First Order Logic on Pathwidth Revisited Again

Michael Lampis
Université Paris-Dauphine, PSL University, CNRS, LAMSADE, 75016, Paris, France

Abstract

Courcelle's celebrated theorem states that all MSO-expressible properties can be decided in linear time on graphs of bounded treewidth. Unfortunately, the hidden constant implied by this theorem is a tower of exponentials whose height increases with each quantifier alternation in the formula. More devastatingly, this cannot be improved, under standard assumptions, even if we consider the much more restricted problem of deciding FO-expressible properties on trees.

In this paper we revisit this well-studied topic and identify a natural special case where the dependence of Courcelle's theorem can, in fact, be improved. Specifically, we show that all FO-expressible properties can be decided with an elementary dependence on the input formula, if the input graph has bounded pathwidth (rather than treewidth). This is a rare example of treewidth and pathwidth having different complexity behaviors. Our result is also in sharp contrast with MSO logic on graphs of bounded pathwidth, where it is known that the dependence has to be non-elementary, under standard assumptions. Our work builds upon, and generalizes, a corresponding meta-theorem by Gajarský and Hliněný for the more restricted class of graphs of bounded tree-depth.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases Algorithmic Meta-Theorems, FO logic, Pathwidth

Digital Object Identifier 10.4230/LIPIcs.ICALP.2023.132

Category Track B: Automata, Logic, Semantics, and Theory of Programming

Related Version *Full Version*: `https://arxiv.org/abs/2210.09899`

Funding Supported by ANR projects ANR-21-CE48-0022 (S-EX-AP-PE-AL) and ANR-18-CE40-0025 (ASSK).

1 Introduction

Algorithmic meta-theorems are general statements of the form "all problems in a certain class are tractable on a particular class of inputs". Probably the most famous and celebrated result of this type is Courcelle's theorem [5], which states that all graph properties expressible in Monadic Second Order (MSO) logic are solvable in linear time on graphs of bounded treewidth. This result has proved to be of immense importance to parameterized complexity theory, because a vast collection of natural NP-hard problems can be expressed in MSO logic (and its variations that allow optimization objectives [1]) and because treewidth is the most well-studied structural graph parameter. Thanks to Courcelle's theorem, we immediately obtain that all such problems are fixed-parameter tractable (FPT) parameterized by treewidth.

Despite its great success, Courcelle's theorem suffers from a significant weakness: the algorithm it guarantees has a running time that is astronomical for most problems. Indeed, a careful reading of the theorem shows that the running time increases as a tower of exponentials whose height is equal to the number of quantifier alternations of the input MSO formula. Hence, even though Courcelle's theorem shows that any MSO formula ϕ can be decided on n-vertex graphs of treewidth tw in time $f(\phi, \mathrm{tw})n$, the function f is *non-elementary*, that is, it cannot be bounded from above by any tower of exponentials of fixed height.

Table 1 Summary of the state of the art for FO and MSO model checking on graphs of bounded treewidth, pathwidth, and tree-depth. Elementary (green cells) indicates that there is an algorithm which, when the corresponding width is bounded by an absolute constant, decides any formula ϕ in time $f(\phi)n^{O(1)}$, where f is a function that can be bounded above by a finite tower of exponentials. For the remaining cases, this is known to be impossible, under standard assumptions, hence it is inevitable to have an $f(\phi)$ that is a tower of exponentials whose height increases with ϕ.

Parameter	FO	MSO
Treewidth	Non-elementary on Trees [13]	Non-elementary on Trees [13]
Pathwidth	Elementary (Theorem 24)	Non-elementary on Caterpillars [13]
Tree-depth	Elementary [14]	Elementary [14]

One could hope that this terrible dependence on ϕ is an artifact of Courcelle's proof technique. Unfortunately, it was shown in a very influential work by Frick and Grohe [13] that this non-elementary dependence on the number of quantifiers of ϕ is best possible (under standard assumptions), even if one considers the severely restricted special case of model-checking First Order (FO) logic on trees. Recall that FO logic is a basic logic formalism that allows us to express graph properties using quantification over the vertices of the graph, while MSO logic also allows quantification over sets of vertices. Since FO logic is trivially a subset of MSO logic and trees have treewidth 1, this result established that Courcelle's theorem is essentially best possible.

Frick and Grohe's lower bound thus provided the motivation for the search for subclasses of bounded-treewidth graphs where avoiding the non-elementary dependence on ϕ may be possible. The obvious next place to look was naturally, pathwidth, which is the most well-known restriction (and close cousin) of treewidth. Unfortunately, Frick and Grohe's paper provided a negative result for MSO model checking also for this parameter. More precisely, they showed that MSO model checking on strings with a total order relation has a non-elementary dependence on the formula (unless P=NP), but such structures can easily be embedded into caterpillars (which are graphs of pathwidth 1) if one allows quantification over sets. Notice, however, that this does not settle the complexity of FO logic for graphs of counstant pathwidth, as it is not clear how one could implement the total ordering relation of a string without access to set quantifiers (we expand on this question further below).

On the positive side, Frick and Grohe's lower bounds motivated the discovery of several meta-theorems with elementary dependence on the formula for other, more restricted variations of treewidth (we review some such results below). Of all these results, the one that is "closest" to treewidth, is the theorem of Gajarský and Hliněný [14], which states that on graphs of constant tree-depth, MSO (and hence FO) model checking has elementary dependence on the input formula. It is known that for all n-vertex graphs G we have $\text{tw}(G) \leq \text{pw}(G) \leq \text{td}(G) \leq \text{tw}(G) \log n$, where $\text{tw}, \text{pw}, \text{td}$ denote the treewidth, pathwidth and tree-depth. In a sense, this positive result seemed to go as far as one could possibly go towards emulating treewidth, while retaining the elementary dependence on the formula and avoiding the lower bound of Frick and Grohe. This state of the art is summarized in Table 1.

Our result. In this paper we revisit this well-studied topic and address the one remaining case of Table 1 where it is still unknown whether it is possible to obtain an elementary dependence on the formula for model checking. We answer this question positively, showing that if we restrict ourselves to graphs of pathwidth p, where p is an absolute constant, then FO formulas with q quantifiers can be decided in time $f(q)n^{O(1)}$, where f is an elementary function of q. More precisely, the function f is at most a tower of exponentials of height $O(p)$. In other words, our result trades the non-elementary dependence on q which is inherent in Courcelle's theorem, with a non-elementary dependence on p. Though this may seem disappointing at first, it is known that this is the best one could have hoped for. In fact, the meta-theorem of [14] also has this behavior (its parameter dependence is a tower of exponentials whose height increases with the tree-depth), and it was shown in [24] that this is best possible (under standard assumptions). Since pathwidth is a more general parameter, we cannot evade this lower bound and our algorithm needs to have a non-elementary dependence on pathwidth, if its dependence on the formula is elementary.

The result we obtain is, therefore, in a sense best possible and fills a natural gap in our knowledge regarding FO model checking for a well-studied graph width. Beyond filling this gap, the fact that we are able to give a positive answer to this question and obtain an algorithm with "good" dependence on the formula is interesting, and perhaps even rather striking, for several reasons. First, in many cases in this domain, it is impossible to obtain an elementary dependence on q, no matter how much we are willing to sacrifice on our dependence on the graph width, as demonstrated by the fact that the lower bounds of Table 1 apply for classes with the smallest possible width (trees and caterpillars). Second, even though FO seems much weaker than MSO in general, the complexities of model checking the two logics seem to be similar (that is, at most one level of exponentiation apart) for most parameters (we review some further examples below). Indeed, a main contribution of [14] was to prove that for graphs of bounded tree-depth, the two logics are actually equivalent. It is therefore somewhat unusual (for this context) that for pathwidth FO has quite different complexity from MSO logic. Third, even though treewidth and pathwidth are arguably the two most well-studied graph widths in parameterized complexity, by and large the complexities of the vast majority of problems are the same for both parameters (for more information on this, see [2] which only recently discovered the first example of a natural problem separating the two parameters). It is therefore remarkable that the complexity of FO model checking is so different for pathwidth and treewidth.

Finally, one aspect of our result that makes it more surprising is that it does not seem to generalize to dense graphs. Meta-theorems that give a non-elementary dependence on the formula by using a restriction of treewidth, generally have a dense graph analogue, using a restriction of clique-width (the dense graph analogue of treewidth). Indeed, this is the case for vertex cover [23] (neighborhood diversity [23], twin cover [16]) but also for tree-depth (shrub-depth [17]). One may have expected something similar to hold in our case. However, the natural dense analogue of pathwidth is linear clique-width and it is already known that FO logic has a non-elementary dependence on threshold graphs [24]. Since threshold graphs have linear clique-width 2, we cannot hope to extend our result to this parameter and it appears that the positive result of this paper is an isolated island of "tractability".

High-level proof overview. Our technique extends and builds upon the meta-theorem of [14] which handles the more restricted case of graphs of bounded tree-depth. We recall that the heart of this meta-theorem is the basic observation that FO logic has bounded counting power: if our graph contains $q+1$ identical parts (for some appropriate definition of "identical"), then deleting one cannot affect the validity of any FO formula with q quantifiers. The approach of [14] is to partition the vertices of the graph depending on their height in the tree-depth decomposition, then identify (and delete) identical vertices in the bottom level. This bounds the degree of vertices one level up, which allows us to partition them into a bounded number of types, delete components of the same type if we have too many, hence bound the degree of vertices one level up, and so on until the size of the whole graph is bounded.

Our approach borrows much of this general strategy: we will appropriately rank the vertices of the graph and then move from lower to higher ranks, at each step bounding the maximum degree of any vertex of the current rank. Besides the fact that ranking vertices into levels is less obvious when given a path decomposition, rather than a tree of fixed height, the main difficulty we encounter is that no matter where we start, we cannot in general easily find identical parts where something can be safely deleted. Intuitively, this is demonstrated by the contrast between the simplest bounded tree-depth graph (a star, where leaves are twins, hence one can easily delete one if we have at least $q+1$) and the simplest bounded pathwidth graph (a path, which contains no twins). In order to handle this more general case, we need to combine the previous approach with arguments that rely on the locality of FO logic.

To understand informally what we mean by this, recall the classical argument which proves that Reachability is not expressible in FO logic. One way this is proved, is to show that a graph G_1 which is a long path (of say, 4^q vertices) and a graph G_2 which is a union of a path and a cycle (of say, $2 \cdot 4^{q-1}$ vertices each) are indistinguishable for FO formulas with q quantifiers. Our strategy is to flip this argument: if we are asked to model check a formula on a long path, we might as well model check the same formula on a simpler (less connected) graph which contains a shorter path and a cycle. Of course, our input graphs will be more complicated than long paths; we will, however, be dealing with long path-like structures, as our graph has small pathwidth. Our strategy is to perform a surgical rewiring operation on the path decomposition, producing the union of a shorter decomposition and a ring-like structure, while still satisfying the same formulas (the reader may skip ahead to Figure 1 to get a feeling for this operation). In other words, the main technical ingredient of our algorithm is inspired by (and exploits) a classical impossibility result on the expressiveness of FO logic. The abstract idea is (in a rough sense) to apply this argument repeatedly, so that if we started with a long path decomposition, we end up with a short path decomposition and many "disconnected rings". Eventually, we will be able to produce some such rings which are identical, delete them, and simplify the graph.

There are, of course, now various technical difficulties we need to overcome in order to turn this intuition into a precise argument. First, when we cut at two points in the path decomposition to extract the part that will form the "ring", we need to make sure that at an appropriate radius around the cut points the decompositions are isomorphic. It is not hard to calculate the appropriate radius we need in the graph (it is known that q-quantifier FO formulas depend on balls of radius roughly 2^q), but a priori two vertices which are close in the graph could be far in the path decomposition. To handle this, we take care when we rank the vertices, so that vertices of lower rank are guaranteed to only appear in a bounded number of bags, hence distances in the path decomposition approximate distances in the graph. Second, we need to calculate how long our decomposition needs to be before we can guarantee that we will be able to find some appropriate cut points. Here we use some counting arguments and pigeonhole principle to show that a path decomposition with length double-exponential in the desired radius is sufficient. Finally, once we find sufficiently many points to rewire and produce sufficiently many "rings", we need to prove that this did not affect the validity of the formula. Then, we are free to delete one, using the same argument as [14] and obtain a smaller equivalent graph. In the end, once we can no longer repeat this process, we obtain a bounded-degree graph, where it is known that FO model checking has an elementary dependence on the formula.

Overall, even though the algorithm we present seems somewhat complicated, the basic ingredients are simple and well-known: the fact that deleting one of many identical parts does not affect the validity of the formula (which is also used in [14]); the fact that FO formulas are not affected if we edit the graph in a way that preserves balls of a small radius around each vertex; and simple counting arguments and the pigeonhole principle.

Paper Organization. We conclude this section below with a short overview of other related work on algorithmic meta-theorems and continue in Section 2 with definitions and notation. The rest of the paper is organized as follows:

1. In Section 3 we present two lemmas, which are standard facts on FO logic, with minor adjustments to our setting. In particular, in Section 3.1 we present the lemma that states that if we have $q+1$ identical parts, it is safe to delete one; and in Section 3.2 we present the lemma that states that if two graphs agree on the local extended neighborhoods around each vertex (for some appropriate radius), then they satisfy the same formulas (that is, FO logic is local). Since these facts are standard, the reader may wish to skip the proofs of Section 3, which are given for the sake of completeness, during a first reading.
2. Then, in Section 4 we present the specific tools we will use to simplify our graph. In Section 4.1 we explain how we rank the vertices of a path decomposition so that each vertex has few neighbors of higher rank (but possibly many neighbors of lower rank). This allows us to process the ranks from lower to higher, simplifying the graph step by step. Then, in Section 4.2 we use some counting arguments to calculate the length of a path decomposition that guarantees the existence of long isomorphic blocks, on which we will apply the rewiring operation. We also show how distances in the graph can be approximated by distances in the path decomposition, if we have bounded the number of occurrences of each vertex in the decomposition. Finally, in Section 4.3 we formally define the rewiring operation and show that if the points where we apply it are in the middle of sufficiently long isomorphic blocks of the decomposition, this operation is safe. We also show that the "rings" it produces can be considered identical, in a sense that will allow us to invoke the lemma of Section 3.1 and delete one.
3. We put everything together in Section 5, where we explain how the lemmas we have presented form parts of an algorithm that ranks the vertices of a graph supplied with a path decomposition and then processes ranks one by one, decreasing the number of occurences of each vertex in the decomposition without affecting the validity of any formula (with q quantifiers). In the end, the processed graph has bounded degree and we invoke known results to decide the formula.

Other related work. Algorithmic meta-theorems are a very well-studied topic in parameterized complexity ([20]) and much work has been devoted in improving and extending Courcelle's theorem. Among such results, we mention the generalization of this theorem to MSO for clique-width, which covers dense graphs [6]. For FO logic, fixed-parameter tractability can be extended to much wider classes of graphs, with the recently introduced notion of twin-width nicely capturing many results in the area [4, 9, 11, 12]. Of course, since all these classes include the class of all trees, the non-elementary dependence on the formula implied by the lower bound of [13] still applies. Meta-theorems have also been given for logics other than FO and MSO, with the goal of either targeting a wider class of problems [18, 21, 22, 28], or achieving better complexity [26]. Kernelization [3, 10, 19] and approximation [8] are also topics where meta-theorems have been studied.

The meta-theorems which are more relevant to the current work are those which explicitly try to improve upon the parameter dependence given by Courcelle, by considering more restricted parameters. We mention here the meta-theorems for vertex cover, max-leaf, and neighborhood diversity [23], twin-cover [16], shrub-depth [17], and vertex integrity [25]. As mentioned, one common aspect of these meta-theorems is that they handle both FO and MSO logic, without a huge difference in complexity (at most one extra level of exponentiation in the parameter dependence), which makes the behavior of FO logic on treewidth somewhat unusual. The only exception, is the meta-theorem on graphs of bounded max-leaf number of [23] which does not generalize to MSO logic. It was later shown that this is with good reason, as MSO logic has a non-elementary dependence even for unlabeled paths [24], which have the smallest possible max-leaf number. This is therefore the only previous result in the literature which mirrors the situation for pathwidth.

A classical result, incomparable to the parameters mentioned above, is the fact that FO model checking is FPT (with an elementary, triple-exponential dependence on the formula) on graphs of bounded degree [27]. We will use this fact as the last step of our algorithm.

The complexity of model checking FO and MSO formulas on structures other than graphs, such as posets [15] and strings has also been investigated. As mentioned, the case of strings is of particular interest to us, because the standard structure that represents a string over a fixed alphabet (a universe that contains the letters of the string, unary predicates that indicate for each letter which character of the alphabet it corresponds to, and a total ordering relation $\prec$ which indicates the ordering of the letters in the string) allows us to easily translate MSO properties of strings into MSO properties of an appropriate caterpillar. Indeed, to embed a string into a caterpillar, we can start with a path with endpoints s, t, and use one vertex of the path to represent each letter in the string. We can attach an appropriate (constant) number of leaves on each vertex to signify which character it represents. The precedence relation $x \prec y$ of the string now becomes the relation "every connected set that contains s and y, also contains x", which is MSO-expressible. Thanks to this simple transformation, the lower bound result of [13] on model checking MSO (and even FO) logic on strings, immediately carries over to graphs of pathwidth 1. Note, however, that the existence of the ordering relation is crucial, as FO model checking on other models of strings (e.g. with a successor relation) has elementary dependence on the formula, as such structures have bounded degree [13]. Hence, it seems that if we focus on FO (rather than MSO) logic, the similarity between model checking on bounded pathwidth graphs and strings becomes much weaker: FO model checking is easier on graphs of bounded pathwidth than on strings with an ordering relation, but harder than on strings with only a successor relation (as the lower bound of [24] for tree-depth applies to pathwidth, and rules out an algorithm with "only" triple-exponential dependence).

2 Definitions and Preliminaries

We use standard graph-theoretic notation and assume the reader is familiar with the basics of parameterized complexity (see e.g. [7]). For a graph $G = (V, E)$, and $S \subseteq V$, we use $G[S]$ to denote the subgraph of G induced by S. When r is a positive integer, we use $[r]$ to denote the set $\{1, \ldots, r\}$, while for two integers s, t, we use $[s, t]$ to denote the set $\{i \in \mathbb{Z} \mid s \leq i \leq t\}$. Note that if $t < s$ then $[s, t] = \emptyset$. We define $\text{tow}(i, n)$ as follows: $\text{tow}(0, n) = n$ and $\text{tow}(i + 1, n) = 2^{\text{tow}(i,n)}$. A function $f : \mathbb{N} \to \mathbb{N}$ is elementary if there exists a fixed i such that for all n we have $f(n) \leq \text{tow}(i, n)$.

We recall the standard notion of path decomposition: a path decomposition of a graph $G = (V, E)$ is an ordered sequence of bags $B_1, B_2, \ldots, B_\ell$, where each B_i is a subset of V, that satisfies the following: (i) $\bigcup_{j \in [\ell]} B_j = V$ and for all $uv \in E$ there exists $i \in [\ell]$ such that $\{u, v\} \subseteq B_i$ (ii) for all $i_1 < i_2 < i_3$, with $i_1, i_2, i_3 \in [r]$ we have $B_{i_1} \cap B_{i_3} \subseteq B_{i_2}$. The width of a path decomposition is the number of vertices in the largest bag (minus one). The pathwidth of a graph G is the smallest width of any path decomposition of G.

First Order Logic. We use a standard form of First Order (FO) logic on graphs, where quantified variables are allowed to range over vertices. To simplify the presentation of some results, we will allow our formulas to also refer to vertex constants, corresponding to some specific vertices of the graph. More formally, the structures on which we will perform model checking are k-terminal graphs as defined below.

▶ **Definition 1.** *For a positive integer k, a k-terminal graph $G = (V, E)$ is a graph supplied with a function $\mathcal{T} : [k] \to V$, called the terminal labeling function. For $i \in [k]$, we say that $\mathcal{T}(i)$ is the i-th terminal of G. The set of terminals is the set T of images of $\mathcal{T}$ in V. Vertices of $V \setminus T$ are called non-terminals.*

Intuitively, terminals will play two roles: on the one hand, we define FO logic on graphs (below) in a way that allows formulas to refer to the terminal vertices; on the other, in some parts of our algorithm we will use a set of terminals that form a separator of the graph and hence allow us to break down the graph into smaller components. Note, however, that Definition 1 does not require the k terminals to be a separator, or have any other particular property.

A formula of FO logic is made up of the following vocabulary: (i) vertex variables, denoted $x_1, x_2, \ldots$ (ii) vertex constants denoted $\ell_1, \ell_2, \ldots$ (iii) existential quantification $\exists$ (iv) the boolean operations $\neg, \vee$ (v) the binary predicates $\sim$ (for adjacency) and $=$ (for equality). More formally, a First Order formula is a formula produced by the following grammar, where x represents a vertex variable and y represents a vertex variable or constant:

$$\phi \to \exists x.\phi \mid \neg\phi \mid \phi \vee \phi \mid y \sim y \mid y = y$$

A FO formula ϕ is called a sentence if every vertex variable x appearing in ϕ is quantified, that is, x appears within the scope of $\exists x$. A variable that is not quantified is called a free variable. For a formula ϕ that contains a free variable x, we will write $\phi[x/\ell_i]$ to denote the formula obtained by replacing every occurrence of x in ϕ by the constant ℓ_i.

The main problem we are concerned with is model checking: given a k-terminal graph G and a sentence ϕ, decide if G satisfies ϕ. We define the semantics of what this means inductively in a standard way, as follows. We say that a k-terminal graph $G = (V, E)$ with labeling function $\mathcal{T}$ models (or satisfies) a formula ϕ, and write $G, \mathcal{T} \models \phi$ (or simply $G \models \phi$ if $\mathcal{T}$ is clear from the context) if and only if we have one of the following:

1. $\phi := (\ell_i = \ell_j)$, where $i, j \in [k]$ and $\mathcal{T}(i)$ is the same vertex as $\mathcal{T}(j)$.
2. $\phi := (\ell_i \sim \ell_j)$, where $i, j \in [k]$ and $\mathcal{T}(i)\mathcal{T}(j) \in E$.
3. $\phi := (\neg\psi)$ and it is not the case that $G, \mathcal{T} \models \psi$.
4. $\phi := (\psi_1 \vee \psi_2)$ and at least one of $G \models \psi_1$, $G \models \psi_2$ holds.
5. $\phi := (\exists x.\psi)$ and there exists $v \in V$ such that $G, \mathcal{T}' \models \psi[x/\ell_{(k+1)}]$, where $\mathcal{T}'$ is the labeling function that sets $\mathcal{T}'(k + 1) = v$ and $\mathcal{T}'(i) = \mathcal{T}(i)$ for $i \in [k]$.

Note that we have not included in our definition universal quantification or other boolean connectives such as $\wedge$. However, this is without loss of generality as $\forall x.\phi$ can be thought of as shorthand for $\neg\exists x.\neg\phi$ and all missing boolean connectives can be simulated using $\neg$ and $\vee$.

Let us also define a kind of isomorphism between labeled graphs that is guaranteed to leave terminal vertices untouched.

▶ **Definition 2.** *A terminal-respecting isomorphism between two k-terminal graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ with terminal labeling functions $\mathcal{T}_1, \mathcal{T}_2$ is a bijective function $f : V_1 \to V_2$ such that (i) for all $u, u' \in V_1$ we have $uu' \in E_1$ if and only if $f(u)f(u') \in E_2$ (ii) for each $i \in [k]$, $f(\mathcal{T}_1(i)) = \mathcal{T}_2(i)$.*

We recall the following basic fact about FO logic which states that isomorphic structures satisfy the same sentences (see e.g. Lemma 9 of [25] for a proof).

▶ **Lemma 3.** *If G_1, G_2 are two k-terminal graphs such there exists a terminal-respecting isomorphism from G_1 to G_2, then, for all FO sentences ϕ we have $G_1 \models \phi$ if and only if $G_2 \models \phi$.*

3 Two Basic Lemmas

The purpose of this section is to establish two basic ingredients that will allow us to simplify the input graph without affecting whether it satisfies any FO formula with at most a given number q of quantified variables. The first lemma (Lemma 5) is rather simple and states that if a graph contains many "identical" components, we can safely remove one. Despite its simplicity, this idea has been sufficient to obtain many of the best currently known meta-theorems with non-elementary dependence in the formula, such as the meta-theorem of [14] for graphs of bounded tree-depth.

The second lemma (Lemma 9) is a variation of standard arguments regarding the locality of FO logic. It states that if we have two graphs which look locally the same, in the sense that for each vertex of one graph there exists a vertex of the other whose r-neighborhood is the same, for some appropriately chosen r, then actually the two graphs are indistinguishable by FO formulas with q quantifiers (even though they are not necessarily isomorphic). As we explained, we intend to use this to allow us to take parts of the graph that resemble "long", low-pathwidth components and cut them up into smaller, disconnected components. The strategy is to eventually produce a large enough number of such components that we can apply Lemma 5 and simplify the graph.

3.1 Identical Parts

We would now like to show that if the given graph contains many (say, at least $q + 1$) "identical" parts, then it is safe to delete one without affecting whether the graph satisfies any FO formula with at most q quantifiers. We first define what we mean that two sets of vertices are identical in a k-terminal graph and then prove that if we can find $q + 1$ such sets in a graph, we can safely delete one without affecting whether any FO formula with at most q quantifiers is satisfied.

▶ **Definition 4.** *Let $G = (V, E)$ be a k-terminal graph with labeling function $\mathcal{T}$ and terminal set T. We say that two disjoint sets of vertices C_1, C_2 are* identical *if there exists a terminal-respecting isomorphism from G to G that maps all vertices of C_1 to C_2 and all vertices of C_2 to C_1, and maps every vertex of $V \setminus (C_1 \cup C_2)$ to itself.*

Before we proceed, let us make two easy observations. First, if C_1, C_2 are identical, it must be the case that $(C_1 \cup C_2) \cap T = \emptyset$, because C_1, C_2 are disjoint and terminal-respecting isomorphisms must map vertices of T to themselves. Second, the relation of being identical is an equivalence relation on a collection of pairwise disjoint sets of vertices, that is, if C_1, C_2, C_3 are disjoint, C_1 is identical to C_2, and C_2 is identical to C_3, then C_1 is identical to C_3 (the fact that the relation is reflexive and symmetric is easy to see).

▶ **Lemma 5.** *Fix a positive integer q. Let $G = (V, E)$ be a k-terminal graph with labeling function $\mathcal{T}$ and terminal set T and suppose that $C_1, C_2, \ldots, C_{q+1}$ are $q + 1$ sets of vertices of G which are pairwise identical. Then, for all FO sentences with at most q quantifiers we have that $G, \mathcal{T} \models \phi$ if and only if $G[V \setminus C_1], \mathcal{T} \models \phi$.*

3.2 Similar Neighborhoods

We now move on to present a lemma that will allow us to claim that two graphs are indistinguishable for FO formulas with q quantifiers if they are locally the same. This is a standard argument in FO logic, going back to Gaifman, though we need to adjust the proof to our purposes to handle terminal vertices appropriately. In particular, we will on the one hand be stricter on the isomorphisms we allow before we consider that the neighborhoods of two vertices are the same (because we only allow terminal-respecting isomorphisms), but on the other, we will only consider the extended neighborhood around a vertex by considering paths that go through non-terminals. This is important, because it allows us to work around the case where, for example, a terminal vertex is connected to everything and hence the diameter of the graph is 2. In such a case, the extended neighborhood of a non-terminal vertex will not trivially contain the whole graph, because we exclude paths that go through the supposed universal terminal.

According to this discussion, we define the notion of a ball of radius r around a vertex v, denoted $B_r(v)$, in a way that only takes into account paths whose internal vertices are non-terminals, as follows.

▶ **Definition 6.** *Let $G = (V, E)$ be a k-terminal graph with terminal labeling function $\mathcal{T}$ and terminal set T, r be a positive integer, and $v \in V$. We define $B_r^G(v)$ (and simply write $B_r(v)$ if G is clear from the context) to be the k-terminal subgraph of G that has labeling function $\mathcal{T}$ and is induced by $T \cup V'$, where V' is the set of all vertices reachable by v via a path of length at most r whose internal vertices are all in $V \setminus T$.*

▶ **Definition 7.** *Let $G_1 = (V_1, E_1), G_2 = (V_2, E_2)$ be two k-terminal graphs, with terminal labeling functions $\mathcal{T}_1, \mathcal{T}_2$ and terminal sets T_1, T_2. For a non-negative integer r, we will say that $v_1 \in V_1$ is* r-similar *to $v_2 \in V_2$, if there exists a terminal-respecting isomorphism from $B_r^{G_1}(v_1)$ to $B_r^{G_2}(v_2)$ that maps v_1 to v_2.*

Note that in the above definition, G_1, G_2 may be the same graph. It is not hard to see that r-similarity is an equivalence relation on the vertices of $V_1 \cup V_2$. Definition 7 allows us to set $r = 0$, in which case we are testing if the graphs induced by $T \cup \{v_1\}$ and $T \cup \{v_2\}$ are isomorphic. Let us also make the following easy observation that decreasing r cannot make two similar vertices dissimilar.

▶ **Observation 8.** *Let G_1, G_2 be two graphs as in Definition 7 and $v_1 \in V(G_1), v_2 \in V(G_2)$ be two vertices which are r-similar. Then, for all non-negative integers $r' \le r$, v_1 is r'-similar to v_2.*

The main lemma of this section is then the following.

▶ **Lemma 9.** *Let q, k be positive integers and set $r = 2^q - 1$. Let G_1, G_2 be two k-terminal graphs that contain some non-terminal vertices, with labeling functions $\mathcal{T}_1, \mathcal{T}_2$ and terminal sets T_1, T_2. Suppose that there exists a bijective mapping $f : V(G_1) \to V(G_2)$ such that (i) for all $i \in [k]$ we have $f(\mathcal{T}_1(i)) = \mathcal{T}_2(i)$ (ii) for all non-terminal vertices $v \in V(G_1) \setminus T_1$ we have that v is r-similar to $f(v) \in V(G_2) \setminus T_2$. Then, for all FO sentences ϕ with at most q quantifiers we have $G_1 \models \phi$ if and only if $G_2 \models \phi$.*

4 Simplification Operations on Path Decompositions

In this section we present the main technical ingredients of our algorithm. In Section 4.1 we show how we can rank the vertices to bound the number of higher-rank neighbors of any vertex; in Section 4.2 we use the pigeonhole principle to show that for sufficiently long path decompositions we can always find long isomorphic blocks; and in Section 4.3 we describe the rewiring operation we will use in these blocks and show that it does not affect the validity of any formula and that it produces identical parts, in the sense of Lemma 5.

4.1 Normalized Path Decompositions

▶ **Definition 10.** *A* ranked *path decomposition of a graph $G = (V, E)$ where all bags have size at most p is a path decomposition together with a ranking function $\rho : V \to \mathbb{N}$ that has the property that no bag B_i of the decomposition contains two vertices $u, v \in B_i$ for which $\rho(u) = \rho(v)$.*

▶ **Lemma 11.** *Given a graph $G = (V, E)$ and a path decomposition of G where each bag contains at most p vertices, it is possible in polynomial time to convert it into a ranked path decomposition with a ranking function $\rho : V \to [8p]$ and the property that for each $i < j$ with $i, j \in [8p]$ we have that for every vertex v with $\rho(v) = i$, there exist at most two vertices u_1, u_2 with $\rho(u_1) = \rho(u_2) = j$ that appear in a bag together with v. Furthermore, the produced decomposition has the property that each bag contains at least one vertex that does not appear in the previous bag.*

We will call the ranked path decompositions that satisfy the properties of the decompositions produced by Lemma 11 *normalized* path decompositions. Since such a decomposition can always be obtained without using too many ranks in the ranking function, we will from now on focus on the case where we are given a normalized decomposition. Furthermore, we will usually use p to denote the maximum rank, rather than the pathwidth; this will not have a significant impact as, according to Lemma 11 we can make sure that the two are at most a constant factor apart.

4.2 Finding Isomorphic Bag Intervals

As mentioned, our high-level strategy will be to identify parts of the graph which are locally isomorphic, so that we can apply Lemma 9 to obtain a simpler (less well-connected) graph, and eventually Lemma 5 in order to decrease the size of the graph. In order to identify such parts, we first define what it means for two blocks of bags of a given decomposition to be isomorphic.

▶ **Definition 12.** *Let $G = (V, E)$ be a k-terminal graph with terminal set T, and $B_1, \ldots, B_\ell$ a ranked path decomposition of G with ranking function $\rho : V \to [p]$. Let s_1, t_1, s_2, t_2 be positive integers with $s_1 \leq t_1$ and $t_1 - s_1 = t_2 - s_2$. We define the* block *corresponding to $[s_1, t_1]$ and write $\mathcal{B}(s_1, t_1)$ to be $\{B_j \mid j \in [s_1, t_1]\}$. We say that $\mathcal{B}(s_1, t_1)$ is* block-isomorphic *to $\mathcal{B}(s_2, t_2)$ if*

1. *For each $j \in [s_1, t_1]$ and rank i we have $|\rho^{-1}(i) \cap B_j| = |\rho^{-1}(i) \cap B_{s_2+(j-s_1)}|$.*
2. *For each $j \in [s_1 + 1, t_1]$ and rank i we have that B_j contains a vertex v with $\rho(v) = i$ such that $v \notin B_{j-1}$ if and only if $B_{s_2+(j-s_1)}$ contains a vertex v' with $\rho(v') = i$ such that $v' \notin B_{s_2+(j-s_1)-1}$.*
3. *The following mapping f is a terminal-respecting isomorphism from $G[T \cup (\bigcup_{j\in[s_1,t_1]} B_j)]$ to $G[T \cup (\bigcup_{j\in[s_2,t_2]} B_j)]$. For each $v \in \bigcup_{j\in[s_1,t_1]} B_j$ we let j_v be the minimum index in $[s_1, t_1]$ such that $v \in B_{j_v}$ and define $f(v)$ to be the (unique) vertex of $B_{s_2+(j_v-s_1)}$ such that $\rho(v) = \rho(f(v))$.*

▶ **Definition 13.** *Let $L \geq 0$ and G, k, T, ℓ, ρ as in Definition 12. For positive integers $s_1, s_2 \in [\ell - L]$ we will write $s_1 \approx_L s_2$ to indicate that $\mathcal{B}(s_1, s_1 + L)$ is block-isomorphic to $\mathcal{B}(s_2, s_2 + L)$.*

Note that the isomorphism of Definition 12 is well-defined, because according to the first condition, if B_j contains a vertex of rank i, then so does $B_{s_2+(j_v-s_1)}$, and such a vertex is unique by the definition of ranked path decomposition. According to Definition 12, two blocks of bags are isomorphic only if the subgraphs induced by the bags they contain (and the terminals of G) are isomorphic under the trivial mapping function which maps each vertex of a bag from one block to the vertex of the corresponding bag of the other block that has the same rank. Despite the fact that this restricts the class of isomorphisms we may consider quite a bit, the block-isomorphism relation is an equivalence relation that does not have too many equivalence classes. In particular, we have the following.

▶ **Lemma 14.** *Let $L \geq 0$, $G = (V, E)$ be a k-terminal graph with terminal set T, and $B_1, \ldots, B_\ell$ a ranked path decomposition of G with ranking function $\rho : V \to [p]$. Let t_1, t_2 be integers such that for all $j, j' \in [t_1, t_2]$ we have $B_j \cap T = B_{j'} \cap T$. Then, the relation $\approx_L$ is an equivalence relation on the set $[t_1, t_2 - L]$ with at most $2^{(L+1)(p^2+2p+kp)}$ equivalence classes.*

Proof. The fact that $\approx_L$ is an equivalence relation is easy to see, as terminal-respecting isomorphisms can be composed to show transitivity. The interesting part of the lemma is then the bound on the number of equivalence classes. We prove this by induction on L.

For $L = 0$, we claim there are at most 2^{p+p^2+kp} equivalence classes of bags (in this case, each block consists of a single bag). Indeed, in order to decide if $\mathcal{B}(s_1, s_1) = \{B_{s_1}\}$ and $\mathcal{B}(s_2, s_2) = \{B_{s_2}\}$ are block-isomorphic, we first need to check if B_{s_1}, B_{s_2} contain vertices of the same ranks, and for this there are 2^p equivalence classes. If they do, then we must check, for each $i_1, i_2 \in [p]$ if the vertices of ranks i_1, i_2 in each of B_{s_1}, B_{s_2} are adjacent, and for this we have $2^{\binom{p}{2}} < 2^{p^2}$ equivalence classes. Finally, since the isomorphism has to be terminal-respecting, we have to check for each rank $i \in [p]$ if the vertex of rank i in each of B_{s_1}, B_{s_2} is connected to each of the k terminals, which gives at most kp edges which may or may not exist. (Note that we have to check these edges, even though the two bags contain the same terminals, because terminal-respecting isomorphisms must also preserve the edges towards terminals outside the bag). Overall we have at most $2^{p+p^2+kp} < 2^{p^2+2p+kp}$ choices. If we make the same choices for two bags, the two bags are block-isomorphic, hence we have bounded the number of equivalence classes for $L = 0$.

Suppose now that $L > 0$ and we have shown that the number of equivalence classes of $\approx_{L-1}$ is at most $2^{L(p^2+2p+kp)}$. Consider two indices s_1, s_2 for which we want to check if $s_1 \approx_L s_2$. We claim that for this it is sufficient to have $s_1 \approx_{L-1} s_2$ and to satisfy certain conditions for the bags B_{s_1+L}, B_{s_2+L} for which we have at most $2^{p^2+2p+kp}$ choices. More precisely, for each rank i, we have three possibilities for the bag B_{s_1+L}: either the bag contains no vertex of rank i; or it contains a vertex of rank i that also appears in B_{s_1+L-1}; or it contains a vertex of rank i that appears for the first time in B_{s_1+L} (and hence this vertex is a non-terminal). Suppose now that for each rank i, the bags B_{s_1+L}, B_{s_2+L} agree on the choice of which of these three possibilities holds (there are $3^p < 2^{2p}$ possibilities in total), and furthermore, that the graphs induced by $B_{s_1+L} \cup T$ and $B_{s_2+L} \cup T$ are isomorphic for the natural terminal-respecting isomorphism that matches vertices of the same rank (at most 2^{p^2+kp} possibilities). Then, if $s_1 \approx_{L-1} s_2$, we now have $s_1 \approx_L s_2$. Therefore, each of the $2^{L(p^2+2p+kp)}$ equivalence classes of $\approx_{L-1}$ has been refined into at most 2^{2p+p^2+kp} equivalence classes, giving that the number of equivalence classes of $\approx_L$ is at most $2^{(L+1)(p^2+2p+kp)}$, as desired. ◀

Now that we know that block-isomorphism has a bounded number of equivalence classes (if k, p, L are bounded), we can try to look for "copies" of the same block in our path decomposition. We observe the following lemma.

▶ **Lemma 15.** *Let L be a non-negative integer, $G = (V, E)$ be a k-terminal graph with terminal set T, and $B_1, \ldots, B_\ell$ a ranked path decomposition of G with ranking function $\rho : V \to [p]$. We define $R = (L+1)(2^{(L+1)(p^2+2p+kp)} + 1)$. Let t_1, t_2 be integers such that for all $j, j' \in [t_1, t_2]$ we have $B_j \cap T = B_{j'} \cap T$. Then, for every $s \in [t_1, t_2 - R]$, there exist $s_1, s_2 \in [s, s + R - (L+1)]$ such that $s_1 + L < s_2$ and $s_1 \approx_L s_2$.*

What we have shown so far is that if we take sufficiently many (at least R) consecutive bags in our decomposition, we will find two blocks of length (roughly) L which are block-isomorphic. Let us now move a step further.

▶ **Lemma 16.** *Let L be a non-negative integer, $G = (V, E)$ be a k-terminal graph with terminal set T, $B_1, \ldots, B_\ell$ a ranked path decomposition of G with ranking function $\rho : V \to [p]$, and t_1, t_2 as defined in Lemma 15. Let q, R be positive integers. We define $R^* = (R+1)(q2^{(R+1)(p^2+2p+kp)} + 1)$. Then, for every $s \in [t_1, t_2 - R^*]$ there exist $q+1$ distinct $s_1, s_2, \ldots, s_{q+1} \in [s, s + R^* - (R+1)]$, such that for any two distinct s_i, s_j with $i, j \in [q+1]$ we have $|s_i - s_j| > R$ and $s_i \approx_R s_j$.*

Note that Lemma 15 and Lemma 16 are non-vacuous only if we find a long enough interval where all bags contain the same terminals, that is if $t_2 - t_1 \geq R$ or $t_2 - t_1 \geq R^*$ respectively. We will take this into account when we use these lemmas in the next section.

At this point we are almost done in our search for appropriate isomorphic parts of the graph. What we have proved is that, if we fix some appropriate radius L, there is some larger radius R^* (double-exponential in L), such that if we look at any interval of the path decomposition of length R^*, we will be able to find $q+1$ isomorphic R-blocks, which are long enough to guarantee the existence of two isomorphic L-blocks inside them. What remains is to ask what value of L will be appropriate for our purposes. Ideally, we would like to calculate a value L that will allow us to preserve the balls around vertices for a suitable radius and apply Lemma 9. However, we can only give such a bound if we know that vertices of our path decomposition do not appear in too many bags.

▶ **Lemma 17.** *Let $G = (V, E)$ be a k-terminal graph with terminal set T and $B_1, \ldots, B_\ell$ a ranked path decomposition of G with the additional property that any non-terminal vertex appears in at most Δ bags of the decomposition. Then, for each $r \geq 0$ and for each non-terminal vertex v, if $v \in B_j$, then each non-terminal vertex of $B_r(v)$ is contained in a bag of $\mathcal{B}(j - r\Delta, j + r\Delta)$.*

4.3 Rewiring Operation

The goal of Section 4.2 was to present the basic tools which will allow us to find isomorphic parts of the input graph. Ideally, we would then like to use Lemma 5 and delete one such part. However, this is in general not possible, as the isomorphism guaranteed by the lemmas of Section 4.2 is not sufficient to obtain identical sets, in the sense of Definition 4. What we need to do, then, is to edit the graph in a way that does not affect the validity of any FO formula with q quantifiers but leverages the isomorphic parts we have found to construct $q+1$ identical parts on which Lemma 5 can be applied. We now present the basic edit operation which will allow us to achieve this for appropriate parameters.

▶ **Definition 18** (Rewiring). *Let $G = (V, E)$ be a k-terminal graph for which we are given a ranked path decomposition with ranking function $\rho : V \to [p]$. Let B_{s_1}, B_{s_2} be two bags of this decomposition, for $s_1 < s_2$. We define the* rewiring *operation on (s_1, s_2) as follows: (i) for every non-terminal vertex $v \in B_{s_1}$ which is adjacent to a non-terminal vertex $u \in$*

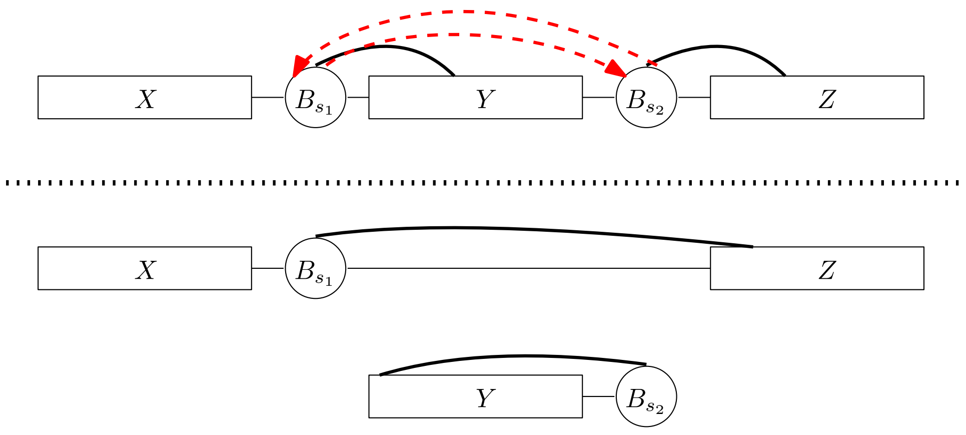

Figure 1 The rewiring operation of Definition 18. Edges from Y to B_{s_1} are rerouted towards B_{s_2}, while edges from Z to B_{s_2} are rerouted towards B_{s_1}. Edges incident on terminals are not modified.

$B_j \setminus (B_{s_1} \cup B_{s_2})$ for some $j \in [s_1+1, s_2-1]$ we delete the edge uv and add to the graph the edge uv', where $v' \in B_{s_2}$ and $\rho(v) = \rho(v')$, if such a v' exists (ii) for every non-terminal vertex $v \in B_{s_2}$ which is adjacent to a non-terminal vertex $u \in B_j \setminus B_{s_2}$ for some $j > s_2$, we delete the edge uv and add to the graph the edge uv', where $v' \in B_{s_1}$ and $\rho(v) = \rho(v')$, if such a v' exists.

Some explanations are in order regarding the motivation of the rewiring operation. We refer the reader to Figure 1. From standard properties of path decompositions, B_{s_1}, B_{s_2} are separators which break down the graph into three parts, call them X, Y, Z, which are respectively vertices which appear in a bag before B_{s_1}, between B_{s_1} and B_{s_2}, and after B_{s_2}. The rewiring operation leaves all edges incident on terminals and all edges incident on X unchanged. What it does is replace edges from Y to B_{s_1} with edges from Y to B_{s_2} and edges from Z to B_{s_2} with edges from Z to B_{s_1}. Intuitively, what this is meant to achieve is to break down the long path-like structure $X - B_{s_1} - Y - B_{s_2} - Z$ into the shorter path-like structure $X - B_{s_1} - Z$ and the ring-like structure $Y - B_{s_2}$. The idea here is that the $Y - B_{s_2}$ part is "disconnected" from the rest of the graph (more precisely, the k terminals separate this part from the rest of the graph, since terminals are not modified by this operation). Hence if we find many isomorphic such parts, they will also be identical in the sense of Definition 4, allowing us to delete one using Lemma 5. This argument is made precise in Lemma 20.

Before we do all these things, however, we need to be sure that the rewiring operation did not affect the validity of any FO formula of at most q quantifiers. The main claim now is that if s_1, s_2 are sufficiently far apart, we have a bound on the number of occurrences of non-terminal vertices in bags, and a sufficiently large block around B_{s_1} is block-isomorphic to a sufficiently large block around B_{s_2}, then the ball of radius $r = 2^q - 1$ around any vertex has remained unchanged. Hence, we can invoke Lemma 9 to conclude that the rewired graph is indistinguishable from the original graph for FO formulas with q quantifiers. Our main tool in proving this will be the following lemma.

▶ **Lemma 19.** *Let $G = (V, E)$ be a k-terminal graph with terminal set T, $B_1, \ldots, B_\ell$ a ranked path decomposition of G with ranking function $\rho : V \to [p]$ with the additional property that any non-terminal vertex appears in at most Δ bags of the decomposition. Fix an integer $q \geq 0$ and let $L = \Delta(2^q - 1)$. Let s_1, s_2 be such that (i) we have $s_1 > 4L$, $s_2 < \ell - 4L$, $s_2 - s_1 > 6L$ (ii) $\mathcal{B}(s_1 - L, s_1 + L)$ is block-isomorphic to $\mathcal{B}(s_2 - L, s_2 + L)$. Let G' be the graph obtained by applying the rewiring operation on (s_1, s_2). Then, for all FO formulas ϕ with at most q quantifiers we have $G \models \phi$ if and only if $G' \models \phi$.*

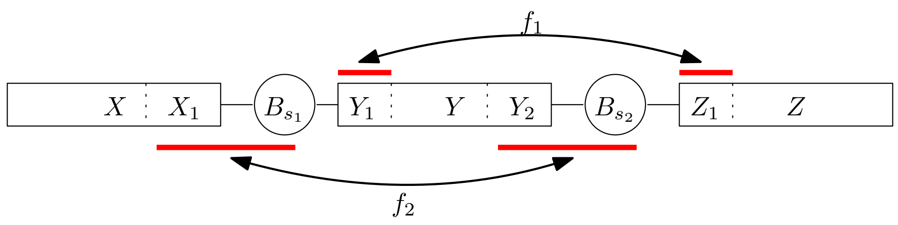

Figure 2 Schematic view of the two mappings of the proof of Lemma 19.

Finally, we argue that if we apply the rewiring operation on two block-isomorphic parts, then we obtain two parts of the graph which are identical in the sense of Definition 4. This will allow us to delete a part of the graph, once we gather sufficiently many identical parts.

▶ **Lemma 20.** *Let R be a positive integer, $G = (V, E)$ be a k-terminal graph with terminal set T, $B_1, \dots, B_\ell$ a ranked path decomposition of G with ranking function $\rho : V \to [p]$ with the property that no non-terminal vertex appears in more than R bags. Let s_1, s_2 be positive integers such that $s_2 - s_1 > 4R$ and $\mathcal{B}(s_1, s_1 + R)$ is block-isomorphic to $\mathcal{B}(s_2, s_2 + R)$. Let $j_1, j_2 \in [0, R-1]$ with $j_1 < j_2$ and let G' be the graph obtained after applying the rewiring operation on $(s_1 + j_1, s_1 + j_2)$ and also on $(s_2 + j_1, s_2 + j_2)$. Let Y_1 be the set of vertices that appear in a bag with index in $[s_1 + j_1 + 1, s_1 + j_2 - 1]$, but not in $B_{s_1+j_1} \cup B_{s_1+j_2}$. Similarly, let Y_2 be the set of vertices that appear in a bag with index in $[s_2 + j_1 + 1, s_2 + j_2 - 1]$, but not in $B_{s_2+j_1} \cup B_{s_2+j_2}$. Then $(Y_1 \cup B_{s_1+j_2}) \setminus T$ is identical to $(Y_2 \cup B_{s_2+j_2}) \setminus T$.*

5 Putting Everything Together

We are now ready to put everything together to obtain our model checking algorithm for FO logic. We formulate a procedure which can either simplify the graph in a way that does not affect the validity of the given formula (or any formula with the same number of quantifiers), or certify that the graph has bounded degree, and hence we can use known algorithms with an elementary dependence on the formula. On a high level, we take as input a graph G, a path decomposition of G, and a formula ϕ with q quantifiers and we will do the following:

1. Use Lemma 11 to normalize the decomposition and obtain a ranking of the vertices. In this ranking, vertices of rank 1 appear in a constant number of bags. We would like to extend this so that every vertex appears in a bounded number of bags. In the remainder we will use the number of bags a vertex appears in as a proxy bound for its degree.
2. Define a function $\Delta(i)$ which defines an acceptable bound for the number of occurrences in distinct bags for a vertex of rank i. This function will be a tower of exponentials of height roughly $2i$, but this is acceptable, since the maximum rank is upper-bounded by a function of the pathwidth, which we consider to be an absolute constant.
3. Examine the graph and check if any vertex of rank i appears in more than $\Delta(i)$ bags. If this is not the case, we can bound the maximum degree of the graph, and we are done.
4. Otherwise, find a vertex v of minimum rank i that appears more than $\Delta(i)$ times. Find a section of the decomposition where v appears, and where all bags contain the same vertices of rank higher than i (if $\Delta(i)$ is large, we can find such a section that is quite long). We label as terminals the vertices of the first and last bag of the section, and the vertices of rank at least i appearing in the section.
5. Now, the remaining vertices of the section appear a bounded (by $\Delta(i-1)$) number of times, and are separated from the rest of the graph by $k = O(p)$ terminals. However, they are quite numerous, as we assumed that v appears too many times. Therefore,

we can invoke the machinery of Section 4.2 to find some isomorphic parts. Note that it is important that vertices of rank at least i (which are now terminals) are common throughout the section, which allows us to invoke Lemmas 15 and 16.

6. Having found many isomorphic parts, we use the tools of Section 4.3 to perform the rewiring operation that will produce $q+1$ identical parts, of which we can remove one. We then "undo" the operation on the remaining parts, and obtain a smaller graph, where v appears in fewer bags, without changing whether ϕ is satisfied.

▶ **Definition 21.** *Let p, q be positive integers. We define the function $\Delta_{p,q}(i)$ as follows: $\Delta_{p,q}(1) = 3p$ and $\Delta_{p,q}(i+1) = 2^{2^{\Delta_{p,q}(i)\cdot 2^{20qp^2}}}$. When p, q are clear from the context, we will write $\Delta(i)$ to denote $\Delta_{p,q}(i)$.*

▶ **Observation 22.** *For each fixed p, i, the function $\Delta_{p,q}(i)$ is an elementary function of q. Furthermore, $\Delta_{p,q}(i)$ is a strictly increasing function of i.*

▶ **Lemma 23.** *There is an algorithm that takes as input an FO formula ϕ with q quantifiers, an n-vertex graph $G = (V, E)$, a normalized ranked path decomposition of G with ranking function $\rho : V \to [p]$, such that G contains a vertex that appears in at least $3p \cdot \Delta(p)$ bags of the decomposition. Then, the algorithm runs in polynomial time and outputs a smaller graph G' and a normalized ranked path decomposition of G' with the same ranking function ρ, such that $G \models \phi$ if and only if $G' \models \phi$.*

▶ **Theorem 24.** *For every fixed p, model checking a formula ϕ on a graph G with pathwidth p can be performed in time $f(\phi)|G|^{O(1)}$, where f is an elementary function.*

6 Conclusions

We have shown that FO model checking for graphs of bounded pathwidth has a complexity behavior that is in sharp contrast with both MSO logic for the same class of graphs and the complexity of FO logic on graphs of bounded treewidth. It may be interesting to improve upon our result by noting that our algorithm's dependence on the pathwidth is a tower of exponentials whose height is $O(\text{pw})$, where the hidden constant is roughly 16. This is in contrast with the meta-theorem of [14], where the height of the tower is roughly equal to the tree-depth. Can the height of the tower in our case can be made $\text{pw} + O(1)$, or is FO model checking on bounded pathwidth truly harder than for bounded tree-depth?

Another interesting research direction would be to explore parameters which lie between pathwidth and treewidth to attempt to trace the frontier of where the arguments of this paper break down. One idea would be to consider tree decompositions with a bounded number of leaf bags, which would generalize pathwidth, and see if, as long as the number of leaf bag and the width of the decomposition is an absolute constant, we can hope for an elementary dependence on the formula for FO model checking.

References

1 Stefan Arnborg, Jens Lagergren, and Detlef Seese. Easy problems for tree-decomposable graphs. *J. Algorithms*, 12(2):308–340, 1991. `doi:10.1016/0196-6774(91)90006-K`.

2 Rémy Belmonte, Eun Jung Kim, Michael Lampis, Valia Mitsou, and Yota Otachi. Grundy distinguishes treewidth from pathwidth. In *ESA*, volume 173 of *LIPIcs*, pages 14:1–14:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020.

3 Hans L. Bodlaender, Fedor V. Fomin, Daniel Lokshtanov, Eelko Penninkx, Saket Saurabh, and Dimitrios M. Thilikos. (meta) kernelization. *J. ACM*, 63(5):44:1–44:69, 2016.

4 Édouard Bonnet, Eun Jung Kim, Stéphan Thomassé, and Rémi Watrigant. Twin-width I: tractable FO model checking. In *FOCS*, pages 601–612. IEEE, 2020.

5 Bruno Courcelle. The monadic second-order logic of graphs. I. recognizable sets of finite graphs. *Inf. Comput.*, 85(1):12–75, 1990.

6 Bruno Courcelle, Johann A. Makowsky, and Udi Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory Comput. Syst.*, 33(2):125–150, 2000.

7 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms.* Springer, 2015.

8 Anuj Dawar, Martin Grohe, Stephan Kreutzer, and Nicole Schweikardt. Approximation schemes for first-order definable optimisation problems. In *LICS*, pages 411–420. IEEE Computer Society, 2006.

9 Zdenek Dvořák, Daniel Král, and Robin Thomas. Testing first-order properties for subclasses of sparse graphs. *J. ACM*, 60(5):36:1–36:24, 2013.

10 Eduard Eiben, Robert Ganian, and Stefan Szeider. Meta-kernelization using well-structured modulators. *Discret. Appl. Math.*, 248:153–167, 2018.

11 Markus Frick. Generalized model-checking over locally tree-decomposable classes. *Theory Comput. Syst.*, 37(1):157–191, 2004.

12 Markus Frick and Martin Grohe. Deciding first-order properties of locally tree-decomposable structures. *J. ACM*, 48(6):1184–1206, 2001.

13 Markus Frick and Martin Grohe. The complexity of first-order and monadic second-order logic revisited. *Ann. Pure Appl. Log.*, 130(1-3):3–31, 2004.

14 Jakub Gajarský and Petr Hliněný. Kernelizing MSO properties of trees of fixed height, and some consequences. *Log. Methods Comput. Sci.*, 11(1), 2015.

15 Jakub Gajarský, Petr Hlinený, Daniel Lokshtanov, Jan Obdrzálek, Sebastian Ordyniak, M. S. Ramanujan, and Saket Saurabh. FO model checking on posets of bounded width. In *FOCS*, pages 963–974. IEEE Computer Society, 2015.

16 Robert Ganian. Improving vertex cover as a graph parameter. *Discret. Math. Theor. Comput. Sci.*, 17(2):77–100, 2015.

17 Robert Ganian, Petr Hliněný, Jaroslav Nešetřil, Jan Obdržálek, and Patrice Ossona de Mendez. Shrub-depth: Capturing height of dense graphs. *Log. Methods Comput. Sci.*, 15(1), 2019.

18 Robert Ganian and Jan Obdržálek. Expanding the expressive power of monadic second-order logic on restricted graph classes. In *IWOCA*, volume 8288 of *Lecture Notes in Computer Science*, pages 164–177. Springer, 2013.

19 Robert Ganian, Friedrich Slivovsky, and Stefan Szeider. Meta-kernelization with structural parameters. *J. Comput. Syst. Sci.*, 82(2):333–346, 2016.

20 Martin Grohe and Stephan Kreutzer. Methods for algorithmic meta theorems. *Model Theoretic Methods in Finite Combinatorics*, 558:181–206, 2011.

21 Dusan Knop, Martin Koutecký, Tomás Masarík, and Tomás Toufar. Simplified algorithmic metatheorems beyond MSO: treewidth and neighborhood diversity. *Log. Methods Comput. Sci.*, 15(4), 2019.

22 Dusan Knop, Tomás Masarík, and Tomás Toufar. Parameterized complexity of fair vertex evaluation problems. In *MFCS*, volume 138 of *LIPIcs*, pages 33:1–33:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019.

23 Michael Lampis. Algorithmic meta-theorems for restrictions of treewidth. *Algorithmica*, 64(1):19–37, 2012. doi:10.1007/s00453-011-9554-x.

24 Michael Lampis. Model checking lower bounds for simple graphs. *Log. Methods Comput. Sci.*, 10(1), 2014.

25 Michael Lampis and Valia Mitsou. Fine-grained meta-theorems for vertex integrity. In *ISAAC*, volume 212 of *LIPIcs*, pages 34:1–34:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.

26 Michal Pilipczuk. Problems parameterized by treewidth tractable in single exponential time: A logical approach. In *MFCS*, volume 6907 of *Lecture Notes in Computer Science*, pages 520–531. Springer, 2011.

27 Detlef Seese. Linear time computable problems and first-order descriptions. *Math. Struct. Comput. Sci.*, 6(6):505–526, 1996.

28 Stefan Szeider. Monadic second order logic on graphs with local cardinality constraints. *ACM Trans. Comput. Log.*, 12(2):12:1–12:21, 2011.