be a graph. Two subgraphs $X, Y \subseteq G$ *touch* if $X \cap Y \neq \varnothing$ or there is an edge in G *linking* X and Y, i.e. with one endpoint in X and the other in Y. A *bramble* in G is a set $\mathcal{B}$ of pairwise touching connected subgraphs of G. A set $S \subseteq V(G)$ is a *cover* for $\mathcal{B}$ if $S \cap V(B) \neq \varnothing$ for all $B \in \mathcal{B}$. The *order* of $\mathcal{B}$ is the minimum cardinality of a cover of $\mathcal{B}$. The *size* of $\mathcal{B}$ is the number $|\mathcal{B}|$ of sets in $\mathcal{B}$.

Brambles provide a dual characterisation of tree-width as shown in the following theorem.

Theorem 2.5 ([30])**.** *A graph G has treewidth at least k if, and only if, G contains a bramble of order at least $k + 1$.*

Brambles will form the basis of our algorithmic part below. However, often it is much easier to work with another obstruction, known as *grids*. A $(k \times l)$-*grid* $G_{k \times l}$ is a graph as in Figure 1. Formally, $G_{k \times l}$ is defined as the graph with

$$
\begin{aligned}
V(G_{k \times l}) &:= \{(i, j) : 1 \leq i \leq k, 1 \leq j \leq l\} \text{ and} \\
E(G_{k \times l}) &:= \{\{(i, j), (i', j')\} : |i - i'| + |j - j'| = 1\}.
\end{aligned}
$$

Grids play a very special role in connection with tree-width as every graph of large tree-width contains a large grid as minor.
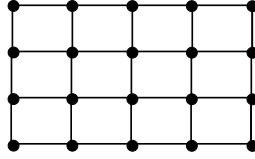
Figure 1: A (4×5)-grid.

Theorem 2.6 (Excluded Grid Theorem [27, 26])**.** *There is a function $f : \mathbb{N} \to \mathbb{N}$ such that any graph of tree-width at least $f(k)$ contains a $(k \times k)$-grid as a minor.*

Unfortunately, the best upper bound on this function f known to date is exponential in k. The results reported in this paper would have much simpler proofs if one could establish a polynomial upper bound for the function f in the previous theorem.

3. Monadic Second-Order Logic

In this section we will introduce monadic second-order logic. Intuitively, *monadic second-order logic* is the extension of first-order logic by quantification over sets of elements. That is, we can use formulas of the form $\exists X \varphi(X)$ which says that there exists a set X which satisfies the formula φ. However, in the context of graphs there are two natural options for what constitutes an element: we can allow quantification over sets of vertices or quantification over sets of edges. This leads to two different logics which are sometimes referred to as MSO_1 and MSO_2, respectively, where MSO_2 allows quantification over sets of edges and vertices whereas MSO_1 only allows quantification over sets of vertices. MSO_2 is much more expressive than MSO_1 as we can easily say that a graph contains a simple path which contains every vertex, a property that is not definable in MSO_1. See below for an example of a formula defining this property. For the purpose of this paper it is convenient to introduce MSO_2 as a logic on the *incidence representation of graphs*, which we will formally define below.

3.1. **Signatures and Structures.** We assume familiarity with basic notions of mathematical logic (see e.g. [9]). A *signature* σ is a finite set of constant symbols $c \in \sigma$ and relation symbols $R \in \sigma$ where each relation symbol is equipped with its arity $ar(R) \in \mathbb{N}$.

A *σ-structure* $\mathfrak{A}$ consists of a finite set A, the *universe* of $\mathfrak{A}$, an r-ary relation $R^{\mathfrak{A}} \subseteq A^r$ for each relation symbol $R \in \sigma$ of arity $r := ar(R)$ and a constant $c^{\mathfrak{A}} \in A$ for each constant symbol $c \in \sigma$. We will denote structures by German letters $\mathfrak{A}, \mathfrak{B}, \mathfrak{D}$ and their universes by corresponding Roman letters A, B, D.

In this paper we will only consider *binary signatures*, where the maximal arity of relation symbols is 2. An example of classes of structures over binary signatures are the skeletons of XML databases, i.e. XML databases where the actual data values are ignored. For instance, the database

```
<libraryholdings>
   <book>
      <author>YM</author>
      <title>EIAS</title>
   </book>
   <book>
```

```
        <author>RD</author>
        <title>GT</title>
    </book>
</libraryholdings>
```

can naturally be modelled as a structure $\mathfrak{D}$ over the signature $\sigma := \{book, author, title\}$, where the elements of the universe D corresponds to the individual tags `<libraryholdings>` etc. and edges represent the child relation.

Another natural interpretation of binary signatures is that structures over these signatures are coloured graphs, i.e. the binary relations represent edges coloured by the relation name and the unary relations represent vertex colours.

To work with logics on graphs we have to specify how we want to represent graphs as logical structures. Let $\sigma_{inc} := \{V, E, \in\}$ be a signature, where V, E are unary and $\in$ is a binary relation symbol. We can view a graph G as a σ_{inc}-structure $\mathfrak{G} := \mathfrak{G}(G)$ with universe $V(G) \dot\cup E(G)$ and $V^{\mathfrak{G}} := V(G)$, $E^{\mathfrak{G}} := E(G)$ and $x \in^{\mathfrak{G}} e$ if $x \in V(G), e \in E(G)$ and x and e are incident in G. This is known as the *incidence representation* of graphs as opposed to the natural representation of graphs G as structures $\mathfrak{G}$ over the signature $\{E\}$, where the universe is $V(G)$ and $E^{\mathfrak{G}} := E(G)$.

We now extend the definition of tree-width from graphs to arbitrary relational structures.

Definition 3.1. Let σ be an at most binary signature. The *tree-width* $\mathrm{tw}(\mathfrak{A})$ of a σ-structure $\mathfrak{A}$ is defined as the tree-width $\mathrm{tw}(\mathcal{G}(\mathfrak{A}))$ of its Gaifman-graph (see Definition 1.1).

In the context of graphs and tree-width it might be worth noting that the tree-width of a graph is the same as the tree-width of its standard or incidence representation.

Definition 3.2. For the rest of this paper we fix a signature $\sigma_{col} := \{V, E, \in, B, R, C_0, C_1\}$, where $\in$ is a binary relation symbol and V, E, B, R, C_0, C_1 are unary.

We define the signature $\sigma_G := \{V, E, \in, C_0, C_1\}$ and $\sigma_{ord} := \sigma_G \cup \{\leq\}$.

3.2. **Definition of Monadic Second-Order Logic.** The class of formulas of *monadic second-order logic* over a signature σ, denoted $\mathrm{MSO}[\sigma]$, is defined as the extension of first-order logic by quantification over sets of elements. That is, in addition to first-order variables, which we will denote by small letters $x, y, ...$, there are unary, or monadic, second-order variables $X, Y, ...$ ranging over sets of elements. Formulas of $\mathrm{MSO}[\sigma]$ are then built up inductively by the rules for first-order logic with the following additional rules: if X is a monadic second-order variable and $\varphi \in \mathrm{MSO}[\sigma \dot\cup \{X\}]$, then $\exists X \varphi \in \mathrm{MSO}[\sigma]$ and $\forall X \varphi \in \mathrm{MSO}[\sigma]$ with the obvious semantics where, e.g., a formula $\exists X \varphi$ is true in a σ-structure $\mathfrak{A}$ with universe A if there is a subset $U' \subseteq A$ such that φ is true in $\mathfrak{A}$ if the variable X is interpreted by U'. We denote this by $(\mathfrak{A}, U') \models \varphi(X)$. If $\varphi(x)$ is a formula with a free first-order variable x, $\mathfrak{A}$ is a structure and $a \in A$, we write $\mathfrak{A} \models \varphi[v]$, or $(\mathfrak{A}, v) \models \varphi$, to say that φ is true in $\mathfrak{A}$ if x is interpreted by a. We write $\varphi(\mathfrak{A})$ for the set $\{v \in A : \mathfrak{A} \models \varphi[v]\}$.

As explained above, when viewed as a logic on graphs, the expressive power of monadic second-order logic depends on whether a graph is represented by its standard representation or by its incidence representation. It has become common terminology to refer to MSO on graphs represented by their standard representation as MSO_1 and to use MSO_2 to indicate that graphs are represented by their incidence structures.

We will follow this terminology. Therefore, if σ is an at most binary signature, we define $\mathrm{MSO}_2[\sigma]$ to be monadic second-order logic over the signature $\sigma \dot\cup \{V, E, \in\}$ where σ-structures are represented as incidence structures in the obvious way. The main theorem stated in the introduction can therefore equivalently be stated as a theorem on structures over a signature $\tau := \{V, E, \in, R_1, \ldots, R_k, U_1, \ldots, U_l, c_1, \ldots, c_s\}$ containing at least two binary relation symbols R_i and two unary relation symbols U_i.

In this paper we will almost exclusively use the incidence representation and therefore agree that MSO always refers to MSO_2 unless explicitly stated otherwise. Also, we will always make the signatures we work with precise to avoid confusion.

We will not distinguish notationally between a graph G and its incidence representation $\mathfrak{G}$ and will simply write G. To simplify the presentation of formulas, we agree on the following notation.

Notation. We will write $\exists X \subseteq V \varphi$ and $\exists F \subseteq E \varphi$ as abbreviation for $\exists X \big((\forall x\, x \in X \rightarrow x \in V) \wedge \varphi \big)$ and $\exists F \big((\forall x\, x \in F \rightarrow x \in E) \wedge \varphi \big)$ to indicate that X is a set of vertices and F is a set of edges. $\forall X \subseteq V$ and $\forall F \subseteq E$ are defined analogously.

We write $e \cap X \neq \varnothing$ for $\exists u \in V(u \in e \wedge u \in X)$ and similarly $e \subseteq X$ for $\forall u(u \in e \rightarrow u \in X)$ to say that X contains an endpoint (both endpoints, resp.) of e. Also, we will use notation such as $X \cap Y \neq \varnothing$, $X \subseteq Y$, ... with the obvious meaning.

We will often use set variables P which are intended to contain the edges of a path in a graph. The following notation helps to simplify formulas speaking about paths. If P is a variable denoting a set of edges then we write $x \in V(P)$ for the formula $\exists e(e \in P \wedge x \in e)$ expressing that x occurs as an endpoint of an edge e in P. Furthermore, we write $\{x, y\} \in P$ for the formula $x \neq y \wedge \exists e \in P(x \in e \wedge y \in e)$ saying that $\{x, y\}$ is an edge in P.

Finally, we write $\exists^{\leq 2} x \varphi$ for the formula $\exists x \exists y \big(\varphi(x) \wedge \varphi(y) \wedge \neg \exists z(z \neq x \wedge z \neq y \wedge \varphi(z)) \big)$ expressing that there are at most two vertices satisfying φ. We will also use $\exists^{=1}, \exists^{\leq 1}$ with the obvious meaning.

3.3. Examples.

Example 3.3. To give an example consider the following MSO-formula φ over the signature σ_{inc}.

$$\exists C_1, C_2, C_3 \subseteq V \Big(\forall x \in V \bigvee_{i=1}^{3} x \in C_i \wedge \forall e \in E \bigwedge_{1 \leq i \leq 3} \neg e \subseteq C_i \Big),$$

where x, y are first-order variables and C_1, C_2, C_3 are second-order variables. The formula expresses in a σ_{inc}-structure G that there are three sets of vertices so that every vertex occurs in at least one of the sets but no edge has both endpoints in the same set. Hence, $G \models \varphi$ if, and only if, G is 3-colourable. ⊣

Example 3.4. As a second example we define a formula φ_{Ham} true in a graph G if, and only if, the graph contains a Hamiltonian path, i.e. a simple path containing every vertex.

The formula $conn(P)$ defined as

$$conn(P) := \forall X \subseteq V \Big[\Big(\begin{matrix} V(P) \cap X \neq \varnothing \wedge \\ \forall e \in P(e \cap X \neq \varnothing \rightarrow e \subseteq X) \end{matrix} \Big) \rightarrow V(P) \subseteq X \Big]$$

says that if X is any set of vertices containing a vertex $x \in V(P)$ which is closed under edges $e \in P$, i.e. if one endpoint of e is in X then both are, then X must contain all vertices

of $V(P)$. Clearly, this formula can only be true of a set P of edges if P induces a connected sub-graph. The next formula expresses that P induces an acyclic graph.

$$ac(P) := \neg \exists s, t \in V \exists P, P' \subseteq E \big(s \neq t \wedge conn(P) \wedge conn(P') \wedge V(P) \cap V(P') = \{s, t\}\big)$$

The formula states that there are no two distinct vertices s and t and two connected sub-graphs P and P' such that s and t are contained both in P and P' but otherwise P and P' are vertex disjoint. Clearly, any cyclic graph contains such s, t, P, P' but no acyclic graph does.

Hence, $conn(P) \wedge ac(P)$ says that P induces a tree. Now the formula

$$path(P) := ac(P) \wedge conn(P) \wedge \forall x \in V \exists^{\leq 2} e(e \in P \wedge x \in e)$$

says that P is a tree and every vertex has degree at most 2 in the graph induced by P. Hence, P is a path. Finally,

$$\varphi_{Ham} := \exists P path(P) \wedge V \subseteq V(P)$$

expresses that the graph contains a Hamiltonian path. Here we crucially need quantification over sets of edges (which is implicit in the incidence encoding of graphs) as the Hamiltonian-path property is not expressible in MSO without edge set quantification.   ⊣

Example 3.5. We now give a much more substantial example which will be used in the proof of the main results of this paper. In particular, we will show that grids can be defined in monadic second-order logic.

We first establish the following characterisation of grids which can then easily be turned into an MSO-formulation.

Let G be a graph and $\mathcal{H}, \mathcal{V}$ be two sets of pairwise vertex disjoint paths, which we think of the horizontal and vertical paths in the grid. Then $\mathcal{H} \cup \mathcal{V}$ is a grid if, and only if, the following conditions are true.

(1) Any two $P \in \mathcal{V}, Q \in \mathcal{H}$ intersect in exactly one vertex and every vertex of the graph is contained in the intersection of two such paths $P \in \mathcal{V}, Q \in \mathcal{H}$.
(2) There are distinct $L, R \subseteq \mathcal{V}$, the left-most and right-most path of the grid, such that every $Q \in \mathcal{H}$ intersects L and R in one endpoint. Analogously, there are distinct $T, B \subseteq \mathcal{V}$, the upper-most and lower-most path of the grid, such that every $P \in \mathcal{H}$ intersects T and B in one endpoint.
(3) Let us define an order $\leq_P$ on the vertex set of a path $P \in \mathcal{V}$ such that $x \leq_P y$, for $x, y \in V(P)$ if x is closer to the endpoint of P in T than y, i.e. if the unique path from y to the endpoint of P in T also contains x. We write $x <_P y$ for the corresponding strict order. Analogously, we define $x \leq_Q y$, for $Q \in \mathcal{H}$ and $x, y \in V(Q)$, if x is closer to endpoint of Q in L, i.e. the unique path from y to the endpoint of Q in L contains x. Again $x <_Q y$ denotes the strict variant.
    Let $P, P' \in \mathcal{V}$ and $Q, Q' \in \mathcal{H}$ and let $x \in V(P \cap Q), x' \in V(P' \cap Q)$ and $y \in V(P \cap Q')$ and $y' \in V(P' \cap Q')$. Then,
    • $x <_P y$ if, and only if, $x' <_{P'} y'$ and
    • $x <_Q x'$ if, and only if, $y <_{Q'} y'$.
    That is, we require that all "horizontal" paths $Q, Q' \in \mathcal{H}$ cross all vertical paths $P, P' \in \mathcal{V}$ in the same order, seen from the top, and that all "vertical" paths $P, P' \in \mathcal{V}$ cross all horizontal paths $Q, Q' \in \mathcal{H}$ in the same order seen from the "left".

It is easily seen that if $\mathcal{H}$ is the set of horizontal paths and $\mathcal{V}$ the set of vertical paths in a grid, then $\mathcal{V}, \mathcal{H}$ satisfy these conditions. Conversely, let $\mathcal{V}, \mathcal{H}$ be two sets of pairwise disjoint paths satisfying conditions 1 to 3 then the graph induced by $\mathcal{V}, \mathcal{H}$ is a grid.

We show next how these conditions can be formalised by a formula $\varphi_{grid}(\mathcal{H}, \mathcal{V})$. To simplify the presentation, we will use second-order variables $\mathcal{P}, \mathcal{Q}, \mathcal{V}, \mathcal{H}$ which we will always ensure to be interpreted by sets of pairwise disjoint paths.

We first define some basic formulas which will be used frequently later on.

The formula

$$set\text{-}o\text{-}dis\text{-}path(\mathcal{P}) := \mathcal{P} \subseteq E \wedge ac(\mathcal{P}) \wedge \forall x \exists^{\leq 2} e \in \mathcal{P}(x \in e)$$

expresses that $\mathcal{P}$ is a set of edges inducing an acyclic sub-graph in which every vertex has degree at most 2. Hence $\mathcal{P}$ must be a set of pairwise vertex disjoint paths.

The next formula

$$maxpath(P, \mathcal{P}) := \mathcal{P} \subseteq E \wedge P \subseteq \mathcal{P} \wedge path(P) \wedge \forall P'\big(P \subseteq P' \wedge P' \subseteq \mathcal{P} \to \neg path(P')\big)$$

states that P is a maximal path in $\mathcal{P}$, hence it one of the paths in the set $\mathcal{P}$ of pairwise disjoint paths. We will write $\exists P \in \mathcal{P}$ as abbreviation for $\exists P\, maxpath(P, \mathcal{P})$ and likewise for $\forall P \in \mathcal{P}$.

Finally,

$$ep(x, P) := path(P) \wedge \exists^{=1} e \in P(x \in e)$$

defines that x is an endpoint of the path P.

Now the conditions above can easily be defined in MSO as follows. The formula $\varphi_0 := set\text{-}o\text{-}dis\text{-}path(\mathcal{V}) \wedge set\text{-}o\text{-}dis\text{-}path(\mathcal{H})$ ensures that $\mathcal{V}$ and $\mathcal{H}$ are interpreted by sets of pairwise vertex disjoint paths.

The formula

$$\varphi_1(\mathcal{V}, \mathcal{H}) := \begin{array}{l} \forall P \in \mathcal{V} \forall Q \in \mathcal{H} \exists^{=1} x \in V(x \in V(P) \cap V(Q)) \wedge \\ \forall x \in V \exists^{=1} P \exists^{=1} Q\big(x \in V(P) \wedge x \in V(Q)\big) \end{array}$$

expresses the first condition above.

The formula $\varphi_2(\mathcal{V}, \mathcal{H}, L, R, T, B)$

$$\varphi_2 := \begin{array}{l} L \in \mathcal{V} \wedge R \in \mathcal{V} \wedge T \in \mathcal{H} \wedge B \in \mathcal{H} \wedge \\ \forall P \in \mathcal{V} \exists x_1, x_2 \in V(P)\big(ep(x_1, P) \wedge ep(x_2, P) \wedge x_1 \in V(T) \wedge x_2 \in V(B)\big) \wedge \\ \forall Q \in \mathcal{H} \exists x_1, x_2 \in V(Q)\big(ep(x_1, Q) \wedge ep(x_2, Q) \wedge x_1 \in V(L) \wedge x_2 \in V(R)\big) \end{array}$$

expresses L, R, T, B satisfy the requirements outlined in Condition 2.

Finally, we define a formula expressing Condition 3. The formula

$$\varphi_P(x, y, P) := \begin{array}{l} \exists p \in V(P) \cap V(T) \wedge \\ \forall P'\big(P' \subseteq P \wedge path(P') \wedge p \in V(P') \wedge y \in V(P') \to x \in V(P')\big) \end{array}$$

defines the ordering $\leq_P$ on a path $P \in \mathcal{V}$. It states that $x \leq_P y$ if every sub-path of P containing the endpoint in T and y also contains x. Analogously, the formula

$$\varphi_Q(x, y, Q) := \begin{array}{l} \exists p \in V(Q) \cap V(L) \wedge \\ \forall Q'\big(Q' \subseteq Q \wedge path(Q') \wedge p \in V(Q') \wedge y \in V(Q') \to x \in V(Q')\big) \end{array}$$

defines the ordering $\leq_Q$ on a path $P \in \mathcal{H}$.

Hence, the formula

$$\varphi_3 := \forall P, P' \subseteq \mathcal{V} \; \forall Q, Q' \subseteq \mathcal{H} \left(\begin{array}{l} \exists x \in V(P \cap Q) \exists x' \in V(P' \cap Q) \\ \exists y \in V(P \cap Q') \exists y' \in V(P' \cap Q') \\ \big(\varphi_P(x,y,P) \leftrightarrow \varphi_P(x',y',P')\big) \wedge \\ \big(\varphi_Q(x,x',Q) \leftrightarrow \varphi_Q(y,y',Q')\big) \end{array} \right)$$

defines Condition 3.

Taken together, the formula

$$\varphi_{grid-border}(L, R, T, B, \mathcal{V}, \mathcal{H}) := \bigwedge_{i=0}^{3} \varphi_i$$

expresses that $\mathcal{V}, \mathcal{H}$ form a grid with borders L, T, R, B, clock-wise from left. The formula φ_{grid} can therefore be defined as $\exists L, T, R, B \subseteq E \; \varphi_{grid\text{-}border}(L, R, T, B, \mathcal{V}, \mathcal{H})$. $\dashv$

As the examples show, once we have established a few basic formulas such as *path* and *ac*, many properties of graphs can very easily be expressed in MSO. We are therefore particularly interested in the problem of deciding whether a given MSO-formula is true in a graph G.

3.4. **MSO-Transductions.**
A useful tool in the proof of our main results in this paper is the concept of logical transduction, which for our purposes play a similar role to many-one reductions in complexity theory. Essentially, a transduction is a way of defining one logical structure inside another. This concept is usually referred to as *interpretations* in model theory, see e.g. [18] for details. However, we will use interpretations in the "wrong" direction and therefore follow Courcelle's notation and call them transductions (see e.g. [5]).

Definition 3.6. Let σ and τ be signatures and let $\overline{X}$ be a tuple of monadic second-order variables. An MSO-*transduction of σ to τ with parameters $\overline{X}$* is a tuple $\Theta := \big(\varphi_{valid}, \varphi_{univ}(x), \varphi_\sim(x,y), (\varphi_R(\overline{x}))_{R \in \tau}\big)$ of MSO$[\sigma \dot\cup \overline{X}]$-formulas, where the arity of $\overline{x}$ in $\varphi_R(\overline{x})$ is $ar(R)$, such that for all σ-structures $\mathfrak{A}$ and interpretations $\overline{Y} \subseteq A$ of $\overline{X}$ with $(\mathfrak{A}, \overline{Y}) \models \varphi_{valid}$, $\varphi_\sim$ defines an equivalence relation on $\varphi_{univ}(\mathfrak{A})$ and if $R \in \tau$ or arity r and $\overline{a} := a_1, \ldots, a_r, \overline{b} := b_1, \ldots, b_r \in A^r$ are tuples such that $\mathfrak{A} \models \varphi_\sim(a_i, b_i)$ for all i then $\mathfrak{A} \models \varphi_R(\overline{a})$ if, and only if, $\mathfrak{A} \models \varphi_R(\overline{b})$.

With any transduction Θ we associate a map taking a σ-structure $\mathfrak{A}$ and $\overline{Y} \subseteq A$ such that $(\mathfrak{A}, \overline{Y}) \models \varphi_{valid}$ to a τ-structure $\mathfrak{B}$ with universe $B := \varphi_{univ}(\mathfrak{A}, \overline{Y})_{/\varphi_\sim(\mathfrak{A}, \overline{Y})} := \{[v]_\sim : (\mathfrak{A}, \overline{Y}) \models \varphi_{univ}(v)\}$ where $[v]_\sim$ denotes the equivalence class of v under the equivalence relation defined by $\varphi_\sim(A, \overline{Y})$. For $R \in \tau$ of arity $r := ar(R)$ we define $R^{\mathfrak{B}} := \{([a_1], \ldots, [a_r]) : (\mathfrak{A}, \overline{Y}) \models \varphi_R(a_1, \ldots, a_r)\}$.

For any σ-structure $\mathfrak{A}$ we define

$$\Theta(\mathfrak{A}) := \{\Theta(\mathfrak{A}, \overline{Y}) : \overline{Y} \subseteq A, (\mathfrak{A}, \overline{Y}) \models \varphi_{valid}\}.$$

If $\mathcal{C}$ is a class of σ-structures then

$$\Theta(\mathcal{C}) := \bigcup \{\Theta(\mathfrak{A}) : \mathfrak{A} \in \mathcal{C}\}.$$

Furthermore, any interpretation Θ also defines a translation of MSO$[\tau]$-formulas φ to MSO$[\sigma]$-formulas φ' by replacing occurrences of relations $R \in \tau$ by their defining formulas

$\varphi_R \in \Theta$ in the usual way (see [18] for details). For notational convenience we define $\Theta(\varphi) := \varphi_{valid} \wedge \varphi'$. The following lemma is then easily proved.

Lemma 3.7. *Let Θ be an MSO-transduction of σ in τ with parameters $\overline{X}$. For any σ-structure $\mathfrak{A}$ and assignment $\overline{Y} \subseteq A$ to $\overline{X}$ such that $(\mathfrak{A}, \overline{Y}) \models \varphi_{valid}$ and any MSO$[\tau]$-formula φ we have $\Theta(\mathfrak{A}, \overline{Y}) \models \varphi$ if, and only if, $(\mathfrak{A}, \overline{Y}) \models \Theta(\varphi)$.*

We will be using the previous lemma as summarised in the next corollary.

Corollary 3.8. *Let $\Theta := (\varphi_{valid}, ...)$ be an MSO-transduction of σ in τ with parameters $\overline{X}$. For any σ-structure $\mathfrak{A}$ and MSO$[\tau]$-formula φ we have*

$$\mathfrak{A} \models \exists \overline{X}(\varphi_{valid} \wedge \Theta(\varphi)) \text{ if, and only if, there exists } \mathfrak{B} \in \Theta(\mathfrak{A}) \text{ s.t. } \mathfrak{B} \models \varphi.$$

Example 3.9. We will define a transduction $\Theta := (\varphi_{valid}, \varphi_{univ}, \varphi_\sim, \varphi_V, \varphi_E, \varphi_\in)$ with parameters $\mathcal{P}, \mathcal{Q}$ from σ_{inc} to σ_{inc} so that for any σ_{inc}-structure $\mathfrak{A}$ and two sets $\mathcal{P}, \mathcal{Q}$ of disjoint paths in $\mathfrak{A}$, $\Theta(\mathfrak{A}, \mathcal{P}, \mathcal{Q})$ is the incidence representation of the intersection graph $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ (see Definition 2.2).

The formula φ_{valid} is simply defined as

$$\varphi_{valid}(\mathcal{P}, \mathcal{Q}) := \textit{set-o-dis-path}(\mathcal{P}) \wedge \textit{set-o-dis-path}(\mathcal{Q})$$

stating that $\mathcal{P}$ and $\mathcal{Q}$ are sets of pairwise disjoint paths (see Example 3.5 for the formula *set-o-dis-path*).

To define the φ_V, φ_E, recall that the vertices of $\mathcal{I} := \mathcal{I}(\mathcal{P}, \mathcal{Q})$ are the paths in $\mathcal{P} \dot\cup \mathcal{Q}$ and that two vertices P, Q are adjacent if the paths intersect. We will represent a path $P \in \mathcal{P} \dot\cup \mathcal{Q}$, and hence the corresponding vertex in $\mathcal{I}$, by the set of edges of $\mathfrak{A}$ occurring only in P and in no other path in $\mathcal{P} \dot\cup \mathcal{Q}$. Note that as $\mathcal{P}$ and $\mathcal{Q}$ are sets of pairwise disjoint paths, such edges must always exist, whereas it could happen that every vertex of P also occurs as a vertex of another path.

Towards this goal, the formula

$$\textit{uni-edge}(x, P) := x \in P \wedge \forall Q\big(P \neq Q \wedge (\textit{maxpath}(Q, \mathcal{P}) \vee \textit{maxpath}(Q, \mathcal{Q})) \rightarrow \neg x \in Q\big),$$

where $P \neq Q$ is an abbreviation for $\exists x \in V(P) \setminus V(Q)$ saying that P and Q are not the same path, states that x is an edge unique to P.

The formula

$$\varphi_{univ}^V(x; \mathcal{P}, \mathcal{Q}) := \begin{aligned}&\exists P\big(\textit{maxpath}(P, \mathcal{P}) \wedge \textit{uni-edge}(x, P)\big) \vee\\ &\exists Q\big(\textit{maxpath}(Q, \mathcal{Q}) \wedge \textit{uni-edge}(x, Q)\big)\end{aligned}$$

defines the set of edges unique to a path in $P \dot\cup Q$. Correspondingly, the formula

$$\varphi_\sim^V(x, y; \mathcal{P}, \mathcal{Q}) := \begin{aligned}&\exists P\big(\textit{maxpath}(P, \mathcal{P}) \wedge \textit{uni-edge}(x, P) \wedge \textit{uni-edge}(y, P)\big) \vee\\ &\exists Q\big(\textit{maxpath}(Q, \mathcal{Q}) \wedge \textit{uni-edge}(x, Q) \wedge \textit{uni-edge}(y, Q)\big)\end{aligned}$$

defines two vertices of $\mathcal{I}$ to be equivalent if the are unique edges of the same path in $\mathcal{P} \dot\cup \mathcal{Q}$.

To define the edges of $\mathcal{I}$, we will represent an edge $\{P, Q\}$ in $\mathcal{I}$ by the set of vertices in $V(P \cap Q)$. The formula

$$\varphi_{univ}^E(x; \mathcal{P}, \mathcal{Q}) := x \in V \wedge \exists P \exists Q\big(\textit{maxpath}(P, \mathcal{P}) \wedge \textit{maxpath}(Q, \mathcal{Q}) \wedge x \in V(Q \cap P)\big)$$

defines the set of all vertices which occur in the intersection of two paths. Correspondingly, the formula $\varphi_\sim^E(x, y; \mathcal{P}, \mathcal{Q})$ defined as

$$\exists P \exists Q\Big(\textit{maxpath}(P, \mathcal{P}) \wedge \textit{maxpath}(Q, \mathcal{Q}) \wedge x \in V(Q \cap P) \wedge y \in V(Q \cap P)\Big)$$

defines two vertices to be equivalent if they occur together in the intersection of the same two paths.

Hence, the vertex set of the incidence representation of $\mathcal{I}$ is defined by $\varphi_{univ}(x; \mathcal{P}, \mathcal{Q}) := \varphi_{univ}^V \vee \varphi_{univ}^E$ and $\varphi_{\sim}(x, y; \mathcal{P}, \mathcal{Q}) := \varphi_{\sim}^V \vee \varphi_{\sim}^V$ and the relations E and V are defined by $\varphi_E(x; \mathcal{P}, \mathcal{Q}) := \varphi_{univ}^E$ and $\varphi_V(x; \mathcal{P}, \mathcal{Q}) := \varphi_{univ}^V$.

All that remains is to define the formula $\varphi_{\in}(x, y, \mathcal{P}, \mathcal{Q})$. But this can easily be done as a vertex x in $\mathcal{I}$, say corresponding to a path $P \in \mathcal{P}$ and therefore represented by the set of unique edges of P, is incident to an edge e of $\mathcal{I}$, corresponding to the intersection of two paths $P' \in \mathcal{P}$ and $Q \in \mathcal{Q}$ and thus represented by the set $V(P') \cap V(Q)$, if $V(P') \cap V(Q') \subseteq V(P)$, i.e. if $e \in V(P)$. This is expressed by the following formula

$$\varphi_{\in}(x, e, \mathcal{P}, \mathcal{Q}) := \begin{array}{l} \varphi_V(x) \wedge \varphi_E(e) \wedge \\ \exists P\big((maxpath(P, \mathcal{P}) \vee maxpath(P, \mathcal{Q})) \wedge uni\text{-}edge(x, P) \wedge e \in P\big). \end{array}$$

This completes the transduction Θ. $\dashv$

4. The Complexity of Monadic Second-Order Logic

The *model checking problem* MC(MSO) for MSO is defined as the problem, given a structure G and a formula $\varphi \in$ MSO, to decide if $G \models \varphi$. By a reduction from the Pspace-complete *Quantified Boolean Formula Problem* (QBF) – the problem to decide whether a quantified Boolean formula is true – we easily get that MC(MSO) is Pspace-hard (see [32]). In fact, the problem is Pspace-complete as membership in Pspace is easily seen.

However, the hardness result crucially uses the fact that the formula is part of the input (and in fact holds on a fixed two-element structure), whereas we are primarily interested in the complexity of checking a fixed formula expressing a graph property in a given input graph. We therefore study model-checking problems in the framework of *parameterized complexity* (see [10] for background on parameterized complexity).

Definition 4.1. Let $\mathcal{C}$ be a class of σ-structures. The *parameterized model-checking problem* p-MC(MSO, $\mathcal{C}$) for MSO on $\mathcal{C}$ is defined as the problem to decide, given $G \in \mathcal{C}$ and $\varphi \in$ MSO$[\sigma]$, if $G \models \varphi$. The *parameter* is $|\varphi|$.

p-MC(MSO, $\mathcal{C}$) is *fixed-parameter tractable* (fpt), if there exists a computable function $f : \mathbb{N} \to \mathbb{N}$ and a $c \in \mathbb{N}$ such that for all $G \in \mathcal{C}$ and $\varphi \in$ MSO$[\sigma]$, $G \models \varphi$ can be decided in time $f(|\varphi|) \cdot |G|^c$. The problem is in the class XP, if it can be decided in time $|G|^{f(|\varphi|)}$.

In Example 3.3 we have seen that the NP-complete 3-Colourability problem is definable in MSO. Hence, MC(MSO, Graphs), the model-checking problem for MSO on the class of all graphs, is not fixed-parameter tractable unless $P = $ NP. However, Courcelle [3] proved that if we restrict the class of admissible input graphs, then we can obtain much better results. Recall the definition of tree-width of structures from Definition 3.1.

Theorem 4.2 ([3])**.** *There is an algorithm which, given a graph G in its incidence representation and an MSO-formula φ, decides "$G \models \varphi$?" in time $f(|\varphi| + \mathrm{tw}(G)) \cdot |G|$.*

Hence, MC(MSO, $\mathcal{C}$) is fixed-parameter tractable on any class $\mathcal{C}$ of structures of tree-width bounded by a constant.

Courcelle's theorem gives a sufficient condition for MC(MSO, $\mathcal{C}$) to be tractable. The obvious counterpart are sufficient conditions for intractability, i.e. what makes MSO-model checking hard? Garey, Johnson and Stockmeyer [13] proved that 3-Colourability remains

NP-hard on the class of planar graphs of degree at most 4. It follows that unless $P = \text{NP}$, MC(MSO, PLANAR) is not fixed-parameter tractable, where PLANAR denotes the class of planar graphs. However, this result only indirectly relates tractability of MSO model-checking on a class $\mathcal{C}$ to its tree-width. It would therefore be interesting to investigate whether Courcelle's theorem can be extended to class of unbounded tree-width or conversely, which bounds on the tree-width of a class $\mathcal{C}$ prohibit tractable MSO-model-checking. As we have seen above, large tree-width of graphs implies the existence of large grid-minors and it is well-known that MSO-model checking is hard on the class of grids. We will make use of this fact below and therefore repeat the statement here.

Definition 4.3. Recall from Definition 3.2 the signature $\sigma_G := \{V, E, \in, C_0, C_1\}$ of coloured grids, where V, E, C_0, C_1 are unary relation symbols and $\in$ is a binary relation symbol. A σ_G-structure G is a *coloured $l \times l$-grid* if its σ_{inc}-reduct $W_{|\{V,E,\in\}}$ is an $l \times l$-grid.

G *encodes a word* $w := w_1 \dots w_n \in \Sigma^n$ *with power* d if $l \geq n^d$, and $C_0 \cap C_1 = \varnothing$ and if $\{v_{1,i} : 1 \leq i \leq l\}$ are the vertices on the bottom row then $v_{1,i} \in C_0$ if $w_i = 0$ and $v_{1,i} \in C_1$ if $w_i = 1$, for all $1 \leq i \leq n$.

The following theorem is a well-known fact about the complexity of MSO.

Theorem 4.4. *For $d \geq 2$ let $\mathfrak{G}_d$ be the class of coloured grids encoding words with power d. Then* MC(MSO, $\mathfrak{G}_d$) *is not in XP unless $P = \text{NP}$.*

The theorem follows immediately from the following lemma, whose proof is standard.

Lemma 4.5. *Let M be a non-deterministic n^d-time bounded Turing-machine. There is a formula $\varphi_M \in \text{MSO}$ such that for all words $w \in \Sigma^\star$, if G is a coloured grid encoding w with power d, then $W \models \varphi_M$ if, and only if, M accepts w. Furthermore, the formula φ_M can be constructed effectively from M. The same holds if M is an alternating Turing-machine with a bounded number of alternations, as they are used to define the polynomial-time hierarchy.*

Proof sketch. The main idea of the proof is to use existential set quantification and the grid to guess the time-space diagram of a successful run R of the Turing-machine M on input w. Figure 2 illustrates this idea.

The grid on the left hand side encodes the word 010 through the three vertices in C_0 and C_1. We can then use existentially quantified monadic second-order variables Q_0, Q_1, Q_2, Q_f, $S_0, S_1, S_\square$ so that Q_s contains a vertex (i, j) if the Turing machine M would be in state q_s after i steps in the run R with the read/write head scanning position j. A vertex (i, j) appears in S_0 if after i steps the tape cell j contains symbol 0, and likewise for $S_1, S_\square$ denoting cells containing 1 and the blank symbol $\square$.

That these existentially quantified variables indeed encode a valid and accepting run of M on input w can easily be formalised in first-order logic, as the content of position (i, j) only depends on the content of $(i-1, j-1), (i-1, j), (i-1, j+1)$ and hence is a local property.

The reason we use a grid encoding a word with *power d* is that we need the grid to be large enough so that we can guess the complete run of the machine M on input w, and if M is n^d time bounded, then it can use up to n^d steps and n^d tape cells. □

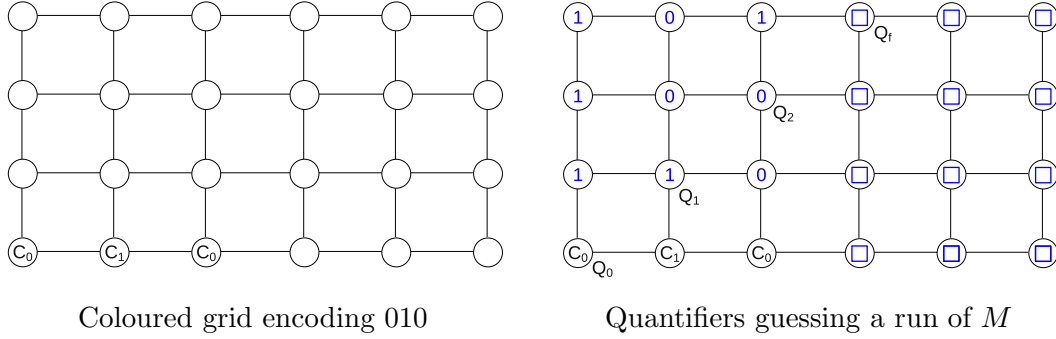Coloured grid encoding 010   Quantifiers guessing a run of M

Figure 2: Guessing a run of a Turing-machine in MSO.

5. A High Level Description of the Main Proof

In this section we give a high level description of the proof of the main theorem 1.3. We want to show that if $\mathcal{C}$ is a class of σ-structures whose tree-width is $(\log^c n, p)$-unbounded, for some large enough c and polynomial p, and which satisfies the conditions of the theorem, then model-checking for MSO is not in XP on the class $\mathcal{C}$. At the core of the proof is a reduction from MSO model-checking on $\mathcal{C}$ to model-checking of MSO on the class of coloured grids which we have already seen to be intractable. We now present a first idea of how to do this. The idea will not work but it helps to illustrate how the theorem is actually proved.

We show intractability of p-MC(MSO,$\mathcal{C}$) by reducing an NP-complete problem P to p-MC(MSO,$\mathcal{C}$) as follows. Given a word w of length n, we choose a graph $G \in \mathcal{C}$ of large enough tree-width. By the excluded grid theorem 2.6, G contains a large grid minor. Such a grid-minor can be defined in monadic second-order logic: we have already seen how to say that a graph is a grid, all we need to do is to extend this to say that a graph contains a grid-minor. This requires some work, but can be done. As $\mathcal{C}$ is closed under colourings, we can use vertex colours to encode the word w in this grid-minor as indicated in the previous section. Hence, given w we have constructed a graph $G \in \mathcal{C}$ of large enough tree-width and from this get a graph G_w with a large grid-minor encoding the word w. Furthermore, this grid-minor encoding w can be defined by MSO-formulas, more precisely there is an MSO-transduction taking the graph G_w and mapping it to the coloured grid H encoding w. Hence, if M is a Turing-machine deciding P, we can now use the formula φ_M constructed in Lemma 4.5 such that $H \models \varphi_M$ if, and only, if M accepts the word w if, and only if, $w \in P$. By definition of transductions, this gives us a formula ψ_M which is true in G_w if, and only if, φ_M is true in H if, and only if, $w \in P$.

Now, using the conditions 1 and 2 of (f,p)-unboundedness, we get that we can always find such graphs G and G_w efficiently. Furthermore, as $\mathcal{C}$ is closed under colourings, G_w is also in $\mathcal{C}$. Hence, if p-MC(MSO,$\mathcal{C}$) was in XP, i.e. $G_w \models \psi_M$ could be decided in time $|G_w|^{f(|\psi_M|)}$, then P could be decided in polynomial time as φ_M does not depend on the input w and the exponent is therefore fixed.

The problem with this approach is that the tree-width of G is only logarithmic in $|G|$ and hence G, and thus G_w, can be of size exponential in its tree-width. Furthermore, the best known bound for the size of grids we are guaranteed to find by the excluded grid theorem is only logarithmic in the tree-width of the graph. Hence, in order to guarantee that G contains a grid of size $|w|$ we would need to construct a graph of tree-width exponential

in the length $|w|$ of w which could therefore be of double exponential size in $|w|$. This completely destroys the argument above, as deciding $G_w \models \psi_M$ in time $|G_w|^{f(|\psi_M|)}$ only yields that we can decide $w \in P$ in time doubly exponential in w and this certainly can be done for NP-problems.

To get the result we want, we need to find grids of size polynomial in the tree-width of G. For, suppose for every graph G we could find a grid of size polynomial in its tree-width. Then, given w we could use the conditions of (f, p)-unboundedness to construct a graph G of tree-width polynomial in w, and hence containing a grid of size $|w| \times |w|$, whose size is bounded by $2^{o(|w|)}$ (this will be explained in detail in Section 8). We could then colour this grid to encode w as before to obtain G_w. Now, if $G_w \models \psi_M$ could be decided in time $|G_w|^{f(|\psi_M|)}$, then this would imply that $w \in P$ could be decided in time $\left(2^{o(w)}\right)^{f(|\psi_M|)}$ which is the same as $\left(2^{f(|\psi_M|) \cdot o(w)}\right)$ and hence in time sub-exponential in w. And sub-exponential solvability of NP-complete problems in case p-MC(MSO, $\mathcal{C}) \in$ XP is exactly what we claim in Theorem 1.3.

Obtaining sub-exponential time algorithms for problems such as TSP or SAT is an important open problem in complexity theory and the common assumption is that no such algorithms exist. This has led to the *exponential-time hypothesis* (ETH) which says that there is no such sub-exponential time algorithm for SAT, a hypothesis widely believed in the community.

Hence, to prove our main result we need to find grids of size polynomial in the tree-width of graphs. The existence of such grids is a major open problem in structural graph theory and remains open to date. Instead of grids we will therefore use a replacement structure for grids, called *grid-like minors*, recently introduced by Reed and Wood [25]. A grid-like minor of order l in a graph G is a pair $\mathcal{P}, \mathcal{Q}$ of sets of pairwise disjoint paths such that their intersection graph $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ contains an $l \times l$-grid as a minor. It was shown in [25] that every graph G contains a grid-like minor of order polynomial in its tree-width (see the next section for details).

Our method for proving Theorem 1.3 is therefore exactly as outlined above, only that instead of defining grid-minors and colouring them appropriately, we will define grid-like minors and colour those appropriately. This, however, is significantly more complicated than the case of grid-minors.

One of the problems is that the grid-like minor is actually a grid-minor of the intersection graph of two sets of pairwise disjoint paths. Hence, to define it in MSO we will have to define these sets of disjoint paths, then define their intersection graph and then define a grid-minor in it. This already is somewhat more complicated than defining pure grids.

The second, and major, challenge is to colour this grid-like minor so that it encodes a word w. For this, we need to colour the vertices and edges of the graph G so that this induces an appropriate colouring of the grid-minor of the intersection graph of two sets of disjoint paths. All this needs to be done in a way that once we have coloured the vertices and edges of G, there are no two different grid-like minors in G for which the colouring induces different words.

For this, we will define a combinatorial structure, called pseudo-walls, and show that every graph G contains a pseudo-wall of order polynomial in the tree-width of G, that we can colour the graph G in a way that it induces a unique colouring of this pseudo-wall, that we can define the pseudo-wall in MSO and, finally, that we can define an appropriately coloured grid in this pseudo-wall. Pseudo-walls and their colourings are defined in Section 6.
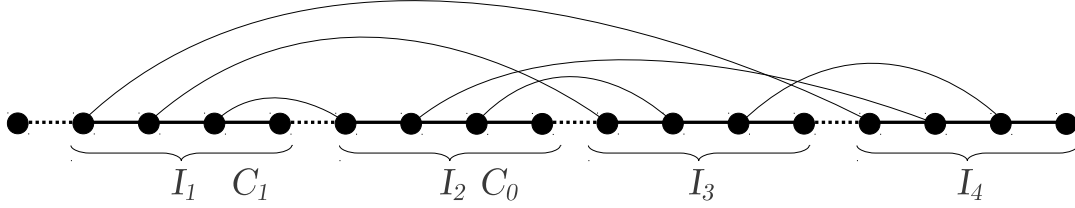
Figure 3: Simple pseudo-wall

Definability of these structures in MSO is proved in Section 7. Finally, we complete the proof in Section 8.

6. Pseudo-Walls in Graphs

This section contains the graph theoretical and algorithmic aspects of the proof outlined in the previous section. We first define the notions of simple and complex pseudo-walls and show that any graph of large enough tree-width can be expanded to a σ_{col}-structure containing either a simple or complex pseudo-wall of large order.

A simple pseudo-wall is a structure as illustrated in Figure 3. Essentially, it consists of a long path L whose edges are coloured either red (solid horizontal lines in the figure) or blue (dashed lines) together with a set $\mathcal{Q}$ of pairwise vertex-disjoint paths (represented by the curved lines in the figure). The first edge of L is blue but the last is red so that this gives the path a direction. Furthermore, the blue edges partition the path into segments formed by the red edges and for any pair of such segments there is a path in $\mathcal{Q}$ linking them. All vertices in a segment have the same colour with respect to C_0, C_1, i.e. they are either all in C_1 or all in C_0 or all uncoloured. Finally, the vertices coloured by $C_0 \cup C_1$ occur to the left of the long path L. This will allow us to define a coloured clique from a simple pseudo-wall where the vertices of the clique are formed by the red segments of L and the edges are defined by the paths in $\mathcal{Q}$. Formally, a simple pseudo-wall is defined as follows.

Definition 6.1 (Simple Pseudo-Wall). A *simple pseudo-wall of order k* is a σ_{col}-structure $\mathfrak{A} := (A, V^{\mathfrak{A}}, E^{\mathfrak{A}}, \in^{\mathfrak{A}}, B^{\mathfrak{A}}, R^{\mathfrak{A}}, C_0^{\mathfrak{A}}, C_1^{\mathfrak{A}})$ defined as follows. Let $h := \binom{k}{2}$.

- $A := \{v_1, \ldots, v_{s+1}, e_1, \ldots, e_s, \} \cup \{u_j^i, e_l^i : 1 \le i \le h, 1 \le j \le s_i + 1, 1 \le l \le s_i\}$ for some $s, s_1, \ldots, s_h > 0$.
- $V^{\mathfrak{A}} := \{v_1, \ldots, v_{s+1}, u_j^i : 1 \le i \le h, 1 \le j \le s_i + 1\}$.
- $E^{\mathfrak{A}} := \{e_1, \ldots, e_s, e_j^i : 1 \le i \le h, 1 \le j \le s_i\}$.
- $\in^{\mathfrak{A}} := \{(a, b) : a \in V, b \in E, a \in b\}$.
- $L := (v_1, e_1, v_2, \ldots, e_s, v_{s+1})$ forms a path of length s.
- There is a tuple $I := (i_1, \ldots, i_k)$ of indices $1 \le i_j \le s$, for $1 \le j \le k$, such that $i_1 := 1$, $i_j < i_{j+1} < i_j + h$, for all $1 \le j < h$, and $i_h < s$. For all $1 \le j \le k$, we call $\{v_{i_j+1}, \ldots, v_{i_{j+1}}\} \subseteq V(P)$ the *j-th interval $\mathcal{I}_j$ of P*, where we set $i_{h+1} := s + 1$. Then $R^{\mathfrak{A}} := \{e_l : l \notin I\}$ and $B^{\mathfrak{A}} := \{e_{i_j} : i_j \in I\}$.
- $C_0^{\mathfrak{A}}, C_1^{\mathfrak{A}} \subseteq \{v_1, \ldots, v_{s+1}\}$ are pairwise disjoint sets such that for all $1 \le j \le k$ and $C \in \{C_0, C_1\}$, either $\mathcal{I}_j \subseteq C^{\mathfrak{A}}$ or $\mathcal{I}_j \cap C^{\mathfrak{A}} = \varnothing$. Furthermore, for all j, if $\mathcal{I}_j \subseteq C_0 \cup C_1$ and $i < j$ then $\mathcal{I}_i \subseteq C_0 \cup C_1$.

- $\mathcal{Q} := \{(u_1^i, e_1^i, \ldots, e_{s_i}^i, v_{s_i}^i) : 1 \leq i \leq h\}$ forms a set of pairwise disjoint paths $P_{i,j}$, $1 \leq i < j \leq k$, such that $P_{i,j}$ links $\mathcal{I}_i$ and $\mathcal{I}_j$, i.e. $u_1^i \in \mathcal{I}_i$, $u_{s_i+1}^i \in \mathcal{I}_j$ and $u_j^i \in^{\mathfrak{A}} e_j^i$ and $u_j^i \in^{\mathfrak{A}} e_{j-1}^i$ for all suitable j.

Let $l \leq k$ be maximal with $\mathcal{I}_l \subseteq C_0 \cup C_1$. The *word w encoded by* $\mathfrak{A}$ is the sequence $w := w_1, \ldots, w_l \in \{0,1\}^*$ with $w_i := 1$ if $\mathcal{I}_i \subseteq C_1^{\mathfrak{A}}$ and $w_i := 0$ if $\mathcal{I}_i \subseteq C_0^{\mathfrak{A}}$.

Note that the intersection graph of the set $\mathcal{Q}$ and the set of paths comprising the intervals forms a complete graph on k vertices. The colouring of intervals by C_1 and C_0, respectively, yields a colouring of this clique in an obvious way. We will show in the next section that if a σ_{col}-structure $\mathfrak{B}$ contains such a simple pseudo-wall $\mathfrak{A}$ encoding a word w as sub-structure, we can use this in a similar way to Section 3 to simulate the run of a Turing machine on input w.

However, we may not always be able to find sufficiently large simple pseudo-walls in a σ_{col}-structure. Instead we may have to settle for a more complicated structure, called complex pseudo-walls.

Definition 6.2 (Complex Pseudo-Wall). A *complex pseudo-wall of order k* is a σ_{col}-structure $\mathfrak{A} := (A, V^{\mathfrak{A}}, E^{\mathfrak{A}}, \in^{\mathfrak{A}}, B^{\mathfrak{A}}, R^{\mathfrak{A}}, C_0^{\mathfrak{A}}, C_1^{\mathfrak{A}})$ defined as follows.

- $A := \{v_1, \ldots, v_{s+1}, e_1, \ldots, e_s,\} \cup \{u_j^i, e_l^i : 1 \leq i \leq r_1 + r_2, 1 \leq j \leq s_i + 1, 1 \leq l \leq s_i\}$ for some $r_1, r_2, s, s_1, \ldots, s_r > 0$.
- $V^{\mathfrak{A}} := \{v_1, \ldots, v_{s+1}\} \cup \{u_j^i : 1 \leq i \leq r_1 + r_2, 1 \leq j \leq s_i + 1\}$.
- $E^{\mathfrak{A}} := \{e_1, \ldots, e_s\} \cup \{e_j^i : 1 \leq i \leq r_1 + r_2, 1 \leq j \leq s_i\}$.
- $L := (v_1, e_1, v_2, \ldots, e_s, v_{s+1})$ forms a path of length s.
- $B^{\mathfrak{A}} := \{e_1\}$.
- $R^{\mathfrak{A}} := \{e_2, \ldots, e_s\}$.
- $C_0^{\mathfrak{A}}, C_1^{\mathfrak{A}} \subseteq \{v_1, \ldots, v_{s+1}\}$ and $C_0^{\mathfrak{A}} \cap C_1^{\mathfrak{A}} = \varnothing$.
- $\mathcal{P} := \{(u_1^i, e_1^i, \ldots, e_{s_i}^i, v_{s_i}^i) : 1 \leq i \leq r_1\}$ and $\mathcal{Q} := \{(u_1^i, e_1^i, \ldots, e_{s_i}^i, v_{s_i}^i) : r_1 < i \leq r_2\}$ form sets of pairwise disjoint paths $P_i := (v_1^i, e_1^i, \ldots, u_{s_i+1}^i)$ and $Q_i := (u_1^i, e_1^i, \ldots, e_{s_i}^i, v_{s_i}^i)$ so that every path $P \in \mathcal{P}$ intersects L in one endpoint of P but has no other vertex with L in common.

  Furthermore, $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ contains an image μ of a complete graph K_{k^2} as topological minor such that if $U := \{v_1, \ldots, v_{s+1}\} \cap (C_0^{\mathfrak{A}} \cup C_1^{\mathfrak{A}})$ then for each $u \in U$ there is a branch set μ_u containing a path $P \in \mathcal{P}$ with one endpoint being u and if $u \neq u' \in U$ then $\mu_u \cap \mu_{u'} = \varnothing$.

Let $i_1, \ldots, i_n$ be the indices of the vertices $v_i \in C_0^{\mathfrak{A}} \cup C_1^{\mathfrak{A}}$. The *word w encoded by* $\mathfrak{A}$ is $w := w_1, \ldots, w_n$ where $w_j := 1$ if $v_{i_j} \in C_1^{\mathfrak{A}}$ and $w_j := 0$ if $v_{i_j} \in C_0^{\mathfrak{A}}$.

Figure 4 illustrates a complex pseudo-wall encoding the word 1010. Here, the horizontal lines and bullets form the path P. The dashed line at the top-left indicates the "blue" edge e_1 and the horizontal solid lines the other edges $e_2, \ldots, e_s$. The vertical lines indicate the paths in $\mathcal{P}$ and the curved lines the paths in $\mathcal{Q}$. The grey areas represent the branch sets of the clique minor. Note that the figure is only an illustration as the paths in $\mathcal{Q}$ as displayed in the figure do not generate an intersection graph with a large clique-minor as required by complex pseudo-walls.

The motivation behind complex pseuo-walls is that the path P is used to encode a word w. The "blue" edge e_1 only serves the purpose of giving the path P an orientation, with e_1 marking the left end of P so that the word encoded in the wall is always read in the
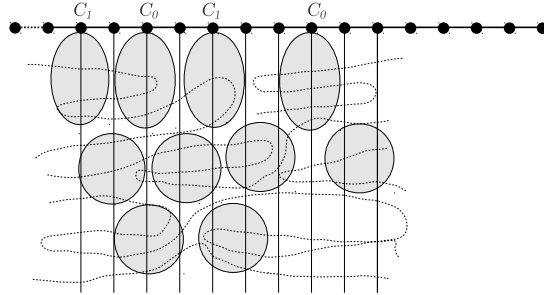
Figure 4: Complex pseudo-wall

correct order. The sets $\mathcal{P}$ and $\mathcal{Q}$ form an intersection graph containing a topological clique minor. The requirement that every coloured vertex occurs in exactly one branch set of this minor ensures that we can assign colours to the branch sets and therefore, given a complex pseudo-wall, we can define from it a vertex coloured clique. Furthermore, we can define an order on the vertices of this clique induced by the order defined by the path L. This will be enough to define a coloured grid in this clique which encodes the same word as the original complex pseudo-wall. Details of this construction will be given in the next section.

Definition 6.3 (pseudo-walls)**.** A σ_{col}-structure $\mathfrak{A}$ is a *pseudo-wall of order k encoding a word w* if it a simple or complex pseudo-wall of order k encoding w.

We will see later that pseudo-walls in σ_{col}-structures can be defined in MSO. The main result of this section is the following theorem showing that any graph G can be expanded, or coloured, to a σ_{col}-structure $\mathfrak{A}$ containing a pseudo-wall of order polynomial in the tree-width of G.

Theorem 6.4. *There is a polynomial-time algorithm and a constant c such that given a graph G such that*

$$\left(\frac{\mathrm{tw}(G)}{\sqrt{\log \mathrm{tw}(G)}}\right)^{\frac{1}{3}} \geq c \cdot m^7 + 1$$

and a word $w \in \{0,1\}^*$ of length at most m computes a σ_{col}-expansion of G containing a pseudo-wall of order m encoding w.

In [25] Reed and Wood consider an alternative to grid-minors as obstructions to small tree-width which they call *grid-like* minors. A grid-like minor of order l in a graph G is a set $\mathcal{P}$ of paths in G such that the intersection graph $\mathcal{I}(\mathcal{P})$ contains a K_l-minor. Reed and Wood's proof is existential, in that it does not directly give a way of computing grid-like minors. In [21], Kreutzer and Tazari show that the individual parts of this proof can be made algorithmic and a polynomial-time algorithm for computing grid-like minors is given.

Grid-like minors are the key to finding pseudo-walls. However, we cannot use Reed and Wood's result directly but have to adapt their proof slightly to get the structures we need. The following is essentially the proof from [25] and the algorithmic components from [21] needed to make it algorithmic, suitably adapted to yield pseudo-walls instead of grid-like minors.

The starting point of the proof are brambles. By Theorem 2.5, every graph G contains a bramble of order $\mathrm{tw}(G) + 1$. However, these can be of size exponential in $|G|$ and, as proved by Grohe and Marx [16], there is an infinite family of graphs where brambles of

optimal order necessarily are of exponential size. However, if we settle for brambles whose order is only polynomial in the tree-width, polynomial size can always be guaranteed. The existence of such brambles was proved in [16], a polynomial-time algorithm for computing them was given in [21].

Theorem 6.5 ([16, 21]). *There exists a polynomial time algorithm which, given a graph G, constructs a bramble in G of size $O(\mathrm{tw}(G))$ and order $\Omega((\frac{\mathrm{tw}(G)}{\sqrt{\log \mathrm{tw}(G)}})^{1/3})$.*

We first need the following lemma, whose simple proof is included for the reader's convenience.

Lemma 6.6 (*Birmelé, Bondy, Reed* [1]). *Let $\mathcal{B}$ be a bramble in a graph G. Then G contains a path intersecting every element in $\mathcal{B}$.*

Proof. Choose a bramble element $B \in \mathcal{B}$ and a vertex $v \in V(B)$. We initialise a path $P := (v)$ and maintain the invariant that for one endpoint u of P there is a bramble element $B \in \mathcal{B}$ such that $V(B) \cap V(P) = \{u\}$. The invariant trivially holds for $P = (v)$. So suppose such a path P has been constructed and let u be the endpoint of P as stated in the invariant. While there still is a bramble element $B' \in \mathcal{B}$ not containing a vertex of P choose a path P' from u to B' in $B \cup B'$ as short as possible. Such as path exists as B and B' touch. As P' is chosen as short as possible, one endpoint of P' is the only element of P' in B'. Further, as u is the only element of P in B, $P \cdot P'$, i.e. the path obtained from adding P' to P at the vertex u is still a path satisfying the invariant. We proceed until there are no bramble elements left which have an empty intersection with P. □

Clearly, if P is a path in G intersecting every element of a bramble $\mathcal{B}$ then the length of P must be at least the order of $\mathcal{B}$.

Lemma 6.7 (*Reed and Wood* [25]). *Let G be a graph containing a bramble $\mathcal{B}$ of order at least kl, for some $k, l \geq 1$. Then G contains l pairwise vertex disjoint disjoint paths $P_1, \ldots, P_l$ s.t. for all $1 \leq i < j \leq l$, G contains k parwise vertex disjoint paths between P_i and P_j.*

Proof. By Lemma 6.6, there is a path $P := (v_1 \ldots v_n)$ in G intersecting every element of $\mathcal{B}$ and hence of length at least kl. For $1 \leq i \leq j \leq n$ let $P_{i,j}$ be the sub-path of P induced by $\{v_i, \ldots, v_j\}$. Let t_1 be the minimal integer such that the sub-bramble $\mathcal{B}_1 := \{B \in \mathcal{B} : B \cap P_{1,t_1} \neq \varnothing\}$ has order k. Given $t_i, \mathcal{B}_i$ with $i < l$, let t_{i+1} be the minimal integer such that the sub-bramble $\mathcal{B}_{i+1} := \{B \in \mathcal{B} : B \cap P_{t_{i+1}, t_{i+1}} \neq \varnothing, B \cap P_{1,t_i} = \varnothing\}$ has order k. Since $\mathcal{B}$ has order kl, in this way we obtain integers $t_1 < t_2 < \cdots < t_l \leq n$. Let $P_i := P_{t_{i-1}+1, t_i}$, where $t_0 := 0$. By construction, the P_i are pairwise disjoint.

Suppose there is a set $S \subseteq V(G)$ of cardinality $|S| < k$ separating some P_i and P_j. Hence, S is neither a hitting set of $\mathcal{B}_i$ nor of $\mathcal{B}_j$ and hence there is $B_i \in \mathcal{B}_i$ and $B_j \in \mathcal{B}_j$ such that $S \cap (B_i \cup B_j) = \varnothing$. As B_i and B_j touch it follows that S does not separate B_i and B_j and therefore does not separate P_i and P_j. Hence, any set separating P_i and P_j must be of cardinality at least k.

By Menger's theorem 2.1, the minimal cardinality of a set separating P_i and P_j is equal to the maximum number of pairwise vertex disjoint paths between P_i and P_j, and hence there are at least k pairwise vertex disjoint paths between P_i and P_j as required. □

A graph G is *d-degenerated* if every subgraph of G contains a vertex of degree at most d. Mader [22] proved that every graph with no K_l-minor is 2^{l-2}-degenerated. Let $d(l)$ be the minimal integer such that every graph with no K_l-minor is $d(l)$-degenerated. Kostocha and, independently, Thomason showed that $d(l) \in \theta(l\sqrt{\log l})$. Bollobás and Thomason [2] proved that there is a constant c so that if a graph has average degree at least cp^2 it contains a K_p as a topological minor. Here we need an algorithmic version of this result, proved in [21].

Theorem 6.8 ([2, 21]). *There is a constant d such that if a graph G has average degree at least dp^2, then G contains K_p as a topological minor. Furthermore, a model of K_p in G can be found in polynomial time.*

We are now ready to prove Theorem 6.4.

Proof of Theorem 6.4. Set $c := d$ where d is the constant from Theorem 6.8. Let $k := \binom{m}{2} \cdot \left(\binom{m}{2} - 1 \right) \cdot d \cdot m^2 + 1$. Let $w := \left(\frac{\mathrm{tw}(G)}{\sqrt{\log \mathrm{tw}(G)}} \right)^{\frac{1}{3}}$. Then $w \geq k \cdot m + 1$.

By Theorem 6.5, we can compute in polynomial time a bramble $\mathcal{B}$ in G of order at least $k \cdot m + 1$. Therefore, by Lemma 6.7, G contains a path A of length $k \cdot m + 1$. Fix one endpoint p of A and let e_0 be the unique edge of A incident to p. Then, $A \setminus \{p\}$ has length at least $k \cdot m$ and can be decomposed into m disjoint paths $P_1, \ldots, P_m$ and, for $1 \leq i < j \leq m$, G contains a set $\mathcal{Q}_{i,j}$ of k disjoint paths between P_i and P_j. The edge e_0 needs to be set aside for the case of simple pseudo-walls below.

For $1 \leq i < j \leq m$ and $1 \leq a < b \leq m$ such that $\{i,j\} \neq \{a,b\}$, let $H_{i,j,a,b} := \mathcal{I}(\mathcal{Q}_{i,j}, \mathcal{Q}_{a,b})$ be the intersection graph of $\mathcal{Q}_{i,j} \cup \mathcal{Q}_{a,b}$.

Complex Pseduo-Walls. Suppose there are i, j, a, b as above such that $H_{i,j,a,b}$ has a subgraph of average degree at least $d \cdot m^2$. We define a σ_{col}-expansion $\mathfrak{A}$ of G which contains a complex pseudo-wall of order m encoding w as follows.

By Theorem 6.8, $H := H_{i,j,a,b}$ contains a K_m as topological minor and we can compute an image of it in polynomial time. Set $L := P_i$. Fix one endpoint of L and let e_1 be the edge incident to it in L. We define $B^{\mathfrak{A}} := \{e_1\}$ and $R^{\mathfrak{A}} := E(L) \setminus \{e_1\}$. This defines a direction on L where the endpoint incident to e_1 is the left-most, or smallest.

This direction induces an ordering $\sqsubset$ on the paths in $\mathcal{Q}_{i,j}$ where for $P, P' \in \mathcal{Q}_{i,j}$ we define $P \sqsubset P'$ if $V(P) \cap V(L)$ is smaller than $V(P') \cap V(L)$. (Note that any $P \in \mathcal{Q}_{i,j}$ has exactly one vertex in common with L, which is its endpoint in L.)

Let $X_1, \ldots, X_m$ be the connected subgraphs in H constituting the image of K_m in H. W.l.o.g. we assume that each X_i contains a path from $\mathcal{Q}_{i,j}$. (There can only be at most one X_i consisting of a single path from $\mathcal{Q}_{a,b}$.) For each $1 \leq i \leq m$ let p_i be the smallest vertex in $V(L)$ with respect to $\sqsubset$ contained in a path in X_i. We order the sets X_i by letting $X_i < X_j$ if $p_i \sqsubset p_j$. W.l.o.g. we assume that $X_1 < X_2 < \cdots < X_m$. Then, $C_0^{\mathfrak{A}} := \{p_i : 1 \leq i \leq |w|, w_i = 0\}$ and $C_0^{\mathfrak{A}} := \{p_i : 1 \leq i \leq |w|, w_i = 1\}$, where $w := w_1, \ldots, w_{|w|}$

It is now immediately clear from the construction, that $\mathfrak{A}$ contains a complex pseudo-wall of order m encoding w: the wall is constituted by L, $\mathcal{P} := \mathcal{Q}_{i,j}$ and $\mathcal{Q} := \mathcal{Q}_{a,b}$ and the colours $R^{\mathfrak{A}}, B^{\mathfrak{A}}, C_0^{\mathfrak{A}}, C_1^{\mathfrak{B}}$.

Simple Pseduo-Walls. Now suppose that the average degree of all sub-graphs of $H_{i,j,a,b}$, where $\{i,j\} \neq \{a,b\}$, is less than $d \cdot m^2$, i.e. all $H_{i,j,a,b}$ are $d \cdot m^2$-degenerated.

Let H be the intersection graph of $\bigcup\{\mathcal{Q}_{i,j} : 1 \leq i < j \leq m\}$. Obviously, H is $\binom{m}{2}$-colourable with each $\mathcal{Q}_{i,j}$ being a colour class. Each colour class has k vertices and each pair of colour classes induce a $d \cdot m^2$-degenerated graph. The following lemma is from [25].

Lemma 6.9 ([25]). *Let $r \geq 2$ and let $V_1, \ldots, V_r$ be the colour classes in an r-colouring of a graph H. Suppose that $|V_i| \geq n := r(r-1)c + 1$, for all $1 \leq i \leq r$, and $H[V_i \cup V_j]$ is c-degenerated for distinct $1 \leq i < j \leq r$. Then there exists an independent set $\{x_1, \ldots, x_r\}$ of H such that each $x_i \in V_i$.*

Furthermore, a simple minimum-degree greedy algorithm will find such an independent set in polynomial time.

Applying the lemma to our setting, with $n = k$ and $r = \binom{m}{2}$ and $c = d \cdot m^2$, we obtain an independent set I in H with one vertex in each colour class and such a set can be found in polynomial time by a simple greedy algorithm. That is, in each set $\mathcal{Q}_{i,j}$ there is one path $Q_{i,j}$ such that $Q_{i,j} \cap Q_{a,b} = \varnothing$ whenever $\{i,j\} \neq \{a,b\}$.

We will now define a σ_{col}-expansion $\mathfrak{A}$ of G containing a simple pseudo-wall of order m encoding w. Consider the long path A constructed above. Recall that there is one edge e_0 of A incident to an endpoint p and that $L \setminus \{p\}$ is partitioned into $P_1, \ldots, P_m$. As $P_1, \ldots, P_m$ are pairwise disjoint, between any P_i and P_{i+1} there is one edge e_i of L not contained in $P_i \cup P_{i+1}$. Let $B^{\mathfrak{A}} := \{e_0, e_1, \ldots e_{m-1}\}$ and $R^{\mathfrak{A}} := E(A) \setminus B^{\mathfrak{A}}$. Furthermore, if $w := w_1, \ldots, w_l$, with $l \leq m$, then $C_0 := \bigcup\{V(P_i) : w_i = 0\}$ and $C_1 := \bigcup\{V(P_i) : w_i = 1\}$.

By construction, $\mathfrak{A}$ contains a simple pseudo-wall of order m encoding w, which is generated by the long path A, the colours $B^{\mathfrak{A}}, R^{\mathfrak{A}}, C_0^{\mathfrak{A}}, C_1^{\mathfrak{A}}$ and the paths in I.

This concludes the proof of Theorem 6.4.   □

7. Intractability of MSO on Pseudo-Walls

The main purpose of this section is to show that MSO is intractable on the class of pseudo-walls. For this purpose, we will lift Lemma 4.5 from grids to pseudo-walls.

To get the result we will exhibit a sequence of MSO-transductions that define coloured grids in pseudo-walls. To simplify the presentation, we will do so in several steps. Obviously, the transductions will be different for simple and complex pseudo-walls. The sequence of transductions works as follows. We will first exhibit a transduction defining coloured grids in coloured ordered cliques. We will then show that there are transductions defining coloured ordered cliques in simple and complex pseudo-walls, where in the latter we will need one further intermediate step.

7.1. **Coloured ordered cliques.** Recall from Definition 3.2 the signatures σ_{ord} and σ_G.

Definition 7.1. A *coloured ordered clique* is a σ_{ord}-structure $\mathfrak{A} := (U, V, E, \in, C_0, C_1, \leq)$ so that

- $(U, V, E, \in)$ is the incidence representation of a complete graph
- $\leq$ is a linear order on V and
- $C_0, C_1 \subseteq V$, $C_0 \cap C_1 = \varnothing$ and $C_0 \cup C_1$ forms an initial subset of $\leq$, i.e. there is a $v \in C_0 \cup C_1$ such that $C_0 \cup C_1 = \{u \in V : u \leq v\}$.

The *order* of $\mathfrak{A}$ is $|V|$. Let $v_1, \ldots, v_n$ be the vertices in $C_0 \cup C_1$ ordered by $\leq$. The word w *encoded by* $\mathfrak{A}$ is $w := w_1, \ldots, w_n$ where $w_i := 1$ if $v_i \in C_1$ and $w_i := 0$ if $w_i \in C_0$.

Lemma 7.2. *There is an* MSO-*transduction* Θ *from* σ_{ord} *to* σ_G *with parameters* $\mathcal{P}, \mathcal{Q}$ *such that if* $\mathfrak{A}$ *is a coloured ordered clique of order* k *encoding a word* w *then* $\Theta(\mathfrak{A})$ *contains a coloured* $(\sqrt{k} \times \sqrt{k})$-*grid encoding* w. *Furthermore, every* $\mathcal{B} \in \Theta(\mathfrak{A})$ *is a grid encoding* w.

Proof. We define the transduction $\Theta := (\varphi_{valid}, \varphi_{univ}, \varphi_{\sim}, \varphi_V, \varphi_E, \varphi_{\in}, \varphi_{C_0}, \varphi_{C_1})$ as follows.

The transduction is quite simple as the grid we seek to define is actually a sub-structure of the given coloured clique. The idea is that the parameters $\mathcal{P}, \mathcal{Q}$ will be enforced to be interpreted by two sets of pairwise vertex disjoint paths, the *vertical* and *horizontal* paths in a grid. All we need to say is that they indeed form a grid, that the bottom row of the grid contains all coloured vertices from left to right in the order given by $\leq$.

So let $\varphi_V(x) := x \in V$, $\varphi_E(x) := x \in E$ and $\varphi_{univ}(x) := \varphi_V \lor \varphi_E$. We set $\varphi_{\sim}(x, y) := x = y$. Furthermore, we define $\varphi_{C_0}(x) := x \in C_0$ and $\varphi_{C_1}(x) := x \in C_1$. What is left to define is φ_{valid}. Recall the formula $\varphi_{grid\text{-}border}(L, R, T, B, \mathcal{P}, \mathcal{Q})$ from Example 3.5 defining that $\mathcal{P}, \mathcal{Q}$ are two sets of pairwise vertex disjoint paths inducing a grid such that bottom, left, top, and right rows are B, L, T, R, respectively. We will also use the formulas *maxpath*, *ep* and *set-o-dis-path* defined in this example.

φ_{valid} will enforce $\mathcal{P}, \mathcal{Q}$ to be interpreted by sets of pairwise disjoint paths inducing a grid whose bottom row contains the coloured vertices in the correct order. As before we will therefore use the notation $Q \in \mathcal{Q}$ as shorthand for $Q \subseteq \mathcal{Q} \land maxpath(Q, \mathcal{Q})$.

$$
\begin{aligned}
\varphi_{valid}(\mathcal{P}, \mathcal{Q}) \quad := \quad &\text{set-o-dis-path}(\mathcal{P}) \land \text{set-o-dis-path}(\mathcal{Q}) \ \land \\
&\exists B, T \in \mathcal{Q} \ \exists L, R \in \mathcal{P} \ \big[\varphi_{grid\text{-}border}(\mathcal{P}, \mathcal{Q}, B, T, R, L) \ \land \\
&\exists x \in V(B) x \in C_0 \cup C_1 \land \forall y (y \in C_0 \cup C_1 \leftrightarrow y \in V(B) \land y \leq x) \ \land \\
&\forall x, y \in V(B) \big(\varphi_{\leq_B}(x, y) \leftrightarrow x \leq y \big) \big]
\end{aligned}
$$

where

$$
\begin{aligned}
\varphi_{\leq_B}(x, y, B) \quad := \quad &\exists u \in V(B) \land ep(u, B) \land \forall y u \leq y \ \land \\
&\forall P \subseteq B(path(P, B) \land u \in V(P) \land y \in V(P) \rightarrow x \in V(P)).
\end{aligned}
$$

The formula $\varphi_{\leq_B}$ states that one endpoint u of B is the $\leq$-smallest element in the structure and then defines a linear order on B where x is smaller than y if the distance from x to u in B is smaller than the distance from y to u. This is formalised by stating that any sub-path of B which contains u and y must also contain x.

φ_{valid} then states that $\mathcal{P}, \mathcal{Q}$ are sets of pairwise disjoint paths defining a grid with bottom row $B \in \mathcal{Q}$ and that the vertices in C_0 and C_1 all occur as an initial subpath on B in the order given by $\leq$. Hence, this grid encodes the same word as the initial structure.

This shows that every structure in $\Theta(\mathfrak{A})$ is a grid encoding w. Furthermore, if $\mathcal{P}, \mathcal{Q}$ are chosen as the vertical and horizontal paths in a $\sqrt{k} \times \sqrt{k}$-grid which exists as a sub-structure of the clique $\mathfrak{A}$, then $\Theta((\mathfrak{A}, \mathcal{P}, \mathcal{Q}))$ has order $\sqrt{k} \times \sqrt{k}$. This concludes the proof. $\qquad\square$

Corollary 7.3. *Let* M *be a non-deterministic* n^d-*time bounded Turing-machine. There is a formula* $\varphi_M \in$ MSO *such that for all words* $w \in \Sigma^\star$, *if* G *is a coloured ordered clique of order* $|w|^d$ *encoding* w, *then* $G \models \varphi_M$ *if, and only if,* M *accepts* w. *Furthermore, the formula* φ_M *can be constructed effectively from* M.

The same holds if M *is an alternating Turing-machine with a bounded number of alternations, as they are used to define the polynomial-time hierarchy.*

Proof. The corollary follows immediately from Lemma 4.5, Corollary 3.8 and the previous Lemma 7.2. $\qquad\square$

7.2. Simple Pseudo-Walls.

Lemma 7.4. *There is an* MSO-*transduction* Θ *from* σ_{col} *to* σ_{ord} *with parameters* $\mathcal{P}, \mathcal{Q}, L$ *such that if* $\mathfrak{A}$ *is a* σ_{col}-*structure containing a simple pseudo-wall of order* k *encoding a word* w *then* $\Theta(\mathfrak{A})$ *contains a coloured ordered clique of order* k *encoding* w *and all* $\mathfrak{B} \in \Theta(\mathfrak{A})$ *are coloured cliques encoding* w.

Proof. We define a transduction $\Theta := (\varphi_{valid}, \varphi_{univ}, \varphi_\sim, \varphi_V, \varphi_E, \varphi_\in, \varphi_{C_0}, \varphi_{C_1}, \varphi_\leq)$ as follows. Recall that a simple pseudo-wall consists of a long path L containing k "blue" edges which partition the path into k sub-paths $P_1, \ldots, P_k$ and a set $\mathcal{Q}$ of pairwise vertex disjoint paths such that for every pair $1 \leq i < j \leq k$ there is a path in $\mathcal{Q}$ linking P_i and P_j.

The parameters $\mathcal{P}, \mathcal{Q}$ will be enforced to be interpreted by sets of pairwise vertex disjoint paths and L will be enforced to be a simple path. The intended interpretation is that L is the long path, $\mathcal{P}$ are the segments of L without the blue edges and $\mathcal{Q}$ are the paths connecting the segments in $\mathcal{P}$. All this will be defined in φ_{valid}. But first we define the other formulas, where as usual we use the notation $P \in \mathcal{P}$ as shortcut for $P \subseteq \mathcal{P} \wedge maxpath(P, \mathcal{P})$.

Note, that the paths in $\mathcal{Q}$ may intersect various segments $P \in \mathcal{P}$. Hence, in principle every vertex of a segment $P \in \mathcal{P}$ can also be contained in some $Q \in \mathcal{Q}$. This means that we cannot take the vertices of $P \in \mathcal{P}$ to represent P in the transduction, as this would make it difficult to guarantee that $\varphi_\sim$ defines an equivalence relation. However, as every segment $P \in \mathcal{P}$ has a non-empty intersection with more than one path in $\mathcal{Q}$ and the paths in $\mathcal{Q}$ are pairwise disjoint, every $P \in \mathcal{P}$ must contain at least one edge not contained in any $Q \in \mathcal{Q}$. Similarly, every $Q \in \mathcal{Q}$ contains an edge not contained in any other path. We will therefore take these unique edges to represent P and Q, resp.

We define formulas $uni\text{-}edge_{\mathcal{P}}(e, P, \mathcal{P}, \mathcal{Q}) := e \in P \wedge \neg \exists Q \in \mathcal{Q} \; e \in E(Q)$ which defines an edge e to be an edge of P not contained in any path in $\mathcal{Q}$. Analogously we define $uni\text{-}edge_{\mathcal{Q}}(e, Q, \mathcal{P}, \mathcal{Q}) := e \in Q \wedge \neg \exists P \in \mathcal{P} \; e \in P$ and set

$$uni\text{-}edge(e, P) := \big(P \in \mathcal{P} \wedge uni\text{-}edge_{\mathcal{P}}(e, P)\big) \vee \big(P \in \mathcal{Q} \wedge uni\text{-}edge_{\mathcal{Q}}(e, P)\big).$$

Let $\varphi_V(x) := \exists P \in \mathcal{P} \; uni\text{-}edge_{\mathcal{P}}(x, P)$ and

$$\varphi_\sim^V(x, y) := \exists P \in \mathcal{P} \; uni\text{-}edge_{\mathcal{P}}(x, P) \wedge uni\text{-}edge_{\mathcal{P}}(y, P).$$

We define $\varphi_E(e) := \exists Q \in \mathcal{Q} \wedge uni\text{-}edge_{\mathcal{Q}}(e, Q)$ and

$$\varphi_\sim^E(x, y) := \exists Q \in \mathcal{Q} \; uni\text{-}edge_{\mathcal{Q}}(x, Q) \wedge uni\text{-}edge_{\mathcal{Q}}(y, Q).$$

Finally, $\varphi_\sim(x, y) := \varphi_\sim^V \vee \varphi_\sim^E$ and $\varphi_{univ} := \varphi_V \vee \varphi_E$. Note that φ_V and φ_E define disjoint sets and therefore $\varphi_\sim$ defines an equivalence relation on φ_{univ}.

To define the colours, we set

$$\varphi_{C_i}(x) := \exists P \in \mathcal{P} \wedge uni\text{-}edge_{\mathcal{P}}(x, P) \wedge \exists u \in V(P) \wedge u \in C_i,$$

for $i \in \{0, 1\}$.

The ordering is defined by

$$\begin{aligned}
\varphi_\leq(x, y) \; := \; & \exists p \in V(L) \exists e \in L \Big(ep(p, L) \wedge p \in e \wedge e \in B \; \wedge \\
& \forall P \subseteq L \big(path(P) \wedge p \in V(P) \wedge y \in E(P) \rightarrow x \in E(P) \big) \Big)
\end{aligned}$$

The formula $\varphi_{\leq}(x, y)$ first defines the endpoint p of the long path L which is incident to a blue edge in L (there is only one blue edge incident to an endpoint) and then defines x to be smaller than y if every sub-path P of L containing p and y also contains x. This defines the natural ordering on L where the blue edge marks the left, i.e. smaller, end.

Finally, we have to define the main formula φ_{valid} which will need to say that the parameters $\mathcal{P}, \mathcal{Q}, L$ indeed define a simple pseudo-wall as required. For this, we need to enforce the following requirements φ_1 to φ_5.

$$\begin{aligned}
\varphi_1 \quad := \quad & path(L) \ \wedge L = B \cup R \ \wedge \\
& \neg \exists u \in V(L) \exists e, e' \in L \big(e \neq e' \wedge e \in B \wedge e' \in B \wedge u \in e \wedge u \in e' \big) \ \wedge \\
& \exists^{=1} p \big(ep(p, L) \wedge \exists e \in L \wedge p \in e \wedge e \in B \big)
\end{aligned}$$

The formula φ_1 says that L is a path which consists exactly of the red and blue edges in the structure. Furthermore, in L no two blue edges $e, e' \in B$ are adjacent, i.e. between any two blue edges there is a red edge, and L has exactly one endpoint which is incident to a blue edge, i.e. the first edge on one end is blue but the last edge is red.

The formula

$$\varphi_2 := set\text{-}o\text{-}dis\text{-}path(\mathcal{P}) \wedge \forall P \subseteq E \Big(\big(P \subseteq L \setminus B \wedge maxpath(P, L \setminus B) \big) \leftrightarrow P \in \mathcal{P} \Big)$$

says that $\mathcal{P}$ contains exactly the connected components of $L \setminus B$, i.e. the segments of L defined by removing the blue edges.

The formula

$$\begin{aligned}
\varphi_3 \quad := \quad & set\text{-}o\text{-}dis\text{-}path(\mathcal{Q}) \wedge \forall P \neq P' \in \mathcal{P} \exists Q \in \mathcal{Q} \exists u, u' \in V(Q) \\
& \big(ep(u, Q) \wedge ep(u', Q) \wedge u \in V(P) \wedge u' \in V(P') \big)
\end{aligned}$$

says that $\mathcal{Q}$ is a set of pairwise vertex disjoint paths and that for any distinct pair $P, P' \in \mathcal{P}$ there is a path in $\mathcal{Q}$ linking P and P', i.e. having one endpoint in P and the other in P'.

Finally, we have to define that the colours are defined properly, i.e. that either all vertices of a path $\mathcal{P}$ have a colour, in this case it is the same colour for all, or none has a colour. Furthermore, we need to say that the coloured paths occur to the left of L, i.e. if a path $P \subseteq L$ contains a coloured vertex then so do all $P' \subseteq L$ which are closer to the end of L marked by a blue edge. This is formalised by the following formula

$$\begin{aligned}
\varphi_4 \quad := \quad & \forall P \in \mathcal{P} \big((V(P) \subseteq C_0 \vee V(P) \cap C_0 = \varnothing) \wedge (V(P) \subseteq C_1 \vee V(P) \cap C_1 = \varnothing) \big) \ \wedge \\
& \exists x \in C_0 \cup C_1 \forall y \in V(L) \big(y \in C_0 \cup C_1 \leftrightarrow \varphi_{\leq}(x, y) \big).
\end{aligned}$$

The last bit we have to specify is that B, R are colours of edges whereas C_0, C_1 are colours of vertices and that all colours are distinct. This is expressed by

$$\varphi_5 := \big(C_0 \subseteq V \wedge C_1 \subseteq V \wedge C_0 \cap C_1 = \varnothing \big) \wedge \big(R \subseteq E \wedge B \subseteq E \wedge R \cap B = \varnothing \big).$$

Putting everything together we get

$$\varphi_{valid} := \bigwedge_{i=1}^{5} \varphi_i.$$

Now, φ_{valid} forces the parameters $\mathcal{P}, \mathcal{Q}, L$ together to define a simple pseudo-wall in the structure and in this case, the various formulas define a coloured ordered clique encoding the same word as the pseudo-wall. Furthermore, the number of vertices in this clique is the same as the number of segments of L. Hence, there is a choice of parameters in $\mathfrak{A}$ where this number is the order k of the pseudo-wall. This concludes the proof. $\qquad \square$

As before, we get the following corollary.

Corollary 7.5. *Let M be a non-deterministic n^d-time bounded Turing-machine. There is a formula $\varphi_M \in \mathrm{MSO}$ such that for all words $w \in \Sigma^\star$, if $\mathfrak{A}$ is a σ_{col}-structure containind a simple pseudo-wall of order $|w|^d$ encoding w, then $\mathfrak{A} \models \varphi_M$ if, and only if, M accepts w. Furthermore, the formula φ_M can be constructed effectively from M.*

The same holds if M is an alternating Turing-machine with a bounded number of alternations, as they are used to define the polynomial-time hierarchy.

7.3. **Complex Pseudo-Walls.** We will now define a transduction from complex pseudo-walls to ordered coloured cliques. As complex pseudo-walls are more complex than simple ones, we will do so in two steps. We first exhibit a transduction with parameters $\mathcal{P}, \mathcal{Q}, L$ that will enforce $\mathcal{P}, \mathcal{Q}, L$ to satisfy the requirements of a complex pseudo-wall and will then generate the intersection graph of $\mathcal{P}$ and $\mathcal{Q}$ where vertices are suitably coloured as prescribed by the definition of a complex pseudo-wall. By definition of a complex pseudo-wall, this intersection graph contains a topological clique-minor. The second transduction, therefore, will generate a coloured ordered clique from this clique minor.

For the first step, we define a transduction

$$\Theta_1 := (\varphi_{valid}, \varphi_{univ}, \varphi_{\sim}, \varphi_V, \varphi_E, \varphi_{\in}, \varphi_{C_0}, \varphi_{C_1}, \varphi_{\leq})$$

from σ_{col} to σ_{ord} with parameters $\mathcal{P}, \mathcal{Q}, L$ as follows.

Again, φ_{valid} will ensure that $\mathcal{P}, \mathcal{Q}$ are interpreted by sets of pairwise vertex disjoint paths, so we will use previous notation such as $Q \in \mathcal{Q}$.

Recall that in the intersection graph $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ the vertices are the paths in $\mathcal{P}$ and $\mathcal{Q}$ and an edge exists between $P \in \mathcal{P}$ and $Q \in \mathcal{Q}$ if they intersect. Hence, in Θ_1 we will represent a path $P \in \mathcal{P}$ by its unique edges (see the previous subsection) and an edge $\{P, Q\}$ by the vertices in the intersection of P and Q.

Thus, we define $\varphi_V(x) := x \in E \wedge \big(\exists P \in \mathcal{P}\, uni\text{-}edge_{\mathcal{P}}(x, P) \vee \exists Q \in \mathcal{Q}\, uni\text{-}edge_{\mathcal{Q}}(x, Q)\big)$ and

$$\varphi_{\sim}^V(x, y) := \begin{array}{l} \exists P \in \mathcal{P}(uni\text{-}edge_{\mathcal{P}}(x, P) \wedge uni\text{-}edge_{\mathcal{P}}(y, P)) \vee \\ \exists Q \in \mathcal{Q}(uni\text{-}edge_{\mathcal{Q}}(x, Q) \wedge uni\text{-}edge_{\mathcal{Q}}(y, Q)). \end{array}$$

Furthermore, we define $\varphi_E(x) := \exists P \in \mathcal{Q} \exists Q \in \mathcal{Q}(x \in V(P) \cap V(Q))$ and

$$\varphi_{\sim}^E(x, y) := \exists P \in \mathcal{P} \exists Q \in \mathcal{Q}\, x \in V(P) \cap V(Q) \wedge y \in V(P) \cap V(Q).$$

Finally, we define $\varphi_{univ}(x) := \varphi_V(x) \vee \varphi_E(x)$ and $\varphi_{\sim}(x, y) := \varphi_{\sim}^V(x, y) \vee \varphi_{\sim}^E(x, y)$.

It is easily seen that $\varphi_{\sim}^V$ and $\varphi_{\sim}^E$ define equivalence relations on the sets defined by $\varphi_V(x)$ and $\varphi_E(x)$, resp., and as these sets are disjoint also on the set defined by φ_{univ}.

Let $\varphi_{\in}(x, e) :=$

$$\exists P \in \mathcal{P} \exists Q \in \mathcal{Q}\Big(e \in V(P) \wedge e \in V(Q) \wedge \big(uni\text{-}edge_{\mathcal{P}}(x, P) \vee uni\text{-}edge_{\mathcal{Q}}(x, Q)\big)\Big)$$

The formula states that e is a vertex in the intersection of a path $P \in \mathcal{P}$ and a path $Q \in \mathcal{Q}$, and therefore representing an edge between P and Q, and x is a unique edge of one of the two paths and hence represents a vertex for P or Q.

We define the colours C_0 and C_1 next. Here, we give a vertex $P \in \mathcal{P}$ the colour C_i if the (uniquely defined) endpoint of the path P in the long path L is in C_i. Recall that in a

complex pseudo-wall, every path P intersect L in exactly one of its endpoints. The colours are therefore defined by the formulas

$$\varphi_{C_i}(x) := \exists P \in \mathcal{P}\big(uni\text{-}edge_{\mathcal{P}}(x, P) \wedge \exists y \in V(P) \cap V(L) \wedge y \in C_i\big),$$

where $i \in \{0, 1\}$. Note that we do not need to state that y is an endpoint of P as P can intersect L only once.

Finally, we define an ordering on the vertices constituted by paths in $\mathcal{P}$. The ordering we aim at is the natural ordering given by L, where a path P is smaller than a path P' if the endpoint of P in L is closer to the blue edge in L than the endpoint of P' in L.

$$
\begin{aligned}
\varphi_{\leq}(x, y) \quad := \quad & \varphi_V(x) \wedge \varphi_V(y) \wedge \exists P, P' \in \mathcal{P}\, uni\text{-}edge_{\mathcal{P}}(x, P) \wedge uni\text{-}edge_{\mathcal{P}}(y, P') \wedge \\
& \exists u, u'\big(u \in V(P) \cap V(L) \wedge u' \in V(P') \cap V(L) \wedge \varphi_{\leq_L}(u, u')\big),
\end{aligned}
$$

where

$$
\begin{aligned}
\varphi_{\leq_L}(u, u') \quad := \quad & \exists e \in L\Big(e \in B \wedge \exists y \in V(L)\big(ep(y, L) \wedge y \in e \wedge \\
& \forall P \subseteq L\big(path(P) \wedge y \in V(P) \wedge u' \in V(P) \rightarrow u \in V(P)\big)\big)\Big).
\end{aligned}
$$

The last part of Θ_1 to be defined is φ_{valid}. Again we will do this in various steps.

$$\varphi_1 \quad := \quad path(L) \wedge (L = B \cup R) \wedge \exists^{=1} e \in B \wedge \exists^{=1} e \in B\big(\exists u(ep(u, L) \wedge u \in e)\big)$$

The formula says that L is a path comprising all red and blue edges and that there is exactly one blue edge and this is the first on the path.

The next formula φ_2 says that only vertices on L are coloured and that no vertex has two colours.

$$\varphi_2 := C_0 \cup C_1 \subseteq V(L) \wedge C_0 \cap C_1 = \varnothing$$

Finally, we need to say that $\mathcal{P}$ and $\mathcal{Q}$ are sets of pairwise disjoint paths and that each path in $\mathcal{P}$ has exactly one endpoint on L and is otherwise vertex disjoint from L. This is expressed in the next formula.

$$\varphi_3 := set\text{-}o\text{-}dis\text{-}path(\mathcal{P}) \wedge set\text{-}o\text{-}dis\text{-}path(\mathcal{Q}) \wedge \forall P \in \mathcal{P}\big(\exists x\, ep(x, P) \wedge V(P) \cap V(L) = \{x\}\big)$$

Now, we set $\varphi_{valid} := \varphi_1 \wedge \varphi_2 \wedge \varphi_3$.

Let $\mathfrak{A}$ be a complex pseudo-wall and let $\mathcal{P}, \mathcal{Q}, L$ be sets of edges such that $(\mathfrak{A}, \mathcal{P}, \mathcal{Q}, L) \models \varphi_{valid}$. Hence, $\mathcal{P}$ and $\mathcal{Q}$ are sets of vertex dispoint paths.

Let

$$
\begin{aligned}
U \quad &:= \quad \{[x]_{/\varphi_{\sim}(\mathfrak{A})} : x \in \varphi_{univ}(\mathfrak{A})\}, \\
V \quad &:= \quad \{[x]_{/\varphi_{\sim}(\mathfrak{A})} : x \in \varphi_V(\mathfrak{A})\}, \\
E \quad &:= \quad \{[x]_{/\varphi_{\sim}(\mathfrak{A})} : x \in \varphi_E(\mathfrak{A})\} \text{ and} \\
\in_U \quad &:= \quad \{([x]_{/\varphi_{\sim}(\mathfrak{A})}, [y]_{/\varphi_{\sim}(\mathfrak{A})}) : (x, y) \in \varphi_{\in}(\mathfrak{A})\}.
\end{aligned}
$$

By construction, $(U, V, E, \in_U)$ is isomorphic to the intersection graph $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ of $\mathcal{P}$ and $\mathcal{Q}$. Furthermore, $C_i := \{\{[x]_{/\varphi_{\sim}(\mathfrak{A})} : x \in \varphi_{C_i}(\mathfrak{A})\}$, for $i \in \{0, 1\}$, define colours of vertices in V and $\leq_V := \{[(x, y)]_{/\varphi_{\sim}(\mathfrak{A})} : (x, y) \in \varphi_{\leq}(\mathfrak{A})\}$ defines a linear order on the subset of the vertices of $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ corresponding to paths in $\mathcal{P}$.

By definition, if $\mathfrak{A}$ is a complex pseudo-wall, then we can choose $\mathcal{P}$ and $\mathcal{Q}$ so that $\mathcal{I}(\mathcal{P}, \mathcal{Q})$ contains a topological clique-minor such that every branch set contains at most one coloured vertex and all coloured vertices occur in a branch set. This is clearly not the

case for all choices of $\mathcal{P}, \mathcal{Q}, L$ satisfying φ_{valid}, but for our purposes it will be enough to know that there is one such choice.

We will now exhibit a second transduction

$$\Theta_2 := (\varphi_{valid}, \varphi_{univ}, \varphi_{\sim}, \varphi_V, \varphi_E, \varphi_{\in}, \varphi_{C_0}, \varphi_{C_1}, \varphi_{\leq})$$

with parameters X, F, T which defines a coloured ordered clique encoding the same word as $\mathfrak{A}$ in some structures in $\Theta_1(\mathfrak{A})$. Here we benefit from the fact that we only need to define topological minors, which makes the next transduction easy to define. The parameters X, F, T have the following intuitive meaning. By definition of topological minors, if K_n is a topological minor of a graph $G \in \Theta_1(\mathfrak{A})$ then there are n vertices $u_1, \ldots, u_n$ in G and for all $1 \leq i < j \leq n$ a path $P_{i,j}$ between u_i and u_j such that if $\{a, b\} \neq \{i, j\}$ then $P_{a,b}$ and $P_{i,j}$ are internally vertex disjoint (they have an endpoint in common if $\{a, b\} \cap \{i, j\} \neq \varnothing$). The parameter X will denote the set $\{u_1, \ldots, u_n\}$ and F will be the union $\bigcup_{i<j} E(P_{i,j})$. Hence, the graph defined by Θ_2 will have X as vertex set and the individual $P_{i,j}$ as edges. To define the colours of the vertices in X we need the last parameter T. T will contain exactly one edge of each path $P_{i,j}$. This will act as a separator: with every $x \in X$ we associate the set of all vertices on the paths $P_{i,j}$ emerging from x up to the edge in T. We will then say that for each x this set contains exactly on coloured vertex and we will take the colour of this vertex as colour of x.

Θ_2 is now formally defined as follows. To define the vertices let $\varphi_V(x) := x \in X$ and $\varphi_{\sim}^V(x, y) := x = y$. To define edges we first need some preparation.

Let
$$F\text{-}path(P, x, x') := \begin{array}{l} P \subseteq F \wedge path(P) \wedge ep(x, P) \wedge ep(x', P) \wedge \\ \neg \exists y \in X(y \in V(P) \wedge y \neq x \wedge y \neq x'). \end{array}$$

The formula says that P is a path whose edges are all from F, whose end points are x and x' and which contains no other vertex from X. Let

$$mp(P, F, X) := P \subseteq F \wedge \exists x, x' \in X \, F\text{-}path(P, x, x').$$

The formula says that P is a path with edge set in F connecting two vertices $x, x' \in X$.

Let $\varphi_E(e) := \exists P \subseteq F\big(mp(P, F, X) \wedge e \in P\big)$ and $\varphi_{\sim}^E(x, y) := \exists P \subseteq F\big(mp(P, F, X) \wedge x \in P \wedge y \in P\big)$. As mentioned above, we will represent edges by paths $P_{i,j}$ in F between vertices in X. $\varphi_E(e)$ says that e is an edge of such a path and $\varphi_{\sim}^E(e, e')$ defines e and e' to be equivalent if they occur on the same path in F. As usual, $\varphi_{univ}(x) := \varphi_V(x) \vee \varphi_E(x)$.

We now define the colours C_0, C_1. First, let

$$branch\text{-}set(x, y) := x \in X \wedge \exists Q \subseteq F\big(maxpath(Q, F \setminus T) \wedge x, y \in V(Q)\big).$$

The formula defines for given $x \in X$ the set of all vertices that can be reached from x by a path with edges of F not containing any edge from T. We can now define $\varphi_{C_i}(x) := \exists y\big(branch\text{-}set(x, y) \wedge y \in C_i\big)$, for $i \in \{0, 1\}$.

Finally, we define $\varphi_{\leq}(x, y) := x \leq y$.

The last part of Θ_2 left to be defined is φ_{valid}. Here we must say that X and F indeed induce a topological clique-minor as indicated above and that T is a separator containing one edge from each path linking two vertices from X.

We first use the formula

$$\varphi_0 := X \subseteq V \wedge F \subseteq E \wedge T \subseteq F \wedge C_0 \cap C_1 = \varnothing$$

to say that the parameters are of the right type.

The formula

$$\varphi_1 := \forall x, x' \in X\Big(x \neq x' \to \exists P \, F\text{-}path(P, x, x') \wedge \forall Q\big(F\text{-}path(Q, x, x') \to P = Q\big)\Big)$$

says that any two distinct vertices in X can be connected by a path in F and that this is unique.

The formula

$$\varphi_2 := \forall e(e \in F \to \exists x, x' \in X \exists P \subseteq F(F\text{-}path(P, x, x') \wedge e \in P))$$

says that every edge of F occurs on a path in F between two vertices of X.

The formula

$$\varphi_3 := \forall x, y, x', y' \in X \; x \neq x' \to \left(\begin{array}{c} \exists P, P' \subseteq F\big(F\text{-}path(P, x, y) \wedge F\text{-}path(P', x', y')\big) \wedge \\ (y \neq y' \to V(P') \cap V(P) = \varnothing) \wedge \\ (y = y' \to V(P') \cap V(P) = \{y\}) \end{array}\right)$$

says that if x, x', y, y' are distinct vertices in X then the paths P, P' linking x to y and x' to y', resp., are pairwise vertex disjoint and if x, x', y, y' are such that $y = y'$ but $x \neq x'$ then the two paths only have y in common.

The formulas $\varphi_0, \varphi_1, \varphi_2, \varphi_3$ together imply that (X, F) induce a topological clique minor as required. We next define a formula saying that T is as required, i.e. T contains one edge from each path connecting two vertices in X and that every edge of T is contained in such a path.

$$\varphi_4 := \begin{array}{l} \forall x, x' \in X \exists P \subseteq F\big(F\text{-}path(P, x, x') \wedge \exists^{=1} e \in P(e \in T)\big) \wedge \\ \forall e \in T \exists x, x' \in X \exists P \subseteq F \; (F\text{-}path(P, x, x') \wedge e \in P) \end{array}$$

What is left to define are the colours and that all vertices in X can be linearly ordered by $\leq$. The latter is easily defined by $\varphi_5 := \forall x(x \in X \to x \leq x)$.

The formula

$$\varphi_6 := \forall x \exists^{\leq 1} y\big(branch\text{-}set(x, y) \wedge y \in C_0 \cup C_1\big) \wedge \forall c \in C_0 \cup C_1 \exists x \in X \, branch\text{-}set(x, c)$$

says that every branch set contains at most one coloured vertex and every coloured vertex is contained in a branch set.

Finally, we need to say that the vertices x whose branch sets contain a coloured vertex are the smallest with respect to $\leq$. This is stated by the formula

$$\varphi_7 := \exists x \in X\big(\varphi_{C_0}(x) \vee \varphi_{C_1}(x) \wedge \forall y \in X(\varphi_{C_0}(y) \vee \varphi_{C_1}(y) \to y \leq x).$$

Let $\varphi_{valid} := \bigwedge_{i=0}^{7} \varphi_i$.

Now, if $\mathfrak{A}$ is a σ_{ord}-structure and $X \subseteq V^{\mathfrak{A}}$ and $F, T \subseteq E^{\mathfrak{A}}$ then $(\mathfrak{A}, X, F, T) \models \varphi_{valid}$ if, and only if, (X, F) determines a topological clique-minor where the branchsets are the components of $F \setminus T$. Furthermore, $\Theta(\mathfrak{A}, X, F, T)$ is a coloured ordered clique encoding the same word as $\mathfrak{A}$ and all $\mathfrak{B} \in \Theta_2(\mathfrak{A}, X, F, T)$ are coloured cliques encoding the same word as $\mathfrak{A}$.

The interpretations Θ_1 and Θ_2 together yield the desired transformation of complex pseudo-walls to coloured ordered grids as stated in the following lemma.

Lemma 7.6. *If $\mathfrak{A}$ is a σ_{col}-structure containing a complex pseudo-wall of order k encoding a word w then $\Theta_2(\Theta_1(\mathfrak{A}))$ contains a coloured ordered clique of order k encoding w. Furthermore, every $\mathfrak{B} \in \Theta_2(\Theta_1(\mathfrak{A}))$ is a coloured ordered clique encoding w.*

Again, using Corollary 7.3 and the formula translation provided by the transductions, we get the following corollary.

Corollary 7.7. *Let M be a non-deterministic n^d-time bounded Turing-machine. There is a formula $\varphi_M \in \mathrm{MSO}$ such that for all words $w \in \Sigma^\star$, if $\mathfrak{A}$ is a σ_{col}-structure containing a complex pseudo-wall of order $|w|^d$ encoding w, then $\mathfrak{A} \models \varphi_M$ if, and only if, M accepts w. Furthermore, the formula φ_M can be constructed effectively from M.*

The same holds if M is an alternating Turing-machine with a bounded number of alternations, as they are used to define the polynomial-time hierarchy.

The following result combines everything we need from this section later on.

Corollary 7.8. *Let M be a non-deterministic n^d-time bounded Turing-machine. There is a formula $\varphi_M \in \mathrm{MSO}$ such that for all words $w \in \Sigma^\star$, if $\mathfrak{A}$ is a σ_{col}-structure containing either a simple or complex pseudo-wall of order $|w|^d$ encoding w, then $\mathfrak{A} \models \varphi_M$ if, and only if, M accepts w. Furthermore, the formula φ_M can be constructed effectively from M.*

The same holds if M is an alternating Turing-machine with a bounded number of alternations, as they are used to define the polynomial-time hierarchy.

Proof. Note that in a complex pseudo-wall there is at most one blue edge $e \in B$ whereas a simple pseudo-wall always contains more than one. So we can easily distinguish in first-order logic between simple and complex pseudo-walls.

Now let
$$\varphi_M := \left(\exists^{=1} e \in B \wedge \varphi_M^c\right) \vee \left(\exists^{\geq 2} e \in B \wedge \varphi_M^s\right),$$
where φ_M^c, φ_M^s are the formulas from Corollary 7.7 and 7.5 respectively.

Then Corollary 7.5 and 7.7 imply that φ_M is indeed true in $\mathfrak{A}$ if, and only if, M accepts w. $\qquad\square$

8. Putting it all together

In this section we conclude the proof of Theorem 1.3 by combining the results obtained in Section 6 and 7. More precisely, we will first show the following lemma, which implies Part 2 of the theorem.

Lemma 8.1. *Let $\mathcal{C}$ be a class of σ_{col}-structures closed under colourings.*

If the tree-width of $\mathcal{C}$ is $(\log^c, p)$-unbounded, for some $c > d \cdot 84$ and polynomial p of degree d, then $\mathrm{MC}(\mathrm{MSO}_2, \mathcal{C})$ is not in XP and hence not fixed-parameter tractable unless Sat *can be solved in sub-exponential time.*

Proof. We show that if $p\text{-MC}(\mathrm{MSO}, \mathcal{C})$ is in XP then the propositional satisfiability problem SAT, i.e. the problem to decide for a formula of propositional logic if it has a satisfying assignment, can be solved in sub-exponential time.

Let w be a propositional logic formula. We can decide whether w is satisfiable as follows.

We first construct a σ_{col}-structure $\mathfrak{A} \in \mathcal{C}$ of tree-width between $c' \cdot m^{84}$ and $c \cdot m^{d \cdot 84}$, where c is the constant from Theorem 6.4 and $c' := \sqrt[d]{c}$. Furthermore, $\mathrm{tw}(\mathfrak{A}) > \log^{d \cdot 84 + \delta} |\mathfrak{A}|$, for some $\delta > 0$. Let G be the σ_{inc}-reduct of $\mathfrak{A}$, i.e. the underlying uncoloured graph of $\mathfrak{A}$.

It follows that
$$
\begin{aligned}
c \cdot m^{d \cdot 84} &> \log^{d \cdot 84 + \delta} |G| \\
\Longleftrightarrow \quad c'' \cdot m^{\frac{1}{y}} &> \log |G| \\
\Longleftrightarrow \quad |G| &< 2^{c'' \cdot m^{\frac{1}{y}}}
\end{aligned}
$$
for some c'' and $y > 1$.

By Theorem 6.4, as $\left(\frac{\mathrm{tw}(G)}{\sqrt{\log \mathrm{tw}(G)}}\right)^{\frac{1}{3}} \geq c \cdot m^{14}$, we can compute in polynomial time a σ_{col}-expansion $\mathfrak{B} \in \mathcal{C}$ of G containing a pseudo-wall encoding w with power 2.

Clearly, SAT can be decided by a non-deterministic Turing-machine M running in time quadratic in the size of the input. Hence, by Corollary 7.8, there is a formula φ_M, depending only on M, such that $\mathfrak{B} \models \varphi_M$ if, and only if, M accepts w if, and only if, w is satisfiable.

By Definition 1.2, we can construct $\mathfrak{A}$, and hence G, in time at most $2^{(c \cdot m^{84})^{\varepsilon}}$ for some $\varepsilon < 1$. By Theorem 6.4, $\mathfrak{B}$ can be constructed in time polynomial in the size of G and thus in time $2^{d' \cdot (c \cdot m^{84})^{\varepsilon}}$, for some constant d'.

Suppose now that $p\text{-MC}(\mathrm{MSO}, \mathcal{C})$ is in XP, i.e. given $G \in \mathcal{C}$ and $\varphi \in \mathrm{MSO}$, we can decide $G \models \varphi$ in time $|G|^{f(|\varphi|)}$, for some computable function $f : \mathbb{N} \to \mathbb{N}$. Hence, we can decide $\mathfrak{B} \models \varphi_M$ in time $|\mathfrak{B}|^{f(|\varphi_M|)}$ and thus in time $|G|^{f(|\varphi_M|)}$. But $|G|^{f(|\varphi_M|)} < 2^{f(|\varphi_M|) \cdot c'' \cdot m^{\frac{1}{y}}} \in 2^{o(|w|)}$. Hence, we can decide whether w is satisfiable in sub-exponential time. This concludes the proof of the lemma. $\qquad\square$

The lemma clearly implies Part 2 of Theorem 1.3. Unsing any other language in the polynomial-time hierarchy instead of SAT we get the first part by exactly the same argument. This concludes the proof of Theorem 1.3.

9. Conclusion and Further Work

In the previous section we have seen that if $\mathcal{C}$ is closed under colourings and its tree-width is not bounded logarithmically, then $\mathrm{MC}(\mathrm{MSO}, \mathcal{C})$ is not in XP unless SAT can be solved in sub-exponential time. What this shows is that Courcelle's theorem cannot be extended beyond logarithmic tree-width in its full generality.

The proof given in this paper shows that in order to apply our theorem to a class $\mathcal{C}$, its tree-width must be $(\log^c, p)$-unbounded for $c > 84 + d$, where d is the degree of p. Using slightly more complex algorithmic results from [21] this bound can be improved slightly to $c > 48 + d$. Furthermore, it is possible to reduce the numbers of colours needed to two binary and one unary relation symbols.

Our result refers to MSO_2, i.e. monadic second-order logic with quantification over sets of edges. If we restrict ourselves to MSO_1 then this logic becomes tractable on the much larger class of graphs of small *clique-width*.

Theorem 9.1 ([4]). *Let $\mathcal{C}$ be a class of graphs of bounded clique-width. Then $\mathrm{MC}(\mathrm{MSO}_1, \mathcal{C})$ is fixed-parameter tractable.*

It would be interesting to study classes of graphs closed under taking induced sub-graphs which have unbounded clique-width. We therefore put forward the following conjecture.

Conjecture 9.2. If $\mathcal{C}$ is a class of graphs whose clique-width is poly-logarithmically unbounded and which is closed under induced sub-graphs, then $\mathrm{MC}(\mathrm{MSO}_1, \mathcal{C})$ is not fixed-parameter tractable.

However, so far no analogue of grid-like minors for clique-width exists and therefore more research on obstructions for clique-width is needed to prove this conjecture.

References

[1] E. Birmelé, J. A. Bondy, and B. Reed. Brambles, prisms and grids. In *Graph theory in Paris*, Trends Math., pages 37–44. Birkhäuser, 2007.

[2] B. Bollobás and A. Thomason. Proof of a conjecture of Mader, Erdös and Hajnal on topological complete subgraphs. *Eur. J. Comb.*, 19(8):883–887, 1998.

[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 and S.-I. Oum. Vertex-minors, monadic second-order logic, and a conjecture by Seese. *Journal of Combinatorial Theory, Series B*, 97(1):91–126, 2007.

[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] H.-D. Ebbinghaus, J. Flum, and W. Thomas. *Mathematical Logic*. Springer, 2nd edition, 1994.

[10] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006. ISBN 3-54-029952-1.

[11] J. Flum and M. Grohe. Fixed-parameter tractability, definability, and model checking. *SIAM Journal on Computing*, 31:113 – 145, 2001.

[12] M. Frick and M. Grohe. Deciding first-order properties of locally tree-decomposable structures. *Journal of the ACM*, 48:1148 – 1206, 2001.

[13] M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some simplified NP-complete problems. In *STOC '74: Proceedings of the sixth annual ACM symposium on Theory of computing*, pages 47–63, New York, NY, USA, 1974. ACM.

[14] M. Grohe. Logic, graphs, and algorithms. In E.Grädel T.Wilke J.Flum, editor, *Logic and Automata – History and Perspectives*. Amsterdam University Press, 2007.

[15] M. Grohe, and S. Kreutzer. Methods for Algorithmic Meta-Theorems. In *Model Theoretic Methods in Finite Combinatorics,* Contemporary Mathematics vol. 588, American Mathematical Society, 2011.

[16] M. Grohe and D. Marx. On tree width, bramble size, and expansion. *J. Comb. Theory, Ser. B*, 99(1):218–228, 2009.

[17] R. Halin. *S*-functions for graphs. *Journal of Geometry*, 8:171–186, 1976.

[18] W. Hodges. *A shorter model theory*. Cambridge University Press, 1997.

[19] S. Kreutzer. Algorithmic meta-theorems. In *Finite and Algorithmic Model Theory*, London Mathematical Society Lecture Notes, No. 379, Cambridge University Press, 2011. See also *Electronic Colloquium on Computational Complexity (ECCC)* 16: 147 (2009)

[20] S. Kreutzer and S. Tazari. Lower bounds for the complexity of monadic second-order logic. In *Logic in Computer Science (LICS)*, 2010.

[21] S. Kreutzer and S. Tazari. On brambles, grid-like minors, and parameterized intractability of monadic second-order logic. In *Symposium on Discrete Algorithms (SODA)*, 2010.

[22] W. Mader. Homomorphieeigenschaften und mittlere Kantendichte von Graphen. *Math. Ann.*, 174:265–268, 1967.

[23] J. A. Makowsky and J. Mariño. Tree-width and the monadic quantifier hierarchy. *Theor. Comput. Sci.*, 1(303):157–170, 2003.

[24] C. H. Papadimitriou and M. Yannakakis. On the complexity of database queries. *J. Comput. Syst. Sci.*, 58(3):407–427, 1999.

[25] B. Reed and D. Wood. Polynomial treewidth forces a large grid-like minor. unpublished. Available at arXiv:0809.0724v3 [math.CO], 2008.

[26] N. Robertson, P. Seymour, and R. Thomas. Quickly excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 62:323 – 348, 1994.

[27] N. Robertson and P. D. Seymour. Graph minors V. Excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 41(1):92–114, 1986.

[28] N. Robertson and P.D. Seymour. Graph minors I – XXIII, 1982 –. Appearing in Journal of Combinatorial Theory, Series B since 1982.

[29] D. J. Rose. Triangulated graphs and the elimination process. *Journal of Mathematical Analysis and Applications*, 32:597–606, 1970.

[30] P. D. Seymour and R. Thomas. Graph searching and a min-max theorem for tree-width. *Journal of Combinatorial Theory, Series B*, 58(1):22–33, 1993.

[31] M. Thorup. All structured programs have small tree width and good register allocation. *Information and Computation*, 142:159–181, 1998.

[32] M. Vardi. On the complexity of relational query languages. In *Proc. of the 14th Symposium on Theory of Computing (STOC)*, pages 137–146, 1982.