Noname manuscript No.
(will be inserted by the editor)

Algorithmic Meta-theorems for Restrictions of Treewidth

Michael Lampis

the date of receipt and acceptance should be inserted later

Abstract Possibly the most famous algorithmic meta-theorem is Courcelle's theorem, which states that all MSO-expressible graph properties are decidable in linear time for graphs of bounded treewidth. Unfortunately, the running time's dependence on the formula describing the problem is in general a tower of exponentials of unbounded height, and there exist lower bounds proving that this cannot be improved even if we restrict ourselves to deciding FO logic on trees.

We investigate whether this parameter dependence can be improved by focusing on two proper subclasses of the class of bounded treewidth graphs: graphs of bounded vertex cover and graphs of bounded max-leaf number. We prove stronger algorithmic meta-theorems for these more restricted classes of graphs. More specifically, we show it is possible to decide any FO property in both of these classes with a singly exponential parameter dependence and that it is possible to decide MSO logic on graphs of bounded vertex cover with a doubly exponential parameter dependence. We also prove lower bound results which show that our upper bounds cannot be improved significantly, under widely believed complexity assumptions. Our work addresses an open problem posed by Michael Fellows.

1 Introduction

Algorithmic metatheorems are general statements of the form "*All problems sharing property P, restricted to a class of inputs I can be solved efficiently*".

A preliminary version of this paper appeared in ESA 2010

M. Lampis
Computer Science Department,
Graduate Center, City University of New York
E-mail: mlampis@gc.cuny.edu

The archetypal, and possibly most celebrated, such metatheorem is Courcelle's theorem which states that every graph property expressible in monadic second-order (MSO_2) logic is decidable in linear time if restricted to graphs of bounded treewidth [6]. Metatheorems have been a subject of intensive research in the last years producing a wealth of interesting results. Some representative examples of metatheorems with a flavor similar to Courcelle's can be found in the work of Frick and Grohe [17], where it is shown that all properties expressible in first order (FO) logic are solvable in linear time on planar graphs, and the work of Dawar et al. [8], where it is shown that all FO-definable optimisation problems admit a PTAS on graphs excluding a fixed minor (see [19] and [20] for more results on the topic). In all these works the defining property P for the problems studied is given in terms of expressibility in a logic language; in many cases metatheorems are stated with P being some other problem property, for example whether the problem is closed under the taking of minors. This approach, which is connected with the famous graph minor project of Robertson and Seymour [25] has also led to a wealth of significant and practical results, including the so called bi-dimensionality theory (see [9] for an overview and also the recent results of [2]).

In this paper we focus on the study of algorithmic metatheorems in the spirit of Courcelle's theorem, where the class of problems we attack is defined in terms of expressibility in a logic language. In this research area, many interesting extensions have followed Courcelle's seminal result: for instance, Courcelle's theorem has been extended to logics more suitable for the expression of optimisation problems [1]. It has also been investigated whether it's possible to obtain similar results for larger graph classes (see [7] for a metatheorem for bounded cliquewidth graphs, [15,16] for corresponding hardness results and [23] for hardness results for graphs of small but unbounded treewidth). Finally, lower bound results have been shown proving that the running times predicted by Courcelle's theorem can not be improved significantly in general [18].

This lower bound result is one of the main motivations of this work, because in some ways it is quite devastating. Though Courcelle's theorem shows that a vast class of problems is solvable in linear time on graphs of bounded treewidth, the "hidden constant" in this running time, that is, the running time's dependence on the input's other parameters, which are the graph's treewidth and the formula describing the problem, is in fact (in the worst case) a tower of exponentials. Unfortunately, in [18] it is shown that this tower of exponentials is unavoidable even if we restrict ourselves to deciding FO logic on trees.

In this paper our aim is to investigate if it is possible to go around this harsh lower bound by restricting the considered class of input graphs further. In other words, we are looking for meta-theorems which would imply that all of FO or MSO logic can be solved in time not only linear in the size of the graph, but also depending more reasonably on the secondary parameters, if we are willing to give up some of the generality of the class of bounded-treewidth graphs. We concentrate on two graph classes: graphs of bounded vertex cover

and graphs of bounded max-leaf number. We note that the investigation of the existence of stronger meta-theorems for these classes has been posed explicitly as an open problem by Fellows in [11].

Though graphs of bounded vertex cover or max-leaf number are considerably more restricted than bounded treewidth graphs, these classes are still interesting from the algorithmic point of view and the complexity of hard problems parameterized by vertex cover or max-leaf number has been investigated in the past ([13], [12]). Furthermore, as mentioned, strong lower bounds are known to apply to slightly more general classes: for bounded feedback vertex set and bounded pathwidth graphs even FO logic is non-elementary, while even for binary trees (thus for graphs of bounded treewidth and max degree) FO logic is at least triply exponential (again by [18]). Bounded vertex cover and bounded max-leaf number evade all these lower bound arguments so it's natural to ask what is exactly the complexity of FO and MSO logic for these classes of graphs?

The main results of this paper show that meta-theorems stronger than Courcelle's can indeed be shown for these classes of graphs. In addition, we show that our meta-theorems for vertex cover cannot be significantly improved under standard complexity assumptions.

Specifically, for the class of graphs of vertex cover bounded by k we show that

- All graph problems expressible with an FO formula ϕ can be solved in time linear in the graph size and singly exponential in k and $|\phi|$.
- All graph problems expressible with an MSO_2 formula ϕ can be solved in time linear in the graph size and doubly exponential in k and $|\phi|$.
- Unless n-variable 3SAT can be solved in time $2^{o(n)}$ (that is, unless the Exponential Time Hypothesis fails), then no $f(k) \cdot poly(|G|)$ algorithm exists to decide MSO logic on graphs of vertex cover k for any $f(k) = 2^{2^{o(k)}}$.
- Unless n-variable 3SAT can be solved in time $2^{o(n)}$, there is no algorithm which can decide if an FO formula ϕ with q quantifiers holds in a graph G of vertex cover k in time $f(k,q)n^c$, for any $f(k,q) = 2^{o(kq)}$.

Furthermore, for the class of graphs of max-leaf number bounded by k we show that

- All graph problems expressible with an FO formula ϕ can be solved in time linear in the graph size, polynomial in k and singly exponential in $|\phi|$.

Our upper bounds rely on techniques different from the standard dynamic programming on decompositions usually associated with treewidth. For max-leaf number we rely on the characterization of bounded max-leaf number graphs from [22] also used heavily in [12] and the fact that FO logic has limited counting power in paths. For vertex cover we exploit an observation that for FO logic two vertices that have the same neighbors are "equivalent" in a sense we will make precise. We state our results in this case in terms of a new graph "width" parameter that captures this graph property more

precisely than bounded vertex cover. We call the new parameter neighborhood diversity, and the upper bounds for vertex cover follow by showing that bounded vertex cover is a special case of bounded neighborhood diversity. Our essentially matching lower bounds on the other hand are shown for vertex cover. In the last section of this paper we prove some additional results for neighborhood diversity, beyond the algorithmic meta-theorems of the rest of the paper, which we believe indicate that neighborhood diversity might be a graph structure parameter of independent interest and that its algorithmic and graph-theoretic properties may merit further investigation.

2 Definitions and Preliminaries

2.1 Model Checking, FO and MSO logic

In this paper we will describe algorithmic meta-theorems, that is, general methods for solving all problems belonging in a class of problems. However, the presentation is simplified if one poses this approach as an attack on a single problem, the model checking problem. In the model checking problem we are given a logic formula ϕ, expressing a graph property, and a graph G, and we must decide if the property described by ϕ holds in G. In that case, we write $G \models \phi$. Clearly, if we can describe an efficient algorithm for model checking for a specific logic, this will imply the existence of efficient algorithms for all problems expressible in this logic. Let us now give more details about the logics we will deal with and the graphs which will be our input instances.

Our universe of discourse will be labeled, colored graphs. Specifically, we assume that the first part of the input is an undirected graph $G(V, E)$, a set of labels L, each associated with a vertex of V and a set of subsets of V, $\mathcal{C} = \{C_1, C_2, \ldots, C_c\}$, which we refer to as color classes. Note that it could be the case that several labels are assigned to the same vertex and that some vertex belongs in several color classes. The interesting case here is unlabeled, uncolored graphs (that is, $L = \mathcal{C} = \emptyset$), but the additional generality in the definition of the problem makes it easier to describe a recursive algorithm. We include labels in our definition to allow our formulas to refer to some constant vertices of the input graph.

The formulas of FO logic are those which can be constructed using vertex variables, denoted usually by $x_i, y_i, \ldots$, vertex labels denoted by l_i, color classes denoted by C_i, the predicates $E(x_i, x_j)$, $x_i \in C_j$, $x_i = x_j$ operating on vertex variables or labels, standard propositional connectives and the quantifiers $\exists, \forall$ operating on vertex variables. The semantics are defined in the usual way, with the $E()$ predicate being true if $(x_i, x_j) \in E$ and labels being interpreted as vertex constants corresponding to the vertices of the graph they are attached to. We also sometimes extend notation slightly by using conditional quantified variables: $\exists x : \psi(x)$ $(\phi(x))$ can be read as shorthand for $\exists x(\psi(x) \wedge \phi(x))$, while $\forall x : \psi(x)$ $(\phi(x))$ is short for $\forall x(\psi(x) \rightarrow \phi(x))$.

For MSO logic the additional property is that we now introduce set variables denoted by X_i and allow the quantifiers and the $\in$ predicate to operate on them. The semantics are defined in the obvious way. If the set variables are allowed to range over sets of vertices only, then the logic is referred to as MSO_1.

A variation is MSO_2 logic. Here, first-order variables are allowed to range over vertices or edges, and second-order variables range over sets of vertices or edges. To keep the presentation simple we will use the letters e_i, F_i for variables which range over edges and sets of edges respectively and we assume that it is clear from the context what domain each variable is quantified over. We also add the incidence predicate $I(v, e)$ which is true if edge e is incident on vertex v. Observe that in the first-order case it does not make a difference if one also allows quantification over edges or not, because any FO formula that uses edge variables can be transformed to an equivalent formula that only uses vertex variables: one simply replaces $\exists e$ with $\exists x \exists y : E(x, y)$ while also replacing $I(v, e)$ with $(v = x) \vee (v = y)$. It is known that this is not possible with MSO in general: there exist MSO_2 expressible properties which are not expressible in MSO_1. However, we will use such a transformation that works for graphs of small vertex cover.

2.2 Bounded Vertex Cover and neighborhood diversity

We will work extensively with graphs of bounded vertex cover, that is, graphs for which there exists a small set of vertices whose removal also removes all edges. We will usually denote the size of a graph's vertex cover by k. Note that there exist linear-time FPT algorithms for finding an optimal vertex cover in graphs where k is small (see e.g. [4]). Recall that an algorithm is called fixed-parameter tractable (FPT) if it runs in time $f(k)n^{O(1)}$ for some function f of the parameter k.

Our technique relies on the fact that in a graph of vertex cover k, the vertices outside the vertex cover can be partitioned into at most 2^k sets, such that all the vertices in each set have exactly the same neighbors outside the set and each set contains no edges inside it. Since we do not make use of any other special property of graphs of small vertex cover, we are motivated to define a new graph parameter, called neighborhood diversity, which intuitively seems to give the largest graph family to which we can apply our method in a straightforward way.

Definition 1 We will say that two vertices v, v' of a graph $G(V, E)$ have the same type iff they have the same colors and $N(v) \setminus \{v'\} = N(v') \setminus \{v\}$, where $N(v)$ denotes the set of neighbors of v.

Lemma 1 *Having the same type is an equivalence relation on the set of vertices of a graph G.*

Proof Obviously the relation is reflexive and symmetric, so we only need to prove that it is transitive. Suppose u and v have the same type and also that

v and w have the same type. First, $N(u) \setminus \{v\} = N(v) \setminus \{u\}$ and $N(v) \setminus \{w\} = N(w) \setminus \{v\}$ from the definition. From this we have $N(u) \setminus \{v, w\} = N(v) \setminus \{u, w\} = N(w) \setminus \{u, v\}$. So, it suffices to show that if v is connected to one of u, w it is connected to the other. But if u, v are connected then u, w are also since v and w have the same type. Now, because u and v have the same type and u, w are connected then v, w are also connected. □

Definition 2 A colored graph $G(V, E)$ has neighborhood diversity at most w, if there exists a partition of V into at most w sets, such that all the vertices in each set have the same type.

Lemma 2 *If an uncolored graph has vertex cover at most k, then it has neighborhood diversity at most $2^k + k$.*

Proof Construct k singleton sets, one for each vertex in the vertex cover and at most 2^k additional sets, one for each subset of vertices of the vertex cover. Place each of the vertices of the independent set in one of these sets, specifically the one which corresponds to its neighborhood in the vertex cover. □

In Section 7 we will show some more results about neighborhood diversity which indicate it may be an interesting parameter in its own right. However, until then our main focus will be graphs of bounded vertex cover. We will prove most of our algorithmic results in terms of neighborhood diversity and then invoke Lemma 2 to obtain our main objective. We will call a partition of the vertex set of a graph G into w sets such that all vertices in every set share the same type a neighborhood partition of width w. We will usually assume that a neighborhood partition of the graph is given to us, because otherwise one can easily be found in linear time by using the mentioned linear-time FPT algorithm for vertex cover and Lemma 2.

2.3 Bounded Max-Leaf Number

We say that a connected graph G has max-leaf number at most l if no spanning tree of G has more than l leaves. The algorithmic properties of this class of graphs have been investigated in the past [10,14,12]. In this paper we rely heavily on a characterization of bounded max-leaf graphs by Kleitman and West [22] which is also heavily used in [12].

Theorem 1 *[22] If a graph G has max-leaf number at most l, then G is a subdivision of a graph on $O(l)$ vertices.*

What this theorem tells us intuitively is that in a graph $G(V, E)$ with max-leaf number l there exists a set S of $O(l)$ vertices such that $G[V \setminus S]$ is a collection of $O(l^2)$ paths. Furthermore, only the endpoints of the paths can be connected to vertices of S in G.

It is well-known that a graph of max-leaf number at most l has a path decomposition of width at most $2l$. Furthermore, it must have maximum degree

at most l. Bounded max-leaf number graphs are therefore a subclass of the intersection of bounded pathwidth and bounded degree graphs (in fact, they are a proper subclass, as witnessed by the existence of say $2 \times n$ grids). Let us mention again that model checking FO formulas on binary trees has at least a triply exponential parameter dependence, so the results we present for graphs of bounded max-leaf number can also be seen as an improvement on the currently known results for FO logic on bounded degree graphs, for this more restricted case.

3 FO Logic for Bounded Vertex Cover

In this Section we show how any FO formula can be decided on graphs of bounded vertex cover number, with a singly exponential parameter dependence. Our main argument is that for FO logic, two vertices which have the same neighbors are essentially equivalent. We will state our results in the more general case of bounded neighborhood diversity and then show the corresponding result for bounded vertex cover as a corollary.

Lemma 3 *Let $G(V, E)$ be a graph and $\phi(x)$ a FO formula with one free variable. Let $v, v' \in V$ be two distinct unlabeled vertices of G that have the same type. Then $G \models \phi(v)$ iff $G \models \phi(v')$.*

Proof Let l be a new label which is not currently used in G. Let G_1 be the labeled graph we obtain from G if we associate l with v and G_2 be the labeled graph we obtain if we associate l with v'. Then the labeled graphs G_1 and G_2 are isomorphic (meaning that there is a one-to-one correspondence between them that also respects the labels). Therefore, $G_1 \models \phi(l)$ iff $G_2 \models \phi(l)$. □

Theorem 2 *Let ϕ be a FO sentence of quantifier depth q. Let $G(V, E)$ be a labeled colored graph with neighborhood diversity at most w and l labeled vertices. Then, there is an algorithm that decides if $G \models \phi$ in time $O((w + l + q)^q \cdot |\phi|)$, assuming that an optimal neighborhood partition is given with the input.*

Proof We will rely heavily on Lemma 3 and describe a recursive algorithm. If $q = 0$ the problem is trivial, so assume $q > 0$. Assume wlog that ϕ is in prenex normal form, $\phi = Qx\psi(x)$ where Q is $\exists$ or $\forall$.

Suppose that V can be partitioned into $V_1, V_2, \ldots V_w$ as required by the definition of neighborhood diversity. Now, by Lemma 3 if $v, v' \in V_i$ for some i, and neither of the two is labeled then $G \models \psi(v)$ iff $G \models \psi(v')$. Thus, it suffices to recursively model check at most $(w+l)$ sentences of $q-1$ quantifiers to decide ϕ: we try replacing x with each of the l labeled vertices or with one arbitrarily chosen nonlabeled representative from each V_i. If x is existentially quantified we decide that $G \models \phi$ if at least one of the resulting sentences is true, while if x is universally quantified we decide that $G \models \phi$ if all of the resulting sentences are true. In the process we introduce a new label. Repeating this process

constructs a computation tree with at most $\prod_{i=0}^{q-1}(w+l+i) = O\left((w+l+q)^q\right)$ leaves. The result of the computation tree can be evaluated in time linear in its size. □

Corollary 1 *There exists an algorithm which, given a FO sentence ϕ with q variables and an uncolored, unlabeled graph G on n vertices with vertex cover at most k, decides if $G \models \phi$ in time $2^{O(kq+q\log q)}|\phi| + O(2^k n)$.*

Proof The second term in the running time comes from the basic FPT algorithm for finding a vertex cover of size k. From this we can construct a neighborhood partition and invoke Theorem 2 and Lemma 2. □

Thus, the running time is (only) singly exponential in the parameters, while a straightforward observation that bounded vertex cover graphs have bounded treewidth and an application of Courcelle's theorem would in general have a non-elementary running time. Of course, a natural question to ask now is whether it is possible to do even better, perhaps making the exponent linear in the parameter. As we will see later on, this is not possible if we accept some standard complexity assumptions.

4 FO Logic for Bounded Max-Leaf Number

In this section we describe a model checking algorithm for FO logic on graphs of small max-leaf number. Because we are not going to solve MSO logic on this class of graphs, we can simplify things by assuming that our graphs only have labels and not colors (i.e. all vertices are initially uncolored). Our main tool is the mentioned observation that all but a small fraction of the vertices have degree 2, and therefore (since we assume without loss of generality that the graph is connected) induce paths. We call a maximal set of connected vertices of degree 2 a topo-edge.

Our main argument is that when a topo-edge is very long (exponentially long in the number of quantifiers of the first-order sentence we are model checking) its precise length does not matter. Readers familiar with classical results regarding Ehrenfeucht-Fraisse games and their use in proving negative results for the expressive power of FO logic on paths will recognize that the technique we use is an extension of this work to graphs of small max-leaf number (for more information on E-F games see for example [21]).

First we define a similarity relation on graphs.

Definition 3 Let G_1, G_2, be two labeled graphs. For a given q we will say that G_1 and G_2 are q-similar and write $G_1 \sim_q G_2$ iff G_1 contains a topo-edge of order at least 2^{q+1} consisting of unlabeled vertices, call it P, and G_2 can be obtained from G_1 by contracting one of the edges of P. We denote the transitive closure of the relation $\sim_q$ as $\sim_q^*$.

Our main technical tool is now the following lemma.

Lemma 4 *Let ϕ be a FO formula with q quantifiers. Then, for any two graphs G_1, G_2 if $G_1 \sim_q G_2$ then $G_1 \models \phi$ iff $G_2 \models \phi$. Therefore, if $G_1 \sim_q^* G_2$ then $G_1 \models \phi$ iff $G_2 \models \phi$.*

Proof We will prove the first statement by induction on q and the second statement follows directly from it. For $q = 0$ the statement is trivial because ϕ can only refer to labeled vertices and G_1, G_2 are identical with respect to these vertices.

Suppose that the statement is true for at most $q-1$ quantifiers. It suffices to show the statement for q quantifiers for a formula ϕ of the form $\exists x \psi(x)$, and the statement then easily follows for formulas which are boolean combinations of formulas of at most q quantifiers. So, suppose that $G_1 \models \exists x \psi(x)$. This means that there exists a vertex in G_1 such that if we label it with a new label l to obtain a graph G_1' (which is G_1 with the label l added) we have $G_1' \models \psi(l)$. Now we must take cases for the vertex where l is placed.

If l is placed on a vertex outside of P then it is not hard to see that $G_2 \models \phi$: we place l on the same vertex on G_2 (and obtain G_2') and now we have $G_1' \sim_{(q-1)} G_2'$ so from the inductive hypothesis $G_2' \models \psi(l)$.

Now the interesting case is when l is placed on a vertex of P. Number the vertices of P from 1 to $|P|$, starting from one of the endpoints of the path induced by P. Partition P into two parts: P_2 contains the last 2^q vertices and P_1 the rest. In G_2 we use the same numbering for the vertices of the path (of course now the numbering is from 1 to $|P| - 1$, since one edge has been contracted).

Suppose that l is placed on a vertex of P_1. We place l on the same vertex in G_2. Now, we have $G_1' \sim_{(q-1)} G_2'$, because in both graphs P has been broken into two paths P' and P''. P' has the same size on both (depending on the position where l was placed) and P'' has size at least 2^q on G_1' and one less than that on G_2'. So, by the inductive hypothesis $G_1' \models \psi(l)$ iff $G_2' \models \psi(l)$.

Finally, if l is placed on a vertex of P_2 we place l on a vertex of G_2 that has the same distance from the end of the path and two q-similar graphs G_1', G_2' are obtained, because the smaller part of the two into which P is broken has the same size on both graphs and the larger has size at least 2^q. So by the inductive hypothesis $G_1' \models \psi(l)$ iff $G_2' \models \psi(l)$.

The converse directions where we know that $G_2 \models \phi$ and need to show that this implies $G_1 \models \phi$ can be established with a similar argument. □

Now we are ready to state our main result of this section.

Theorem 3 *Let G be a graph on n vertices with max-leaf number k and ϕ a FO formula with q quantifiers. Then, there exists an algorithm for deciding if $G \models \phi$ running in time poly$(n) + 2^{O(q^2 + q \log k)}$.*

Proof By applying Theorem 1 we know that G can be partitioned into a set of at most $O(k)$ vertices of degree at least 3 and a collection of paths. By applying Lemma 4 we know that there exists a G' such that $G \sim_q^* G'$ and G' consists of the same $O(k)$ vertices of degree at least 3 and at most $O(k^2)$ paths

whose length is at most 2^{q+1}. Of course, G' can be found in time polynomial in n.

Now, we can apply the straightforward algorithm to model check ϕ on G'. The trivial algorithm takes time $O(|V|^q) = O((k^2 2^{q+1})^q)$ giving the promised running time. □

5 MSO Logic for Bounded Vertex Cover

Here we will follow a similar strategy as in Section 3 proving that if there is a very large number of vertices of a certain type in our graph then it is safe to delete some of them without affecting the truth of the MSO sentence we are trying to model check. To do this we first define another kind of similarity relation on graphs.

Definition 4 Let G_1, G_2, be two labeled colored graphs. For given integers q_S, q_V we will say that G_1 and G_2 are (q_S, q_V)-similar and write $G_1 \sim_{(q_S,q_V)} G_2$ iff G_2 can be obtained by G_1 by deleting an unlabeled vertex u and G_1 contains at least $2^{q_S} q_V$ additional unlabeled vertices of the same type as u. We denote the transitive closure of the relation $\sim_{(q_S,q_V)}$ as $\sim^*_{(q_S,q_V)}$.

Lemma 5 *Let ϕ be a MSO$_1$ formula with q_S set quantifiers and q_V vertex quantifiers. Then, for any two graphs G_1, G_2 if $G_1 \sim_{(q_S,q_V)} G_2$ then $G_1 \models \phi$ iff $G_2 \models \phi$. Therefore, if $G_1 \sim^*_{(q_S,q_V)} G_2$ then $G_1 \models \phi$ iff $G_2 \models \phi$.*

Proof We will prove the first statement by induction on $q_S + q_V$ and the second statement will immediately follow. For $q_V = 0$ the statement is trivial since without vertex variables the formula may only refer to the labeled vertices where G_1 and G_2 are identical so the statement is proved for $q_S + q_V = 0$.

Suppose that we have proved the statement for formulas with at most q quantified (vertex and set) variables and we are given a formula ϕ with q_S set variables and q_V vertex variables, where $q_S + q_V = q + 1$. We are also given two graphs G_1, G_2 such that $G_1 \sim_{(q_S,q_V)} G_2$. The two interesting cases are $\phi = \exists x \psi(x)$ and $\phi = \exists X \psi(X)$ (i.e. ϕ begins with an existentially quantified vertex or set variable) because the universal quantification case and boolean combinations of simpler formulas follow directly if we deal with these.

First, assume that $\phi = \exists X \psi(X)$ and that $G_1 \models \phi$. So, there exists a set S_1 of vertices of G_1 such that assigning a new color C to these, thus obtaining a new colored graph G'_1, gives us $G'_1 \models \psi(C)$. Let T be the type of vertices of G_1 where if we delete a vertex we obtain G_2 (recall that $|T| \geq 2^{q_S} q_V + 1$). We select a set S_2 of vertices of G_2 as follows: for every type other than T we select the same number of vertices as S_1 has selected from this type in G_1. From T, if S_1 contains at most half the vertices of type T in G_1 we place the same number of vertices from that type of G_2 in S_2. Otherwise, we select from type T one vertex less than S_1 contains from that type in G_1. We thus obtain a graph G'_2 by coloring all the vertices of S_2 with a new color C. Informally, we can say that G'_1 and G'_2 are the same except that one of the types of G'_1

has one more vertex than the corresponding type of G'_2. Note that we have made sure that this type has at least half the vertices of T. The claim now is that $G'_1 \sim_{(q_S-1,q_V)} G'_2$. To see this, observe that we can obtain G'_2 from G'_1 by deleting a vertex which had type T in G_1. If $|S_1 \cap T| = |S_2 \cap T|$ that vertex is one which did not receive the new color C, but this happens if at most half the vertices did, meaning its type contains at least $\lceil (2^{q_S} q_V + 1)/2 \rceil = 2^{q_S-1} q_V + 1$ vertices in G'_1. Otherwise, the vertex we can delete is one that received the new color, which means in this case its type again contains at least $2^{q_S-1} q_V + 1$ vertices. From the inductive hypothesis we now get $G'_1 \models \psi(C)$ iff $G'_2 \models \psi(C)$, which gives $G_1 \models \phi$ iff $G_2 \models \phi$. Similar arguments can be applied if we start with the assumption $G_2 \models \phi$.

Second, if $\phi = \exists x \psi(x)$ and $G_1 \models \phi$, there exists a vertex of G_1 such that assigning to it a new label l, thus obtaining a new graph G'_1, we have $G'_1 \models \psi(l)$. We assign the label l to a vertex of the same type in G_2, obtaining G'_2. Now, we have $G'_1 \sim_{(q_S,q_V-1)} G'_2$, because the number of unlabeled vertices in the type where G_1 and G_2 differ has been decreased by at most one. Therefore, it is now at least $2^{q_S} q_V \geq 2^{q_S}(q_V - 1) + 1$. By inductive hypothesis we get $G'_2 \models \psi(l)$ so $G_2 \models \phi$. Similar arguments can again be applied if we initially assume that $G_2 \models \phi$. □

Theorem 4 *Let G be a graph on n vertices with neighborhood diversity at most w and ϕ be a MSO$_1$ formula with q_S set quantifiers and q_V vertex quantifiers. Then, given a neighborhood partition of G, there exists an algorithm which can decide if $G \models \phi$ in time $2^{O(2^{q_S} w q_V + q_V \log q_V)}$.*

Proof Using Lemma 5 we can assume that no type has more than $2^{q_S} q_V$ vertices, otherwise we can delete one vertex and get an equivalent graph. Thus, the total number of vertices is at most $w 2^{q_S} q_V$.

The trivial MSO$_1$ model checking algorithm on a graph on n vertices would take time $O((2^n)^{q_S} \cdot n^{q_V} \cdot |\phi|)$ (try all possible cases for each set variable and each vertex variable). Using the above bound on n gives the promised running time. □

Corollary 2 *There exists an algorithm which, given a MSO$_1$ sentence ϕ with q variables and an uncolored, unlabeled graph G with vertex cover at most k, decides if $G \models \phi$ in time $2^{2^{O(k+q)}} + O(2^k n)$.*

Again, this gives a dramatic improvement compared to Courcelle's theorem, though exponentially worse than the case of FO logic. This is an interesting point to consider because for treewidth there does not seem to be any major difference between the complexities of model checking FO and MSO$_1$ logic.

The natural question to ask here is once again, can we do significantly better? For example, perhaps the most natural question to ask is, is it possible to solve this problem in $2^{2^{o(k+q)}}$? As we will see later on, the answer is no, if we accept some standard complexity assumptions.

Finally, let us briefly discuss the case of MSO_2 logic. In general this logic is more powerful than MSO_1, so it is not straightforward to extend Theorem 4 in this case. However, if we are not interested in neighborhood diversity but just in vertex cover we can observe that all edges in a graph with vertex cover of size k have one of their endpoints in one of the k vertices of the vertex cover. Thus, any edge set X can be written as the union of k edge sets. In turn, each of these k edge sets can easily be replaced by vertex sets, without loss of information, since we already know one of the endpoints of each of these edges. Using this trick we can replace every edge set variable in an MSO_2 sentence with k vertex set variables. This leads to a $2^{2^{O(kq)}}$ algorithm for MSO_2 logic on graphs of bounded vertex cover.

Lemma 6 *Let ϕ be an MSO_2 sentence with q quantifiers and G be a graph of vertex cover k. Then, there exists an MSO_1 sentence ϕ' with $O(kq)$ quantifiers and a graph G' with vertex cover k and k labeled vertices such that $G \models \phi$ iff $G' \models \phi'$.*

Proof We'll first argue that edge and edge-set variables can be removed from ϕ. G' will simply be G with k labels $l_1, \ldots, l_k$, each attached to a different vertex of the vertex cover. Suppose wlog that ϕ is in prenex normal form, and the edge-set variables which appear in ϕ are $F_1, \ldots, F_m$, while the edge variables which appear are $e_1, \ldots, e_p$. For the former, we replace their quantifications QF_i, where Q is $\exists$ or $\forall$, with k new quantified set variables for each: $QX_{i,1}QX_{i,2}\ldots QX_{i,k} : (\wedge_{1\leq j\leq k}\forall x : x \in X_{i,j}(E(x, l_j)))$. For edge variables, we replace Qe_i with $Qx_{e_i}Qy_{e_i} : E(x_{e_i}, y_{e_i})$ where x_{e_i}, y_{e_i} are new vertex variables.

We continue by replacing every occurence of $e_i \in F_j$ with the formula $\vee_{1\leq i'\leq k}(x_i = l_{i'} \wedge y_i \in X_{j,i'})$, while we replace each occurence of $I(x, e_i)$ with $(x = x_{e_i} \vee x = y_{e_i})$. Thus, we are left with an MSO_1 formula. It is not hard to see that the new formula has at most k quantifiers for every quantifier of the old formula. Its total size is also at most $O(k|\phi|)$.

Now ϕ' is equivalent to ϕ because every valuation of the edge and edge-set variables of ϕ corresponds to a valuation of the new variables of ϕ' and vice-versa.

□

Corollary 3 *There exists an algorithm which, given a MSO_2 sentence ϕ with q variables and an uncolored, unlabeled graph G with vertex cover at most k, decides if $G \models \phi$ in time $2^{2^{O(kq)}} + O(2^k n)$.*

6 Lower Bounds for Parameterizations by Vertex Cover

In this Section we will prove some lower bound results for the model checking problems we are dealing with for vertex cover. Our proofs rely on a construction which reduces SAT to a model checking problem on a graph with small vertex cover.

Given a propositional 3-CNF formula ϕ_p with n variables and m clauses, we want to construct a graph G that encodes its structure, while having a small vertex cover. The main problem is encoding numbers up to n with graphs of small vertex cover but this can be achieved by using the binary representation of numbers. We will begin by constructing a colored graph and then briefly describe how the reduction can be strengthend to apply to uncolored graphs as well. Without loss of generality we will assume that n is a power of 2 (dummy variables can be added to ϕ_p if necessary).

We begin constucting a graph by adding $7 \log n$ vertices, call them $u_{(i,j)}, 1 \leq i \leq 7, 1 \leq j \leq \log n$. Add all edges of the form $(u_{(i,j)}, u_{(k,j)})$ (so we now have $\log n$ disjoint copies of K_7). Let $N_i = \{u_{(i,j)} \mid 1 \leq j \leq \log n\}$.

For every variable x_i in ϕ_p add a new vertex to the graph, call it v_i. Define for every number i the set $X(i) = \{j \mid$ the j-th bit of the binary representation of i is 1$\}$. Add the edges $(v_i, u_{(1,j)}), j \in X(i)$, that is, connect every variable vertex with the vertices of N_1 that correspond to the binary representation of its index. Let $U = \{v_i \mid 1 \leq i \leq n\}$ be the vertices corresponding to variables.

For every clause c_i in ϕ_p add a new vertex to the graph, call it w_i. If the first literal in c_i is a positive variable x_k then add the edges $(w_i, u_{(2,j)}), j \in X(k)$. If the first literal is a negated variable $\neg x_k$, add the edges $(w_i, u_{(3,j)}), j \in X(k)$. Proceed in a similar way for the second and third literal, that is, if the second literal is positive connect w_i with the vertices that correspond to the binary representation of the variable in N_4, otherwise in N_5. For the third literal do the same with N_6 or N_7. Let $W = \{w_i \mid 1 \leq i \leq m\}$ be the vertices corresponding to clauses.

Finally, set the color classes to be $\{N_1, N_2, \ldots, N_7, U, W\}$.

Now, looking at the graph it is easy to see if a vertex v_i corresponds to a variable that appears positive in the clause represented by a vertex w_i. They must satisfy the formula

$$pos(v_i, w_j) = \bigvee_{k=2,4,6} \forall x : x \in N_1 \, (\exists y : y \in N_k((E(v_i, x) \leftrightarrow E(w_j, y)) \wedge E(x, y))))$$

It is not hard to define $neg(v_i, w_j)$ in a similar way. Now it is straightforward to check if ϕ_p was satisfiable:

$$\phi = \exists S(\forall x : x \in S(x \in U)) \wedge (\forall w : w \in W(\exists x : x \in U \\ (((pos(x, w) \wedge x \in S) \vee (neg(x, w) \wedge x \notin S)))))$$

Clearly, ϕ holds in the constructed graph iff ϕ_p is satisfiable. S corresponds to the set of variables set to true in a satisfying assignment. It is relatively easy to eliminate the colors and labels from the construction above. Colors can be reduced to labels by adding a labeled vertex for each color and connecting all the vertices that had that color to the labeled vertex. Finally, labels can also be eliminated by attaching a different FO-definable gadget to each labeled vertex. In particular, observe that all the vertices in our construction now have

degree at least two. Thus, attaching a leaf to a vertex can be seen as labeling it (this can be expressed in FO logic). Similarly, we attach two leaves to the next vertex we want to label and so on. We only need a constant number of labels to simulate the constant number of color classes our construction uses. Therefore the lower bounds given below apply to the natural form of the problem.

Lemma 7 *$G \models \phi$ iff ϕ_p is satisfiable. Furthermore, ϕ has size $O(1)$ and G has a vertex cover of size $O(\log n)$.*

Proof Follows from the description of the construction. □

Theorem 5 *Let G a graph with vertex cover k. Then, there exists a fixed MSO formula ϕ such that, unless 3-SAT can be solved in time $2^{o(n)}$, there is no algorithm which decides if $G \models \phi$ in time $O(2^{2^{o(k)}} \cdot poly(n))$.*

Proof We have already observed that the construction we described has $k = O(\log n)$. Since the construction can clearly be performed in polynomial time, an algorithm running in time $O(2^{2^{o(k)}} \cdot poly(n))$ would imply an algorithm for SAT running in $2^{o(n)} \cdot poly(n)$. □

Note that, since the formula used in Theorem 5 is fixed, it is also implied that a $O(2^{2^{o(k+q)}} \cdot poly(n))$ algorithm would also give a sub-exponential algorithm for SAT. Thus, Theorem 5 essentially matches the results of Corollary 2.

Theorem 6 *Let ϕ be a FO formula with q_v vertex quantifiers and G a graph with vertex cover k. Then, unless 3-SAT can be solved in time $2^{o(n)}$, there is no algorithm which decides if $G \models \phi$ in time $O(2^{o(kq_v)} \cdot poly(n))$.*

Proof We use the same construction, but begin our reduction from Weighted 3-SAT, a well-known W[1]-hard parameterized problem. Suppose we are given a 3-CNF formula and a number w and we are asked if the formula can be satisfied by setting exactly w of its variables to true. The formula ϕ we construct is exactly the same, except that we replace the $\exists S$ with $\exists x_1 \exists x_2 \ldots \exists x_w (\bigwedge_{1 \leq i < j \leq w} x_i \neq x_j)$ and all occurences of $x \in S$ with $\bigvee_{1 \leq i \leq w} x = x_i$. It is not hard to see that the informal meaning of ϕ now is to ask whether there exists a set of exactly w distinct variables such that setting them to true makes the formula true.

We now have $q_v = w + O(1)$ so an algorithm running in time $2^{o(kq_v)} \cdot poly(n)$ would imply an algorithm for Weighted 3-SAT running in $2^{o(w \log n)} \cdot poly(n) = n^{o(w)}$, and thus, by the results of [3] that there exists a sub-exponential algorithm for 3-SAT. □

7 Neighborhood Diversity

In this Section we give some general results on the new graph parameter we have defined, neighborhood diversity. We will use $nd(G), tw(G), cw(G)$ and

$vc(G)$ to denote the neighborhood diversity, treewidth, cliquewidth and minimum vertex cover of a graph G. We will call a partition of the vertex set of a graph G into w sets such that all vertices in every set share the same type a neighborhood partition of width w.

First, some general results

Theorem 7 *1. Let $V_1, V_2, \ldots, V_w$ be a neighborhood partition of the vertices of a graph $G(V, E)$. Then each V_i induces either a clique or an independent set. Furthermore, for all i, j the graph either includes all possible edges from V_i to V_j or none.*

2. For every graph G we have $nd(G) \leq 2^{vc(G)} + vc(G)$ and $cw(G) \leq nd(G) + 1$. Furthermore, there exist graphs of constant treewidth and unbounded neighborhood diversity and vice-versa.

3. There exists an algorithm which runs in polynomial time and given a graph $G(V, E)$ finds a neighborhood partition of the graph with minimum width.

Proof For the first statement, to show that every V_i induces either a clique or an independent set, we may assume that $|V_i| \geq 3$, otherwise the statement is trivial. Suppose that some V_i includes at least one edge (u, v). Consider another vertex $w \in V_i$. The vertex w has the same type as u, therefore (w, v) must be an edge. Similarly, (w, u) must also be an edge, and generally all other vertices in V_i are connected to both u and v. Finally, if w, w' are two vertices of V_i other than u, v it must be the case that (w, w') is an edge, because (u, w') is an edge and u and w have the same type. Another way to see this observation is to say that the property of two vertices having the same type is an equivalence relation as observed in Lemma 1.

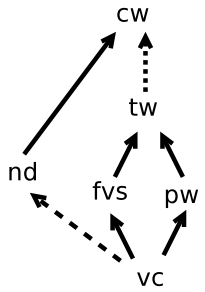
For the edges between V_i and V_j, suppose that there exists at least an edge (u, v) between them and let $w \in V_i$, $w' \in V_j$. v has the same type as w', therefore (u, w') must be an edge. Now, w has the same type as u so (w, w') must also be an edge, and once again this is true for any w, w'.

We have already shown the first part of the second statement. For the part with cliquewidth, we remind the reader that the graphs of cliquewidth k are those which can be constructed by repeated application of the following operations: introducing a new vertex with a label in $\{1, \ldots, k\}$, joining all vertices of label i with all vertices of label j, renaming all vertices of label i to label j and taking disjoint union of two graphs of cliquewidth at most k. We must show how to construct a graph in such a way starting from a neighborhood partition of width w, using at most $w + 1$ labels. The labels in $\{1, \ldots, w\}$ will only be used for the vertices of the corresponding set in the partition, while the extra label will be used to construct the cliques. For each V_i, if V_i is an independent set introduce $|V_i|$ new vertices with label i. If V_i is a clique repeat $|V_i|$ times: introduce a new vertex of label $w + 1$, join all vertices of label i to $w + 1$ and rename $w + 1$ to i. After all the vertices have been introduced, for all i, j for which the graph had all edges between V_i and V_j join the vertices labeled i with those labeled j.

To see why treewidth is incomparable to neighborhood diversity consider the examples of a complete bipartite graph $K_{n,n}$ and a path on n vertices.

Finally, let us argue why neighborhood diversity is computable in polynomial time. As mentioned already, the property of two vertices having the same type is an equivalence relation. The number of sets in an optimal partition is equal to the number of equivalence classes, so we simply need to determine these. It is easy to see that one can check if two vertices have the same type in polynomial time, so dividing the vertices into equivalence classes can also be done in polynomial time by checking all pairs of vertices. □

Taking into account the observations of Theorem 7 we summarize what we know about the graph-theoretic and algorithmic properties of neighborhood diversity and related measures in Figure 1.



	FO	MSO	MSO_2
Cliquewidth	$tow(w)$	$tow(w)$	$tow(w)$
Treewidth	$tow(w)$	$tow(w)$	$tow(w)$
Vertex Cover	$2^{O(w)}$	$2^{2^{O(w)}}$	$2^{2^{O(w)}}$
Neighborhood Diversity	$poly(w)$	$2^{O(w)}$	Open

Fig. 1 A summary of the relations between neighborhood diversity and other graph widths. Included are cliquewidth, treewidth, pathwidth, feedback vertex set and vertex cover. Arrows indicate generalization, for example bounded vertex cover is a special case of bounded feedback vertex set. Dashed arrows indicate that the generalization may increase the parameter exponentially, for example a graph of treewidth w has cliquewidth at most $O(2^w)$ and this is known to be tight. The table summarizes the best known model checking algorithm's dependence on each width for the corresponding logic.

There are several interesting points to make here. First, though this work is motivated by a specific goal, beating the lower bounds that apply to graphs of bounded treewidth by concentrating on a special case, it seems that the results which can achieved are at least somewhat better; it is possible to prove stronger meta-theorems by focusing on a class which is not necessarily smaller than bounded treewidth, only different. However, this class is a special case of another known width which generalizes treewidth as well, namely cliquewidth. Since the lower bound results which apply to treewidth apply to cliquewidth as well, this work can perhaps be viewed more appropriately as an improvement on the results of [7] for bounded cliquewidth graphs when restricting our attention to the more special case of bounded neighborhood diversity.

Second, there is the case of MSO_2 logic. The very interesting hardness results shown in [15,16] demonstrate that the tractability of MSO_2 logic is in a sense the price one has to pay for the additional generality that cliquewidth provides over treewidth. It is natural to ask if these results can be strengthened to apply to neighborhood diversity or MSO_2 logic can be shown to be tractable when parameterized by neighborhood diversity.

Though we cannot yet fully answer the above question related to MSO_2, we can offer some first indications that this direction might merit further investigation. In [15] it is shown that MSO_2 model checking is not fixed-parameter tractable when the input graph's cliquewidth is the parameter by considering three specific MSO_2-expressible problems and showing that they are W-hard. The problems considered are Hamiltonian cycle, Graph Chromatic Number and Edge Dominating Set. We can show that these three problems admit FPT algorithms on graphs of small neighborhood diversity (for Hamiltonian cycle this is in fact an easy consequence of an old result from [5]). Since small neighborhood diversity is a special case of small cliquewidth, where these problems are hard, this result could be of independent interest.

Theorem 8 *Given an n-vertex graph G whose neighborhood diversity is w, there exist algorithms running in time $O(f(w) \cdot poly(n))$ that decide Hamiltonian cycle, Graph Chromatic Number and Edge Dominating Set.*

Proof We will make use of an auxiliary graph G' on w vertices. Each vertex of G' corresponds to a set in an optimal neighborhood partition of G and two vertices of G' have an edge iff the corresponding sets of the partition of G have all possible edges between them.

For Hamiltonian cycle we rely on the results of [5]. There it is shown that the TSP problem is FPT parameterized by the number of cities even when one has to visit each city a number of times given in the input. We take G' and add a self-loop to every vertex that corresponds to a clique in G. The number of times we want to visit each vertex of G' is equal to the size of the corresponding set of the neighborhood partition. We set the cost of each edge to 1 and each non-edge to 2 and solve the resulting TSP instance on G'. G is Hamiltonian iff there is a TSP tour on G' with cost n.

Let us now show how to solve graph coloring. Observe that if a set V_i of a neighborhood partition of G induces an independent set, we can delete all of its vertices but one, without affecting the graph's chromatic number, because there always exists an optimal coloring where all the vertices of V_i take the same color. So, we can assume without loss of generality that all the sets V_i of a neighborhood partition of G induce cliques (some of them of order one).

We are now going to reformulate the problem, using the fact that in any coloring of G every color class intersects each set of the neighborhood partition in at most one vertex (since all the sets of the partition induce cliques). In other words, every color class essentially coincides with an independent set of G'. Let $\mathcal{I}$ be the set of all independent sets of G' and let V_p be the set of vertices of G', each of which represents a set in the neighborhood partition of G. Consider the following ILP with variables $x_I,\ I \in \mathcal{I}$ (i.e. at most 2^w variables):

$$\min \sum_{I \in \mathcal{I}} x_I$$
$$\text{s.t. } \forall v \in V_p : \sum_{I : v \in I} x_I = |V_p|$$

Intuitively, in the above ILP the variables x_I encode how many different color classes coincide with the independent set I of G' in a coloring of G.

We argue that the optimal solution to this problem is exactly the chromatic number of G. First, suppose that there exists a coloring of G with c colors. Every color class induces an independent set, so by looking at the collection of sets of the neighborhood partition that the class intersects we have that every color class corresponds to some $I \in \mathcal{I}$. Several color classes may correspond to the same set I, so we set x_I to be equal to the number of color classes that correspond to the set I. It should be easy to see then that $\sum x_I = c$. The requirement that $|V_i| = \sum_{I:i \in I} x_I$ is satisfied because every vertex belongs in exactly one color class.

For the other direction, suppose that there exist integers x_I which satisfy the ILP and $\sum x_I = s$. We can produce a coloring of G with s colors: as long as there exists an I with $x_I > 0$ select a new color and arbitrarily pick exactly one uncolored vertex from each V_i with $i \in I$. Color these vertices with the new color, and set $x_I := x_I - 1$. It is not hard to see that this algorithm will use exactly s colors. It produces a valid coloring because in the beginning we have $|V_i| = \sum_{I:i \in I} x_I$ and the equality continues to hold at each step if we only count the uncolored vertices on the left-hand side of the equation.

Thus, intuitively we have reformulated the problem as one of selecting the independent sets that will form the color classes. The main observation now is that the possible choices for the independent sets are only 2^w. Thus, we can recast the problem as an Integer Linear Program with at most 2^w variables and w constraints for the equations $|V_i| = \sum_{I:i \in I} x_I$. It follows from a seminal result of Lenstra ([24]) that solving this can be performed in FPT time.

In the edge dominating set problem, we are asked to find a set of edges of minimum size such that all other edges share an endpoint with one of the edges we selected. This problem is equivalent to the minimum maximal matching problem, where we are trying to find a minimum size independent set of edges that cannot be extended by picking another edge of the graph. To see why the optimal solution to the edge dominating set problem can always be transformed to a matching, suppose that we have a solution S which includes two edges $(u, v), (u, v')$. Now, if all the neighbors of v' are incident on an edge of S we can simply remove (u, v') from S and improve the size of the solution. If there is a neighbor w of v' that is not incident on an edge of S we can replace (u, v') with (w, v') in S. To see why a solution to the edge dominating set problem can always be transformed to a matching that is maximal, suppose that the matching we got was not maximal. Then there would be two unmatched vertices connected by an edge, which would imply that this edge is not dominated.

Our algorithm will proceed as follows: for every vertex cover V' of G' repeat the following (there are at most 2^w vertex covers to be considered): from V' infer a vertex cover of G by placing into the vertex cover all the vertices that belong in a type whose corresponding vertex is in V'. Also place in the vertex cover all but one (arbitrarily chosen) vertex of every vertex type that induces a clique but whose corresponding vertex is not in V'. Denote the resulting

vertex cover of G by V''. Find a maximum matching on the graph induced by V'', call it M_1. Take the bipartite graph induced by the unmatched vertices of V'' and $V \setminus V''$ and find a maximum matching there, call it M_2. The solution produced is $M_1 \cup M_2$. After repeating this for all vertex covers of G', pick the smallest solution.

Now we need to argue why this solution is optimal. Let S be an optimal solution for G. We say that a set of the neighborhood partition V_i is full if all of its vertices are incident on edges of S. If we take in G' the corresponding vertices of the full sets of G, they must form a vertex cover of G', otherwise there would be two neighboring vertices with neither having any edge of S incident to it, which would mean that S is not maximal. This is a vertex cover of G' considered by our algorithm, since our algorithm considers all vertex covers of G', call it V'. Let V'' be again the vertex cover of G our algorithm derived from V' by also including a minimal number of vertices from each remaining clique. Let V^* be the set of vertices of G incident on some edge of S, which must also be a vertex cover of G. Without loss of generality we will assume that $V'' \subseteq V^*$, because the two vertex covers of G agree on taking all vertices of the full sets and V'' takes a minimal number of vertices from every other clique. Even if V^* leaves out a different vertex from some clique because all the vertices of the clique have the same neighbors we can apply an exchanging argument and transform S appropriately without increasing its size so that both sets leave out the same vertex.

Now note that $|M_2| \leq |V''| - 2|M_1|$. So our algorithm's solution has size at most $|V''| - |M_1|$. On the other hand the optimal solution S includes some edges with both endpoints in V'', call this set S_1. Because M_1 is a maximum matching, $|S_1| \leq |M_1|$. From what we have so far, the fact that all vertices of V^* are matched by S and the fact that V'' is a vertex cover, so $V^* \setminus V''$ induces no edges we have $|V^*| = |V^* \cap V''| + |V^* \setminus V''| = |V''| + |V''| - 2|S_1| \geq 2|V''| - 2|M_1|$. This implies that $|S| \geq |V''| - |M_1|$ which concludes the proof. □

8 Conclusions and Open Problems

In this paper we presented algorithmic meta-theorems for more restricted inputs, which improve the running times implied by previously known meta-theorems. In this way we have partially explored the trade-off which can be achieved between running time and generality. This is an interesting area for further investigations and much more can be done.

For bounded max-leaf number the complexity of MSO logic is unknown. Quite likely, it is possible to improve upon Courcelle's theorem for this case as well, but the problem remains open. Also, it would be nice to obtain a lower bound for FO logic in the case of max-leaf showing that it is impossible to achieve $2^{o(q^2)}$, i.e. that the exponent must be quadratic in the number of quantifiers of the FO formula. For neighborhood diversity the most interesting open problem is the complexity of MSO_2.

Going further, it would also make sense to investigate whether restricting the model checking problem to graphs of bounded vertex cover or max-leaf number can also allow us to solve logics wider than MSO_2. Some indications that this may be possible are given in [13]

Acknowledgement: I would like to thank the anonymous reviewers for offering various suggestions to improve this paper and catching errors in the proofs of Theorems 7 and 8.

References

1. Stefan Arnborg, Jens Lagergren, and Detlef Seese. Easy problems for tree-decomposable graphs. *J. Algorithms*, 12(2):308–340, 1991.
2. Hans Bodlaender, Fedor Fomin, Daniel Lokshtanov, Saket Saurabh Eelko Penninkx, and Dimitrios Thilikos. (Meta) kernelization. In *FOCS*, 2009.
3. Jianer Chen, Xiuzhen Huang, Iyad A. Kanj, and Ge Xia. Linear FPT reductions and computational lower bounds. In László Babai, editor, *STOC*, pages 212–221. ACM, 2004.
4. Jianer Chen, Iyad A. Kanj, and Ge Xia. Improved parameterized upper bounds for vertex cover. In Rastislav Kralovic and Pawel Urzyczyn, editors, *MFCS*, volume 4162 of *Lecture Notes in Computer Science*, pages 238–249. Springer, 2006.
5. S.S. Cosmadakis and C.H. Papadimitriou. The traveling salesman problem with many visits to few cities. *SIAM Journal on Computing*, 13:99, 1984.
6. Bruno Courcelle. The Monadic Second-Order Logic of Graphs. I. Recognizable Sets of Finite Graphs. *Inf. Comput.*, 85(1):12–75, 1990.
7. 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.
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. Erik D. Demaine and MohammadTaghi Hajiaghayi. The bidimensionality theory and its algorithmic applications. *Comput. J.*, 51(3):292–302, 2008.
10. Vladimir Estivill-Castro, Michael R. Fellows, Michael A. Langston, and Frances A. Rosamond. FPT is P-Time Extremal Structure I. In Hajo Broersma, Matthew Johnson, and Stefan Szeider, editors, *ACiD*, volume 4 of *Texts in Algorithmics*, pages 1–41. King's College, London, 2005.
11. Michael R. Fellows. Open problems in parameterized complexity, AGAPE spring school on fixed parameter and exact algorithms, 2009.
12. Michael R. Fellows, Daniel Lokshtanov, Neeldhara Misra, Matthias Mnich, Frances A. Rosamond, and Saket Saurabh. The Complexity Ecology of Parameters: An Illustration Using Bounded Max Leaf Number. *Theory Comput. Syst.*, 45(4):822–848, 2009.
13. Michael R. Fellows, Daniel Lokshtanov, Neeldhara Misra, Frances A. Rosamond, and Saket Saurabh. Graph layout problems parameterized by vertex cover. In Seok-Hee Hong, Hiroshi Nagamochi, and Takuro Fukunaga, editors, *ISAAC*, volume 5369 of *Lecture Notes in Computer Science*, pages 294–305. Springer, 2008.
14. Michael R. Fellows and Frances A. Rosamond. The Complexity Ecology of Parameters: An Illustration Using Bounded Max Leaf Number. In S. Barry Cooper, Benedikt Löwe, and Andrea Sorbi, editors, *CiE*, volume 4497 of *Lecture Notes in Computer Science*, pages 268–277. Springer, 2007.
15. Fedor V. Fomin, Petr A. Golovach, Daniel Lokshtanov, and Saket Saurabh. Clique-width: on the price of generality. In Claire Mathieu, editor, *SODA*, pages 825–834. SIAM, 2009.
16. F.V. Fomin, P.A. Golovach, D. Lokshtanov, and S. Saurabh. Intractability of clique-width parameterizations. *SIAM Journal on Computing*, 39(5):1941–1956, 2010.

17. Markus Frick and Martin Grohe. Deciding first-order properties of locally tree-decomposable structures. *J. ACM*, 48(6):1184–1206, 2001.
18. Markus Frick and Martin Grohe. The complexity of first-order and monadic second-order logic revisited. *Ann. Pure Appl. Logic*, 130(1-3):3–31, 2004.
19. Martin Grohe. Logic, graphs, and algorithms. *Electronic Colloquium on Computational Complexity (ECCC)*, 14(091), 2007.
20. Petr Hlinený, Sang il Oum, Detlef Seese, and Georg Gottlob. Width parameters beyond tree-width and their applications. *Comput. J.*, 51(3):326–362, 2008.
21. N. Immerman. *Descriptive complexity*. Springer Verlag, 1999.
22. D.J. Kleitman and D.B. West. Spanning trees with many leaves. *SIAM Journal on Discrete Mathematics*, 4:99, 1991.
23. Stephan Kreutzer and Siamak Tazari. On brambles, grid-like minors, and parameterized intractability of monadic second order logic. In *SODA*, 2010.
24. HW Lenstra Jr. Integer programming with a fixed number of variables. *Mathematics of operations research*, pages 538–548, 1983.
25. Neil Robertson and Paul D. Seymour. Graph minors. I-XXIII. *J. Comb. Theory, Ser. B*, 1983-2004.