Logical Methods in Computer Science
Volume 15, Issue 4, 2019, pp. 12:1–12:32
https://lmcs.episciences.org/

Submitted Dec. 07, 2018
Published Dec. 04, 2019

SIMPLIFIED ALGORITHMIC METATHEOREMS BEYOND MSO: TREEWIDTH AND NEIGHBORHOOD DIVERSITY

DUŠAN KNOP, MARTIN KOUTECKÝ, TOMÁŠ MASAŘÍK, AND TOMÁŠ TOUFAR

Department of Theoretical Computer Science, Faculty of Information Technology, Czech Technical University in Prague, Prague, Czech Republic
e-mail address: knop@kam.mff.cuni.cz

Faculty of IE&M, Technion – Israel Institute of Technology, Haifa, Israel
Computer Science Institute, Charles Univesity, Prague, Czech Republic
e-mail address: koutecky@iuuk.mff.cuni.cz

Department of Applied Mathematics, Charles University, Prague, Czech Republic
Faculty of Mathematics, Informatics and Mechanics, University of Warsaw, Poland
e-mail address: masarik@kam.mff.cuni.cz

Computer Science Institute, Charles University, Prague, Czech Republic
e-mail address: toufi@iuuk.mff.cuni.cz

Abstract. This paper settles the computational complexity of model checking of several extensions of the monadic second order (MSO) logic on two classes of graphs: graphs of bounded treewidth and graphs of bounded neighborhood diversity.

A classical theorem of Courcelle states that any graph property definable in MSO is decidable in linear time on graphs of bounded treewidth. Algorithmic metatheorems like Courcelle's serve to generalize known positive results on various graph classes. We explore and extend three previously studied MSO extensions: global and local cardinality constraints (CardMSO and MSO-LCC) and optimizing the fair objective function (fairMSO).

First, we show how these extensions of MSO relate to each other in their expressive power. Furthermore, we highlight a certain "linearity" of some of the newly introduced extensions which turns out to play an important role. Second, we provide parameterized algorithms for the aforementioned structural parameters. On the side of neighborhood diversity, we show that combining the linear variants of local and global cardinality constraints is possible while keeping the linear (FPT) runtime but removing linearity of either makes this impossible assuming FPT $\neq$ W[1]. Moreover, we provide a polynomial time (XP) algorithm for the most powerful of studied extensions, i.e. the combination of global and local constraints. Furthermore, we show a polynomial time (XP) algorithm on graphs of bounded treewidth for the same extension. In addition, we propose a general procedure for deriving XP algorithms on graphs on bounded treewidth using Constraint Satisfaction Problems (CSPs). This shows an alternative approach to standard dynamic programming formulations.

Key words and phrases: MSO extensions, metatheorem, parameterized complexity, neighborhood diversity, treewidth.

Preliminary version appeared in the proceedings of the WG 2017 conference [29].

LOGICAL METHODS IN COMPUTER SCIENCE
DOI:10.23638/LMCS-15(4:12)2019

1. Introduction

It has been known since the '80s that various NP-hard problems are solvable in polynomial time by dynamic programming on trees and "tree-like" graphs. This was famously captured by Courcelle [10] in his theorem stating that any property definable in Monadic Second Order (MSO) logic is decidable in linear time on graphs of bounded treewidth. Subsequently, extensions to stronger logics and optimization versions were devised [3, 14] while still keeping linear runtime.

However, several interesting problems do not admit an MSO description and are unlikely to be solvable in linear time on graphs of bounded treewidth due to hardness results. In the language of parameterized complexity, Courcelle's theorem runs in *fixed-parameter tractable* (FPT) time, that is, in time $f(||\varphi||, \tau)n^{\mathcal{O}(1)}$, where n is the number of vertices of the input graph, τ its treewidth, φ is an MSO formula, and f is a computable function. On the other hand, unless FPT=W[1], the "hard" (specifically, W[1]-hard) problems have algorithms running at best in XP time $n^{g(||\varphi||,\tau)}$, for some computable function $g \in \omega(1)$. This led to an examination of extensions of MSO which allow greater expressive power.

Another research direction was to improve the computational complexity of Courcelle's theorem, since the function f grows as an exponential tower in the number of quantifier alternations of the MSO formula. However, Frick and Grohe [21] proved that this is unavoidable unless P = NP which raises a question: is there a (simpler) graph class where MSO model checking can be done in single-exponential (i.e., $2^{k^{\mathcal{O}(1)}}$) time? This was answered in the affirmative by Lampis [34], who introduced graphs of bounded neighborhood diversity. The classes of bounded treewidth and bounded neighborhood diversity are incomparable: for example, paths have unbounded neighborhood diversity but bounded treewidth, and vice versa for cliques. Bounded treewidth has become a standard parameter with many practical applications (cf. a survey [6]); bounded neighborhood diversity is of theoretical interest [1, 2, 7, 19, 22, 24, 38] because it can be viewed as representing the simplest of dense graphs.

Courcelle's theorem proliferated into many fields. Originating among automata theorists, it has since been reinterpreted in terms of finite model theory [37], database programming [25], game theory [28] and linear programming [32].

1.1. Related work.

For a recent survey of algorithmic metatheorems see e.g. Grohe et al. [26].

Objective functions. A linear optimization version of Courcelle's theorem was given by Arnborg, Lagergren and Seese [3]. An extension to further objectives was given by Courcelle and Mosbah [14]. Kolman, Lidický and Sereni [33] introduce MSO with a fair objective function (fairMSO) which, for a given MSO formula $\varphi(F)$ with a free edge set variable F, minimizes the maximum degree in the subgraph given by F, and present an XP algorithm. This is justified by the problem being W[1]-hard, as was later shown by Masařík and Toufar [38], who additionally gave an FPT algorithm on graphs of bounded neighborhood diversity for MSO_1 and an FPT algorithm on graphs of bounded vertex cover for MSO_2. Those results were extended to graphs of bounded twin cover by Knop, Masařík and Toufar [30].

Extended logics. Along with MSO, Courcelle also considered *counting* MSO (cMSO) where predicates of the form "$|X| \equiv p \mod q$" are allowed, with the largest modulus q constant. Szeider [41] introduced MSO with *local cardinality constraints* (MSO-LCC)

and gave an XP algorithm deciding it on graphs of bounded treewidth. MSO-LCC can express various problems, such as General Factor, Equitable r-Coloring or Minimum Maximum Outdegree, which are known to be W[1]-hard on graphs of bounded treewidth. Ganian and Obdržálek [23] study CardMSO, which is incomparable with MSO-LCC in its expressive power; they give an FPT algorithm on graphs of bounded neighborhood diversity.

1.2. **Our contribution.** The contribution of the paper is twofold. First, we survey and enrich the so far studied extensions of MSO logic – fairMSO, CardMSO, and MSO-LCC. We do this in Section 2.1. Second, we study the parameterized complexity of the associated model checking problem for various combinations of these MSO extensions. We completely settle the parameterized complexity landscape for the model checking problems with respect to the parameters treewidth and neighborhood diversity; for an overview of the complexity landscape refer to Figures 1 and 1. We postpone formal definitions of logic extensions and corresponding model checking to Subsection 2.1.

While both MSO-LCC and CardMSO express certain *cardinality* constraints, the constraints of CardMSO are inherently *global* and *linear*, yet the constraints of MSO-LCC are *local* and *non-linear*. This leads us to introduce two more fragments and to rename the aforementioned ones: CardMSO becomes $\mathsf{MSO}^{\mathsf{G}}_{\mathsf{lin}}$, MSO-LCC becomes $\mathsf{MSO}^{\mathsf{L}}$ and we additionally have $\mathsf{MSO}^{\mathsf{G}}$, $\mathsf{MSO}^{\mathsf{L}}_{\mathsf{lin}}$, $\mathsf{MSO}^{\mathsf{GL}}_{\mathsf{lin}}$ and $\mathsf{MSO}^{\mathsf{GL}}$. By this we give a complete landscape for all possible combinations of types of constraints: global/local and linear/non-linear.

In the following, we do not differentiate between the logics MSO_1 (allowing quantification over vertex sets) and MSO_2 (additionally allowing quantification over edge sets); see a detailed explanation in Subsection 2.1. For now, it suffices to say that our positive result for graphs of bounded treewidth holds for the appropriate extension of MSO_2, while all remaining results (positive for graphs of bounded neighborhood diversity and negative for graphs of bounded vertex cover number) hold for the appropriate extensions of MSO_1.

For graphs of bounded treewidth we give an XP algorithm for the logic $\mathsf{MSO}^{\mathsf{GL}}$, which is a composition of $\mathsf{MSO}^{\mathsf{G}}$ and $\mathsf{MSO}^{\mathsf{L}}$ and thus represents the most expressive fragment under our consideration.

Theorem 1.1. $\mathsf{MSO}^{\mathsf{GL}}$ *model checking is XP parameterized by* $\mathrm{tw}(G)$ *and* $||\varphi||$.

This result is also significant in its proof technique. We connect a recent result of Kolman, Koutecký and Tiwary [32] about the polytope of satisfying assignments with an old result of Freuder [20] about the solvability of the constraint satisfaction problem (CSP) of bounded treewidth. This allows us to formulate the proof of Theorem 1.1 essentially as providing a CSP instance with certain properties. We also briefly discuss which problems can be modeled using $\mathsf{MSO}^{\mathsf{GL}}$, deriving the following as a consequence:

Corollary 1.2. *Let G be a graph of treewidth τ and with $n = |V(G)|$.*

The following problems have algorithms with runtime $n^{f(\tau)}$: General Factor, Minimum Maximum Outdegree, Capacitated Dominating Set, Capacitated Vertex Cover, Vector Dominating Set, Generalized Domination.

The following problems have algorithms with runtime $n^{f(\tau+k)}$:

- Equitable k-Coloring, Equitable connected k-Partition, k-Balanced Partitioning,
- Graph Motif, *where k is the number of colors.*

Theorem 1.1 is complemented from the negative side by the following hardness result.

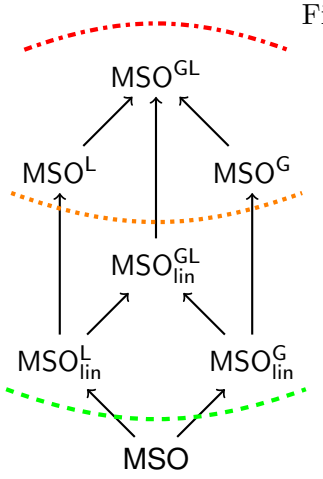

Figure 1: **MSO extensions.** A partial order of MSO extensions considered. An arrow denotes generalizations; e.g., $\mathsf{MSO}^{\mathsf{L}}_{\mathsf{lin}}$ generalizes MSO and is generalized by $\mathsf{MSO}^{\mathsf{GL}}_{\mathsf{lin}}$. Green (dashed) line separates logics whose model checking is FPT parameterized by $\mathrm{tw}(G)$ (Courcelle [10]) from those whose model checking is W[1]-hard (both $\mathsf{MSO}^{\mathsf{L}}_{\mathsf{lin}}$ and $\mathsf{MSO}^{\mathsf{G}}_{\mathsf{lin}}$ capture the W[1]-hard Equitable r-Coloring problem). Orange (dotted) line separates logics whose model checking is FPT parameterized by $\mathrm{nd}(G)$ (Theorem 1.4) from those whose model checking is W[1]-hard (Theorems 1.3 and 1.6). The model checking of all logics below the red (dashed-dotted) line is XP parameterized by both $\mathrm{tw}(G)$ (Theorem 1.1) and $\mathrm{nd}(G)$ (Theorem 1.5).

Theorem 1.3. $\mathsf{MSO}^{\mathsf{G}}$ *model checking is* ***W[1]-hard*** *parameterized by* $\mathrm{vc}(G)$ *and* $||\varphi||$*, where* $\mathrm{vc}(G)$ *is the vertex cover number of the input graph* G*.*

For graphs of bounded neighborhood diversity we give two positive results; the logic $\mathsf{MSO}^{\mathsf{GL}}_{\mathsf{lin}}$ is a composition of $\mathsf{MSO}^{\mathsf{L}}_{\mathsf{lin}}$ and $\mathsf{MSO}^{\mathsf{G}}_{\mathsf{lin}}$.

Theorem 1.4. $\mathsf{MSO}^{\mathsf{GL}}_{\mathsf{lin}}$ *model checking is* ***FPT*** *parameterized by* $\mathrm{nd}(G)$ *and* $||\varphi||$*.*

Theorem 1.5. $\mathsf{MSO}^{\mathsf{GL}}$ *model checking is* ***XP*** *parameterized by* $\mathrm{nd}(G)$ *and* $||\varphi||$*.*

We complement the above results with another hardness result.

Theorem 1.6. $\mathsf{MSO}^{\mathsf{L}}$ *model checking is* ***W[1]-hard*** *parameterized by* $\mathrm{vc}\,(G)$ *and* $||\varphi||$*.*

Interestingly, our finding about hardness being caused by nonlinearity carries over to a generalization of the Set Cover problem.

Multidemand Set Multicover
Input: Universe $U = [k]$, set of multidemands $D_1, \ldots, D_k \subseteq [n]$, a covering system $\mathcal{F} = \{F_1, \ldots, F_f\} \subseteq 2^U$, an integer $r \in \mathbb{N}$
Task: Find integer multiplicities $m_1 + \cdots + m_f = r$ such that for all $i \in [k]$, $(\sum_{j:u_i \in F_j} m_j) \in D_i$.

We show that the (non)linearity of the multidemands is crucial. On one hand, we have the following hardness.

Theorem 1.7. Multidemand Set Multicover *is* ***W[1]-hard*** *parameterized by* k*, already when* $|\mathcal{F}| = k$*.*

On the other hand, consider the Weighted Set Multicover problem. It is a weighted variant of Multidemand Set Multicover where each multidemand D_i is the interval $[0, \delta_i]$ for some $\delta_i \in \mathbb{N}$. Recently, Bredereck et al. [8] showed that this variant is fixed-parameter tractable when parameterized by k, which has applications in computational social choice and elsewhere. Moreover, it is not difficult to see that even the more general case when each D_i is a discrete interval is in FPT, by a small modification of the approach of Bredereck et al.

nd, vc	MSO	fairMSO	$\mathsf{MSO}^{\mathsf{L}}_{\mathsf{lin}}$	$\mathsf{MSO}^{\mathsf{L}}$
MSO	FPT [34]	FPT [38]		W[1]-h, Thm 1.6
$\mathsf{MSO}^{\mathsf{G}}_{\mathsf{lin}}$	FPT [23]		FPT, Thm 1.4	
$\mathsf{MSO}^{\mathsf{G}}$	W[1]-h, Thm 1.3			XP, Thm 1.5
tw	MSO	fairMSO	$\mathsf{MSO}^{\mathsf{L}}_{\mathsf{lin}}$	$\mathsf{MSO}^{\mathsf{L}}$
MSO	FPT [10]	W[1]-hard [38]		XP [41]
$\mathsf{MSO}^{\mathsf{G}}_{\mathsf{lin}}$	W[1]-hard [23]			
$\mathsf{MSO}^{\mathsf{G}}$				XP, Thm 1.1

Table 1: Complexity of various logic fragments generalizing MSO parameterized by $||\varphi||$ and in addition by vertex cover (vc), neighborhood diversity (nd), or treewidth (tw). Positive results (FPT, XP) spread to the left and up. W[1]-hardness spreads to the right and down. Green background (lighter gray in b-w print) stands for FPT fragments, while orange (darker gray) stands for W[1]-hard. A cell represents a union of studied fragments, i.e., the cell indexed by $\mathsf{MSO}^{\mathsf{G}}$ and $\mathsf{MSO}^{\mathsf{L}}$ corresponds to the $\mathsf{MSO}^{\mathsf{GL}}$ fragment.

2. Preliminaries

For two integers a, b we define a set $[a, b] = \{x \in \mathbb{Z} \mid a \leq x \leq b\}$; we use $[b]$ to denote the set $[1, b]$. We write vectors in bold font, i.e., $\mathbf{x} \in \mathbb{R}^n$, and denote by x_i, $i \in [n]$, its i-th coordinate.

For a vertex $v \in V$ of a graph $G = (V, E)$, we denote by $N_G(v)$ the set of neighbors of v in G, that is, $N_G(v) = \{u \in V \mid \{u, v\} \in E\}$; the subscript G is omitted when clear from the context. For a rooted tree T, $N^{\downarrow}_T(v)$ denotes the *down-neighborhood* of v, i.e., the set of children of v. For a graph $G = (V, E)$ a set $U \subseteq V$ is a *vertex cover* of G if for every edge $e \in E$ it holds that $e \cap U \neq \emptyset$. The size of a minimum vertex cover of graph G is denoted by $\mathrm{vc}(G)$. For more graph theory notation cf. the book of Matoušek and Nešetřil [39].

2.1. MSO and its Extensions.

Let us shortly introduce MSO over graphs. In *first-order logic* (FO) we have variables for the elements $(x, y, \ldots)$, equality for variables, quantifiers $\forall, \exists$ ranging over elements, and the standard Boolean connectives $\neg, \wedge, \vee, \implies$. The *monadic second order logic* (MSO) extends first order logic using so called monadic variables. Graph MSO has the binary relational symbol E encoding edges, and traditionally comes in two flavors, MSO_1 and MSO_2, differing by the objects we are allowed to quantify over: in MSO_1 these are the vertices and vertex sets, while in MSO_2 we can additionally quantify over edges and edge sets. For example, the 3-colorability property can be expressed in MSO_1 as follows:

$$\exists X_1, X_2, X_3 \quad [\forall x\,(x \in X_1 \vee x \in X_2 \vee x \in X_3) \wedge \bigwedge\nolimits_{i=1,2,3} \forall x, y\,(x \notin X_i \vee y \notin X_i \vee \{x, y\} \notin E)\,]$$

For a formula φ, we denote by $||\varphi||$ the *size* of φ, that is, the number of symbols in a chosen encoding of φ.

Regarding MSO_1 *and* MSO_2. We use standard notation for relational structures (cf. Libkin [37]): a *vocabulary* σ is a collection of constant and predicate symbols, and a σ*-structure* is a tuple $\mathcal{A} = \big(A, \{c_i\}, \{P_i\}\big)$, where the set A is the *universe*, $\{c_i\}$ are *constant symbols* ($c_i \in A$) and $\{P_i\}$ are finitely many *relations* (or *predicates*), each of arity r_i ($P_i \subseteq A^{r_i}$). (More precisely, c_i and P_i are the realizations of the symbols in the vocabulary σ, but we identify a symbol and its realization to avoid notational overload.) Then, a graph $G = (V, E)$ is typically modeled as a σ_1-structure $\big(V, \emptyset, \{E\}\big)$ with σ_1 containing one binary relation E. We can view G as a σ_2-structure $\mathcal{A} = \big(V \cup E, \emptyset, \{I, L_V, L_E\}\big)$, where σ_2 contains a binary relation I representing incidence in G and two unary predicates L_V, L_E distinguishing vertices and edges, respectively, with realizations defined as $I = \{\{v, e\} \mid v \in e, e \in E\}$, $L_V = V$ and $L_E = E$. Given a vocabulary σ, we define the logic $\mathsf{MSO}[\sigma]$ inductively as usual by letting its terms be the variables and constant symbols, its atomic formulae (atoms) be either $t_1 = t_2$ for two terms t_1, t_2, or $P(t_1, \ldots, t_k)$ for any k-ary predicate P and terms $t_1, \ldots, t_k$, and its formulae be quantified boolean combinations of atoms. Thus, $\mathsf{MSO}[\sigma_1]$ is the logic with a binary predicate E, and $\mathsf{MSO}[\sigma_2]$ is the logic with two unary predicates L_V, L_E and one binary predicate I. For short, we denote $\mathsf{MSO}_1 = \mathsf{MSO}[\sigma_1]$ and $\mathsf{MSO}_2 = \mathsf{MSO}[\sigma_2]$. The treewidth of a relational structure $\mathcal{A}$ is the treewidth of its *Gaifman graph* $G(\mathcal{A}) = (V^{\mathcal{A}}, E^{\mathcal{A}})$ where $V^{\mathcal{A}} = A$ and $E^{\mathcal{A}} = \{\{u, v\} \mid \exists P_i \exists \mathbf{r} \in P_i : u, v \in \mathbf{r}\}$. It is known that the treewidth of a graph does not change when viewed as a σ_1-structure, and increases by at most 1 when viewed as a σ_2-structure [31]. Even though MSO_2 is strictly stronger than MSO_1 (hamiltonicity is expressible in MSO_2 but not in MSO_1 [37]), on graphs with bounded treewidth their power is equal [11, 12].

For this reason, when considering graphs of bounded treewidth, we will focus on MSO_2, and all the extended logics such as $\mathsf{MSO}^{\mathsf{GL}}_{\mathsf{lin}}$ etc. will be extensions of MSO_2. On the other hand, on graphs of bounded neighborhood diversity, MSO_2 is strictly more powerful than MSO_1. However, because model checking of an MSO_2 formula already over a *clique* is not even in XP unless $\mathsf{E} = \mathsf{NE}$ [13, 35], when dealing with graphs of bounded neighborhood diversity we will always refer to MSO_1 and $\mathsf{MSO}^{\mathsf{GL}}_{\mathsf{lin}}$ will stand for an extension of MSO_1.

We consider two orthogonal ways to extend MSO logic. In what follows φ is a formula with ℓ free set-variables.

Global cardinality constraints. We introduce a new type of atomic formulae called *global cardinality constraints* (*global constraints* for short). An MSO formula with c global cardinality constraints contains ℓ-ary predicates $R_1, \ldots, R_c$ where each predicate takes as argument only the free variables of φ. The input to the model checking problem is a graph $G = (V, E)$ on n vertices and an implicit representation (see below) of a tuple $(R_1^G, \ldots, R_c^G)$, where $R_i^G \subseteq [n]^\ell$.

To define the semantics of the extension, it is enough to define the truth of the newly introduced atomic formulae. A formula $R_i(X_1, \ldots, X_\ell)$ is true under an assignment $\mu\colon \{X_1, \ldots, X_\ell\} \to 2^V$ if and only if $(|\mu(X_1)|, \ldots, |\mu(X_\ell)|) \in R_i^G$. We allow a relation R_i^G to be represented either as a linear constraint $a_1|X_1| + \cdots + a_m|X_m| \leq b$, where $(a_1, \ldots, a_m, b) \in \mathbb{R}^{m+1}$, or, much more generally, by an arbitrary algorithm $\mathcal{A}(R_i^G)$ such that $\mathcal{A}(R_i^G)$ decides whether $(|X_1|, \ldots, |X_\ell|) \in R_i^G$ in time $n^{\mathcal{O}(1)}$ for any tuple in $[0, n]^\ell$.

For example, suppose we want to satisfy a formula $\varphi(X_1, X_2)$ with two sets for which $|X_1| \geq |X_2|^2$ holds. Then, we solve the $\mathsf{MSO}^{\mathsf{G}}$ model checking problem with a formula $\varphi' := \varphi \wedge [|X_1| \geq |X_2|^2]$, that is, we write the relation as a part of the formula, as this is a more convenient way to think of the problem. However, formally the relation is a part of the input, represented by the obvious algorithm computing $|X_2|^2$, comparing it with $|X_1|$, and returning the result.

Local cardinality constraints. Local cardinality constraints are additional cardinality requirements such that every variable assignment has to satisfy the cardinality constraint for every vertex and for every free variable. Specifically, we want to control the size of $\mu(X_i) \cap N(v)$ for every v; we define a shorthand $S(v) = S \cap N(v)$ for a subset $S \subseteq V$ and vertex v. *Local cardinality constraints* for a graph $G = (V, E)$ on n vertices and a formula φ with ℓ free variables are mappings $\alpha_1, \ldots, \alpha_\ell$, where each α_i is a mapping from V to $2^{[n]}$.

We say that an assignment μ *obeys local cardinality constraints* $\alpha_1, \ldots, \alpha_\ell$ if for every $i \in [\ell]$ and every $v \in V$ it holds that $|\mu(X_i)(v)| \in \alpha_i(v)$.

The logic that incorporates both of these extensions is denoted $\mathsf{MSO}^{\mathsf{GL}}$. Let φ be an $\mathsf{MSO}^{\mathsf{GL}}$ formula with c global cardinality constraints. Then the $\mathsf{MSO}^{\mathsf{GL}}$ model checking problem has input:

- graph $G = (V, E)$ on n vertices,
- relations $R_1^G, \ldots, R_c^G \subseteq [n]^\ell$, and,
- mappings $\alpha_1, \ldots, \alpha_\ell$.

The task is to find an assignment μ that obeys local cardinality constraints and such that φ is true under μ by the semantics defined above.

The $\mathsf{MSO}^{\mathsf{GL}}$ logic is very powerful and, as we later show, it does not admit an FPT model checking algorithm neither for the parameterization by neighborhood diversity, nor for the parameterization by treewidth. It is therefore relevant to consider the following weakenings of the $\mathsf{MSO}^{\mathsf{GL}}$ logic:

$\mathsf{MSO}^{\mathsf{G}}$ **:** Only global cardinality constraints are allowed.

$\mathsf{MSO}^{\mathsf{L}}$(originally MSO-LCC [41])**:** Only local cardinality constraints are allowed.

$\mathsf{MSO}^{\mathsf{G}}_{\mathsf{lin}}$(originally CardMSO [22])**:** The cardinality constraints can only be linear; that is, we allow constraints in the form $[e_1 \geq e_2]$, where e_i is a linear expression over $|X_1|, \ldots, |X_\ell|$.

$\mathsf{MSO}^{\mathsf{L}}_{\mathsf{lin}}$ **:** Only local cardinality constraints are allowed; furthermore every local cardinality constraint α_i must be of the form $\alpha_i(v) = [l_i^v, u_i^v]$, (i.e., an interval) where $l_i^v, u_i^v \in [n]$. Those constraints are referred to as *linear local cardinality constraints.*

fairMSO **:** Further restriction of $\mathsf{MSO}^{\mathsf{L}}_{\mathsf{lin}}$; now we only allow $\alpha_i(v) = [u_i^v]$.

$\mathsf{MSO}^{\mathsf{GL}}_{\mathsf{lin}}$ **:** A combination of $\mathsf{MSO}^{\mathsf{L}}_{\mathsf{lin}}$ and $\mathsf{MSO}^{\mathsf{G}}_{\mathsf{lin}}$; both local and global constraints are allowed, but only in their linear variants.

The model checking problem for the considered fragments is defined in a natural way analogously to $\mathsf{MSO}^{\mathsf{GL}}$ model checking.

Pre-evaluations. Many techniques used for designing MSO model checking algorithms fail when applied to MSO extensions. A common workaround is first transforming the given $\mathsf{MSO}^{\mathsf{GL}}$ formula into an MSO formula by fixing the truth values of all global constraints to either `true` or `false`. Once we determine which variable assignments satisfy the transformed MSO formula, we can by other means (e.g. integer linear programming or constraint

satisfaction) ensure that they obey the constraints imposed by fixing the values to `true` or `false`. This approach was first used for CardMSO by Ganian and Obdržálek [23]. We formally describe this technique as *pre-evaluations*:

Definition 2.1 (Pre-evaluation)**.** *Let φ be an $\mathsf{MSO}^{\mathsf{GL}}$ formula. Denote by $C(\varphi)$ the list of all global constraints. A mapping $\beta\colon C(\varphi) \to \{\texttt{true}, \texttt{false}\}$ is called a* pre-evaluation function on φ. *The* MSO *formula obtained by replacing each global constraint $c_i \in C(\varphi)$ by $\beta(c_i)$ is denoted by $\beta(\varphi)$ and is referred to as a* pre-evaluation of φ.

Definition 2.2 (Assignment and Pre-evaluation Compliance)**.** *A variable assignment μ of an $\mathsf{MSO}^{\mathsf{GL}}$ formula φ* complies *with a pre-evaluation function β if every global constraint $c_i \in C(\varphi)$ evaluates to $\beta(c_i)$ under the assignment μ.*

2.2. Treewidth and Neighborhood Diversity.

Treewidth. For notions related to the treewidth of a graph and nice tree decompositions, in most cases we stick to the standard terminology as given by Kloks [27]; the only deviation is in the leaf nodes of the nice tree decomposition where we assume that the bags are empty.

Definition 2.3 (Tree decomposition, Treewidth)**.** *A* tree decomposition *of a graph $G = (V, E)$ is a pair $(T, \mathcal{B})$, where T is a tree and $\mathcal{B}$ is a mapping $\mathcal{B} : V(T) \to 2^V$ satisfying*

- *for any $\{u, v\} \in E$, there exists $a \in V(T)$ such that $u, v \in B(a)$,*
- *if $v \in B(a)$ and $v \in B(b)$, then $v \in B(c)$ for all c on the a-b path in T.*

We call the vertices of the tree nodes *and the sets $B(a)$ we call* bags.

The treewidth *$tw((T, \mathcal{B}))$* of a tree decomposition *$(T, \mathcal{B})$ is the size of the largest bag of $(T, \mathcal{B})$ minus one. A graph G has* treewidth *τ ($\mathrm{tw}(G) = \tau$) if it has a tree decomposition of treewidth τ.*

Observe that for every graph G we have $\mathrm{tw}(G) \leq \mathrm{vc}(G)$, since it admits a tree decomposition with T being a path $P_{|V(G)|-\mathrm{vc}(G)}$. In this decomposition we have (in any order) bags of the form $U \cup \{v\}$ for each vertex $v \in V \setminus U$, where U is a vertex cover of G with $|U| = \mathrm{vc}(G)$. It is easy to verify that the constructed decomposition is a tree decomposition of G. Thus, the class of graphs of bounded treewidth is more general than the class of graphs of bounded vertex cover.

Definition 2.4 (Nice tree decomposition)**.** *A* nice tree decomposition *is a tree decomposition with T rooted and binary, where the root is denoted r and each node is one of the following types:*

- Leaf node*: a leaf a of T with $B(a) = \emptyset$.*
- Introduce node*: an internal node a of T with one child b for which $B(a) = B(b) \cup \{v\}$ for some $v \in B(a)$; for short we write $a = b * (v)$*
- Forget node*: an internal node a of T with one child b for which $B(a) = B(b) \setminus \{v\}$ for some $v \in B(b)$; for short $a = b \dagger (v)$*
- Join node*: an internal node a with two children b and c with $B(a) = B(b) = B(c)$; for short $a = \Lambda(b, c)$.*

For a vertex $v \in V$, we denote by $top(v)$ the topmost node of a nice tree decomposition that contains v in its bag. For any graph G on n vertices, a nice tree decomposition of G with width $\mathrm{tw}(G)$ and at most $8n$ nodes can be computed in time $\mathcal{O}(n)$ [5, 27]. (This is done by first using Bodlaender's algorithm [5] to compute an optimal tree decomposition with at most n nodes in linear time, and then transforming it into a nice decomposition using an algorithm of Kloks [27] while keeping the number of nodes bounded by $8n$.)

Given a graph $G = (V, E)$ and a subset of vertices $V' = \{v_1, \ldots, v_d\} \subseteq V$, we denote by $G[V']$ the subgraph of G induced by V'. Given a tree decomposition $(T, \mathcal{B})$ and a node $a \in V(T)$, we denote by T_a the subtree of T rooted in a, and by G_a the subgraph of G induced by all vertices in bags of T_a, that is, $G_a = G[\bigcup_{b \in V(T_a)} B(b)]$.

Neighborhood diversity. We say that two (distinct) vertices u, v are of the same *neighborhood type* if they share their respective neighborhoods, that is when $N(u) \setminus \{v\} = N(v) \setminus \{u\}$. Let $G = (V, E)$ be a graph. We call a partition of vertices $\mathcal{T} = \{T_1, \ldots, T_\nu\}$ a *neighborhood decomposition* if, for every $i \in [\nu]$, all vertices of T_i are of one neighborhood type.

Definition 2.5 (Neighborhood diversity [34])**.** *A graph $G = (V, E)$ has* neighborhood diversity ν *($\mathrm{nd}(G) = \nu$) if its unique minimal neighborhood decomposition is of size ν. Moreover, this decomposition can be computed in linear time.*

We call the sets $T_1, \ldots, T_\nu$ *types*. Note that every type induces either a clique or an independent set in G and two types are either joined by a complete bipartite graph or no edge between vertices of the two types is present in G. We call a type which is a clique a *clique type* and a type which is an independent set an *independent type*. Thus, we introduce the notion of a *type graph* $T_{\mathcal{T}}(G)$. The vertices of $T_{\mathcal{T}}(G)$ are the types $T_1, \ldots, T_\nu$ and two types T_i, T_j are joined by an edge if T_i and T_j are joined by a complete bipartite graph in G. If the decomposition $\mathcal{T}$ is clear from the context, we omit the subscript $\mathcal{T}$.

Observe that for every graph $G = (V, E)$ we have $\mathrm{nd}(G) \leq 2^{\mathrm{vc}(G)} + \mathrm{vc}$. Indeed, we can construct a decomposition $\mathcal{T}$ witnessing this as follows. Let U be a vertex cover of G with $|U| = \mathrm{vc}(G)$. We put singleton $\{u\}$ to $\mathcal{T}$ for every $u \in U$ and then we add sets $\{v \in V \setminus U \mid N_G(v) = X\}$ to $\mathcal{T}$ for every $X \subseteq U$. It is easy to verify that $\mathcal{T}$ is a neighborhood decomposition of G. Thus, neighborhood diversity is a more general structural graph parameter than vertex cover, since the class of cliques has neighborhood diversity 1 while unbounded size vertex cover.

2.3. **Parameterized Complexity.** Let Σ be a finite alphabet. A *parameterized language* is a language $P \subseteq \Sigma^* \times \mathbb{N}$. The associated parameterized problem is then to decide whether the input (x, k) belongs to P or not; the value k is the *parameter*. A parameterized language P belongs to the class FPT (is fixed-parameter tractable) if there is an algorithm deciding P in $f(k) \cdot \mathrm{poly}(|x|)$ time, where $f \colon \mathbb{N} \to \mathbb{N}$ is a computable function. A parameterized language P belongs to the class XP if there is an algorithm deciding P in $|x|^{f(k)}$ time for a computable function $f \colon \mathbb{N} \to \mathbb{N}$. Clearly, FPT is a subclass of XP.

Let P and Q be two parameterized languages. A *parameterized reduction* from P to Q is an algorithm that on input (x, k) computes an instance (y, ℓ) in $f(k) \cdot \mathrm{poly}(|x|)$ time such that

- $(x, k) \in P$ if and only if $(y, \ell) \in Q$ and
- $\ell \leq g(k)$ for some computable function $g \colon \mathbb{N} \to \mathbb{N}$.

Roughly speaking, a parameterized problem Q is W[1]-hard if there exists a parameterized from Clique parameterized by the solution size to Q. Under a common assumption FPT $\neq$ W[1] if a parameterized problem is W[1]-hard, then it is unlikely to be in FPT. For further detail please refer to e.g. [15].

3. Graphs of Bounded Neighborhood Diversity

For graphs of bounded neighborhood diversity we prove two negative results (Theorems 1.3 and 1.6) and two positive results (Theorems 1.4 and 1.5).

3.1. Theorems 1.3 and 1.6: W[1]-hardness of $\mathsf{MSO}^{\mathsf{L}}$ and $\mathsf{MSO}^{\mathsf{G}}$.

We begin with a definition of an auxiliary problem:

Local Cardinality Constrained Subset (LCC Subset)
Input: Graph $G = (V, E)$ with $|V| = n$ and a function $f\colon V \to 2^{[0,n-1]}$.
Task: Find a set $U \subseteq V$ such that $|U(v)| \in f(v)$ for each vertex $v \in V$.

Obviously LCC Subset is equivalent to $\mathsf{MSO}^{\mathsf{L}}$ with an empty formula φ. We call an LCC Subset instance *uniform* if, on G with neighborhood decomposition $\mathcal{T}$, the demand function f can be written as $f\colon \mathcal{T} \to 2^{[0,n-1]}$, that is, the vertices of the same type have the same demand set. We show that already uniform LCC Subset is W[1]-hard by a reduction from the W[1]-hard k-Multicolored Clique problem [15].

k-Multicolored Clique *Parameter: k*
Input: k-partite graph $G = (V_1 \dot\cup \cdots \dot\cup V_k, E)$, where V_a is an independent set for every $a \in [k]$.
Task: Find a clique of size k.

We refer to a set V_a as to a *colorclass* of G. We may assume that all of the colorclasses are of the same size and that the number of edges between any two colorclasses is the same. This follows from the reduction from k-Clique to k-Multicolored Clique, where we take k copies of the vertex set of the original graph and connect two vertices in the new instance if and only if they are in different copies and their pre-images are adjacent (i.e., the degree of every vertex to any other colorclass is the same as in the original graph).

Our proof is actually a simplified proof of W[1]-hardness for the Target Set Selection problem [16] for the parameter neighborhood diversity.

Theorem 3.1. LCC Subset *is W[1]-hard parameterized by the vertex cover number already in the case when $f(v) = \{0\}$ for all v not belonging to the vertex cover.*

Proof. Let $G = (V_1 \cup \cdots \cup V_k, E)$ be the instance graph for k-Multicolored Clique. We naturally split the set of edges E into sets $E_{\{a,b\}}$ by which we denote the edges between colorclasses V_a and V_b. We denote n the (common) size of colorclasses in G, and we denote m the number of edges between any two colorclasses. Fix $N > n$, say $N = n^2$, and distinct $a, b \in [k]$.

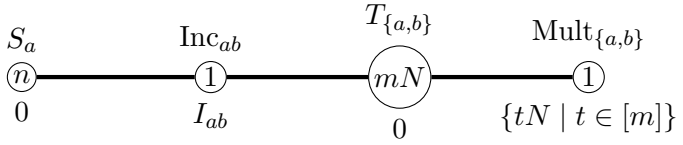

Figure 2: An overview of the decomposition of a gadget used in the proof of Theorem 3.1. Numbers inside nodes denote the number of vertices in the independent set represented by the node. Below each node a description of the respective set of admissible numbers is shown.

Description of the reduction. We numerate vertices in each color class V_a for $a \in [k]$ using numbers in $[n]$, that is, we fix a bijection $\mu_a \colon V_a \to [n]$ for each $a \in [k]$. We also numerate the edges between color classes a and b by numbers in $[m]$. Let $\varepsilon_{\{a,b\}} \colon E_{\{a,b\}} \to [m]$ be the numeration function for distinct $a, b \in [k]$. We set

$$I_{ab} = \left\{ \mu_a(v) + N \cdot \varepsilon_{\{a,b\}}(e) \mid v \in e, e \in E_{\{a,b\}} \right\}.$$

We build the graph H using the following groups of vertices (refer to Figure 2):

- an independent set S_a of size n for each color class V_a and set $f(v) = \{0\}$ for every $v \in S_a$,
- an independent set $T_{\{a,b\}}$ of size mN for each edge set $E_{\{a,b\}}$, with $f(v) = \{0\}$ for every $v \in T_{\{a,b\}}$,
- a single vertex $\mathrm{Mult}_{\{a,b\}}$ with $f(\mathrm{Mult}_{\{a,b\}}) = \{tN \mid t \in [m]\}$ for each $\{a, b\} \in \binom{[k]}{2}$, and
- a single vertex Inc_{ab} with $f(\mathrm{Inc}_{ab}) = I_{ab}$ for each $a, b \in [k]$ with $a \neq b$.

Finally, we add all possible edges between S_a and Inc_{ab}, between Inc_{ab} and $T_{\{a,b\}}$, and between $T_{\{a,b\}}$ and $\mathrm{Mult}_{\{a,b\}}$, thus forming complete bipartite subgraphs between the respective sets of vertices. It is straightforward to check that the $\binom{k}{2}$ vertices $\mathrm{Mult}_{\{a,b\}}$ together with $k(k-1)$ vertices Inc_{ab} form a vertex cover of H. It follows that $\mathrm{vc}(H) = \binom{k}{2} + k(k-1)$. For an overview of the reduction please refer to Figure 2.

Correctness of the reduction. Suppose there is a clique of size k in G with vertex set $\{v_1, \ldots, v_k\}$. We assume that $v_i \in V_i$ for all $i \in [k]$. We select $\mu_a(v_a)$ vertices in the set S_a and $N \cdot \varepsilon_{\{a,b\}}(\{v_a, v_b\})$ vertices in the set $T_{\{a,b\}}$ for all distinct $a, b \in [k]$. It is straightforward to check that this is a solution respecting the demands in H.

For the opposite direction suppose there is a solution U respecting demands in H. First note that none of vertices $\mathrm{Mult}_{\{a,b\}}$, Inc_{ab} is selected as their neighborhood demands are set to 0. Denote $s_a = |U \cap S_a|$ and $t_{\{a,b\}} = |T_{\{a,b\}} \cap U|$. Now observe that since the demand of vertex $\mathrm{Mult}_{\{a,b\}}$ is fulfilled, there are $t_{\{a,b\}} = tN$ vertices for some $t \in [m]$. Let e_{ab} denote the edge in $E_{\{a,b\}}$ with numeration $\varepsilon_{\{a,b\}}(e_{ab}) = t$ for each distinct $a, b \in [k]$. Let v_a denote the vertex in V_a with numeration $\mu_a(v_a) = s_a$ for every $a \in [k]$. We now want to prove that respecting demands of Inc_{ab} vertices implies that $\{v_a, v_b\} \in E_{\{a,b\}}$ for all $a, b \in [k]$, that is, the vertices $\{v_a \mid a \in [k]\}$ for a (multicolored) clique in G. Since the demand of vertex Inc_{ab} is fulfilled, it follows that the vertex v_a must be incident to the edge e_{ab}. Symmetrically, since the demand for the vertex Inc_{ba} is fulfilled, we get that the vertex v_b is incident to the edge e_{ab}. Combining all of these together we infer that $\{v_1, \ldots, v_k\}$ defined in this way form a clique in the graph G. This concludes the proof. □

Note that Theorem 1.6 follows easily from Theorem 3.1, since the LCC Subset problem is expressed by an empty $\mathsf{MSO}^{\mathsf{L}}_{\mathsf{lin}}$ formula. Furthermore, we get the following consequence of the presented proof.

Corollary 3.2. LCC Subset *is* **W[1]-hard** *parameterized by neighborhood diversity even if*
- *the instance of* LCC Subset *is uniform for the given decomposition and*
- *all of the types in the given decomposition are independent sets.*

Proof. Observe that neighborhood diversity of the graph resulting from the construction presented in the proof of Theorem 3.1 has neighborhood diversity at most $\binom{k}{2} + k(k-1) + \binom{k}{2} + k$. To see this note that we can introduce a type with one vertex for every vertex in the vertex cover of the constructed graph (recall there are $\binom{k}{2} + k(k-1)$ many of these). Furthermore, all of the vertices in (independent sets) Inc_{ab} (for distinct fixed $a, b \in [k]$) have the same neighborhood (in the vertex cover) and thus form a single type; which is an independent set and recall that for all such vertices v we have $f(v) = 0$. The same holds for vertices in $\mathrm{Mult}_{\{a,b\}}$ (again for fixed $a, b \in [k]$). This finishes the proof. □

As we mentioned in the introduction, our hardness result has consequences for the hardness of the Multidemmand Set Multicover problem. We now use Corollary 3.2 as the basis of our reduction.

Proof of Theorem 1.7. Given a uniform instance of LCC Subset on a graph G with $\mathrm{nd}(G) = \nu$ with every type being an independent set, let $U = [\nu]$, $\mathcal{F} = \{N(v) \mid \forall v \in T(G)\}$ and let $D_i = f(i)$. Now, if there exists an $r \in [n]$ such that $(U, \mathcal{F}, (D_1, \ldots, D_\nu), r)$ is a Yes instance of Multidemand Set Multicover, then the given LCC Subset instance is a Yes instance, and otherwise it is a No instance. Consequently, one can test all of the possible values of r in polynomial time and thus obtain the answer for the graph G. □

Having showed the hardness of $\mathsf{MSO}^{\mathsf{L}}$ parameterized by $\mathrm{nd}(G)$, let us now turn our attention to the proof of Theorem 1.3, i.e., hardness of $\mathsf{MSO}^{\mathsf{G}}$ parameterized by $\mathrm{nd}(G)$.

Proof of Theorem 1.3. Let $(G = (V, E), f, k)$ be an instance of the LCC Subset problem parameterized by the vertex cover number resulting from Theorem 3.1. Let $C \subseteq V$ be the vertex cover in G. Note that it follows from the proof of Theorem 3.1 that we may assume that the independent set $V \setminus C$ is divided into $\mathcal{O}(k)$ groups, where each group shares the neighborhood in C. Observe further that indeed the graph G is bipartite (i.e., the set C is also an independent set), in particular, the largest clique subgraph of G is of size 2.

By Theorem 3.1 we know that it is **W[1]-hard** to find a subset $X \subseteq V \setminus C$ such that $|X(v)| \in f(v)$ for all $v \in C$. Our goal now is to build an $\mathsf{MSO}^{\mathsf{G}}$ formula expressing exactly this.

First we take G and construct a graph G' by, for each $v \in C$, attaching a $K_{2+\eta(v)}$ to $N(v)$, where $\eta\colon C \to [k]$ is a bijective mapping. We will call the clique $K_{2+\eta(v)}$ a *marker* because it will allow us to recognize exactly the vertices of $N(v)$. Note that markers are the only cliques present in G' of size at least 3. Note further that by this we have added $\mathcal{O}(k)$ cliques of size $\mathcal{O}(k)$ and thus the resulting graph has vertex cover of size $\mathcal{O}(k^2)$.

Let us describe some auxiliary formulae which we then use to define the desired formula φ. We reserve X for the set that will represent the set X from the LCC Subset problem.

- Z **is a clique:**

$$\texttt{clique}(Z) := (\forall x, y \in Z)(x \neq y \implies xy \in E)$$

- u **and** v **are of the same neighborhood type:**

$$\texttt{same}(u, v) := (\forall w \in V)(w = u \vee w = v \vee (wu \in E \iff wv \in E))$$

- Z **is a type:**

$$\texttt{type}(Z) := (Z \neq \emptyset) \wedge (\forall u, v \in Z)(\texttt{same}(u, v)) \wedge (\forall u \in Z, v \notin Z)(\neg\texttt{same}(u, v))$$

- Z **is** $\eta(v)$**-th marker:**

$$\texttt{marker}_v(Z) := (|Z| = 2 + \eta(v)) \wedge \texttt{clique}(Z) \wedge \texttt{type}(Z)$$

- Z **is** $N(v)$**:**

$$\texttt{neigh}_v(Z) := \texttt{type}(Z) \wedge (\exists Q \subseteq V)(\texttt{marker}_v(Q) \wedge (\forall u \in Z, w \in Q)(uw \in E))$$

- Z **is exactly** X_v**:**

$$\texttt{sel-neigh}_v(Z, X) := (\exists Z_v)(\texttt{neigh}_v(Z_v) \wedge Z = Z_v \cap X)$$

Now $\varphi(X, (X_v)_{v \in C}) := \bigwedge_{v \in C}\big(\texttt{sel-neigh}_v(X_v, X) \wedge |X_v| \in f(v)\big)$. □

3.2. **Theorem 1.4: FPT algorithm for $\mathsf{MSO}^{\mathsf{GL}}_{\mathsf{lin}}$ on neighborhood diversity.** Essentially, we are modifying the algorithm of Ganian and Obdržálek [23] for $\mathsf{MSO}^{\mathsf{G}}_{\mathsf{lin}}$ model checking so that it can deal with the additional constraints introduced by $\mathsf{MSO}^{\mathsf{L}}_{\mathsf{lin}}$. We use integer linear programming (ILP). By the result of Lenstra [36] ILP can be solved in FPT-time parameterized by the number of integral variables.

3.2.1. *Signatures and Shapes.* Before we move on to proving Theorem 1.4 we first need to introduce some notation.

Definition 3.3. *Let φ be an $\mathsf{MSO}^{\mathsf{GL}}_{\mathsf{lin}}$ formula with free set variables $X_1, \ldots, X_\ell$, let $G = (V, E)$ be a graph with $\mathrm{nd}(G) = \nu$ and types $T_1, \ldots, T_\nu$, and let $\mu\colon \{X_1, \ldots, X_\ell\} \to 2^V$ be a variable assignment. The* signature of μ *is the mapping $S_\mu\colon [\nu] \times 2^{[\ell]} \to \mathbb{N}$ defined by*

$$S_\mu(j, I) = \left|\bigcap_{i \in I} \mu(X_i) \cap T_j\right|$$

for $j \in [\nu]$ and $I \subseteq [\ell]$.

Clearly, if we have two variable assignments μ and μ' with the same signature, then $G, \mu \models \varphi$ if and only if $G, \mu' \models \varphi$.

However, for MSO formulae and graphs of bounded neighborhood diversity, much more is true. Informally speaking, the formula cannot distinguish between two cardinalities if both of them are large. This is formally stated in the next lemma, which is a direct consequence of [34, Lemma 5]:

Lemma 3.4. *Let φ be an MSO formula with free set variables $X_1, \ldots, X_\ell$ that has q_S set quantifiers and q_e element quantifiers. Let G be a graph with $\mathrm{nd}(G) = \nu$ and let $t = 2^{q_S} \cdot q_e$. Suppose that μ and μ' are two variable assignments such that for every $I \subseteq [\ell]$, $j \in [\nu]$ we have either*

- $S_\mu(j, I) = S_{\mu'}(j, I)$, *or*
- *both* $S_\mu(j, I), S_{\mu'}(j, I) > t$.

Then $G, \mu \models \varphi$ *if and only if* $G, \mu' \models \varphi$.

The last lemma leads to the following definition.

Definition 3.5 (Shape)**.** *Let* φ*,* G*, and* t *be as before. A* shape *of a variable assignment* $\mu\colon \{X_1, \ldots, X_\ell\} \to 2^V$ *is the mapping* $sh_\mu\colon [\nu] \times 2^{[\ell]} \to [0, t] \cup \{\uparrow\}$ *defined by*

$$sh_\mu(j, I) = \begin{cases} S_\mu(j, I) & \text{if } S_\mu(j, I) \leq t \\ \uparrow & \text{if } S_\mu(j, I) > t \end{cases}.$$

Since t depends only on the formula φ, the total number of shapes can be bounded by some function of $||\varphi||$ and $\mathrm{nd}(G)$. Note that there are mappings from $[\nu] \times 2^{[\ell]}$ to $[0, t] \cup \{\uparrow\}$ that do not correspond to a shape of any variable assignment μ for a particular graph G. For example, if $sh(j, I) = \uparrow$ for some j and I but $|T_j| < t$, clearly there is no assignment of such a shape sh.

It is worth noting that Lemma 3.4 cannot be used directly, as the global linear constraints allow us to distinguish small differences in cardinalities, even if the cardinalities are large; consider for example the constraint $[|X_1| = |X_2| + 1]$. We use the approach outlined in Subsection 2.1, Pre-evaluations. This approach relies on Definitions 2.1 and 2.2. We simply guess all possible outcomes of the (global) cardinality constraints (the number of such outcomes is clearly bounded by $2^{||\varphi||}$) and later ensure that our assignment obeys those constraints by an Integer Linear Program.

Definition 3.6. *A shape* sh *is* admissible *with respect to a pre-evaluation* β *if for any variable assignment* μ *of the shape* sh *we have* $G, \mu \models \beta(\varphi)$.

3.2.2. *Unifying Local Linear Constraints.* Here we show how to change the local linear constraints and the neighborhood diversity decomposition in a such way that

- the new instance is equivalent to the former one,
- the size of the new decomposition is bounded in terms of $\mathrm{nd}(G)$ and $\|\varphi\|$, and
- vertices of the same type in the newly obtained neighborhood diversity decompostion also have exactly the same local linear constraints.

This, in turn, allows us to prove the main theorem in a much simpler setting.

Single Local Cardinality Constraint. We first show how to alter the given decompostion with respect to one local cardinality constraint α which is the core of the represented reduction. Then, we show how this can be used to alter the neighborhood diversity decomposition when more local cardinality constraints $(\alpha_1, \ldots, \alpha_\ell)$ are given.

Let G be a graph and $\mathcal{T}$ its neighborhood diversity decomposition. A type $T \in \mathcal{T}$ is said to be *nonuniform* with respect to local linear cardinality constraint α if there exist vertices $u, v \in T$ with $\alpha(u) \neq \alpha(v)$, otherwise T is said to be *uniform*. As already mentioned, the purpose of this section is to alter the given instance into an equivalent uniform one. In order to be able to do so we have to change the neighborhood diversity decomposition (i.e., the type graph). A neighborhood diversity decomposition $\hat{\mathcal{T}}$ is a *refinement* of $\mathcal{T}$ if for every $\hat{T} \in \hat{\mathcal{T}}$ there exists a type $T \in \mathcal{T}$ such that $\hat{T} \subseteq T$. We define $\nu_\alpha(\mathcal{T})$ as the number of nonuniform types in $\mathcal{T}$ with respect to α.

Proposition 3.7. *Let $G = (V, E)$ be a graph and let $\mathcal{T}$ be a neighborhood diversity decomposition of G. For every $T \in \mathcal{T}$ and for every $X \subseteq V$ there exists a nonnegative integer z such that for every vertex $w \in T$*

- *it holds that $|X(w)| = z$ if T is an independent set and*
- *it holds that $|X(w)| \in \{z, z+1\}$ if T is a clique.*

Proof. First assume that T is an independent set, then $N(v) = N(w)$ for all $v, w \in T$.

For the second case assume that T is a clique and let $M = N(v) \setminus \{w\}$. Now $N(v) = M \cup \{w\}$ and $N(w) = M \cup \{v\}$ and, as the number $|M \cap X|$ contributes to both $X(v)$ and $X(w)$, $\big||X(v)| - |X(w)|\big| \leq 1$ must hold. □

Now, we show how to refine the neighborhood diversity decomposition with respect to one local linear cardinality constraint.

Lemma 3.8. *Given a graph $G = (V, E)$, neighborhood diversity decomposition $\mathcal{T}$ of G, and local linear cardinality constraint α. Let $T \in \mathcal{T}$ be a nonuniform type. There exists a partition $\mathcal{T}'$ of T and local linear cardinality constraint α' such that the following holds*

(1) *$|\mathcal{T}'| \leq 4$,*
(2) *if $T' \in \mathcal{T} \setminus \{T\}$ is a uniform type with respect to α, then T' remains uniform with respect to α',*
(3) *$\nu_{\alpha'}((\mathcal{T} \setminus \{T\}) \cup \mathcal{T}') < \nu_\alpha(\mathcal{T})$, and*
(4) *for each $X \subseteq V$, X satisfies α if and only if X satisfies α'.*

Proof. Let us first argue about an independent type T. In this case it suffices to set $\alpha'(u) = \bigcap_{v \in T} \alpha(v)$ for each $u \in T$. Now $X \subseteq V$ satisfies α if and only if X satisfies α' as the value $|N(T) \cap X|$ has to be the same for all vertices of T and thus has to be in $\alpha'(v)$ for $v \in T$ by Proposition 3.7.

Let T be a clique type of $\mathcal{T}$. We define

$$\mathtt{l} = \max_{v \in T} \min \alpha(v) \text{ and} \tag{3.1}$$

$$\mathtt{u} = \min_{v \in T} \max \alpha(v) \ . \tag{3.2}$$

If $\mathtt{u} \leq \mathtt{l} - 2$, then α cannot be satisfied by Proposition 3.7. Define the new local linear constraint $\alpha'(v) = \alpha(v) \cap [\mathtt{l} - 1, \mathtt{u} + 1]$ for every $v \in T$ and define $\alpha'(v) = \alpha(v)$ for every $v \in V \setminus T$. We get that:

- $\alpha'(v) \subseteq [\mathtt{l} - 1, \mathtt{u} + 1]$ for each $v \in T$ and
- $[\mathtt{l}, \mathtt{u}] \subseteq \alpha'(v)$ for each $v \in T$.

This yields at most four possibilities for $\alpha'(v)$ for $v \in T$; namely $\alpha'(v)$ is one of the sets $[\mathtt{l}-1, \mathtt{u}], [\mathtt{l}-1, \mathtt{u}+1], [\mathtt{l}, \mathtt{u}]$, or $[\mathtt{l}, \mathtt{u}+1]$. We can refine T into at most 4 subtypes such that all the vertices of a subtype of T have the same $\alpha'(v)$. As all newly introduced types are uniform with respect to α', we have replaced a nonuniform type T with at most 4 uniform types (while we have kept all other types untouched). We have proven (1)–(3); in order to prove (4) we use the following claim.

Clearly, all subtypes of T are uniform with respect to α' and for each $X \subseteq V$ it holds that if X satisfies α', then X satisfies α, since $\alpha'(v) \subseteq \alpha(v)$ for every vertex $v \in V$. Thus, it remains to show the converse.

Claim 3.9. *Let $p \in [n]$ and let $\mathtt{l}$ be defined as in (3.1). If there exists $v \in T$ such that the following conditions are fulfilled*

- $p \in \alpha(v)$ *and*
- $p \leq \mathtt{l} - 2$,

then for each X *satisfying* α *it holds that* $p \neq |X(v)|$.

Proof of Claim 3.9. Let z be as in Proposition 3.7, that is, each $w \in T$ must have z or $z+1$ in $\alpha(w)$. Suppose for a contradiction that $|X(v)| = p$ and let s be a vertex with $\alpha(s) \subseteq \{\mathtt{l}, \ldots\}$ (such s exists from the definition of $\mathtt{l}$). As $p \leq \mathtt{l} - 2$, it follows that X cannot satisfy $\alpha(s)$. There are two possible options $\{p-1, p\}$ and $\{p, p+1\}$ for the value of z from Proposition 3.7. Observe that $\{p-1, p, p+1\} \cap \alpha(s) = \emptyset$. This finishes the proof of the claim. □

Let $X \subseteq V$ satisfy α; otherwise there is nothing to prove. By the above claim and its symmetric version for $p \geq \mathtt{u} + 2$ it follows that $\mathtt{l} - 1 \leq X(v) \leq \mathtt{u} + 1$. By the definition of α' it follows that X satisfies α'. □

Local Cardinality Constraints. We now apply the above lemma to all local cardinality constraints $(\alpha_1, \ldots, \alpha_\ell)$ in the given instance.

Lemma 3.10. *Given a graph* $G = (V, E)$ *with* $\mathrm{nd}(G) = \nu$ *and with local linear cardinality constraints* $(\alpha_1, \ldots, \alpha_\ell)$*, there exists a neighborhood decomposition* $\mathcal{T}$ *of* G *of size at most* $\nu 4^\ell$ *and local linear cardinality constraints* $(\alpha'_1, \ldots, \alpha'_\ell)$ *such that:*

- *each type* $T \in \mathcal{T}$ *is uniform with respect to* α_i *for all* $i \in [\ell]$*, and,*
- *for each* $(X_1, \ldots, X_\ell) \subseteq V^\ell$*,* X_i *satisfies* α_i *for all* $i \in [\ell]$ *if and only if* X_i *satisfies* α'_i *for all* $i \in [\ell]$*.*

Proof. The proof goes by repeatedly applying Lemma 3.8. We start with the neighborhood decomposition $\hat{\mathcal{T}}$ of size ν that is guaranteed by $\mathrm{nd}(G) = \nu$, and with the local linear cardinality constraints $(\alpha_1, \ldots, \alpha_\ell)$.

First let $i = 1$, and as long as there is a type $T \in \hat{\mathcal{T}}$ that is nonuniform with respect to α_1 we do the following. We apply Lemma 3.8 to the type T, the local linear cardinality constraint α'_1 and decomposition $\mathcal{T}'$ resulting from the previous application of the lemma (using α_1 and $\hat{\mathcal{T}}$ in the first iteration). Note that in such an iteration we leave all of the other local cardinality constraints $(\alpha_2, \ldots, \alpha_\ell)$ intact. Clearly, after we are done we have a neighborhood decomposition $\mathcal{T}'$ of size at most $4 \cdot \nu$ and local linear cardinality constraints $(\alpha'_1, \alpha_2, \ldots, \alpha_\ell$ such that every type $T \in \mathcal{T}'$ is uniform with respect to α'_1.

Then, continuing with $i \in [2, \ell]$, we do the same, finally resulting in a decomposition $\mathcal{T}$ of size $\nu 4^\ell$ and local linear cardinality constraints $(\alpha'_1, \ldots, \alpha'_\ell)$. Note that the only side effect of an invocation of Lemma 3.8 is a refinement of some type T and observe that if T is uniform with respect to α'_j, then so is any of its subtypes in the refinement. Consequently, every type $T \in \mathcal{T}$ is uniform with respect to α'_i for all $i \in [\ell]$. □

Uniform Instance. Let G be a graph and let $(\alpha_1, \ldots, \alpha_\ell)$ be local cardinality constraints. For a given neighborhood diversity decomposition $\mathcal{T}$ we say that $(\alpha_1, \ldots, \alpha_\ell)$ are *uniform* if $\nu_{\alpha_i}(T) = 0$ for all $T \in \mathcal{T}$ and all $i \in [\ell]$.

3.2.3. *Uniform Instance Theorem.*

Theorem 3.11. *There exists an algorithm that, for given an* $\mathsf{MSO}^{\mathsf{GL}}_{\mathsf{lin}}$ *formula* φ *with free set variables* $X_1, \ldots, X_\ell$*, graph* $G = (V, E)$ *with neighborhood diversity decomposition* $\mathcal{T}$*, and uniform local linear constraints* $(\alpha_1, \ldots, \alpha_\ell)$ *decides whether there exists an assignment* μ *such that* $G, \mu \models \varphi$ *and* $|\mu(X_i) \cap N(v)| \in \alpha_i(v)$ *for every* $v \in V$ *and every* $i \in [\ell]$*. The algorithm terminates in time* $f(\|\varphi\|, \nu)n^{\mathcal{O}(1)}$ *for some computable function* f*. Moreover, if such an assignment exists, the algorithm outputs one.*

Proof. Let ν be the size of $\mathcal{T}$.

The algorithm works as follows. For every pre-evaluation function β and every mapping $sh\colon [\nu] \times 2^{[\ell]} \to [0, t] \cup \{\uparrow\}$, we test whether sh is admissible. This can be done by picking an arbitrary variable assignment μ of shape sh (if there exists such an assignment) and testing whether $G, \mu \models \beta(\varphi)$ by an FPT model checking algorithm for MSO formulae [34].

If the shape sh is admissible with respect to β, we need to find a variable assignment μ such that

- μ complies with β,
- μ has shape sh, and
- μ satisfies the local linear constraints.

We find such an assignment μ by the following integer linear program (which is infeasible if μ does not exist). We begin the description of the integer linear program with a description of all its variables:

- for every $I \subseteq [\ell], j \in [\nu]$, we introduce an integer variable x_I^j (these correspond to $S_\mu(j, I)$ of the variable assignment μ we are about to find),
- for every $i \in [\ell]$, $j \in [\nu]$, we introduce an auxiliary variable y_i^j corresponding to $|\mu(X_i) \cap T_j|$, and
- for every $i \in [\ell]$, we add an auxiliary variable z_i corresponding to $|\mu(X_i)|$ (technically variables y_i^j and z_i are redundant as they are projections of the x variables, but they will simplify the presentation).

To ensure that μ has the required properties, we add the following constraints:

$$\sum_{I \subseteq [\ell]} x_I^j = |T_j| \qquad \text{for every } j \in [\nu] \tag{0}$$

$$y_i^j = \sum_{\{i\} \subseteq I \subseteq [\ell]} x_I^j \qquad \text{for every } j \in [\nu] \text{ and every } i \in [\ell] \tag{a1}$$

$$z_i = \sum_{j=1}^{\nu} y_i^j \qquad \text{for every } i \in [\ell] \tag{a2}$$

$$x_I^j = sh(I, j) \qquad \text{for every } j \in [\nu], I \subseteq [\ell] \text{ such that } sh(I, j) \neq \uparrow \tag{sh1}$$

$$x_I^j > t \qquad \text{for every } j \in [\nu], I \subseteq [\ell] \text{ such that } sh(I, j) = \uparrow \tag{sh2}$$

We note that if the variables x are integral, then all auxiliary variables are integral as well (as we obtain them only by summing up integers). Thus, these variables can be real and thus do not contribute to the total number of integral variables (i.e., only the x variables do). The constraints (0) ensure that variables x_I^j encode a variable assignment $\mu(x)$ for

the graph G. This is as follows: There are exactly x_I^j vertices form type T_j in the set $\bigcap_{i\in I}\mu(X_i)$. Now, it is not hard to see that constraints (0) ensure that every vertex of type T_j is placed (possibly to none of X_i's). The constraints (a1) and (a2) set auxiliary variables y_i^j and z_i to the desired values. The constraints (sh1) and (sh2) guarantee that $\mu(x)$ has the shape sh.

We for convenience denote the i-th linear cardinality constraint for type T_j by $\alpha_{i,j}$ and furthermore we set $\mathrm{lb}_j^i := \min\{\alpha_{i,j}(v) \mid v \in T_j\}$ and $\mathrm{ub}_j^i := \max\{\alpha_{i,j}(v) \mid v \in T_j\}$.

If T_j is an independent type, we need to ensure that for every $v \in T_j$ we have $|\mu(X_i) \cap N(v)| \in \alpha_{i,j}(v)$. It is easy to see that the quantity $|\mu(X_i) \cap N(v)|$ is the same for every $v \in T_j$ and it can be expressed as

$$\sum_{j':\{j',j\}\in E(T_G)} \left|\mu(X_i) \cap T_{j'}\right| .$$

By the definition of auxiliary variables y_I^j, we have that $|\mu(X_i) \cap T_{j'}| = y_i^{j'}$, so the local linear condition for the variable X_i can be rewritten as

$$\mathrm{lb}_j^i \leq \sum_{j':\{j',j\}\in E(T_G)} y_i^j \leq \mathrm{ub}_j^i . \tag{lli}$$

If T_j is a clique type, we have to be slightly more careful, since the quantity $|\mu(X_i) \cap N(v)|$ depends on whether v is in $\mu(X_i)$ or not. The set $N(v)$ does not include v itself, so if $|\mu(X_i)\cap N(v')| = |\mu(X_i)\cap N(v)|+1$ for every $v \in T_j \cap \mu(X_i)$, then $v' \in T_j \setminus \mu(X_i)$. Similarly as before, we have equations

$$|\mu(X_i) \cap N(v)| = \left(\sum_{j':\{j',j\}\in E(T_G)} \left|\mu(X_i) \cap T_{j'}\right|\right) - 1$$

for $v \in T_j \cap \mu(X_i)$, and

$$|\mu(X_i) \cap N(v)| = \sum_{j':\{j',j\}\in E(T_G)} \left|\mu(X_i) \cap T_{j'}\right|$$

for $v \in T_j \setminus \mu(X_i)$.

This means that we need to add the constraint

$$\mathrm{lb}_j^i \leq \sum_{j':\{j',j\}\in E(T_G)} y_i^{j'} \leq \mathrm{ub}_j^i \tag{llc1}$$

if $|\mu(X_i) \cap T_j| \geq 1$ and add the constraint

$$\mathrm{lb}_j^i \leq \sum_{j':\{j',j\}\in E(T_G)} y_i^{j'} - 1 \leq \mathrm{ub}_j^i \tag{llc2}$$

if $|T_j \setminus \mu(X_i)| \geq 1$.

Fortunately, we can deduce whether the conditions $|\mu(X_i) \cap T_j| \geq 1$ or $|T_j \setminus \mu(X_i)| \geq 1$ hold already from the shape sh. If we have

$$\sum_{I\,:\,I\ni i} sh(I,j) > 0 \qquad\qquad (\forall i \in [\ell]) , \tag{cllc1}$$

then $\mu(X_i)$ necessarily intersect T_j, whereas if we have

$$\sum_{I\,:\,I \not\ni i} sh(I,j) > 0 \qquad (\forall i \in [\ell])\ , \tag{cllc2}$$

then there exists vertex in $T_j \setminus \mu(X_i)$.

This means that the local linear constraints for type T_j and variable X_i can be enforced by adding constraint (llc1) if (cllc1) holds, and by adding constraint (llc2) if (cllc2) holds.

It remains to add all of the constraints arising from the pre-evaluation β. However, this is not a problem, since we know that every such condition is linear and as such can be easily added to the so far constructed MILP. Note that any constraint pre-evaluated in β is of the form $\sum_{i=1}^{\ell} c_i \cdot |X_i| \leq b$. We can assume that all constants c_i are integers, since otherwise we can multiply all of them by greatest common divisor. Furthermore, observe that b can be assumed to be integral as well, since the left-hand side is now integral and thus upper-bounding it by b is the same as upper-bounding it by $\lfloor b \rfloor$. Now, based on β we either add the constraint $\sum_{i=1}^{\ell} c_i \cdot |X_i| \leq b$ or the constraint $\sum_{i=1}^{\ell} c_i \cdot |X_i| \geq b+1$ to the above constructed MILP.

Let us turn our attention to the analysis of the running time of the algorithm. There are at most

- $(t+2)^{\nu}$ different shapes and
- $2^{||\varphi||}$ pre-evaluation functions.

Since t depends only on the number of quantifiers in the formula φ, both numbers can be bounded by a function of $||\varphi||$ and ν. For each such combination of a shape and a pre-evaluation, we construct an ILP with $\nu 2^{\ell}$ integer variables, so this ILP can be solved in time FPT time with respect to $||\varphi||$ and ν by the aforementioned result of Lenstra [36]. □

Proof of Theorem 1.4. The theorem is a simple consequence of Theorem 3.11 and Lemma 3.10. □

3.3. **Theorem 1.5: XP algorithm for $\mathsf{MSO}^{\mathsf{GL}}$.** The main idea behind the proof of Theorem 1.5 is as follows. We use the advantage of XP time to guess all sizes b_I^T of the sets $T \cap \bigcap_{i \in I} \mu(X_i)$ for a possible assignment μ, for every type $T \in \mathcal{T}$, and for every $I \subseteq [\ell]$. Clearly, the number of possible assignments of b_I^T can be upper-bounded by $n^{|\mathcal{T}|2^{\ell}}$. This immediately allows us to verify global cardinality constraints and in particular φ. Note that at this point this is possible, since the vertices in T are equivalent with respect to MSO logic and thus the MSO-model-checking algorithm of Lampis [35] can be used to verify φ on the (now labeled) graph G. Here the labeling represents the assignment μ respecting the guessed values b_I^T. If φ is satisfied, we proceed in a similar way as in the proof of Theorem 1.4. Now, we have to check whether there exists an assignment μ that obeys all local cardinality constraints for each vertex in G. Observe that this can be done locally, as the only thing that matters in the neighboring types is the number of vertices in the sets $X_1, \ldots, X_{\ell}$ (especially it is independent of the actually selected vertices). This however, results in a computation of lower- and upper-bounds on b_I^T in a similar but simpler way to Lemma 3.8. Finally, if all b_I^T fulfill both lower- and upper-bounds, we can use these values to computer an assignment μ that on the one hand satisfies φ (and thus the global constraints) and on the other hand satisfies all the local cardinality constraints (see Lemma 3.12). If such b_I^T's do not exist it is impossible to simultaneously satisfy the local and the global cardinality constraints.

3.3.1. *Extended Numerical Assignments.* Fix a formula φ with ℓ free variables $X_1, \ldots, X_\ell$. Let $G = (V, E)$ be a graph and let $\mathcal{T}$ be its neighborhood diversity decomposition. The *extended numerical assignment* is a function $\sigma\colon 2^{\{X_1,\ldots,X_\ell\}} \times \mathcal{T} \to [|V(G)|]$. We say that σ is *valid for G and $\mathcal{T}$* if

$$\sum_{I \in 2^{\{X_1,\ldots,X_\ell\}}} \sigma(I, T) \leq |T|$$

for each type $T \in \mathcal{T}$. The crucial thing is that the extended numerical assignment plays the same role for the purpose of a design of the XP algorithm as pre-evaluations for the FPT algorithm presented in Section 3.2. We formalize this by showing that knowing σ it is possible to decide whether φ holds for G or not. Before we do so, we have to introduce one more formalism. We say that an assignment $\mu\colon \{X_1, \ldots, X_\ell\} \to 2^V$ is a *realisation* of a valid extended numerical assignment σ if

$$\left| \left(\bigcap_{i \in I} \mu(X_i) \right) \cap T \right| = \sigma(I, T)$$

for every $I \subseteq \{X_1, \ldots, X_\ell\}$ and every type $T \in \mathcal{T}$.

Lemma 3.12. *Fix a formula φ with ℓ free variables $X_1, \ldots, X_\ell$. Let G be a graph, let $\mathcal{T}$ be its neighborhood diversity decomposition, and let σ be a valid extended numerical assignment. Then either each realization μ of σ satisfies φ or no realization of σ satisfies φ.*

Proof. Let assignments μ and μ' be realizations of σ. We will show that $G, \mu \models \varphi$ if and only if $G, \mu' \models \varphi$. This is not hard to see. Note that the global cardinality constraints $R_1, \ldots, R_c$ contained in φ depend solely on the number of vertices contained in some X_i's and some types of $\mathcal{T}$ which is, however, prescribed by σ and thus both μ, μ' have to agree on this. Now, it is possible to evaluate the validity of all of the constraints $R_1, \ldots, R_c$ and (as in the case of pre-evaluations) replace them by the constants `true` or `false` in φ yielding a simplified formula $\tilde{\varphi}$. The lemma now follows from the fact that $\tilde{\varphi}$ is an MSO formula and both μ and μ' yield the same labeled graph. □

By this we have shown that it makes sense to write $G, \sigma \models \varphi$. Observe that there are at most $|V(G)|^{|\mathcal{T}| \cdot 2^\ell}$ (valid) extended numerical assignments for G.

3.3.2. *Satisfying Local Cardinality Constraints.* Fix a valid extended numerical assignment σ with $G, \sigma \models \varphi$. Note that in such a case there is at least one realisation of σ. Now, we would like to resolve whether among all of the possible realizations of σ there is at least one realization that obeys the local cardinality constraints $(\alpha_1, \ldots, \alpha_\ell)$.

Fix a type $T \in \mathcal{T}$ and define the function $s\colon \{1, \ldots, \ell\} \to \mathbb{N}$ by

$$s(i) = \left| \bigcup_{S \in \mathcal{T} : \{S,T\} \in E(T_G)} (S \cap \mu(X_i)) \right|.$$

Recall that if T is a clique in G, it has a loop in the corresponding type graph T_G. Now, following Proposition 3.7 we say that $(\alpha_1, \ldots, \alpha_\ell)$ are *possibly satisfied by σ* if

- $s(i) \in \alpha_i(v)$ for all $i \in [\ell]$ and $v \in T$, where T is an independent set in G and
- $\{s(i) - 1, s(i)\} \cap \alpha_i(v) \neq \emptyset$ for all $i \in [\ell]$ and $v \in T$, where T is a clique in G.

Observe that if T is an independent set in G and $(\alpha_1, \ldots, \alpha_\ell)$ are possibly satisfied by σ, then every assignment μ realizing σ fulfills $(\alpha_1, \ldots, \alpha_\ell)$ for every vertex in T. This is, however, not true when T is a clique in G. In this case we let t_i^+ be the number of vertices in T with $\{s(i)-1, s(i)\} \cap \alpha_i(v) = \{s(i)-1\}$, we let t_i^- be the number of vertices in T with $\{s(i)-1, s(i)\} \cap \alpha_i(v) = \{s(i)\}$, and we let $t_i^\pm$ be $|T| - (t_i^+ + t_i^-)$. Note that t_i^+ is the number of vertices in T that must belong to X_i, t_i^- is the number of vertices in T that cannot belong to X_i, and $t_i^\pm$ is the number of vertices in T that may or may not belong to X_i (again this directly follows from Proposition 3.7). Thus, we arrive at the following claim.

Lemma 3.13. *If $(\alpha_1, \ldots, \alpha_\ell)$ are possibly satisfied by σ and for each type $T \in \mathcal{T}$ forming a clique in G we have*

$$t_i^+ \leq s(i) \leq t_i^+ + t_i^\pm ,$$

then there is an assignment μ realizing σ and fulfilling all of $(\alpha_1, \ldots, \alpha_\ell)$. □

Proof of Theorem 1.5. Let G be a graph with $n = |V(G)|$ and neigborhood diversity ν. There are $n^{\nu \cdot 2^\ell}$ possible (valid) extended numerical assignments for G. We loop through all of them and for each such σ we

(1) check if $G, \sigma \models \varphi$,
(2) check if σ possibly satisfies $(\alpha_1, \ldots, \alpha_\ell)$, and
(3) verify the conditions given in Lemma 3.13.

If all three of the above conditions are satisfied, we accept σ and say Yes, otherwise we reject σ and proceed to next σ. If there is no σ left, we say No. It is not hard to see that the above procedure takes $O(n \cdot n^{\nu \cdot 2^\ell})$ time and can be simply extended so that it actually returns the sought assignment μ in the same time. □

4. Theorem 1.1: XP algorithm for $\mathsf{MSO}^{\mathsf{GL}}$ on bounded treewidth

We believe that the merit of Theorem 1.1 lies not only in being a very general tractability result, but also in showcasing a simplified way to prove a metatheorem extending MSO. Our main tool is the constraint satisfaction problem (CSP). The key technical result of this section is Theorem 4.6, which relates MSO and CSP on graphs of bounded treewidth. Let us shortly describe it.

Notice that in the $\mathsf{MSO}^{\mathsf{GL}}$ model checking problem, we wish to find a satisfying assignment of some formula φ which satisfies further constraints. Simply put, Theorem 4.6 says that it is possible to restrict the set of satisfying assignments of a formula $\varphi \in \mathsf{MSO}_2$ with CSP constraints under the condition that these additional constraints are structured along the tree decomposition of G. This allows the proof of Theorem 1.1 to simply be a CSP formulation satisfying this property.

We believe that the key advantage of our approach, when compared with prior work, is that it is *declarative*: it only states what a solution looks like, but does not describe how it is computed. This makes the proof cleaner, and possible extensions easier.

We consider a natural optimization version of $\mathsf{MSO}^{\mathsf{GL}}$:

Weighted $\mathsf{MSO}^{\mathsf{GL}}$
Input: An $\mathsf{MSO}^{\mathsf{GL}}$ model checking instance, weights $\mathbf{w}^1, \ldots, \mathbf{w}^\ell \in \mathbb{Z}^n$.
Task: Find an assignment $X_1, \ldots, X_\ell$ satisfying the $\mathsf{MSO}^{\mathsf{GL}}$ model checking instance and minimizing $\sum_{j=1}^{\ell} \sum_{v \in X_j} w_v^j$.

4.1. **CSP, MSO and treewidth.**

Definition 4.1 (CSP)**.** *An instance $I = (V, \mathcal{D}, \mathcal{H}, \mathcal{S})$ of CSP consists of*

- *a set of* variables z_v*, one for each $v \in V$; without loss of generality we assume that $V = [|V|]$,*
- *a set $\mathcal{D}$ of finite* domains $D_v \subseteq \mathbb{Z}$*, one for each $v \in V$,*
- *a set of* hard constraints $\mathcal{H} \subseteq \{C_U \mid U \subseteq V\}$ *where each hard constraint $C_U \in \mathcal{H}$ with a* scope $U = \{i_1, \ldots, i_k\}$ *and $i_1 < \cdots < i_k$, is a $|U|$-ary relation $C_U \subseteq D_{i_1} \times \cdots \times D_{i_k}$,*
- *a set of* weighted soft constraints $\mathcal{S} \subseteq \{w_U \mid U \subseteq V\}$ *where each $w_U \in \mathcal{S}$ with a* scope $U = \{i_1, \ldots, i_k\}$ *and $i_1 < \cdots < i_k$ is a function $w_U : D_{i_1} \times \cdots \times D_{i_k} \to \mathbb{R}$.*

For a vector $\mathbf{z} = (z_1, \ldots, z_{|V|})$ and a set $U = \{i_1, \ldots, i_k\} \subseteq V$ with $i_1 < \cdots < i_k$, we define the projection of $\mathbf{z}$ *on U as $\mathbf{z}|_U = (z_{i_1}, \ldots, z_{i_k})$. A vector $\mathbf{z} \in \mathbb{Z}^{|V|}$* satisfies the hard constraint $C_U \in \mathcal{H}$ *if and only if $\mathbf{z}|_U \in C_U$. We say that a vector $\mathbf{z}^\star = (z^\star_1, \ldots, z^\star_{|V|})$ is* a feasible assignment *for I if $\mathbf{z}^\star \in D_1 \times \cdots \times D_{|V|}$ and $\mathbf{z}^\star$ satisfies every hard constraint $C \in \mathcal{H}$, and write* $\mathrm{Feas}(I) = \{\mathbf{z}^\star \mid \mathbf{z}^\star$ *is a feasible assignment for $I\}$. The* weight *of $\mathbf{z}^\star$ is $w(\mathbf{z}^\star) = \sum_{w_U \in \mathcal{S}} w_U(\mathbf{z}^\star|_U)$. To solve a CSP instance I means to find a feasible assignment $\mathbf{z}$ which minimizes the weight $w(\mathbf{z}^*)$.*

We denote by D_I the maximum size of all domains, that is, $D_I = \max_{u \in V} |D_u|$, and we omit the subscript I if the instance is clear from the context. We denote by $\|\mathcal{D}\|$, $\|\mathcal{H}\|$ and $\|\mathcal{S}\|$ the length *of $\mathcal{D}$, $\mathcal{H}$ and $\mathcal{S}$, respectively, and define it as $\|\mathcal{D}\| = \sum_{v \in V} |D_v|$, $\|\mathcal{H}\| = \sum_{C_U \in \mathcal{H}} |C_U|$ and $\|\mathcal{S}\| = \sum_{w_U} |w_U|$; here $|w_U|$ denotes the size of the subset of $D_{i_1} \times \cdots \times D_{i_k}$ for which the function w_U is nonzero.*

Definition 4.2 (Constraint graph, Treewidth of CSP)**.** *For a CSP instance $I = (V, \mathcal{D}, \mathcal{H}, \mathcal{S})$ we define the* constraint graph *$G(I)$ of I as $G = (V, E)$ where*

$$E = \{\{u, v\} \mid (\exists C_U \in \mathcal{H}) \vee (\exists w_U \in \mathcal{S}) \text{ s.t. } \{u, v\} \subseteq U, u \neq v\}.$$

The treewidth of a CSP instance *I, $tw(I)$, is defined as* $\mathrm{tw}(G(I))$*. When we talk about $G(I)$ we use the terms "variable" and "vertex" interchangeably.*

Freuder [20] proved that CSPs of bounded treewidth can be solved quickly. We use a natural weighted version of this result.

Proposition 4.3 ([20])**.** *For a CSP instance I of treewidth τ and maximum domain size D, a minimum weight solution can be found in time $\mathcal{O}(D^\tau |V| + \|\mathcal{H}\| + \|\mathcal{S}\|)$.*

Modeling after the terminology regarding extended formulations of polytopes [9], we introduce the notion of a *CSP extension*.

Definition 4.4 (CSP extension)**.** *Let $I = (V_I, \mathcal{D}_I, \mathcal{H}_I, \mathcal{S}_I)$ be a CSP instance. We say that $J = (V_J, \mathcal{D}_J, \mathcal{H}_J, \mathcal{S}_J)$ is an* extension of *I (or that J* extends *I) if $V_I \subseteq V_J$ and* $\mathrm{Feas}(I) = \{\mathbf{z}^\star|_{V_I} \mid \mathbf{z}^\star \in \mathrm{Feas}(J)\}$.

By Proposition 4.3, we can solve CSP instances of small treewidth efficiently. Our motivation for introducing CSP extensions is that we are able to formulate a CSP instance I expressing what we need, but having large treewidth. However, if an extension J of I exists with small treewidth, solving J instead suffices.

Let φ be an MSO_2 formula with ℓ free set variables and let G a σ_2-structure with a universe of size n. We say that a binary vector $\mathbf{y} \in \{0,1\}^{n\ell}$ *satisfies* φ ($G, \mathbf{y} \models \varphi$) if it is the characteristic vector of a satisfying assignment μ, that is, if $v \in \mu(X_i) \Leftrightarrow y^i_v = 1$ and $G, \mu \models \varphi$. For a vector $\mathbf{s}$, let the *support of* $\mathbf{s}$ be $\mathrm{supp}(\mathbf{s}) = \{i \mid s_i \neq 0\}$, that is, the set of

its nonzero indices. The following definition characterizes sets which are structured along a given tree decomposition of a graph. The subsequent theorem then shows that, provided a CSP instance whose constraints are structured in this way along a tree decomposition of G, there exists a compact and tree-structured CSP extension.

Definition 4.5 (Local scope property)**.** *Let $\ell, m \in \mathbb{N}$, G be a σ_2-structure, $(T, \mathcal{B})$ be a nice tree decomposition of G, and S be a set of vectors of elements indexed by $U := (V(G) \times [\ell]) \cup (V(T) \times [m])$. We say that S has the* local scope property *if*

$$\forall \mathbf{s} \in S \; \exists a \in V(T) : \; supp(\mathbf{s}) \subseteq \big(\{(v,i) \mid v \in B(a), i \in [\ell]\} \cup \{(b,j) \mid b \in N_T^{\downarrow}(a), j \in [m]\}\big).$$

We extend the definition to a set S containing not only vectors but also mappings indexed in the same way, where for $U' \subseteq U$ and a mapping $w_{U'} : \mathbb{Z}^{U'} \to \mathbb{R}$, we define $supp(w_{U'}) = U'$.

The heart of our proof of Theorem 1.1 is the following theorem.

Theorem 4.6. *Let $I = (V, \mathcal{D}, \mathcal{H}, \mathcal{S})$ be a CSP instance, G a σ_2-structure, φ an MSO_1 formula with ℓ free variables, $(T, \mathcal{B})$ a nice tree decomposition of G of width τ, an integer $k \in \mathbb{N}$, and a set of hard constraints $\mathcal{H}'$ so that V and $\mathcal{H}$ satisfy*

$$V = \{y_v^i \mid v \in V(G), i \in [\ell]\} \cup \{x_a^j \mid a \in V(T), j \in [k]\} \text{ and } \mathcal{H} = \{\mathbf{y} \mid G, \mathbf{y} \models \varphi\} \cup \mathcal{H}'.$$

If $\mathcal{H}' \cup \mathcal{S}$ have the local scope property, then there is a computable function f and an algorithm computing in time $f(\|\varphi\|, \tau) \cdot |V| + \|\mathcal{H}' + \mathcal{S}\|)$ a CSP instance $J = (V_J, \mathcal{D}_J, \mathcal{H}_J, \mathcal{S}_J)$ which extends I, and,

- $\mathrm{tw}(J) \leq f(||\varphi||, \tau) + 2k$,
- $\|\mathcal{H}_J\| + \|\mathcal{S}_J\| \leq f(||\varphi||, \tau) \cdot |V| + (\|\mathcal{H}'\| + \|\mathcal{S}\|)$,
- $D_J = D_I$.

Before giving the proof, we will show how Theorem 4.6 implies Theorem 1.1.

4.2. **CSP instance construction.**

Proof of Theorem 1.1. As before, we first note that there are at most $2^{||\varphi||}$ different pre-evaluations $\beta(\varphi)$ of φ, so we can try each and choose the result with minimum weight. Let a pre-evaluation $\beta(\varphi)$ be fixed from now on.

Let $(T, \mathcal{B})$ be a nice tree decomposition of G. We will now construct a CSP instance I satisfying the conditions of Theorem 4.6, which will give us its extension J with properties suitable for applying Freuder's algorithm (Proposition 4.3).

Let y_v^i be the variables as described above in Theorem 4.6; we use the constraint $G, \mathbf{y} \models \beta(\varphi)$ to enforce that each feasible solution complies with the pre-evaluation $\beta(\varphi)$. Now we will introduce additional CSP variables and constraints in two ways to assure that the local and global cardinality constraints are satisfied. Observe that we introduce the additional CSP variables and constraints in such a way that they have the local scope property (Definition 4.5), that is, their scopes will always be limited to the neighborhood of some node $a \in V(T)$.

Global cardinality constraints. In addition to the original $\mathbf{y}$ variables, we introduce, for each node $a \in T$ and each $j \in [\ell]$, a variable s_a^j with domain $[n]$. We refer to the set of these variables as to s-variables. The meaning of this variable is $s_a^j = |X_j \cap V(G_a)|$. Thus, in the root node r, s_r^j is exactly $|X_j|$. To enforce the desired meaning of the variables $\mathbf{s}$, we add the following hard constraints:

$$\begin{aligned} s_a^j &= 0 && \forall \text{ leaves } a \\ s_a^j &= s_b^j + y_v^j && \forall a = b * (v) \\ s_a^j &= s_b^j && \forall a = b \dagger (v) \\ s_a^j &= s_b^j + s_{b'}^j - \sum_{v \in B(a)} y_v^j && \forall a = \Lambda(b, b') \end{aligned}$$

To enforce the cardinality constraints themselves, we add:

$$\begin{aligned} (s_r^1, \ldots, s_r^\ell) &\in R && \forall R : \beta(R) = \texttt{true} \\ (s_r^1, \ldots, s_r^\ell) &\in ([n]^\ell \setminus R) && \forall R : \beta(R) = \texttt{false} \end{aligned}$$

Local cardinality constraints. For every node $a \in V(T)$, every $j \in [\ell]$ and every vertex $v \in B(a)$, introduce a variable λ_a^{vj} with domain $[n]$, with the meaning $\lambda_a^{vj} = |N_{G_a}(v) \cap X_j|$. We refer to these as to λ-variables. Their meaning is enforced by setting:

$$\begin{aligned} \lambda_a^{vj} &= \sum_{\substack{u \in B(a): u \neq v \\ uv \in E}} y_u^j && \forall a = b * (v) \\ \lambda_a^{uj} &= \lambda_b^{uj} + y_v^j && \forall a = b * (v), u \in B(a), uv \in E \\ \lambda_a^{vj} &= \lambda_b^{vj} && \forall a = b \dagger (u), u \neq v, v \in B(a) \\ \lambda_a^{vj} &= \lambda_b^{vj} + \lambda_{b'}^{vj} - \sum_{\substack{u \in B(a): u \neq v \\ uv \in E}} y_u^j && \forall a = \Lambda(b, b') \end{aligned}$$

Now the local constraints themselves are enforced by setting:

$$\lambda_{\mathrm{top}(v)}^{vj} \in \alpha_j(v) \qquad \forall v \in V, \quad j \in [\ell] \ .$$

Objective function. In order to express the objective function we add soft constraints $\mathcal{S} = \{w_{\{y_v^j\}} \mid v \in V, j \in [\ell]\}$ where $w_{\{y_v^j\}}(y_v^j) = w_v^j$ if $y_v^j = 1$ and is 0 otherwise.

In order to apply Theorem 4.6, let us determine the parameters of the CSP instance I we have constructed, namely the number of extra variables per node k, the maximum domain size D_I, and the lengths of the additional constraints $\|\mathcal{H}'\|$ and $\|\mathcal{S}\|$.

We have introduced ℓ s-variables per node, and $\ell\tau$ λ-variables per node. Thus, $k = (\tau + 1)\ell$. Since each s- and λ-variable corresponds to a size of some vertex subset, its value is upper bounded by n, and thus $D_I = n$. Let

$$N = \sum_{j=1}^{c} |R_j^G| + \sum_{j=1}^{\ell} \sum_{v \in V(G)} |\alpha_j(v)|$$

be the input length of the global and local cardinality constraints.

Let $\mathcal{H}'$ be the set of all CSP constraints defined above to enforce the global and local constraints of the $\mathsf{MSO}^{\mathsf{GL}}$ model checking instance. Then, Theorem 4.6 implies that we can compute a CSP instance J which is an extension of I and has

- $\mathrm{tw}(J) \leq f(||\varphi||,\tau)+2k \leq f(||\varphi||,\tau)+(\tau+1)\ell \leq f'(||\varphi||,\tau)$ for some computable function f',
- $\|\mathcal{H}_J\| + \|\mathcal{S}_J\| \leq f(||\varphi||,\tau)\cdot|V| + (\|\mathcal{H}'\| + \|\mathcal{S}\|) \leq f(||\varphi||,\tau)\cdot n + N$, and
- $D_J = D_I = n$.

Finally, applying Proposition 4.3 to J solves it in time $n^{f'(||\varphi||,\tau)}+N$, finishing the proof of Theorem 1.1. □

Conditional cardinality constraints. As Szeider [41] points out, it is easy to extend his XP result for $\mathsf{MSO}^{\mathsf{L}}$ in such a way that the local cardinality constraint $|X(v)| \in \alpha(v)$ is conditioned on the fact that $v \in X$. Observe that our approach can be extended in such a way as well by enforcing

$$y_v^j = 1 \implies \lambda^{vj}_{\mathrm{top}(v)} \in \alpha_j(v) \qquad\qquad \forall v \in V, \quad j \in [\ell] \ .$$

(Formally, the above is a binary relation of the variables y_v^j and $\lambda^{vj}_{\mathrm{top}(v)}$ defined as $R = \{(0,i) \mid i \in [n]\} \cup \{(1,i) \mid i \in \alpha_j(v)\}$.) Moreover, in our setting with multiple set variables, we can even condition on an arbitrary predicate $\psi(v, X_1, \ldots, X_m)$ describing how vertex v relates to the set variables.

4.3. **Applications (Corollary 1.2).** Let us briefly sketch some consequences of Theorem 1.1. We focus on showing how to encode various W[1]-hard (w.r.t. treewidth) problems using the notions we have provided. The parameterized complexity statements which follow are not very surprising and in many cases were known. Still, we believe that our approach captures and summarizes them nicely.

Local constraints. While introducing $\mathsf{MSO}^{\mathsf{L}}$, Szeider [41] points out that the problems General Factor, Equitable k-Coloring and Minimum Maximum Outdegree are expressible in $\mathsf{MSO}^{\mathsf{L}}$. Let us now observe that using the extension to conditional local constraints, we can also express the problems Capacitated Dominating Set, Capacitated Vertex Cover, Vector Dominating Set and Generalized Domination.

Take for example the Capacitated Dominating Set problem. There, we are given a graph $G = (V, E)$ together with a capacity function $c : V \to \mathbb{N}$, and our goal is to find a subset $D \subseteq V$ and a mapping $f : V \setminus D \to D$ such that for each $v \in D$, $|f^{-1}(v)| \leq c(v)$. Essentially, $f(w) = v$ means that the vertex w is dominated by the vertex v, and the condition $|f^{-1}(v)| \leq c(v)$ ensures that the mapping f respects the capacities. Recall that $\mathsf{MSO}^{\mathsf{L}}$ here generalizes MSO_2, so we view G as the σ_2-structure whose universe V_I contains both vertices and edges of the graph it represents. Then, we let $\varphi(D, F)$ and α be a formula and local cardinality constraints, respectively, enforcing that:

- $D \subseteq V$,
- $F \subseteq E$,
- each $v \in V$ is either in D or has a neighbor in F,
- each $e \in F$ has a neighbor in D, and,

- if $v \in D$, then $|N(v) \cap F| \in \alpha_F(v) = [0, c(v)]$.

Then D encodes a dominating set and F can be used to construct the mapping f since for each edge $e \in F$, at most one of its endpoints is not in D, and each $v \in D$ sees at most $c(v)$ edges from F.

Let us define the remaining problems; their $\mathsf{MSO}^{\mathsf{L}}$ formulations are analogous to the one above. The Vector Dominating Set problem is similar to Capacitated Dominating Set, except now each vertex v has a *demand* $d(v)$ and if $v \notin D$, then it must have at least $d(v)$ neighbors in D. In Generalized Domination, we are given two sets $\sigma, \rho \subseteq \mathbb{N}$ and for each vertex v in D or in $V \setminus D$, it must hold that $|N(v) \cap D| \in \sigma$ or $|N(v) \cap D| \in \rho$, respectively. Finally, the Capacitated Vertex Cover problem is the following. We are given a capacitated graph, and the task is to find a vertex cover $C \subseteq V$ and an assignment $f : E \to C$ such that for each v, $|f^{-1}(v)| \leq c(v)$.

Global constraints. Ganian and Obdržálek [23] introduce $\mathsf{MSO}^{\mathsf{G}}_{\mathsf{lin}}$ and show that also this logic expresses Equitable k-Coloring, and moreover the Equitable connected k-Partition problem. Interestingly, they also discuss the complexity of the k-Balanced Partitioning problem, where the goal is to find an equitable (all parts of size differing by at most one) k-partition and, moreover, minimize the number of edges between partites. They provide an FPT algorithm for graphs of bounded vertex cover, but are unable to express the problem in $\mathsf{MSO}^{\mathsf{G}}_{\mathsf{lin}}$, and thus pose as an open problem the task of finding a more expressive formalism which would capture this problem. They also state that no parameterized algorithm exists for graphs of bounded treewidth, but that is no longer true due to the results of van Bevern et al. [4]. On the other hand, the question of capturing k-Balanced Partitioning by some MSO extension stands. Here we show that it can be expressed as an instance of weighted $\mathsf{MSO}^{\mathsf{G}}_{\mathsf{lin}}$ model checking over σ_2-structures (thus this is not applicable to graphs of bounded neighborhood diversity where we only have algorithms for σ_1-structures).

Let φ be an $\mathsf{MSO}^{\mathsf{G}}$ formula with k free vertex set variables $X_1, \ldots, X_k$ and one free edge set variable Y. We use φ to express that $X_1, \ldots, X_k$ is an equitable k-partition; this is easily done using the global constraints. Furthermore, we enforce that Y is the set of edges with one endpoint in X_i and another in X_j for any $i \neq j$. For a satisfying assignment $X_1, \ldots, X_k, Y$, let $(\mathbf{x}, \mathbf{y}) \in \{0,1\}^{k|V|} \times \{0,1\}^{|E|}$ be its characteristic vector. To minimize the number of edges between partites, it suffices to minimize $\mathbf{y}$. This also clearly extends to the case studied in the literature when edges are assigned weights.

Fellows et al. [18] study the Graph Motif problem from the perspective of parameterized complexity, especially on graphs with bounded widths. In Graph Motif, we are given a vertex-colored graph G with χ colors and a multiset of colors M, and the task is to find a *motif*, that is, a connected subset of vertices $S \subseteq V$ such that the multiset of colors of S is exactly M. This problem is most naturally expressed when the vertex-colored graph G is encoded as a σ-structure with $\sigma = (V \cup E, \emptyset, \{L_V, L_E, L_1, \ldots, L_\chi\})$ (i.e., σ is σ_2 extended by unary predicates $L_1, \ldots, L_\chi$). Since we have not explicitly phrased our results for such structures, Graph Motif does not directly fit any of our notions. However, the extension of our results (specifically, Theorem 1.1) to such structures is straightforward since it is known that Courcelle's theorem can be extended and then for example global constraints over the set $S \cap L_i$ are treated like global constraints over any other set (there is no change to the CSP constraints required in the proof of Theorem 1.1).

Then, let us consider the number of colors χ a parameter and introduce additional unary relations (labels) $L_1, \ldots, L_\chi$. It is easy to see that Graph Motif is encoded by the following $\mathsf{MSO}^{\mathsf{G}}_{\mathsf{lin}}$ formula $\varphi(S)$:

$$\varphi(S) \equiv \text{connected}(S) \wedge \bigwedge_{i=1}^{\chi} [|S \cap L_i| = \text{mult}(i, M)],$$

where connected(S) is a formula which holds if S is connected, and mult$(i, M) \in \mathbb{N}$ is the multiplicity of color i in the motif M.

4.4. **Proof of Theorem 4.6.** Fix objects and quantities as in the statement of Theorem 4.6, that is, $I = (V, \mathcal{D}, \mathcal{H}, \mathcal{S})$ is a CSP instance, G is a σ_2-structure, n is the size of the universe of G, φ is an MSO_2 formula with ℓ free variables, $(T, \mathcal{B})$ is a nice tree decomposition of G of width τ, $k \in \mathbb{N}$ is such that V has $\ell \cdot |V(G)|$ variables y_v^i and $k \cdot |V(T)|$ variables x_a^j, $\mathcal{H} = \{\mathbf{y} \mid G, \mathbf{y} \models \varphi\} \cup \mathcal{H}'$ and $\mathcal{H}' \cup \mathcal{S}$ has the local scope property.

We give a brief outline of the proof of Theorem 4.6, which proceeds in three stages:

(1) Using a recent result of Kolman, Koutecký and Tiwary [32] we construct a linear program (LP) whose constraint matrix has bounded treewidth (to be defined) and whose integer solutions correspond to feasible assignments of φ.
(2) We view this LP as an integer linear program (ILP) and construct an equivalent CSP instance J' of bounded treewidth. Thus, J' is an extension of $I' = ([n \cdot \ell], \mathcal{D}_{I'}, \mathcal{H}_{I'}, \emptyset)$ where $\mathcal{D}_{I'} = \{D_i \mid D_i = \{0, 1\}, i \in [n \cdot \ell]\}$ and $\mathcal{H}_{I'} = \{\mathbf{y} \mid G, \mathbf{y} \models \varphi\}$.
(3) We show that if $\mathcal{H}'$ and $\mathcal{S}$ have the local scope property, it is possible to add new constraints derived from $\mathcal{H}'$ and $\mathcal{S}$ to the instance J' which results in instance J (with $\text{Feas}(J) \subseteq \text{Feas}(J')$), such that J is an extension of I.

Stage 1: LP. We begin by extending our notion of treewidth to matrices.

Definition 4.7 (Treewidth of a matrix)**.** *Given a matrix $\mathbf{A} \in \mathbb{Z}^{m \times n}$, we define its* Gaifman graph $G = G(\mathbf{A})$ *as follows. Let $V(G) = [n]$. Let $\{i, j\} \in E(G)$ if and only if there is an $r \in [n]$ with $\mathbf{A}[r, i] \neq 0$ and $\mathbf{A}[r, j] \neq 0$. The* (primal or Gaifman) treewidth of a matrix $\mathbf{A}$ *is then* $\text{tw}(\mathbf{A}) := \text{tw}(G(\mathbf{A}))$.

Let $P_\varphi(G) = \text{conv}\{\mathbf{y} \mid G, \mathbf{y} \models \varphi\}$ be the polytope of satisfying assignments of φ on G, also called the MSO *polytope*; $\text{conv}(X)$ denotes the convex hull of a set X. A result of Kolman et al. [32] shows that there exists a polytope closely related to $P_\varphi(G)$ with many useful properties:

Proposition 4.8 ([32, Theorem 4])**.** *Let $G = (V, E)$ be a σ_2-structure, let n be the size of the universe of G, $(T, \mathcal{B})$ be a nice tree decomposition of G of width τ, and φ be an MSO_2 formula with ℓ free variables.*

Then there exists an LP $\mathbf{Ay} + \mathbf{Dz} + \mathbf{Cw} = \mathbf{d}, \mathbf{z}, \mathbf{w} \geq 0$, a set $\mathcal{C}$, a function $\eta : \mathcal{C} \times V(T) \times V \times [\ell] \to \{0, 1\}$ and a tree decomposition $(T, \mathcal{B}^*)$ of the Gaifman graph $G(\mathbf{A}\ \mathbf{D}\ \mathbf{C})$ such that the following claims hold:

(1) *The polytope $P = \{(\mathbf{y}, \mathbf{z}, \mathbf{w}) \mid \mathbf{Ay} + \mathbf{Dz} + \mathbf{Cw} = \mathbf{d}, \mathbf{z}, \mathbf{w} \geq \mathbf{0}\}$ is a 0/1-polytope and $P_\varphi(G) = \{\mathbf{y} \mid \exists \mathbf{z}, \mathbf{w} : (\mathbf{y}, \mathbf{z}, \mathbf{w}) \in P\}$.*
(2) *For any integer point $(\mathbf{y}, \mathbf{z}, \mathbf{w}) \in P$, for any $t \in \mathcal{C}$, $b \in V(T)$, $v \in B(b)$ and $i \in [\ell]$, equalities $z_b^t = 1$ and $\eta(t, b, v, i) = 1$ imply that $y_v^i = 1$.*

(3) (a) *The treewidth of* $(T, \mathcal{B}^*)$ *is* $\mathcal{O}(|\mathcal{C}|^3)$,
(b) *for every node* $b \in V(T)$, $\bigcup_{t\in\mathcal{C}}\{z_b^t\} \subseteq B^*(b)$.
(4) *There is a computable function* f *such that* $|\mathcal{C}| \leq f(\tau, ||\varphi||)$ *and* $\mathbf{A}, \mathbf{D}, \mathbf{C}, \mathbf{d}, \eta$ *can be computed in time* $\mathcal{O}(|\mathcal{C}|^3 \cdot n)$.

Stage 2: Viewing ILP as CSP. By $\mathbf{a}_j$ we denote the j-th row of a matrix $\mathbf{A}$. Let us connect ILP and CSP:

Definition 4.9. *A CSP instance* I *is equivalent to an ILP* $\mathbf{Ax} \leq \mathbf{b}\, \mathbf{x} \in \mathbb{Z}^n$ *if*

$$\{\mathbf{x} \in \mathbb{Z}^n \mid \mathbf{Ax} \leq \mathbf{b}\} = \mathrm{Feas}(I).$$

Proposition 4.10. *Let* $\mathbf{Ax} \leq \mathbf{b}$, $\mathbf{x} \in \mathbb{Z}^n$ *be an ILP with* $\mathbf{A} \in \mathbb{Z}^{m\times n}$, $\tau = \max_{j=1}^{m} |supp(\mathbf{a}_j)|$ *and* $D = \max_{i=1}^{n} |u_i - \ell_i|$, *where* u_i *and* ℓ_i *are an upper and lower bound on* x_i, *that is,* $\mathbf{u} \leq \mathbf{x} \leq \boldsymbol{\ell}$ *holds. Then, an equivalent CSP instance* I *can be constructed in time* $\mathcal{O}(D^\tau \cdot mn)$, *and* $G(\mathbf{A}) = G(I)$.

Proof. Let $V = \{x_1, \ldots, x_n\}$. For every $i \in [n]$, let $D_i = [\ell_i, u_i]$ and $\mathcal{D} = \{D_i \mid i \in [n]\}$. Observe that $\max_i |D_i| = \|\mathbf{u} - \boldsymbol{\ell}\|_\infty = D$. Regarding hard constraints $\mathcal{H}$, observe that every row $\mathbf{a}_j$ of $\mathbf{A}$ contains at most τ non-zeros. Let $U_j = \mathrm{supp}(\mathbf{a}_j) = \{i_1, \ldots, i_k\}$, where $k \leq \tau$, and let $x_c = 0$ for all $c \notin U_j$. Let C_{U_j} be the set of assignments from $D_{i_1} \times \cdots \times D_{i_k}$ to $x_{i_1}, \ldots, x_{i_k}$ that satisfy $\mathbf{a}_j\mathbf{x} \leq b_j$; obviously $|C_{U_j}| \leq D^k$ and it can be constructed in time $\mathcal{O}(D^k)$. Then, $\mathcal{H} = \{C_{U_j} \mid j = 1, \ldots, m\}$. It is easy to verify that the feasible assignments of I correspond to integer solutions of $\mathbf{Ax} \leq \mathbf{b}$ and that $G(\mathbf{A}) = G(I)$. □

Proof of Theorem 4.6. We apply Proposition 4.8 to obtain an ILP $\mathbf{Ay} + \mathbf{Dz} + \mathbf{Cw} = \mathbf{d}$, and use Proposition 4.10 to get an equivalent CSP instance J'. Recall that $(T, \mathcal{B})$ is a nice tree decomposition of G and $(T, \mathcal{B}^*)$ is a tree decomposition of $G(\mathbf{A}\ \mathbf{D}\ \mathbf{C})$ (and thus $G(J')$) as described in Proposition 4.8, part (3). Let I' be a CSP instance over variables $\mathbf{y}$ with $\mathcal{H} = \{\mathbf{y} \mid G, \mathbf{y} \models \varphi\}$. Clearly, J' is an extension of I' by the fact that the polytope P is an extension of $P_\varphi(G)$.

Now we will add the variables $\mathbf{x}$ to J' and add constraints in such a way that the resulting instance J will be an extension of I, and that it satisfies the claim of Theorem 4.6.

Stage 3: Adding variables and constraints. We introduce auxiliary binary variables $f_v^{i,a}$ for each $a \in V(T)$, $i \in [\ell]$ and $v \in B(a)$, and we let $f_v^{i,a} = \sum_{t\in\mathcal{C}} z_a^t \cdot \eta(t, a, v, i)$. For any subset U of variables of I, let U_a be the set U where each variable y_v^i is replaced by $f_v^{i,a}$. Then, for every constraint $C_U \in \mathcal{H}'$ and $w_U \in \mathcal{S}$ let $a \in V(T)$ be the node in the definition of the local scope property which satisfies

$$\mathrm{supp}(\mathbf{s}) \subseteq \big(\{(v, i) \mid v \in B(a), i \in [\ell]\} \cup \{(b, j) \mid b \in N_T^{\downarrow}(a), j \in [m]\}\big),$$

and add to J' a constraint obtained by replacing the scope U with U_a. Denote the resulting instance J.

By property (2) of Proposition 4.8, $f_v^{i,a} = y_v^i$ for each $a \in V(T)$ with $v \in B(a)$. Thus, for any constraint C_U or w_U, replacing its scope with U_a does not change the set of feasible assignments, and J is an extension of I.

By the equivalence of $\mathbf{Ay} + \mathbf{Dz} + \mathbf{Cw} = \mathbf{d}$ and J', we have that $\|\mathcal{H}_{J'}\| + \|\mathcal{S}_{J'}\| \leq f(||\varphi||, \tau) \cdot n$, and thus $\|\mathcal{H}_J\| + \|\mathcal{S}_J\| \leq f(||\varphi||, \tau) \cdot n + \|\mathcal{H}'\| + \|\mathcal{S}\|$. The variables contained

in J and not I are the $\mathbf{z}, \mathbf{w}$ and $\mathbf{f}$ variables. Since they are all binary we also have that $D_J = D_I$. It remains to show that $\mathrm{tw}(J) \leq f(||\varphi||, \tau) + 2k$. A lemma will help us see that:

Lemma 4.11. *Let $T = (I, F)$ be a rooted binary tree, let $(T, \mathcal{B})$ be a tree decomposition of a graph $G = (V, E)$ of width κ, and let $H = (V \cup W, E \cup Y)$ be a supergraph of G such that:*

- *$W = \bigcup_{a \in I} W_a$, $Y = \bigcup_{ab \in F} Y_{ab}$, all W_a are pair-wise disjoint, and all Y_{ab} are pair-wise disjoint,*
- *$|W_a| \leq \kappa'$ for all $a \in I$,*
- *if $a \in I$ has only child b, then $\bigcup Y_{ab} \subseteq (B(a) \cup W_a \cup W_b)$, and,*
- *if a has two children b, b', then $\bigcup (Y_{ab} \cup Y_{ab'}) \subseteq (B(a) \cup W_a \cup W_b \cup W_{b'})$.*

Then there is a tree decomposition $(T, \mathcal{B}')$ of H of width at most $\kappa + 2\kappa'$.

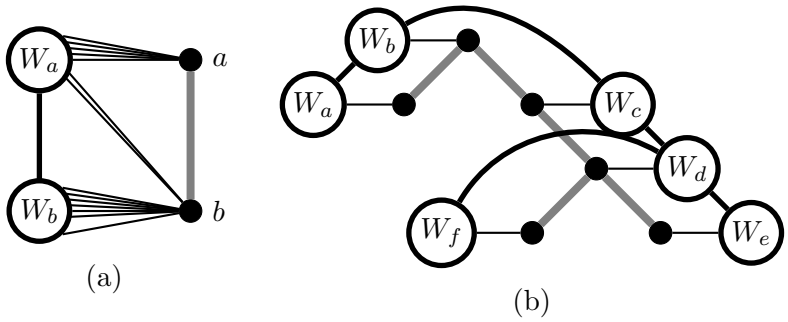

Figure 3: The situation of Lemma 4.11. Part (a) depicts a single edge $ab \in F$ and the requirement that edges from vertices in W_a only connect to vertices in W_a, W_b, $B(a)$ or $B(b)$. Part (b) depicts how W and Y relate to the whole of T. Black points correspond to I, grey edges to F, $W_a, \ldots, W_f$ are self-explanatory, and all remaining edges correspond to sets of edges Y_{ij}.

Proof. Let $\mathcal{B}'$ be obtained from $\mathcal{B}$ by, for every edge $ab \in F$, adding W_a to the bags $B(a)$ and $B(b)$. We will verify that $(T, \mathcal{B}')$ is a tree decomposition of H of width at most $\kappa + 2\kappa'$; we shall denote by B' the bags of $(T, \mathcal{B}')$. The conditions of a tree decomposition obviously hold for all vertices and edges of G, so we only check it for new vertices and edges.

Edge condition. Let $uv \in Y_{ab}$ be an edge in $H \setminus G$ with $u \in W_a$. Either $v \in B(a)$ and then $\{u, v\} \subseteq B(a) \cup W_a \subseteq B'(a)$, or $v \in W_b$ and then $\{u, v\} \subseteq B(b) \cup W_a \subseteq B'(b)$.

Connectedness condition. Let $v \in W_a$ and let a have children b, b', with possibly $b = b'$. Notice that v does not appear in the bag of any node above a and any node below b and b'. Since we have added W_a to all of a, b and b', the connectedness condition holds.

We have added to each node b (except the root) two sets W_a, W_b where a is the parent of b, and because $|W_a| + |W_b| \leq 2\kappa'$, $\mathrm{tw}((T, \mathcal{B}')) \leq \kappa + 2\kappa'$. □

Let us consider how the constraint graph $G(J)$ relates to $G(J')$. Since J is obtained by adding new variables and constraints, this corresponds to $G(J)$ being a supergraph of $G(J')$. The vertices $W = V(G(J)) \setminus V(G(J'))$ can be partitioned into sets W_a for every node $a \in V(T)$, and $|W_a| \leq k$. Moreover, the new edges $Y = E(G(J)) \setminus E(G(J'))$ can also be partitioned into sets Y_{ab} for each $ab \in E(T)$, such that for each $uv \in Y_{ab}$ we have

$\{u, v\} \subseteq (W_a \cup W_b)$, because, for each node $a \in V(T)$, the new constraints only contain variables associated with node a and its neighbors. The tree decomposition $(T, \mathcal{B}^*)$ of $G(J')$ is such that we are precisely in the situation of Lemma 4.11 with $G := G(J')$, $H := G(J)$, $\kappa := \mathrm{tw}(J') = f'(||\varphi||, \tau)$ and $\kappa' := k + \ell\tau$, which then implies that $G(J)$ has a tree decomposition $(T, \mathcal{B}')$ of width $f'(||\varphi||, \tau)) + 2k + \ell\tau \leq f(||\varphi||, \tau)) + 2k$. □

5. Conclusions

Limits of MSO *extensions, other logics, and metatheorems.* We have defined extensions of MSO and extended positive and negative results for them. There is still some unexplored space in MSO extensions: Szeider [41] shows that $\mathsf{MSO}^{\mathsf{L}}$ where some of the sets of local cardinality constraints are quantified is NP-hard already on graphs of treewidth 2. We are not aware of a comparable result for $\mathsf{MSO}^{\mathsf{G}}$, and no results of this kind are known for graphs of bounded neighborhood diversity. Also, we have not explored other logics, as for example the modal logic considered by Pilipczuk [40]. Also, one merit of algorithmic metatheorems is in generalizing existing results. However, many problems [19, 22] are FPT on bounded neighborhood diversity which are not expressible in any of the studied logics. So we ask for a metatheorem generalizing as many such positive results as possible.

Complementary Parameters and Problems. Unlike for treewidth, taking the complement of a graph preserves its neighborhood diversity. Thus our results apply also in the complementary setting, where, given a graph G and a parameter $p(G)$, we are interested in the complexity (with respect to $p(G)$) of deciding a problem P on the *complement* of G. While the complexity stays the same when parameterizing by neighborhood diversity, it is unclear for sparse graph parameters such as treewidth. It was shown very recently [17] that the Hamiltonian Path problem admits an FPT algorithm with respect to the treewidth of the complement of the graph. This suggest that at least sometimes this is the case and some extension of Courcelle's theorem deciding properties of the complement may hold.

Acknowledgements. This research was supported by the project 338216 of GA UK and the grant SVV–2017–260452. D. Knop acknowledges support by the OP VVV MEYS funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics". M. Koutecký was partially supported by a postdoctoral fellowship at the Technion funded by the Israel Science Foundation grant 308/18, by Charles University project UNCE/SCI/004, and by the project 17-09142S of GA ČR. T. Masařík was supported by the project GA17-09142S of GA ČR.

References

[1] Sancrey R. Alves, Konrad K. Dabrowski, Luérbio Faria, Sulamita Klein, Ignasi Sau, and Uéverton dos Santos Souza. On the (parameterized) complexity of recognizing well-covered (r, l)-graphs. In *COCOA*, volume 10043 of *Lecture Notes in Computer Science*, pages 423–437. Springer, 2016. doi:10.1007/978-3-319-48749-6.

[2] N.R. Aravind, Subrahmanyam Kalyanasundaram, Anjeneya S. Kare, and Juho Lauri. Algorithms and hardness results for happy coloring problems. 2017. arXiv:1705.08282.

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

[4] René van Bevern, Andreas E. Feldmann, Manuel Sorge, and Ondřej Suchý. On the parameterized complexity of computing balanced partitions in graphs. *Theory of Computing Systems*, 57(1):1–35, 2015. doi:10.1007/s00224-014-9557-5.

[5] Hans L. Bodlaender. A linear time algorithm for finding tree-decompositions of small treewidth. In *STOC*, pages 226–234, 1993. doi:10.1145/167088.167161.

[6] Hans L. Bodlaender. Treewidth: characterizations, applications, and computations. In *WG*, volume 4271 of *Lecture Notes in Computer Science*, pages 1–14. Springer, 2006. doi:10.1007/11917496_1.

[7] Édouard Bonnet and Florian Sikora. The graph motif problem parameterized by the structure of the input graph. *Discrete Applied Mathematics*, 231:78–94, 2017. doi:10.1016/j.dam.2016.11.016.

[8] Robert Bredereck, Piotr Faliszewski, Rolf Niedermeier, Piotr Skowron, and Nimrod Talmon. Elections with few candidates: Prices, weights, and covering problems. In *ADT*, volume 9346 of *Lecture Notes in Computer Science*, pages 414–431, 2015. doi:10.1007/978-3-319-23114-3_25.

[9] Michele Conforti, Gérard Cornuéjols, and Giacomo Zambelli. Extended formulations in combinatorial optimization. *Annals OR*, 204(1):97–143, 2013. doi:10.1007/s10479-012-1269-0.

[10] Bruno Courcelle. The monadic second-order logic of graphs I: Recognizable sets of finite graphs. *Information and Computation*, 85:12–75, 1990. doi:10.1016/0890-5401(90)90043-H.

[11] Bruno Courcelle. The monadic second-order logic of graphs VI: On several representations of graphs by relational structures. *Discrete Applied Mathematics*, 54(2–3):117–149, 1994. doi:10.1016/0166-218X(94)90019-1.

[12] Bruno Courcelle. The monadic second-order logic of graphs XIV: Uniformly sparse graphs and edge set quantifications. *Theoretical Computer Science*, 299(1–3):1–36, 2003. doi:10.1016/S0304-3975(02)00578-9.

[13] Bruno Courcelle, Johann A. Makowsky, and Udi Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory of Computing Systems*, 33(2):125–150, 2000. doi:10.1007/s002249910009.

[14] Bruno Courcelle and Mohamed Mosbah. Monadic second-order evaluations on tree-decomposable graphs. *Theoretical Computer Science*, 109(1–2):49–82, 1993. doi:10.1016/0304-3975(93)90064-Z.

[15] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

[16] Pavel Dvořák, Dušan Knop, and Tomáš Toufar. Target set selection in dense graph classes. In *ISAAC*, volume 123 of *LIPIcs*, pages 18:1–18:13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018. doi:10.4230/LIPIcs.ISAAC.2018.18.

[17] Pavel Dvořák, Dušan Knop, and Tomáš Masařík. Anti-path cover on sparse graph classes. In *MEMICS*, volume 233 of *Electronic Proceedings in Theoretical Computer Science*, pages 82–86. Open Publishing Association, 2016. doi:10.4204/EPTCS.233.8.

[18] Michael R. Fellows, Guillaume Fertin, Danny Hermelin, and Stéphane Vialette. Upper and lower bounds for finding connected motifs in vertex-colored graphs. *Journal of Computer and System Sciences*, 77(4):799–811, 2011. doi:10.1016/j.jcss.2010.07.003.

[19] Jiří Fiala, Tomáš Gavenčiak, Dušan Knop, Martin Koutecký, and Jan Kratochvíl. Parameterized complexity of distance labeling and uniform channel assignment problems. *Discrete Applied Mathematics*, 248:46–55, 2018. doi:10.1016/j.dam.2017.02.010.

[20] Eugene C. Freuder. Complexity of K-tree structured constraint satisfaction problems. In *AAAI*, pages 4–9. AAAI Press, 1990. URL: http://dl.acm.org/citation.cfm?id=1865499.1865500.

[21] Markus Frick and Martin Grohe. The complexity of first-order and monadic second-order logic revisited. *Annals of Pure and Applied Logic*, 130(1–3):3–31, 2004. doi:10.1016/j.apal.2004.01.007.

[22] Robert Ganian. Using neighborhood diversity to solve hard problems. 2012. arXiv:1201.3091.

[23] 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. doi:10.1007/978-3-642-45278-9.

[24] Luisa Gargano and Adele A. Rescigno. Complexity of conflict-free colorings of graphs. *Theoretical Computer Science*, 566:39–49, 2015. doi:10.1016/j.tcs.2014.11.029.

[25] Georg Gottlob, Reinhard Pichler, and Fang Wei. Monadic datalog over finite structures with bounded treewidth. In *PODS*, pages 165–174, 2007. doi:10.1145/1265530.1265554.

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

[27] Ton Kloks. *Treewidth: Computations and Approximations*, volume 842 of *Lecture Notes in Computer Science*. Springer, 1994. doi:10.1007/BFb0045375.

[28] Joachim Kneis, Alexander Langer, and Peter Rossmanith. Courcelle's theorem – A game-theoretic approach. *Discrete Optimization*, 8(4):568–594, 2011. doi:10.1016/j.disopt.2011.06.001.

[29] Dušan Knop, Martin Koutecký, Tomáš Masařík, and Tomáš Toufar. Simplified algorithmic metatheorems beyond MSO: treewidth and neighborhood diversity. In *WG*, volume 10520 of *Lecture Notes in Computer Science*, pages 344–357. Springer, 2017. doi:10.1007/978-3-319-68705-6_26.

[30] Dušan Knop, Tomáš Masařík, and Tomáš 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. doi:10.4230/LIPIcs.MFCS.2019.33.

[31] Phokion G. Kolaitis and Moshe Y. Vardi. Conjunctive-query containment and constraint satisfaction. *Journal of Computer and System Sciences*, 61(2):302–332, 2000. doi:10.1006/jcss.2000.1713.

[32] Petr Kolman, Martin Koutecký, and Hans R. Tiwary. Extension complexity, MSO logic, and treewidth, 2016. Short version presented at SWAT 2016. URL: http://arxiv.org/abs/1507.04907.

[33] Petr Kolman, Bernard Lidický, and Jean-Sébastien Sereni. On Fair Edge Deletion Problems, 2009. URL: https://kam.mff.cuni.cz/~kolman/papers/kls09.pdf.

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

[35] Michael Lampis. Model checking lower bounds for simple graphs. *Logical Methods in Computer Science*, 10(1):1–21, 2014. doi:10.2168/LMCS-10(1:18)2014.

[36] Hendrik W. Lenstra Jr. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):538–548, 1983. doi:10.1287/moor.8.4.538.

[37] Leonid Libkin. *Elements of Finite Model Theory*. Springer-Verlag, Berlin, 2004. doi:10.1007/978-3-662-07003-1.

[38] Tomáš Masařík and Tomáš Toufar. Parameterized complexity of fair deletion problems. *Discrete Applied Mathematics*, in press, 2019. doi:10.1016/j.dam.2019.06.001.

[39] Jiří Matoušek and Jaroslav Nešetřil. *Invitation to Discrete Mathematics (2. ed.)*. Oxford University Press, 2009.

[40] Michał 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. doi:10.1007/978-3-642-22993-0.

[41] Stefan Szeider. Monadic second order logic on graphs with local cardinality constraints. *ACM Transactions on Computational Logic*, 12(2):12:1–12:21, 2011. doi:10.1145/1877714.1877718.