

MASARYK UNIVERSITY
FACULTY OF INFORMATICS



Graphs, Their Width, and Logic

HABILITATION THESIS
(Collection of Articles)

Jan Obdržálek

Brno, May 2017

Acknowledgement

First, I would like to thank my co-authors and colleagues over the years. I have much appreciated the enriching experience of our research collaboration. Secondly, I would like to thank my Brno colleagues Vojtěch Řehák and Tomáš Brázdil, for many stimulating discussions not only about research, but also about teaching and many other topics. I would especially like to thank Petr Hliněný for his guidance, support, and giving me the opportunity to learn from him, and his former students Robert Ganian and Jakub Gajarský who taught me much about doing research. I would also like to thank Antonín Kučera for giving me the opportunity to stay in academia in Brno. Finally, but most importantly, my heartfelt gratitude goes to my wife Martina and our children Hana, Vítěk and Šimon, for their love, patience and unconditional support.

Abstract

Graphs are a formalism well suited to model many types of relations and processes from various areas – from road networks to computer programs to complex biological systems. Having a graph as a model, we would naturally like to solve problem like finding a shortest path between two vertices in the case of the road networks. For many of these problems, like the shortest path, there are indeed well-known efficient algorithms. However, many other problems on graphs are NP-hard or worse.

One way to obtain efficient algorithms for such problems is to restrict the class of input graphs. Graph width measures, the best known example being treewidth, provide us with just such a restriction. The famous result of Courcelle states that every property expressible in monadic second-order logic is decidable in linear time on graphs of bounded treewidth. Results like Courcelle’s theorem are called often *algorithmic meta-theorems*, as they can quickly tell us that a problem can be solved efficiently.

In this thesis I describe some of my contributions to the area of algorithmic meta-theorems and graph width measures. Each algorithmic meta-theorem has two ingredients: a logic and a class of structures. The logics we are interested in this thesis are the well known first-order and monadic second-order logics. As for the input structures, the research included in this thesis branches into two directions. The first one is the area of width measures, where one of my main goals has been to find a good width measure for directed graphs. Parity games, which can be viewed as labelled directed graphs and are closely related to yet another logic, modal μ -calculus, are also discussed. The other branch of research is concerned with obtaining efficient results for dense graphs, i.e. graphs with “many” edges. Unlike for sparse graphs (and graphs of bounded treewidth are sparse), not much is known about dense graph model checking.

The thesis is structured as a collection of published articles with a commentary. This commentary serves multiple purposes. It briefly introduces the research area, places our results in context of earlier (or, where appropriate, later) works, and summarizes my contribution.

Contents

I COMMENTARY	1
1 Introduction	3
2 Preliminaries	7
2.1 <i>Graphs</i>	7
2.2 <i>Logics on graphs</i>	7
2.3 <i>Model checking and parameterized complexity</i>	9
2.4 <i>LinEMSO optimization problems</i>	11
2.5 <i>Modal μ-calculus and parity games</i>	12
3 Undirected graph width measures	15
3.1 <i>Treewidth</i>	15
3.1.1 <i>Game characterization of treewidth</i>	16
3.2 <i>Clique-width and Rank-width</i>	17
3.3 <i>MSO lower bounds</i>	19
3.3.1 <i>MSO₂ lower bounds</i>	19
3.3.2 <i>MSO₁ lower bounds</i>	21
3.4 <i>Undirected width as a measure of directed graphs</i>	22
4 Directed graph width measures	25
4.1 <i>Early cops-and-robber based measures</i>	25
4.2 <i>DAG-width</i>	26
4.3 <i>Kelly-width – an improvement on DAG-width</i>	28
4.4 <i>Hardness of MSO₁ model checking</i>	29
4.5 <i>Measures with efficient MSO₁ model checking</i>	30
5 Existence of powerful digraph width measures	35
5.1 <i>Characterizing game definable measures</i>	35
5.2 <i>Non-existence of ideal measures</i>	37
5.3 <i>From topological minors to subdigraphs</i>	38
6 FO logic on dense graphs	39
6.1 <i>Interval graphs</i>	40
6.2 <i>Posets and existential FO</i>	42
6.3 <i>Posets FO</i>	44
6.4 <i>Interpreting dense graph classes</i>	45

Bibliography	49
II COLLECTION OF ARTICLES	57
Articles in the collection	59

PART I
COMMENTARY

1 Introduction

In mathematics, graph theory considers graphs to be relational structures consisting of objects, called *vertices*, any two of which may be in some sense pairwise “related” (this relationship is modelled by putting an *edge* between two such vertices). Graphs come in many forms and flavours, depending on the kind of relationship we intend to model – they can be directed or undirected, edge or vertex labelled etc. Over the time graphs proved to be an ideal formalism to model many types of relations and processes in various fields, including computer science, biology, chemistry, social sciences, and electrical engineering.

To give just a few examples, in chemistry graphs can naturally model molecules, where atoms are vertices and edges the bonds between them. In sociology, graphs are used to model and study social networks. In chip design graphs model the different components of a chip and the signal paths between them, whereas in travel graphs can describe road or flight networks.

But modelling and description is only the first part. Once we have the model, we would like to know answers to questions like “what is the fastest a combination of at most three connecting flights which will get me from Brno, Czech Republic, to Huntsville, Alabama?” Finding efficient algorithms to answer such questions is one of the primary goals of computer science.

For many problems definable on graphs we know elegant and efficient algorithms, which perform well in practice. Among the well-known algorithms of this kind are, e.g., the algorithms for finding the minimum spanning tree (Borůvka’s algorithm from 1926 [12] can be made to run in time $\mathcal{O}(m \cdot \log n)$, where n is the number of vertices and m the number of edges), or algorithms for finding single source shortest paths (Dijkstra’s algorithm can be made to run in time $\mathcal{O}(m + n \cdot \log n)$). See, e.g., [18] for a reference.

Unfortunately, many other graph problems turn out to be NP-hard [43]. Some of the better known problems in this category are problems like LONGEST PATH (finding a simple path of maximum length), VERTEX COVER (finding a set of vertices such that each edge is incident to at least one vertex in this set), DOMINATING SET (finding a set of vertices such that each vertex not in this set is adjacent to a vertex in the set) or HAMILTONIAN PATH (finding a path which visits each vertex exactly once), itself a special case of the well known TRAVELLING SALESMAN PROBLEM. Their NP-hardness suggests that most likely there are no efficient algorithms for these problems.

Fortunately, the situation is not so bleak. If, instead of all graphs, one restricts the input instances to just trees, many of these problems become trivially solvable in polynomial time. This intuition was generalized in the very successful concept of graph *treewidth*, which was introduced in the context of the Graph Minors project by Robertson and Seymour [86, 87]. Almost immediately after its introduction, treewidth turned out to be very useful for efficiently solving many graph problems (including NP-hard ones). In a nutshell, treewidth measures tree-likeness of a graph. Trees themselves, for example, have treewidth one and series-parallel graphs have treewidth two. Many graphs occurring in practical applications have small treewidth. This comes as no big surprise as one often deals with hierarchical structures that are inherently similar to trees. Examples include problems in VLSI design, evolution theory, interval routing, and the control-flow graphs of structured programs. See [8, 10, 11, 51] for surveys. Following on the heels of treewidth came many other “width measures” for undirected graphs, the most successful ones probably being clique-width [20, 22] and rank-width [83].

While treewidth has been very successful, it is a property of *undirected graphs*. What about directed graphs, which often occur in practice? One can, of course, just forget the orientation of edges and apply the concept of treewidth directly to digraphs. However, with this simplistic approach we seem to ignore too much. For example, in directed acyclic graphs (DAGs) it is easy to find a longest path. Nevertheless, DAGs have unbounded treewidth if we forget the edge directions.

In the search for a “truly directed” width measure inspired by treewidth several suggestions have been made, starting with directed treewidth [55], and being complemented few years later by several new approaches including directed path-width [3], entanglement [6], D-width [88], DAG-width [81, 4, 5] (defined by the author of this thesis) and Kelly-width [52].

Among the many problems which one may want to solve on a class of graphs the *model checking* problem, which asks whether a logical formula holds true on a given graph, holds a special place as a fundamental problem in computer science. The reason is that finding an efficient algorithm for the model checking problem automatically establishes efficient solvability for whole class of problems – those expressible in the given logic. Results of this kind are nowadays called *algorithmic meta-theorems* – see [64] for a survey.

For the well-known first-order logic (FO) the model checking problem is known to be PSPACE-complete when the formula is part of the input, and polynomial-time solvable when the formula is fixed in advance. However, once restricted to graphs of bounded treewidth, the situation changes dramatically. Famous theorem of Courcelle [19] states that every property

definable in monadic second-order (MSO) logic (which subsumes FO) can be checked in linear time on graphs of bounded treewidth. Another well-known result, by Courcelle, Makowski and Rotics [21], asserts that the same conclusion holds for graphs of bounded clique-width when quantification is restricted to vertices and their subsets.

The question is for which graph classes (or, more generally, classes of input structures) can we obtain similarly efficient algorithms as in the two cases above? Both these algorithms run in time $f(|\varphi|) \cdot n^c$, where φ is the input formula and c is a constant independent of the formula. We call such algorithms *fixed-parameter tractable* (FPT). (See [26] for a thorough treatment of parameterized complexity.) Over the past decade this line of research has been very active and led to several important results on (mainly) undirected graphs, which culminated in the recent result of Grohe, Kreutzer and Siebertz [49] stating that FO model checking is fixed-parameter tractable on all nowhere dense classes of graphs, the most general known class of sparse graphs.

On the other hand not much is known about the complexity of the FO model checking problems on dense classes of graphs. However, with the completion of the sparse graphs project by [49], this field has recently started to attract the attention of researchers [40, 41, 13, 36, 34, 35].

In addition to the model checking problem for FO and MSO, another problem has a special place in this thesis – the problem of solving (finding the winner) parity games. These are infinite, two player games played on a directed graph labelled by natural numbers. See Section 2.5 or [80, 46] for more detail. There are several reasons why parity games deserve their place. First, the problem of solving parity games is equivalent to the μ -calculus model checking problem. μ -calculus is one of the most important logics in systems verification, subsuming most of the other logics used for this purpose. Moreover μ -calculus is the bisimulation-invariant fragment of MSO. Second reason is the intriguing complexity-theoretic status of the problem. We know that solving parity games is in $\text{NP} \cap \text{co-NP}$, but membership in P is unknown. This has been a long-standing important open problem in the area. Thirdly, parity games are played on labelled digraphs, and therefore the task of finding a winner is a natural digraph problem. The fourth reason is that this problem was instrumental to stimulating research into digraph width measures – the introduction of both DAG-width [81, 4, 5] and Kelly-width [52] was motivated by the need to find a large class of digraphs on which parity games can be solved in polynomial time. The final reason for including the problem of solving parity games in this thesis is a personal

one: it was this problem which gradually made me switch to doing research in the areas described in this thesis.

Thesis organization and my contribution We start with Chapter 2 which introduces basic concepts and terminology of graphs and logic required in the rest of the thesis. In Chapter 3 we introduce the undirected width measures of treewidth and clique-width, including our results on lower-bounds for MSO_1 model checking [42]. A polynomial algorithm for parity games on graphs of bounded treewidth [79] is also presented there.

Next we turn our attention to width measures for directed graphs. Chapter 4 discusses the most important digraph width measures, with a focus on my notion of DAG-width [81, 5]. Also the hardness of MSO_1 model checking on directed graphs [38] is discussed here, together with an algorithm for parity games on graphs of bounded (directed) clique-width [82]. Since neither DAG-width nor other digraph width measures have all the nice properties of treewidth, in Chapter 5, recounting results from [39], we investigate whether any such measure can actually exist.

Finally in Chapter 6 we discuss the current state in the FO model checking of dense graphs, showing the existence of FPT algorithms for interval graphs, partially ordered sets, and graphs interpretable in the class of graphs of bounded degree. This chapter contains our recent results for interval graphs [41], partially ordered sets [36, 34] and graphs interpretable in graphs of bounded degree [35].

2 Preliminaries

2.1 Graphs

We use standard graph theoretic notation (see, e.g., [25]). The graphs (both undirected and directed) that we consider in this thesis are *simple*, i.e. they do not contain loops and parallel edges. Given a graph G , we let $V(G)$ denote its vertex set and $E(G)$ denote its edge set, if G is undirected. We usually denote a directed graph (*digraph*) by D and its arc set by $A(D)$. Given a directed graph D , the *underlying undirected graph* $U(D)$ of D is an undirected graph on the vertex set $V(D)$ and $\{u, v\}$ is an edge of $U(D)$ if and only if either $(u, v) \in A(D)$ or $(v, u) \in A(D)$. A digraph D is an *orientation* of an undirected graph G if $U(D) = G$. If H is a sub(di)graph of G , then we denote this fact by writing $H \subseteq G$.

For a vertex pair u, v of a digraph D , a sequence $P = (u = x_0, \dots, x_r = v)$ is called *directed (u, v) -path* of length $r > 0$ in D if the vertices x_0, \dots, x_r are pairwise distinct and $(x_i, x_{i+1}) \in A(G)$ for every $0 \leq i < r$. A *directed cycle* is defined analogously with the modification that $x_0 = x_r$. A digraph D is *acyclic* (DAG) if D contains no directed cycle.

2.2 Logics on graphs

In this section we informally introduce the logics we will consider in this thesis. For a more thorough and formal treatment we refer the reader to, e.g., [28].

In order to express graph properties using logic, we shall view graphs as finite *relational structures* (i.e. without functional symbols) consisting of a finite set, the *domain*, and of a collection of relations over the domain. A simple graph G can be represented as a relational structure in two major ways; either as the domain $V(G)$ with the binary *adjacency relation* between its vertices, or with the domain formed by a disjoint union of $V(G)$ and $E(G)$ and the binary relation being the *incidence* between vertices and edges. For the logics FO and MSO_1 we will consider the former, while the latter will be useful in the context of the MSO_2 logic.

We start by defining the first-order logic (FO) of undirected graphs first.

Definition 2.1 (FO language). *The language of FO on graphs contains the logical expressions that are built from the following elements:*

- variables x, y, x_1, \dots, x_k for elements (vertices)
- the adjacency predicate $\text{adj}(x, y)$,

- equality for variables, the connectives $\wedge, \vee, \neg, \rightarrow$,
- vertex quantification $\forall x, \exists x$.

Notice that FO can deal with both graphs and digraphs, simply depending on whether the adjacency relation is symmetric or not. For clarity, we replace the relational symbol $adj(u, v)$ with $arc(u, v)$ when dealing explicitly with digraphs.

The set of well-formed FO formulas and the semantics are given in the usual way. A decision graph property \mathcal{P} is FO-definable if there exists an FO formula φ such that \mathcal{P} holds for an arbitrary graph G if and only if $G \models \varphi$, i.e., φ holds on G .

Example 2.2. For undirected graphs, the property of each vertex having a degree at most two can be expressed by the following closed FO formula

$$\forall x, y_1, y_2, y_3. adj(x, y_1) \wedge adj(x, y_2) \wedge adj(x, y_3) \implies y_1 = y_2 \vee y_2 = y_3 \vee y_3 = y_1$$

An example of a graph property which is not FO-definable is the property of being 3-colourable. To increase the expressive power, we introduce a restricted version of second-order quantification – over the subsets of the domain. This way we obtain so called *monadic second-order* (MSO) logic. Depending on the domain, there are two variants of this logic: MSO_1 and MSO_2 .

We start with the logic MSO_1 first. This logic extends the FO logic by allowing quantification over the sets of vertices, in addition to quantification over vertices (and therefore also edges) already present in FO.

Definition 2.3 (MSO_1 language). *The language of MSO_1 on graphs contains the logical expressions that are built from the following elements:*

- variables x, y, x_1, \dots, x_k for elements (vertices)
- variables X, Y, X_1, \dots, X_k for sets of vertices
- the predicates $adj(x, y)$ and $x \in X$,
- equality for variables, the connectives $\wedge, \vee, \neg, \rightarrow$,
- vertex quantification $\forall x, \exists x$ and vertex set quantification $\forall X, \exists X$.

The semantics and other notions are again defined in the usual way. The greater expressive power of MSO_1 allows us to define properties which are not FO-definable:

Example 2.4. For undirected graphs, the property of being 3-colourable can be expressed by the MSO_1 formula

$$\exists V_1, V_2, V_3 [\forall v (v \in V_1 \vee v \in V_2 \vee v \in V_3) \wedge \bigwedge_{i=1,2,3} \forall v, w (v \notin V_i \vee w \notin V_i \vee \neg adj(v, w))].$$

The most general extension of FO we consider is the MSO_2 logic. Here we allow quantification not only over sets of vertices, but also over sets of edges. To do so, we must extend our domain to also contain the edges (as discussed above). In addition, we swap the predicate $\text{adj}(x, y)$, expressing that two vertices are adjacent, for the incidence relation predicate $\text{inc}(x, e)$:

Definition 2.5 (MSO_2 language). *The language of MSO_2 on graphs contains the logical expressions that are built from the following elements:*

- variables $x, y, x_1, \dots, x_k, e, e_1, \dots, e_l$ for elements (vertices and edges)
- variables X, Y, X_1, \dots, X_k for sets of vertices and edges
- the predicates $\text{inc}(x, e)$ and $x \in X$,
- equality for variables, the connectives $\wedge, \vee, \neg, \rightarrow$,
- element quantification $\forall x, \exists x$ and set quantification $\forall X, \exists X$.

As an example of a property not expressible in MSO_1 , but expressible in MSO_2 , is existence of a Hamiltonian path. (The basic idea behind the formula expressing Hamiltonicity is stating that there exists a set of edges forming a path and visiting each vertex exactly once.)

Logics with labels Sometimes it is useful to consider graphs equipped with *labels* (sometimes called *colours*) from some finite set Γ . (The labels can be attached to vertices and/or edges.) In logic these labels can be naturally modelled by unary predicates. We use the notation $\text{MSO}_2\text{-}\Gamma$ to denote such an extension of the logic MSO_2 .

2.3 Model checking and parameterized complexity

Since we can use logic to express graph properties, the obvious problem is whether we can algorithmically check whether a graph satisfies given formula. I.e., given a graph G and a formula φ , we would like to determine whether $G \models \varphi$. Here $G \models \varphi$ is read as “ G models φ ” or “ φ holds on G ”, and the process of testing this statement is called *model checking*.

Definition 2.6 (The model checking problem). *The model checking problem for a logic \mathcal{L} and a class of structures \mathcal{C} is the following decision problem: Given a formula φ of \mathcal{L} and a graph $G \in \mathcal{C}$, decide whether $G \models \varphi$. We denote this problem $\text{MC}(\mathcal{L}, \mathcal{C})$. If \mathcal{C} is the class of all graphs, we write just $\text{MC}(\mathcal{L})$.*

Most of the research on algorithms for FO model checking has focused on graphs. For general graphs there is a naive brute-force algorithm that

takes as an input an n -vertex graph G and a formula φ and determines whether $G \models \varphi$ in time $n^{O(|\varphi|)}$ by enumerating all the possible ways to instantiate the variables of φ . On the other hand, the problem is PSPACE-complete (see e.g. [45]) and encodes the CLIQUE problem¹. Therefore the problem admits no algorithm with running time $f(\varphi)n^{o(|\varphi|)}$ for any function f [71], assuming the Exponential Time Hypothesis [53] (ETH). Thus, assuming the ETH, the naive algorithm is the best possible, up to constants in the exponent. Furthermore, FO model checking remains PSPACE-complete on any fixed graph containing at least two vertices (again, see [45]). Hence, it is futile to look for restricted classes of graphs in which FO model checking can be done in polynomial time without restricting φ .

For that reason research has focused on obtaining algorithms with running time $f(\varphi)n^{O(1)}$ on restricted classes of graphs (and other structures). While such algorithms are not polynomial-time algorithms, due to unlimited f , they nevertheless significantly outperform brute-force. To capture these differences in running times it is handy to make use of the framework of parameterized complexity.

Parameterized complexity Following Downey–Fellows [26], a parameterized problem \mathcal{Q} is defined as a subset of $\Sigma \times \mathbb{N}_0$, where Σ is a finite alphabet and $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$. The following is folklore:

Definition 2.7 (parameterized tractability, [26]). *A parameterized problem \mathcal{Q} is fixed-parameter tractable if there is an algorithm that given $\langle x, k \rangle \in \Sigma \times \mathbb{N}_0$ decides whether $\langle x, k \rangle$ is a yes-instance of \mathcal{Q} in time $f(k) \cdot p(|x|)$, where f is some computable function of the parameter k alone, p is a polynomial and $|x|$ is the size measure of the input. The class of fixed-parameter tractable problems is denoted by FPT.*

Furthermore, let XP denote the class of parameterized problems \mathcal{Q} that admit an algorithm running in time $\mathcal{O}(|x|^{f(k)})$ for some computable function f , i.e. with polynomial run-time for every fixed value of the parameter k . We refer to such an algorithm for \mathcal{Q} as to an XP-time algorithm with respect to k .

Looking back at the FO model checking problem, in the parameterized setting it can be expressed as follows:

<p>MC(FO, \mathcal{C})</p> <p>Input: A first-order sentence φ and a graph $G \in \mathcal{C}$.</p> <p>Question: Is it true that $G \models \varphi$, i.e., is G a model of φ?</p>	<p>Parameter: φ</p>
---	---

1. To determine, given a graph G and $k \in \mathbb{N}$, whether G contains a clique of size k .

For a logic \mathcal{L} and a graph class \mathcal{C} the problem $\text{MC}(\mathcal{L}, \mathcal{C})$ is

- a) in FPT if there is an algorithm that for $G \in \mathcal{C}$ decides whether $G \models \varphi$ in time $f(|\varphi|) \cdot n^c$, where $n = |G|$ and c is a constant independent of φ , or
- b) in XP if such algorithm runs in time $\mathcal{O}(n^{f(|\varphi|)})$.

2.4 LinEMSO optimization problems

The model checking problem is a *decision* problem – given a graph G and a formula φ , we ask whether $G \models \varphi$. However, in practice the problems we want to solve are actually optimization problems (e.g. computing maximum dominating set). With a bit of extra work, logic can be used to describe such problems. Using the logic MSO, one obtains e.g. the class of LinEMSO optimization problems [2, 21]. Here we present the simpler definition of LinEMSO₁ as given in [21], while the LinEMSO₂ is defined along the same lines. Consider any MSO₁ formula $\psi(X_1, \dots, X_p)$ with free set variables, and state the following problem on an input (di)graph G :

$$\text{opt} \{ f_{lin}(W_1, \dots, W_p) : W_1, \dots, W_p \subseteq V(G), G \models \psi(W_1, \dots, W_p) \},$$

where *opt* can be min or max, and f_{lin} is a linear evaluation function. It is

$$f_{lin}(W_1, \dots, W_p) = \sum_{i=1}^p \sum_{j=1}^m \left(a_{i,j} \cdot \sum_{x \in W_i} f_j(x) \right) \quad (2.1)$$

where m and $a_{i,j}$ are (integer) constants and f_j are (integer) weight functions on the vertices of G . Typically f_{lin} is just the cardinality function.

Example 2.8. *The MAXIMUM INDEPENDENT SET problem can be expressed as*

$$\psi(X) \equiv \forall v, w (v \notin X \vee w \notin X \vee \neg \text{edge}(v, w))$$

with $f_{lin}(X) = \max(|X|)$.

Example 2.9. *The MINIMUM DIRECTED DOMINATING SET problem can be expressed as*

$$\psi(X) \equiv \forall z (z \in X \vee \exists x \in X. \text{arc}(x, z))$$

with $f_{lin}(X) = \min(|X|)$.

2.5 Modal μ -calculus and parity games

Modal μ -calculus, introduced by Kozen [63], is an extension of propositional modal logic with least fixpoint operator μ and greatest fixpoint operator ν . (We do not give a full definition here and refer the reader to [46] for a thorough treatment.) It is used to describe properties of labelled transition systems² (e.g. safety, liveness or termination) and can therefore be used in modelling and verification of both software and hardware systems. Its importance is further signified by the fact that it subsumes most of the other temporal logics used for these purposes – e.g. CTL* and its fragments LTL and CTL (see, e.g., [17, 46]). Modal μ -calculus can also be evaluated on graphs, using the well known Kripke semantics. As to its expressive power, it has been shown that μ -calculus is the bisimulation-invariant fragment of MSO [54].

Example 2.10. *The formula φ below is true in every vertex (state) s.t. all paths starting at this vertex contain a vertex labelled p :*

$$\mu Z. p \vee \square Z$$

The problem of determining, given a system \mathcal{A} and a μ -calculus formula φ , whether or not \mathcal{A} satisfies φ can be turned into a *parity game* – see e.g. [46, 80] (the converse is also true). Parity game is an infinite two-player game played on a directed graph where the vertices are labelled by priorities (from the set \mathbb{N}). The players take turns pushing a token along edges of the graph. The winner is determined by the parity of the least priority occurring infinitely often in this infinite play. We refer the reader to author’s thesis [80] or [46] for a thorough treatment of parity games.

The exact complexity of solving parity games is an open problem that has received a large amount of attention. It is known [29] that the problem is in $\text{NP} \cap \text{co-NP}$ (even $\text{UP} \cap \text{co-UP}$ [56]), but no polynomial-time algorithm is known. The early algorithms, e.g. [74], had an exponential run-time dependence on the number of priorities. There were several improvements (e.g. [57]), but it was only in 2003 when the first truly sub-exponential algorithm appeared – the randomized algorithm by Björklund et al. [7], with the complexity bounded by $2^{\mathcal{O}(\sqrt{n \log n})}$. In 2006 Jurdziński et al. [58] came with a very simple deterministic sub-exponential algorithm of complexity roughly matching the upper bound of the randomized algorithm. Lacking

2. Labelled transition systems are basically directed graph with edges labelled by a finite set of labels. One way to think about them is consider them as graph with multiple edge relations, one per each label.

any substantial progress research in this area focused on various subclasses of game graphs (see, e.g., our contribution in [79, 5, 82]). A very recent significant progress is the first known FPT (when parameterized by the number of priorities) algorithm for parity games by Calude et al. [15]. However the existence of a polynomial-time algorithm is still open.

3 Undirected graph width measures

3.1 Treewidth

If one looks for graph classes where MSO model checking can be done efficiently, it is perhaps natural to start with the class of all trees (for which the efficient algorithm is straightforward). The next step is to consider “tree-like” graphs, i.e. graphs which are sufficiently close to being a tree to allow for efficient model checking. The parameter of *treewidth*, introduced in the context of the Graph Minors project by Robertson and Seymour [86, 87], turned out to be just such a concept.

Definition 3.1 (treewidth, [86]). *A tree-decomposition of a graph G is a pair (T, \mathcal{X}) , where T is a tree and $\mathcal{X} = (X_t)_{t \in V(T)}$ is a family of subsets of V (called “bags”) such that*

- $\bigcup_{d \in V(D)} X_d = V$.
- for each edge $e = uv \in E(G)$, there is $t \in V(T)$ such that $\{u, v\} \subseteq X_t$,
- if $t \in V(T)$, and if $t', t'' \in V(T)$ are two nodes in distinct components of $T - t$, then $X_{t'} \cap X_{t''} \subseteq X_t$ (“interpolation”),

The width of (T, \mathcal{X}) is defined as $\max\{|X_t| - 1 \mid t \in V(T)\}$. The smallest width over all tree-decompositions of the graph G is the treewidth $tw(G)$ of G .

It is not hard to check that the class of graphs of treewidth 1 is the class of all forests, and the class of graphs of treewidth 2 is the well known class of series-parallel graphs. To generalize this intuition, we can say the lower the treewidth, the closer is a graph to being a tree.

We say that class of graphs \mathcal{C} has *bounded treewidth*, if there is a $k \in \mathbb{N}$ such that for all $G \in \mathcal{C}$ we have $tw(G) \leq k$ (we will implicitly assume a similar definition for all other measures). Bodlaender proved in [9] that the treewidth of a graph (and the witnessing tree-decompositions) can be computed in FPT time.

Theorem 3.2 ([9]). *There is an algorithm which, given a graph G , constructs a tree-decomposition of $k = tw(G)$ in time*

$$2^{\mathcal{O}(k^2)} \cdot |G|$$

One of the big reasons behind the popularity of treewidth is the fact, that the MSO_2 model checking problem can be solved efficiently on graphs of bounded treewidth – the famous Courcelle’s MSO_2 theorem:

Theorem 3.3 (Courcelle’s MSO_2 Theorem, [19]). *The MSO_2 model checking problem is fixed parameter tractable when parameterized by the length of formula and the treewidth of the input graph. I.e. given a graph G of width k and an MSO_2 formula φ , the problem can be solved time $f(|\varphi|, k) \cdot |G|$, where f is a computable function.*

The proof of Courcelle’s theorem involves a construction of bottom-up tree automaton, which runs on the tree-decomposition of the graph.

Practical significance of Courcelle’s theorem Even though Theorem 3.3 seems to give us efficient algorithms for all problems expressible in MSO_2 , in practice this is not really so. The catch here is the function f . As shown by Frick and Grohe [33], this is not even an elementary function in the rank of the formula φ . Therefore Theorem 3.3 is more often used to ascertain that an FPT algorithm for a given problem exists, and a specific algorithm is then designed by hand to solve the problem. Nevertheless, there have been surprisingly successful attempts to implement Courcelle’s algorithm, most notably [70].

Note that LinEMSO_2 optimization problems can also be efficiently solved on graphs of bounded treewidth:

Theorem 3.4 ([2]). *For every integer t and MSO_2 formula ψ , every ψ - LinEMSO_2 optimization problem is fixed-parameter tractable on digraphs of treewidth t , with the parameters t and $|\psi|$.*

3.1.1 Game characterization of treewidth

One of the nice features of treewidth is its alternative characterization in terms of a cops-and-robber game. This proved to be very useful, often significantly simplifying some of the proofs. The characterization is possible thanks to nice structural properties of treewidth.

The original cops-and-robber game was introduced in [90]. In this game the robber stands on a vertex of the graph, and can at any time run at a great speed to any other vertex along a path of the graph. He is not permitted to run through a cop, however. There are k cops, each of whom at any time either stands on a vertex or is in a helicopter. The goal of the player controlling the cops is to land a cop via a helicopter onto a vertex currently occupied by the robber, and the robber’s objective is to elude capture. (The point of the helicopter is that cops can move anywhere in the graph but, while moving, they do not interfere with the robber.) The robber can see the helicopter land-

ing and may run to a new vertex before it actually lands. See e.g. [39] for a formal definition.

The following theorem (often called the *treewidth duality theorem*) relates cops-and-robber games and treewidth:

Theorem 3.5 ([90]). *Graph G has treewidth k iff the minimum number of cops required to win the cops-and-robber game is $k + 1$.*

3.2 Clique-width and Rank-width

As we have seen, for tree-like graphs, which are well captured by the notion of treewidth, we can efficiently decide the MSO_2 model checking problem. Can we possibly get a similar result for a much richer class of graphs? Notice that graphs of bounded treewidth are, by definition, sparse (i.e., they contain relatively few edges compared to the number of vertices). This suggests we should turn our attention to classes which contain cliques, the ultimate non-sparse graphs. For cliques the model checking problem is trivial, since all vertices are exactly the same. Building upon this intuition is the notion of clique-width, introduced by Courcelle, Engelfriet and Rozenberg [20] (even though the name “clique-width” itself was first used only later in [22]).¹

Definition 3.6 (clique-width, [22]). *Let k be a positive integer. A pair (G, γ) is a k -labelled graph if G is a simple graph and $\gamma : V(G) \rightarrow \{1, 2, \dots, k\}$ is a mapping. A k -expression is a well formed expression t built using the four operators defined below. Let $1 \leq i, j \leq k$. Then*

- $[i]$ is a nullary operator which represents a graph with a single vertex labelled i ,
- $\eta_{i,j}$, for $i \neq j$, is a unary operator which adds edges between all pairs of vertices where one is labelled i and the other is labelled j ,
- $\rho_{i \rightarrow j}$ is a unary operator which changes the labels of all vertices labelled i to j , and
- \oplus is a binary operator which represents disjoint union of two k -labelled graphs.

Each k -expression t naturally generates a k -labelled simple graph $G = G[t]$. The smallest k such that there exists a k -expression generating G is the clique-width of G .

1. Independently, Wanke [91] developed the notion of NLC-width, which is within a factor of 2 of clique-width.

It is not hard to show that cliques and complete bipartite graphs have clique-width 2 (as indeed do all cographs). With some degree of vagueness one can argue that clique-width measures how close a graph is to being a complete bipartite graph. Moreover, while cliques can have arbitrarily high treewidth, bounding treewidth also bounds clique-width, as witnessed by the following theorem:

Theorem 3.7 ([22, 91]). *Every graph of treewidth at most k has clique-width at most $2^{k+1} + 1$.*

As for the model checking, Courcelle, Makowski and Rotics [21] proved the following

Theorem 3.8 ([21]). *The MSO_1 model checking problem is fixed parameter tractable on (any class of) graphs of bounded clique-width.*

Notice that, compared to Theorem 3.3, we use less expressive logic, MSO_1 , instead of MSO_2 . This is no accident. As was proved already in [21] (see also [69] for an alternative proof), having an efficient algorithm for MSO_2 model checking on graphs of bounded clique-width is not possible under standard complexity-theoretic assumptions.

Theorem 3.9 ([21], [69]). *The MSO_2 model checking problem on cliques is not in XP, unless $\text{EXP} = \text{NEXP}$.*

Similarly to treewidth, the MSO_1 model checking result can be extended to LinEMSO_1 optimization problems:

Theorem 3.10 ([21]). *For every integer t and MSO_1 formula ψ , every ψ - LinEMSO_1 optimization problem is fixed-parameter tractable on digraphs of clique-width t , with the parameters t and $|\psi|$.*

Rank-width One of the drawbacks of clique-width is that it cannot be efficiently computed for a given graph (there is no clique-width counterpart of Theorem 3.2.) Fortunately, there is the related notion of *rank-width*, defined by Oum and Seymour [83], which keeps many desirable properties of clique-width, while not suffering from this drawback.

It turns out that for graphs of bounded rank-width the decomposition can be computed efficiently.

Theorem 3.11 ([50]). *For every parameter t there is an $O(n^3)$ -time FPT algorithm that, for a given n -vertex graph G , either finds a rank-decomposition of G of width at most t , or confirms that the rank-width of G is more than t .*

As established already in the original rank-width paper [83], the rank-width of a graph is bounded if, and only if, its clique-width is bounded.

Theorem 3.12 ([83]). *Let G be a graph. Then*

$$rw(G) \leq cw(G) \leq 2^{rw(G)+1} - 1$$

This theorem also, in fact, proves the existence of an efficient MSO₁ model checking algorithm for graphs of bounded rank-width – via clique-width and Theorem 3.8.

3.3 MSO lower bounds

As we have seen, Courcelle’s theorem (Theorem 3.3) is a fast and relatively easy way of establishing that a problem can be solved efficiently on graphs of bounded treewidth. However, one may ask how far this result could be generalized. That is, is there another reasonable graph class of unbounded treewidth such that MSO₂ model checking remains tractable on this class? In the previous section (Theorem 3.9) we have seen that the class of graphs of bounded clique-width is already too rich for this to work. So we are looking for a graph class strictly between treewidth and clique-width.

3.3.1 MSO₂ lower bounds

The first “lower bound” to Courcelle’s theorem, by Makowski and Mariño, appeared in [73]. In that paper the authors show that if a class of graphs has unbounded treewidth and is closed under topological minors, then model checking for MSO₂ is not fixed-parameter tractable unless $P = NP$. More recently, a stronger lower bound result by Kreutzer—not requiring the class to be closed under minors—appeared in [65]. In that paper, Kreutzer used the following version of “unbounding” the treewidth of a graph class:

Definition 3.13 (Kreutzer and Tazari [65, 66]). *The treewidth of a class \mathcal{C} of graphs is strongly unbounded by a function $f: \mathbb{N} \rightarrow \mathbb{N}$ if there is $\epsilon < 1$ and a polynomial $p(x)$ s.t. for all $n \in \mathbb{N}$ there is a graph $G_n \in \mathcal{C}$ with the following properties:*

- i) *the treewidth of G_n is between n and $p(n)$ and is greater than $f(|G_n|)$, and*
- ii) *given n , the graph G_n can be constructed in time 2^{n^ϵ} .*

The degree of the polynomial p is called the gap-degree of \mathcal{C} (with respect to f). The treewidth of \mathcal{C} is strongly unbounded poly-logarithmically if it is strongly unbounded by $\log^c n$, for all $c \geq 1$.

In other words, saying that treewidth of \mathcal{C} is *strongly unbounded* means that

- (i) there are no big gaps between the treewidth of witness graphs (those certifying that the treewidth of n -vertex graphs in \mathcal{C} is greater than $f(n)$), and
- (ii) we can compute such witnesses effectively—in sub-exponential time w.r.t. n .

The main result of [65] is the following theorem:

Theorem 3.14 (Kreutzer [65]). *Let Γ be a fixed set of (at least two) colours, and \mathcal{C} be a class of graphs such that*

- (1) *the treewidth of \mathcal{C} is strongly unbounded poly-logarithmically;*
- (2) *\mathcal{C} is closed under Γ -colourings (i.e., if $G \in \mathcal{C}$ and G' is obtained from G by colouring some vertices or edges by colours from Γ , then $G' \in \mathcal{C}$); and,*
- (3) *\mathcal{C} is constructible (i.e., given a witness graph in \mathcal{C} , a certain substructure can be computed in polynomial time).*

Then $\text{MC}(\text{MSO}_2\text{-}\Gamma)$, the MSO_2 model checking problem on all Γ -coloured graphs from \mathcal{C} , is not in XP (and hence not in FPT), unless all problems in the polynomial-time hierarchy can be solved in sub-exponential time.

This would, of course, mean that the Exponential Time Hypothesis (ETH) [53] fails. The results of [65] have been improved by Kreutzer and Tazari in [67], where the constructability requirement (3) was dropped. A further improvement by the same authors appeared in [66]. The main result in [66] can be stated as follows:

Theorem 3.15 (Kreutzer and Tazari [66]). *Let \mathcal{C} be a class of graphs such that*

- (1) *the treewidth of \mathcal{C} is strongly unbounded poly-logarithmically; and*
- (2') *\mathcal{C} is closed under taking subgraphs, i.e. $G \in \mathcal{C}$ and $H \subseteq G$ implies $H \in \mathcal{C}$.*

Then $\text{MC}(\text{MSO}_2)$, the MSO_2 model checking problem on \mathcal{C} , is not in XP unless all problems in the polynomial-time hierarchy can be solved in sub-exponential time.

Note that (2'), to be closed under subgraphs, is a strictly weaker condition than previous (2), to be closed under Γ -colourings (of edges, too).

3.3.2 MSO₁ lower bounds

While Kreutzer and Tazari settled the extents of tractability for the MSO₂ logic, the issue still remained open for the weaker logic MSO₁. We therefore decided to investigate this issue for MSO₁ logic with a fixed set of vertex labels [42]. The role of *vertex labels* in our paper is similar to that of colours in [65, 67], but weaker in the sense that the labels are not assigned to edges.² In contrast to the work by Kreutzer and Tazari, we assumed a different set of problems—those expressible by MSO₁- L on graphs with vertex labels from a fixed finite set L —to be efficiently solvable on a graph class in order to derive an analogous conclusion. Note that there exist classes \mathcal{C} of L -labelled graphs of unbounded treewidth on which MC(MSO₁- L), the MSO model checking problem on \mathcal{C} , is polynomial time solvable, e.g. classes of bounded clique-width or rank-width (see Section 3.2). But it is important to realize that these classes are *not* closed under taking subgraphs. The main result of [42] is the following theorem:

Theorem 3.16 ([42]). *Assume a (suitable but fixed) finite label set L , and a graph class \mathcal{G} satisfying the following two properties:*

- a) \mathcal{G} is closed under taking subgraphs and under L -vertex-labelings,
- b) the treewidth of \mathcal{G} is densely unbounded poly-logarithmically

Then MC(MSO₁- L , \mathcal{G}), the MSO₁- L model checking problem on all L -vertex-labelled graphs from \mathcal{G} , is not in XP unless the non-uniform Exponential Time Hypothesis fails.

Densely unbounded treewidth The theorem above uses a different definition of unbounding than [65, 66]:

Definition 3.17 (Densely unbounded treewidth [42]). *For a graph class \mathcal{G} , we say that the treewidth of \mathcal{G} is densely unbounded by a function g if there is a constant $\gamma > 1$ such that, for every $m \in \mathbb{N}$, there exists a graph $G \in \mathcal{G}$ whose treewidth is $tw(G) \geq m$ and $|V(G)| < \mathcal{O}(g^{-1}(m^\gamma))$. The constant γ is called the gap-degree of this property.*

Remark 3.18. *Compared to Definition 3.13 one can easily check that if the treewidth of a class \mathcal{G} is strongly unbounded by a function g , then the treewidth is densely*

2. The reason we used the term labels, and not colours, is to be able to clearly distinguish between vertex-labelled graphs and the coloured graphs used in [65, 67], where colours are assigned to edges and vertices.

unbounded by g with the same gap-degree, and the witnessing graphs G of Definition 3.17 can be computed for all m efficiently—in sub-exponential time w.r.t. m . Hence the notion of being densely unbounded is weaker in this respect.

The use of a different “unbounding” definition allowed us to simplify the proofs, for the cost of using a stronger complexity-theoretic assumption, namely the non-uniform ETH instead of the ordinary ETH.

Also our result applies to $\text{MSO}_1\text{-}L$ model checking on L -vertex-labelled graphs, while the result of [66] applies to MSO_2 over unlabelled graphs. There are problems that can be expressed in $\text{MSO}_1\text{-}L$ and not in MSO_2 and vice versa (take RED-BLUE DOMINATING SET vs. HAMILTONIAN CYCLE, for instance). If, however, the set of labels L is fixed for both, $\text{MSO}_1\text{-}L$ has much weaker expressive power than $\text{MSO}_2\text{-}L$ due to missing edge-set quantifications. In particular, note that the efficient model checking results for treewidth and clique-width (Theorems 3.3 and 3.8) that deal with MSO -definable properties handle unlabelled as well as (vertex-)labelled inputs with equal ease.

Moreover, if we assume that the label set L is potentially unbounded, then we obtain a stronger result (getting us even closer to [66]):

Theorem 3.19 ([42]). *$\text{MSO}_1\text{-}L$ model checking with vertex labels L (L depending on the formula size) is not tractable for a graph class satisfying (a) and (b) of Theorem 3.16 unless every problem in the polynomial-time hierarchy is in $\text{DTIME}(2^{o(n)})/\text{SUBEXP}$.*

3.4 Undirected width as a measure of directed graphs

Even though the measures described in the previous sections were designed as width measures of undirected graphs, there is nothing which prohibits us from using them also on digraphs – we can just forget the orientation of the edges. In the case of treewidth this works exactly as intended. E.g. to solve the directed Hamiltonian path on graphs of bounded treewidth one can use basically the same algorithm as for undirected graphs. The only modification would be to check the orientation of each considered arc. Similarly Courcelle’s MSO_2 theorem (Theorem 3.3) would hold for the oriented variant of the MSO_2 logic. (Which is obtained by replacing the predicate *adj* with *arc*.)

Parity games on graphs of bounded treewidth A good example of a problem where using an undirected width measure allows us to efficiently solve a problem which is otherwise difficult to solve on general graphs is the prob-

lem of finding a winner in a parity game (see Section 2.5 for definitions and the relationship between parity games and modal μ -calculus).

To solve parity games (which are played on directed graphs) one may be tempted to devise the following approach: Define the winner using an MSO formula, and then apply Courcelle's theorem [19] to this formula and the input graph. The problem with this approach is that [19] considers the formula to be fixed, whereas in parity games the size of the formula describing the winning condition depends on the number of priorities, which can be as high as the number of vertices. Coupled with the non-elementary dependence of the algorithm presented in [19] on the size of the formula (see Section 3.1), we do not get an FPT algorithm.

Fortunately, this does not mean we cannot get an efficient algorithm by other means. Using standard dynamic programming approach, in [79] I was able to devise an FPT algorithm for solving parity games on graphs of bounded treewidth:

Theorem 3.20 ([79]). *Let \mathcal{G} be a parity game and \mathcal{T} a tree decomposition of \mathcal{G} of width k . Then we can find the winner of the parity game \mathcal{G} in time $\mathcal{O}(n \cdot (k+1)^2 \cdot d^{2(k+1)^2})$, where n is the number of vertices of \mathcal{G} and $d \leq n$ the number of priorities.*

My algorithm was many years later improved by Fearnley and Schewe [30] to run in time $\mathcal{O}(n \cdot k^{k+5} \cdot (d+1)^{3k+5})$.

For some measures forgetting edge orientation does not help Unlike the case of treewidth discussed above, bounding the *undirected* clique-width or rank-width of the underlying undirected graph does not generally help us to solve directed graph problems.

Proposition 3.21. *If $P \neq NP$, then there exist MSO_1 -definable digraph properties that have **no** XP-time algorithms with respect to undirected clique-width or rank-width.*

This follows from the observation that there exist directed problems which are NP-complete even on tournaments (orientations of complete graphs), i.e. on digraphs of undirected clique-width 2 and rank-width 1. If such a problem had an XP-time algorithm, then this would immediately imply $P = NP$. An example of this phenomenon is the problem of partitioning a tournament into two acyclic subtournaments [16]: that problem is both MSO_1 -definable and is NP-complete.

Proposition 3.21 is therefore in *sharp contrast* to the situation with treewidth, where bounding the treewidth of the underlying undirected

graph allows all the algorithmic machinery to work also on digraphs. A brief informal explanation of this antagonism lies in the fact that a “bag” in a tree-decomposition has bounded size and so there is only a bounded number of possible orientations of the edges in it, while a single $\eta_{i,j}$ (edge-addition) operation in a clique-width expression creates a bipartite clique of an arbitrary size which admits an unbounded number of possible orientations.

Articles in the collection

[42] R. Ganian, P. Hliněný, A. Langer, J. Obdržálek, P. Rossmanith, and S. Sikdar. “Lower bounds on the complexity of MSO_1 model-checking”. In: *J. Comput. Syst. Sci.* 80.1 (2014), pp. 180–194. doi: 10.1016/j.jcss.2013.07.005

MSO₁ lower bounds, Section 3.3.2.

[79] J. Obdržálek. “Fast Mu-calculus Model Checking when Tree-width is Bounded”. In: *CAV 2003*. Vol. 2725. LNCS. Springer, 2003, pp. 80–92. doi: 10.1007/978-3-540-45069-6_7

The algorithm for parity games on graphs of bounded treewidth, Section 3.4.

4 Directed graph width measures

Despite treewidth being very successful, from both algorithmic and graph-theoretic standpoint, width measure for undirected graphs, in early 2000's the situation was significantly different for directed graphs – there were hardly any “useful” measures for these graphs. In this chapter I describe our contribution to defining a good width measure for digraphs.

4.1 Early cops-and-robber based measures

Directed treewidth The first digraph width measure to be defined as a directed counterpart of treewidth was the *directed treewidth* [55] of Johnson, Robertson, Seymour and Thomas. This measure is defined so that directed acyclic digraphs have width 0. The game characterization is the same as for treewidth, with two important differences:

1. the robber must follow the direction of edges
2. the robber must stay in the same (cop-free) strongly connected component

The formal definition of directed treewidth is somewhat unwieldy and we therefore do not include it here and refer the reader to [55]. However we include the (slightly loose) relationship between the number of cops needed to catch the robber and the directed treewidth of a graph.

Theorem 4.1 ([55]). *Let G be a graph. Then the directed treewidth of G ($dtw(G)$), and the number of cops needed to catch the robber on G are within a factor of 3 of each other.*

On the algorithmic side the authors of [55] showed that on graphs of bounded directed treewidth problems like HAMILTONIAN PATH can be solved in XP time.

Other early measures In 2004–2005 several new directed measures appeared: directed pathwidth [3] (a direct counterpart to pathwidth of undirected graphs), entanglement [6] and D-width [88]. These measures, for various reasons, never gained sufficient traction and we will not discuss them in the rest of this thesis. Interested reader can find more information, e.g., in the thesis of Rabinovich [85].

4.2 DAG-width

Trying to use directed treewidth to efficiently solve parity games (see Section 2.5), I realized that this measure is, compared to (undirected) treewidth too cumbersome to use. DAG-width, first published in [81], was my attempt to fix some of the issues of directed treewidth. Independently, Berwanger, Dawar, Hunter and Kreutzer came with exactly the same notion, which they published in [4] just a few months after [81]. Not to further duplicate our efforts we decided to produce the journal version of the paper together as [5], and this version is now the authoritative reference.

One of the guiding ideas behind DAG-width was to design a measure whose variant of directed cops-and-robber game most closely corresponds to the (undirected) game for treewidth: The robber is visible to the cops, and can move infinitely fast.

The definition of the DAG-width mirrors that of treewidth (cf. Definition 3.1).

Definition 4.2 (DAG-decomposition [5]). *A DAG-decomposition of a graph G is a pair (D, \mathcal{X}) where D is a DAG and $\mathcal{X} = (X_d)_{d \in V(D)}$ is a family of subsets of V such that*

$$(D1) \bigcup_{d \in V(D)} X_d = V.$$

$$(D2) \text{ For all vertices } d \preceq_D d' \preceq_D d'', X_d \cap X_{d''} \subseteq X_{d'}.$$

$$(D3) \text{ For all edges } (d, d') \in E(D), X_d \cap X_{d'} \text{ guards } X_{\succeq d'} \setminus X_d, \text{ where } X_{\succeq d'} \text{ stands for } \bigcup_{d'' \succeq_D d'} X_{d''}. \text{ For any root } d, X_{\succeq d} \text{ is guarded by } \emptyset.$$

The width of a DAG-decomposition (D, \mathcal{X}) is defined as $\max\{|X_d| \mid d \in V(D)\}$. The DAG-width of a digraph G , $\text{dagw}(G)$, is defined as the minimal width of any of its DAG-decompositions.

The new concept here is that of *guarding*.

Definition 4.3. *Let $G = (V, E)$ be a digraph and $W, V' \subseteq V$. Then W guards V' if for all $(u, v) \in E$ it is true that $u \in V'$ implies $v \in V' \cup W$.*

This is directed counterpart of the following property enjoyed by treewidth: given a node t , the removal of vertices in the bag X_t disconnects the graph G according to the tree-decomposition. In the directed case this disconnection works only in one way: “bottom-up”.

The game characterization of DAG-width is precise. Moreover, the monotone and non-monotone cop strategies are in this case equivalent. (The cops play *monotonely* if they never revisit a vertex they have already left.)

Theorem 4.4 ([5]). *For any digraph G , there is a DAG-decomposition of G of width at most k if, and only if, k cops have a (monotone) winning strategy in the cops-and-robber game on G .*

Computing a DAG-decompositions for a graph of bounded DAG-width essentially corresponds to computing monotone winning strategies in the monotone cops-and-robber game, which gives us the following:

Proposition 4.5 ([5]). *Given a digraph G of DAG-width at most k , a DAG-decomposition of G of width at most k can be computed in time $\mathcal{O}(|G|^{\mathcal{O}(k)})$.*

On the other hand, a recent result [1] shows that computing the DAG-width is surprisingly hard, unlike for treewidth.

Theorem 4.6 ([1]). *Let G be a graph and $k \in \mathbb{N}$. The problem of deciding whether $\text{dagw}(G) \leq k$ is PSPACE-complete.*

Additionally the authors of [1] also managed to prove that there are graphs G such that the optimal DAG-decomposition contains a super-polynomial number of bags.

Solving parity games On the algorithmic side, using DAG-width as parameter one can efficiently solve the same problems which were shown in [55] to be efficiently solvable for directed treewidth. Other problems can be solved as well. Actually the main motivation of both [81] and [4] was to come up with a *directed* width measure which can help us to efficiently solve parity games. (See Section 2.5.) The need for a directed measure was already pointed out in [79]: while it is trivial to efficiently solve parity games on directed acyclic graphs, these graphs can have arbitrarily high treewidth. That means that the bounds provided by Theorem 3.20 are far from optimal.

In [5] we presented an algorithm similar in spirit to our algorithm for treewidth [79]. That algorithm relies on the fact that in a tree decomposition (of the underlying undirected graph), the set of k vertices in any bag of the decomposition guards all entries and exits to the part of the graph below this bag. In the case of a DAG decomposition, while the k -element set guards all exits from the subgraph below it, there may be an unlimited number of edges going into this subgraph. This is the main challenge that our algorithm for DAG-width addresses. Our result is summarized by the following theorem:

Theorem 4.7 ([5]). *For each k , there is a polynomial p and an algorithm running in time $\mathcal{O}(p(n))$ which determines the winner of parity games on all digraphs of n vertices with DAG-width at most k .*

4.3 Kelly-width – an improvement on DAG-width

The discussion of DAG-width would not be complete without mentioning Kelly-width, which was introduced by Hunter and Kreutzer in [52]. It was intended to improve upon DAG-width by reducing the size of the decompositions. The original definition is through the *elimination ordering*, which is modelled after the elimination ordering used to give an alternative characterization of treewidth.

The game characterization of Kelly-width is the same as for DAG-width, with the following two changes

1. the robber is *invisible* to the cops, and
2. the robber is *lazy* – he does not move until a helicopter is about to land on the vertex occupied by him.

Similarly to the case of DAG-width, monotone and non-monotone strategies are equivalent:

Theorem 4.8. *For any digraph G , the Kelly-width of G , $kellyw(G)$, is $\leq k$ if, and only if, k cops have a (monotone) winning strategy in the cops-and-robber game on G , where the robber is lazy and invisible.*

Unlike DAG-width, the Kelly-width can be computed efficiently. The main reason is the fact that the size of the elimination ordering is small (linear).

Theorem 4.9. *The Kelly-width of a graph with n vertices can be determined in time $\mathcal{O}^*(2^n)$ and space $\mathcal{O}^*(2^n)$, or in time $\mathcal{O}^*(4^n)$ and polynomial space.*

Here the \mathcal{O}^* notation hides polynomial factors.

Remark 4.10 (Common properties of DAG-width and Kelly-width). *In addition to being characterized by games, both DAG- and Kelly-width share some other common properties:*

- *Acyclic digraphs (DAGs) have DAG-width 0 and Kelly-width 1 (cf. Proposition 4.11).*
- *If we replace each edge of a graph of treewidth k by a pair of opposite arcs, then the resulting digraph has DAG-width k and Kelly-width $k + 1$.*

Relations between widths

1. $dagw(G) \leq k \implies dtw(G) \leq 3k + 1$ [5]

$$2. \text{ kellyw}(G) \leq k \implies \text{ dtw}(G) \leq 6k - 2 \text{ [52]}$$

$$3. \text{ kellyw}(G) = k + 1 \implies \text{ dagw}(G) \leq 72k^2 + 42k + 18 \text{ [1]}$$

On the other hand, bounding directed treewidth does not bound any of the other measures. The last open question is whether bounding DAG-width bounds Kelly-width or not. For a complete overview of the relationship among various digraph width measures see [1].

4.4 Hardness of MSO_1 model checking

While MSO model checking is FPT for treewidth and clique-width, the situation is markedly different in the directed case. Indeed for all the digraph width measures introduced in the previous section—including directed treewidth [55], DAG-width [5], and Kelly-width [52]—there is likely no FPT algorithm. The following is folklore:

Proposition 4.11 ([39]). *Assume a digraph width measure δ achieving only bounded values on the class of all directed acyclic graphs, and let \mathcal{C} be a class of graphs of bounded δ -width, If $\text{P} \neq \text{NP}$, then $\text{MC}(\text{MSO}_1, \mathcal{C})$ is not in XP.*

Proof. Let \mathcal{P} be any NP-complete MSO_1 -definable property of undirected graphs, say, 3-colourability. We construct the digraph property \mathcal{P}' by replacing every occurrence of the predicate $\text{adj}(x, y)$ in \mathcal{P} by $(\text{arc}(x, y) \vee \text{arc}(y, x))$. Clearly, an undirected graph has the property \mathcal{P} if and only if any digraph D that is an orientation of G has the property \mathcal{P}' . If we could solve the model checking problem in XP time, then property \mathcal{P}' would be decidable on all DAGs in polynomial time. Hence for any input graph G , we could decide whether $G \models \mathcal{P}$ in polynomial time by first constructing an acyclic orientation D of G , and then deciding whether $D \models \mathcal{P}'$ (in polynomial time). This would imply that $\text{P} = \text{NP}$. \square

In our quest for a “good” directed counterpart to treewidth we tried to find out how “bad” the situation really is. It turns out that the theorem above can be made even stronger, by replacing directed acyclic graphs with an even more restricted class of graphs. In [38] we defined and examined two such restrictions. The first one is called DAG-depth, and was inspired by the well known notion of tree-depth – see the book [77] of Nešetřil and Ossona de Mendez. Our definition of DAG-depth is based upon the alternative inductive definition of tree-depth from their paper [76, Lemma 2.2]:

Definition 4.12 (DAG-depth [38]). The *DAG-depth* $ddp(G)$ of a digraph G is inductively defined as follows: If $|V(G)| = 1$, then $ddp(G) = 1$. If G has a single reachable fragment, then $ddp(G) = 1 + \min\{ddp(G - v) \mid v \in V(G)\}$. Otherwise, $ddp(G)$ equals the maximum over the DAG-depth of the reachable fragments of G .

Among the various properties of DAG-depth (see [38, Corollary 3.12]) we know that DAG-depth of a graph is at least the logarithm of the length of its longest directed path. What that means is there are DAGs of arbitrary DAG-depth.

The second measure, called K-width (not to be confused with Kelly-width), is, in a sense, complementary to DAG-width. While DAG-depth attempts, among other things, to bound the length of the longest directed path in a graph, K-width bounds the number of distinct paths:

Definition 4.13 (K-width [38]). The *K-width* of a digraph G is the maximum number of distinct (not necessarily disjoint) directed s - t paths in G over all pairs of distinct vertices $s, t \in V(G)$.

Similarly to DAG-depth, K-width can be arbitrarily large on DAGs. To get an intuition how these measures interact, check the series of simple examples demonstrating differences between the measures in Table 4.1.

We can now strengthen the statement of Proposition 4.11.

Theorem 4.14 ([38]). *There exists an MSO_1 sentence φ such that the $\text{MC}(\text{MSO}_1)$ problem is NP-hard even on DAGs that are of K-width 1 and DAG-depth 2.*

4.5 Measures with efficient MSO_1 model checking

Directed clique-width Theorem 4.11 suggests that if we want to have a digraph with measure with an efficient MSO_1 model checking, we need to look beyond the game-based measures (as DAGs can be easily searched by a few cops). A natural candidate for such a measure would be directed clique-width. While clique-width was originally defined for undirected graphs [22], the Definition 3.6 the definition readily extends to digraphs; simply replace the operator $\eta_{i,j}$ creating undirected edges by the operator $\alpha_{i,j}$ creating directed edges (arcs) from each vertex with label i to each vertex with label j .

Indeed, the MSO_1 model checking can be done efficiently for graphs of bounded directed clique-width. In full generality one gets the following (cf. Theorem 3.8):

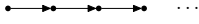
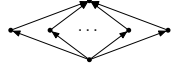





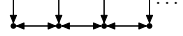
Graph family	DAG-depth	K-width	DAG-width
	∞	1	1
	3	∞	1
	∞	∞	1
	3	1	2
	∞	1	2
	3	∞	2
	∞	1	3
	∞	∞	3

Table 4.1: Families of graphs demonstrating various possible combinations of the listed width measures being bounded and unbounded. Adapted from [38].

Theorem 4.15 ([21] for the undirected case). *For every integer t and MSO_1 formula ψ , every ψ -LinEMSO₁ optimization problem is fixed-parameter tractable on digraphs of clique-width t , with the parameters t and $|\psi|$.*

While the original proof of the undirected version by Courcelle, Makowsky, and Rotics [21] has been followed by at least two different published proofs in [37, 62], none of these published proofs explicitly includes the directed case. Nevertheless, a formal proof of the directed case is a simple translation of any one of these previously published arguments into digraph terms. In [38] we provide a short alternative proof of Theorem 4.15 via a reduction of the directed version into the undirected one with vertex labels [21].

Parity games and directed clique-width Being already able to solve parity games on graphs of bounded treewidth [79] and DAG-width [5], my attention turned to directed clique-width. This choice is natural, as there are DAGs of arbitrarily high clique-width and digraphs of fixed clique-width but arbitrarily high DAG-width. The (positive) algorithmic result for parity games and directed clique-width was presented in [82].

As is the case of treewidth (see Section 3.4), the result is not a consequence of the general result saying that MSO logic is decidable in linear time on graphs of bounded clique-width (Theorem 3.8). The reasons are the same as for treewidth (cf. Section 3.4).

The algorithm described in [82] is, in spirit, similar to the one for graphs of bounded treewidth and bounded DAG-width. However there are many conceptual differences, as there is no small set of vertices forming an interface between the already processed subgraph and the rest of the graph. To the contrary, in one step one can connect an unbounded number of edges. Unfortunately, keeping the results for all the individual vertices of one colour would take too much space. This is the main obstacle which needed to be dealt with in [82]. The result is then summarized by the following theorem:

Theorem 4.16 ([82]). *Let \mathcal{G} be a parity game of directed clique-width k and t the associated k -expression corresponding to \mathcal{G} . Then there is an algorithm which solves the parity game \mathcal{G} in time polynomial in n .*

Bi-rank-width While the definition of clique-width works “as is” also on digraphs, in the case of rank-width slightly more work needs to be done to translate it to the realm of directed graphs. The right notion is that of *bi-rank-width*, defined by Kanté in [59, 60].

Bi-rank-width is related to directed clique-width in the sense that one is bounded on a directed graph class if and only if the other is (cf. Theorem 3.12). This of means that the MSO_1 model checking algorithm for directed clique-width (Theorem 4.15) can be used for bi-rank-width. Finally, as in the case of (undirected) rank-width, the decompositions can be computed efficiently.

Articles in the collection

[5] D. Berwanger, A. Dawar, P. Hunter, S. Kreutzer, and J. Obdržálek. “The DAG-width of directed graphs”. In: *J. Comb. Theory, Ser. B* 102.4 (2012), pp. 900–923. doi: 10.1016/j.jctb.2012.04.004

Section 4.2.

[38] R. Ganian, P. Hliněný, J. Kneis, A. Langer, J. Obdržálek, and P. Rossmanith. “Digraph width measures in parameterized algorithmics”. In: *Discrete Appl. Math.* 168 (2014), pp. 88–107. doi: 10.1016/j.dam.2013.10.038

Section 4.4.

[82] J. Obdržálek. “Clique-Width and Parity Games”. In: *CSL'07*. Vol. 4646. LNCS. Springer, 2007, pp. 54–68. doi: 10.1007/978-3-540-74915-8_8

The algorithm for solving parity games on graphs of bounded directed clique-width, Section 4.5.

5 Existence of powerful digraph width measures

Game-based measures As we have seen in the previous chapter, the most important directed graph measures fall into two separate groups. The first group consists of “game-based” measures like DAG-width and Kelly-width. These measures share some nice structural properties and are closely related to treewidth of undirected graphs. However, for these measures we have no efficient model checking algorithms for the MSO logic. In the words of [39], they are not *powerful*.

Powerful measures In the other group are the, mutually closely related, measures of directed clique-width and bi-rank-width. From Propositions 4.11 and 4.15, it seems that directed clique-width and bi-rank-width are more suitable candidates for a good digraph width measure, since MSO_1 model checking can be done efficiently on classes of graphs where one of these parameters is bounded. Unfortunately, clique-width and bi-rank-width do not possess the nice structural properties common to the various treewidth-like measures, such as being subgraph- or contraction-monotone. This is due to symmetric orientations of complete graphs all having clique-width two, while their subdigraphs include all digraphs, even those with arbitrarily high clique-width. This seems to be a drawback and a possible reason why clique-width- and rank-width-like measures are not so widely accepted.

Can we take the better of each of the two worlds? To us this seemed as the ultimate natural question and we therefore decided to investigate it in a greater detail. Unfortunately, the answer to this question, as published in [39] and summarized by Theorems 5.5 and 5.6, is negative.

5.1 Characterizing game definable measures

Introducing directed minors. In the realm of undirected graphs, characterizability by a cops-and-robber game is closely related to monotonicity under taking minors. Recall that a graph H is a *minor* of a graph G if it can be obtained by a sequence of applications of three operations: vertex deletion, edge deletion and edge contraction. (See e.g. [25].) A measure is *monotone* under taking minors if the measure of a minor is never larger than the measure of the graph itself. Note that treewidth is monotone under taking minors. The relationship between cops-and-robber games and taking a minor

of a graph should now be obvious: taking a subgraph can never improve robber's chances of evading k cops and, since the robber can move infinitely fast and cops use helicopters, neither can edge contraction.

It is therefore only natural to expect that a "good" digraph-width measure, characterizable by a directed cops-and-robber game, should also be (at least nearly) monotone under some notion of a directed minor. However, at the time of writing [39], there was no definition of a directed minor which would be useful for our purposes. We therefore came up with a definition of *directed topological minor* which seems to fit the bill:

Definition 5.1 ([39]). *A digraph H is a directed topological minor of D if there exists a sequence of digraphs D_0, \dots, D_r such that $D_0 \subseteq D$ and $D_r \cong H$, and for all $0 \leq i \leq r - 1$, one can obtain D_{i+1} from D_i by contracting a 2-contractible arc.*

(Arc is 2-contractible if its contraction does not result in the creation of a new directed path between vertices of degree at least 3.) Robustness of the definition is justified by the following proposition:

Proposition 5.2 ([39]). *Given a digraph D , let D' be obtained from D by a sequence of vertex deletions, arc deletions and contractions of 2-contractible arcs (in any order). Then D' is a directed topological minor of D .*

Monotonicity of the measures. We can now argue that the property of being closed under taking directed topological minors is indeed a *natural requirement* for any cops-and-robber based digraph width measure.

Theorem 5.3. [39] *Let D be a digraph such that in the DAG-width (Kelly-width) game, $k \geq 1$ cops are enough to catch the robber. Let H be a directed topological minor of D . Then at most $k + 2$ cops are needed to catch the robber on H in that same game.*

(We actually believe that a bound of $k + 1$ cops is enough in Theorem 5.3, and perhaps even a bound of k cops is sufficient if $k \geq 3$.)

A digraph width measure δ is *closed under taking directed topological minors* if there is an absolute constant c such that, for each digraph D , the δ -width of any directed topological minor of D is at most $\delta(D) + c$. By Theorem 5.3 this is indeed so for the major existing measures. We moreover give the following relaxed definition to make our negative results slightly stronger:

Definition 5.4. *A digraph width measure δ is weakly closed under taking directed topological minors if there exists a computable function w such that, for each digraph D , the δ -width of any directed topological minor of D is at most $w(\delta(D))$.*

5.2 Non-existence of ideal measures

To state the main result of [39], we need to define two essential properties a “ideal” digraph width measure should possess:

Not treewidth-bounding Good measure should not be comparable to the treewidth of the underlying undirected graph. To be more precise, we do not want the width measure δ to upper-bound the treewidth of the underlying undirected graph. Such δ would not help solve any more inputs than what we already can do with traditional undirected measures.

Efficiently directable Good measure should not “keep computationally excessive” information in the orientation of edges. A digraph width measure δ is *efficiently directable* if, given any undirected graph G , one can orient its edges in time polynomial in $|G|$ to obtain a digraph with near-optimal δ -width. DAG-width, Kelly-width, digraph clique-width, and bi-rank-width are all efficiently directable.

The main results of [39] can now be stated as follows:

Theorem 5.5 ([39]). *Let δ be a digraph width measure with the following properties*

- a) δ is not treewidth-bounding;
- b) δ is weakly closed under taking directed topological minors;
- c) δ is efficiently directable.

Then δ is not powerful unless $P = NP$.

This can be further strengthened by removing the assumption of efficient directability as follows.

Theorem 5.6 ([39]). *Let δ be a digraph width measure with the following properties*

- a) δ is not treewidth-bounding;
- b) δ is weakly closed under taking directed topological minors.

Then δ is not powerful unless $NP \subseteq P/\text{poly}$.

A small price to pay for the stronger formulation of Theorem 5.6 is the need for a stronger complexity assumption, namely that $NP \not\subseteq P/\text{poly}$ instead of $NP \neq P$. Recall that P/poly denotes the polynomial-time complexity class with a *polynomially-bounded advice function*, i.e. the class of languages that have polynomial-size circuits. By the Karp-Lipton theorem [61], $NP \subseteq P/\text{poly}$ would imply that the polynomial hierarchy collapsed to the level Σ_2^P (which is not considered likely).

5.3 From topological minors to subdigraphs

The results of the previous section can be strengthened even further. Our results about the lower bounds for MSO_1 from [42], which we have already discussed in Section 3.3.2, can, without too much effort, be extended to the directed case [42]. The result of [42] differs from Theorem 5.6 in several aspects:

1. it requires directed width measure to be closed under *subdigraphs* and not *directed topological minors*
2. relaxes *unbounded* treewidth by *poly-logarithmically unbounded* treewidth
3. uses $\text{MSO}_1\text{-}L$ instead of MSO_1

The main result then reads:

Theorem 5.7 ([39]). *Let L be a finite set of labels, $|L| \geq 47m$ and δ be a digraph width measure with the following properties:*

- a) δ is monotone under taking subdigraphs;
- b) there exists $d \in \mathbb{N}$ such that the treewidth of the undirected graph class $\{U(D) : \delta(D) \leq d\}$ is densely unbounded poly-logarithmically
- c) for all L -vertex-labelled digraphs D and all sentences $\varphi \in \text{MSO}_1\text{-}L$, the problem of deciding whether $D \models \varphi$ is solvable in time $\mathcal{O}(|D|^{f(\delta(D), |\varphi|)})$ for some computable f .

unless non-uniform ETH fails.

Articles in the collection

[39] R. Ganian, P. Hliněný, J. Kneis, D. Meister, J. Obdržálek, P. Rossmanith, and S. Sikdar. “Are there any good digraph width measures?” In: *J. Comb. Theory, Ser. B* 116 (2016), pp. 250–286. doi: 10.1016/j.jctb.2015.09.001

Whole chapter, except for Section 5.3.

[42] R. Ganian, P. Hliněný, A. Langer, J. Obdržálek, P. Rossmanith, and S. Sikdar. “Lower bounds on the complexity of MSO_1 model-checking”. In: *J. Comput. Syst. Sci.* 80.1 (2014), pp. 180–194. doi: 10.1016/j.jcss.2013.07.005

Section 5.3.

6 FO logic on dense graphs

As we have seen in Section 2.3, the FO model checking problem is PSPACE-complete on general graphs and admits no algorithm with running time $f(\varphi)n^{o(|\varphi|)}$ for any function f , assuming the ETH. Therefore much research has focused on graph classes where the FO model checking problem can be solved more efficiently.

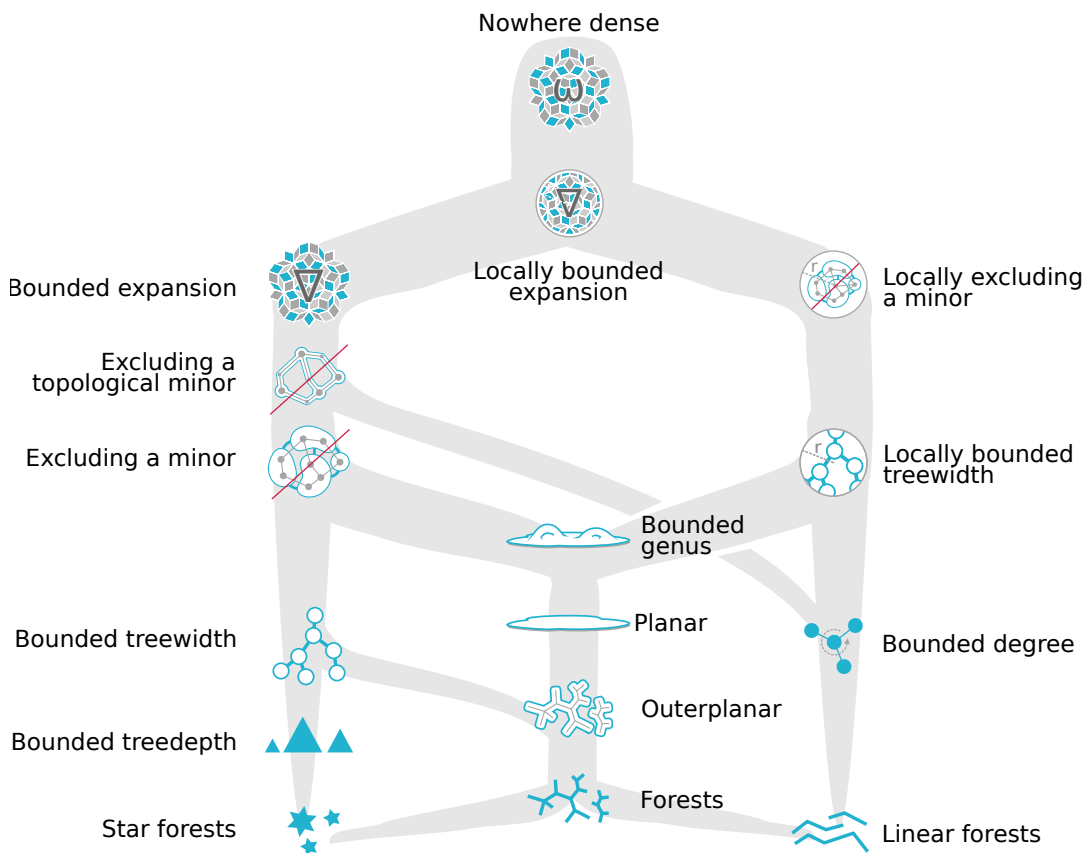


Figure 6.1: The hierarchy of sparse graph classes. Picture courtesy of Felix Reidl.

A great success story is the sequence of results for *sparse* graph classes. (See Figure 6.1 for the hierarchy of sparse graph classes.) It all started with the result of Seese [89], who in 1994, using Gaifman's theorem (see [28]),

showed that FO model checking is FPT on any class of graphs of bounded degree. The use of Gaifman's theorem is possible thanks to the fact that vertex neighbourhoods are quite restricted when the degree is bounded. In the following years, Seese's result was extended first to graphs of locally bounded treewidth [32] and graphs excluding a minor [31], and then to graphs locally excluding a minor [23]. Later, when the theory of sparse graphs was developed by Nešetřil and Ossona de Mendez (see [78]), results started appearing also for the newly defined classes [24, 27, 48]. This line of research culminated in the result of Grohe, Kreutzer and Siebertz [49] for nowhere-dense graph classes, which captures *all* subgraph-closed sparse graph classes on which FO model checking is fixed parameter tractable.

Dense graph classes Unlike for sparse graphs, there were essentially no known results for dense graphs classes. We decided to start filling this gap and began identifying those (dense) graph classes for which the FO model checking is also fixed parameter tractable. Since there is no established theory of dense graph classes, the results we obtained do not follow a single line of research, but rather span multiple areas and approaches.

6.1 Interval graphs

The first non-trivial results on dense graph classes we managed to obtain were for the well known class of interval graphs [41]. These are the intersection graphs of intervals on the real line. I.e. a graph G is an interval graph, if there exists a set \mathcal{I} of intervals of the real line such that $V(G) = \mathcal{I}$ and $E(G)$ contains all pairs of intervals which have a non-empty intersection (see, e.g., [25]). If we restrict the length of intervals, we get the following definition:

Definition 6.1 (*L*-interval graphs). *For a set L of reals, an interval graph is called an L -interval graph if it is an intersection graph of intervals with lengths from L . If, additionally all intervals are subintervals of $[0, d]$ for some real d , we speak about (L, d) -interval graphs.*

Clique-width of interval graphs For example, the well-known (and studied) unit interval graphs are $\{1\}$ -interval graphs. When restricted to unit interval graphs, one can easily deduce the existence of a linear time algorithm for testing FO properties. This follows from the result of Lozin [72] asserting that every proper hereditary subclass of unit interval graphs, in particular, the class of unit interval graphs with bounded radius, has bounded clique-width. The result then follows from the results of Courcelle et al. [21] (see

Theorem 3.8), using Gaifman’s theorem. This can be generalized to finite sets L of rational numbers:

Proposition 6.2 ([41]). *Let L be a finite set of positive rational numbers. For any $d > 0$, the class of (L, d) -interval graphs has bounded clique-width.*

From Proposition 6.2 and Gaifman’s theorem, one can approach the FO model checking problem on L -interval graphs for finite sets L of rationals. By Gaifman’s theorem, every FO model checking instance can be reduced to model checking of basic local FO sentences, i.e. to FO model checking on L -interval graphs with bounded radius. Since L -interval graphs with radius d are $(L, (2d + 1) \max L)$ -interval graphs and so have bounded clique-width, the latter can be solved in linear time by [21]. Combining this with the neighbourhood covering technique from [32], which can be adapted to run in linear time in the case of L -interval graphs given with their interval representation, we obtain the following.

Corollary 6.3 ([41]). *Let L be a finite set of positive rational numbers and φ an FO sentence. There exists a linear time algorithm that decides whether an L -interval graph G satisfies φ if the input graph G is given by its L -representation with the left end points of the intervals sorted.*

However, Proposition 6.2 is just a fortunate special case, since aside of rational lengths one can prove the following.

Proposition 6.4 ([41]). *For any irrational $q > 0$ there is d such that the class of $(\{1, q\}, d)$ -interval graphs has unbounded clique-width.*

This is an improvement on an earlier result of [44] which shows that the class of all unit interval graphs has unbounded clique-width.

Algorithms for L -interval graphs The main algorithmic result of [41] says that every fixed FO property can be tested in time $O(n \log n)$ for n -vertex L -interval graphs when L is any fixed finite set of reals and an L -interval representation is given on the input.

Theorem 6.5 ([41]). *For every finite subset L of reals and every FO sentence φ , there exists an algorithm running in time $O(n \log n)$ that decides whether an input n -vertex L -interval graph G given by its L -representation satisfies φ .*

To prove this result, we used a well-known characterization of FO properties by Ehrenfeucht-Fraïssé games. This allowed us to convert the input graph into an L -interval graph and a representation of this graph such that

every vertex of the new graph has at most $K_0 \cdot \lceil \max L \rceil$ neighbours. In particular, the maximum degree of the new graph is bounded. The final step is applying the result of Seese [89] stating that every FO property can be decided in linear time for graphs with bounded maximum degree.

Negative results The result of Theorem 6.5 cannot be pushed much further. If L is an (infinite) set that is dense in some open set, then L -interval graphs can be used to model arbitrary graphs [41]. Since the parameterized FO model checking problem is AW[*]-complete¹ for general graphs, the following is straightforward.

Theorem 6.6 (Corollary 6.2 of [41]). *If L is a subset of non-negative reals that is efficiently dense in some non-empty open set, then FO model checking is AW[*]-complete on L -interval graphs when parameterized by the formula size.*

In addition, we showed that unit interval graphs allow an efficient polynomially bounded MSO interpretation of all graphs and a successor FO interpretation of all graphs. This means that Theorem 6.5 cannot be extended to MSO_1 .

Corollary 6.7 ([41]). *MSO_1 model checking is para-NP-hard on unit interval graphs.*

Note that the aforementioned result of Lozin [72] states that every proper hereditary subclass of unit interval graphs has bounded clique-width, and hence MSO_1 model checking on this class can be carried out in linear time [21].

6.2 Posets and existential FO

So far in this thesis we have focused on one kind of algebraic structures: finite graphs. In his 2007 survey paper [47] Grohe noted that “*it would also be very interesting to study the complexity of model checking problems on finite algebraic structures such as groups, rings, fields, lattices, et cetera*”. From this perspective it is particularly interesting to investigate model checking problems on *partially ordered sets* (posets), since posets can be seen both as dense graphs and as algebraic structures. Motivated by Grohe’s survey [47], Bova, Ganian and Szeider [13, 14] initiated the study of FO model checking on posets.

Definition 6.8. *A poset \mathcal{P} is a pair (P, \leq^P) where P is a set and \leq^P is a reflexive, antisymmetric, and transitive binary relation over P . The size of a poset $\mathcal{P} = (P, \leq^P)$ is $\|\mathcal{P}\| := |P|$.*

1. And therefore not FPT – see [26].

Despite similarities between posets and graphs (e.g., in Hasse diagrams), the existing FO model checking results from graphs do not seem to transfer well to posets, perhaps due to lack of usable notions of “locality” and “sparsity” there (which makes it difficult to employ Gaifman’s theorem). This feeling is supported by several negative results in [13], too.

The main result of Bova et al. [13] then is that the model checking problem for the existential fragment of FO (\exists -FO) can be efficiently solved when parameterized by the *width* of the poset, i.e. the size of its largest antichain.

Theorem 6.9 ([13]). *Let $\mathcal{P} = (P, \leq^P)$ be a poset and φ a formula. Then the \exists -FO model checking problem is solvable in time $f(|\varphi|) \cdot n^{g(w)}$ where $n = |P|$ is the size of a poset and w its width.*

In the language of parameterized complexity, this means that the problem is FPT in the size of the formula, but only XP with respect to the width of the poset. Note that this is not an easy result since, for instance, posets of fixed width can have unbounded clique-width [13].

The proof in [13] goes by first showing that the model checking problem for the existential fragment of FO is equivalent to the embedding problem for posets (which can be thought as analogous to the induced subgraph problem, see below), and then reducing the embedding problem to a suitable family of instances of the homomorphism problem of certain semilattice structures.

Let $\mathcal{Q} = (Q, \leq^Q)$ and $\mathcal{P} = (P, \leq^P)$ be two posets. An *embedding* from \mathcal{Q} to \mathcal{P} is an injective function $e : Q \rightarrow P$ such that, $q \leq^Q q'$ if and only if $e(q) \leq^P e(q')$ for every $q, q' \in Q$. The *embedding problem* for posets is thus defined as:

<p>EMBEDDING</p> <p>Input: Two posets $\mathcal{Q} = (Q, \leq^Q)$ and $\mathcal{P} = (P, \leq^P)$.</p> <p>Question: Is there an embedding from \mathcal{Q} into \mathcal{P}?</p>	<p>Parameter: $\text{width}(\mathcal{P}), \ \mathcal{Q}\$</p>
---	--

Proposition 6.10 ([13]). *POSET \exists -FO-MODEL CHECKING is fixed-parameter tractable if and only if so is EMBEDDING. Moreover, there is a polynomial parameter reduction from EMBEDDING to POSET \exists -FO-MODEL CHECKING.*

Improved algorithms for \exists -FO While the algorithm of [13] is FPT in the size of the formula, its only XP with respect to the width of the poset. In [36], we managed to significantly improve the running time of the model checking algorithm – our algorithm is FPT in *both* the size of the formula and the width of the poset:

Theorem 6.11 ([36]). *POSET \exists -FO-MODEL CHECKING is fixed-parameter tractable in the formula size and the width of an input poset; precisely, solvable in time $h(|\varphi|, w) \cdot O(n^2)$ where n is the size of a poset and w its width.*

We used the same reduction of existential FO model checking to the embedding problem from [13], but our subsequent solution to embedding is faster and, at the same time, much more straightforward and easier to follow.

In fact we produced two different algorithms for the embedding problem. The first of these two algorithms is a natural, and easy to understand, polynomial-time reduction to a CSP (Constraint Satisfaction Problem) instance closed under min polymorphisms, giving an $O(n^4)$ dependence of the running time on the size of the poset.

Theorem 6.12 ([36]). *Let $\mathcal{Q} = (Q, \leq^Q)$ and $\mathcal{P} = (P, \leq^P)$ be two posets. Then the embedding problem from \mathcal{Q} into \mathcal{P} is fixed-parameter tractable, more precisely, it can be solved in time $O(\text{width}(\mathcal{P})^{|\mathcal{Q}|} \cdot |\mathcal{Q}|^4 \cdot |\mathcal{P}|^4)$.*

The second algorithm has even better, quadratic, time complexity and works by reducing the embedding problem to a restricted variant of the MULTICOLOURED CLIQUE problem, which is then efficiently solved.

Theorem 6.13 ([36]). *Let $\mathcal{Q} = (Q, \leq^Q)$ and $\mathcal{P} = (P, \leq^P)$ be two posets. Then the embedding problem from \mathcal{Q} into \mathcal{P} is fixed-parameter tractable, more precisely, it can be solved in time $O(\text{width}(\mathcal{P})^{|\mathcal{Q}|} \cdot |\mathcal{Q}|^3 \cdot |\mathcal{P}|^2)$.*

It is this second theorem which, together with Theorem 6.10, gives us the result of Theorem 6.11.

6.3 Posets FO

While the results for the existential fragment of FO logic were interesting, the existence of an FPT algorithm for general FO model checking on posets of bounded width remained open. In [34] we resolved this question in the positive for coloured posets:

Theorem 6.14 ([34]). *Let $\mathcal{P} = (P, \leq^P)$ be a poset of width w , with elements coloured by $\lambda : P \rightarrow \Lambda$ where Λ is a finite set, and let φ be an FO sentence in negation normal form. There is an algorithm which decides whether $\mathcal{P} \models \varphi$ in FPT time $f(w, \varphi) \cdot \|\mathcal{P}\|^2$.*

Our algorithm is based on a new locality lemma for posets. More concretely, we show that for every poset \mathcal{P} and formula φ one can efficiently iteratively construct a directed graph D such that

- (a) the vertex set of D are the elements of \mathcal{P} ,
- (b) every element of \mathcal{P} has bounded out-degree in D , and
- (c) it is possible to determine whether $\mathcal{P} \models \varphi$ by checking whether φ holds on sub-posets of \mathcal{P} induced by constant-radius balls in D .

The statement of our lemma sounds very similar to that of Gaifman’s locality theorem, the crucial differences being that the digraph D is not the Gaifman graph of \mathcal{P} and that D depends on the quantifier rank of φ . Indeed, constant radius balls in the Gaifman graph of constant width posets typically contain the entire poset. Thus a naive application of Gaifman’s theorem would reduce the problem of deciding whether \mathcal{P} is a model of φ to itself. The crucial difficulty we have to overcome is that we have to make the digraph D “dense enough” so that (c) holds, while keeping it “sparse enough” so that the vertices in D still have bounded out-degree. The latter is necessary to ensure that constant radius balls in D have constant size, making it feasible to use the naive model checking algorithm for determining whether φ holds on sub-posets of \mathcal{P} induced by constant-radius balls in D . The construction of the graph D and the proof that it indeed has the desired properties relies on a delicate inductive argument thoroughly exploiting properties of posets of bounded width.

Application to interval graphs The power of Theorem 6.14 can be demonstrated by applying it to interval graphs, obtaining the alternative proof of Theorem 6.5 for L -interval graphs.

Theorem 6.15 ([34]). *Let φ be a graph FO sentence. Assume G is an interval graph given along with its k -fold proper interval representation \mathcal{I} . Then the FO model checking problem $G \models \varphi$, parameterized by k and φ , is FPT.*

This theorem allows us to match the result of [41], except for the precise runtime. On the other hand, the formulation of Theorem 6.15 is more general than [41]. We can, for instance, in the same way derive fixed-parameter tractability also for FO model checking of well-studied *k -proper interval graphs*, introduced in [84] as those having an interval representation such that no interval is properly contained in more than k other intervals.

6.4 Interpreting dense graph classes

While the results of previous sections were encouraging, they did not show a way to systematically study dense graph classes for which the FO model

checking problem is efficiently solvable. The question was if there is a natural way to arrive at new graph classes admitting FPT algorithms for FO model checking. The notion of *graph interpretation* seems to provide us with a way to achieve this goal.

In a simplified setting – given a graph G and an FO formula $\psi(x, y)$ with two free variables, we can define a graph $H = I_\psi(G)$ on the same vertex set as G and the edge set determined by $\psi(x, y)$: a pair of distinct vertices u, v is an edge of H iff $G \models \psi(u, v)$ or $G \models \psi(v, u)$. We then say that H is *interpreted* in G using ψ . A graph class \mathcal{D} is *interpretable* in a graph class \mathcal{C} if there exists an FO formula $\psi(x, y)$ such that every member of \mathcal{D} is interpreted in some member of \mathcal{C} using ψ .

In this context we ask the following question:

Question 6.16. *Let \mathcal{C} be a graph class admitting an FPT algorithm for FO model checking, and \mathcal{D} be a graph class interpretable in \mathcal{C} . Does there exist an FPT algorithm for FO model checking on \mathcal{D} ?*

It might seem that a definite easy answer is ‘yes’, based on the following natural property of interpretations: if $H \in \mathcal{D}$ is interpreted in $G \in \mathcal{C}$ using formula $\psi(x, y)$, and our question is to decide whether $H \models \varphi$, it is a standard routine to construct a sentence φ' such that $H \models \varphi$ if and only if $G \models \varphi'$. Then $G \models \varphi'$ is decided by the FPT algorithm given for \mathcal{C} . However, the difficulty lies in the fact that our inputs come from \mathcal{D} , without any reference to the respective members of \mathcal{C} in which they are interpreted. Even if the interpretation formula $\psi(x, y)$ is fixed and known beforehand, we have generally no efficient way of obtaining the respective member $G \in \mathcal{C}$ for an input $H \in \mathcal{D}$. Thus, Question 6.16 can be reduced to the following:

Question 6.17. *Let \mathcal{C}, \mathcal{D} be graph classes such that \mathcal{D} is interpretable in \mathcal{C} . Does there exist an integer s and a polynomial-time algorithm \mathcal{A} such that; given $H \in \mathcal{D}$ as input, \mathcal{A} outputs $G \in \mathcal{C}$ and an FO formula $\psi(x, y)$ of size at most s such that H is interpreted in G using ψ ?*

An answer to Question 6.17 is far from being obvious. Take, for example, the following particular FO interpretation: A graph H is the *square* of a graph G if the edges of H are those pairs of vertices which are at distance at most 2 in G . Then the problem; given H find G such that H is the square of G , is NP-hard [75]. This shows that it is important to choose a suitable interpretation formula ψ (avoiding the hard cases) in an answer to Question 6.17.

The results We answered both Questions 6.16 and 6.17 in the positive for the case when \mathcal{C} is a class of graphs of bounded degree. The results are based

on a new notion of near- k -twin relation, which generalizes the folklore twin-vertex relation, and is related also to the neighbourhood diversity parameter of [68].

Definition 6.18 (near- k -twin [35]). *For a graph G and $k \in \mathbb{N}$, the near- k -twin relation of G is the relation ρ_k on $V(G)$ defined by $(u, v) \in \rho_k \iff |N(u) \triangle N(v)| \leq k$.*

Considering, e.g., k a small parameter and G a large graph then, intuitively, two vertices of G are near- k -twins if they have “almost the same” neighbourhood. Now if the near- k -twin relation is an equivalence of bounded index, then we can use it to decompose the vertex set of the graph G and efficiently find an interpretation of G in a graph of bounded degree.

Definition 6.19 (near-uniform [35]). *A graph G is (k_0, p) -near-uniform if there exists $k \leq k_0$ for which near- k -twin relation of H is an equivalence of index at most p .*

A graph class \mathcal{D} is (k_0, p) -near-uniform if every member of \mathcal{D} is (k_0, p) -near-uniform, and \mathcal{D} is near-uniform if there exist integers k_0, p such that \mathcal{D} is (k_0, p) -near-uniform.

We then give an efficient FO model checking algorithm for the near-uniform graph classes.

Theorem 6.20 ([35]). *Let \mathcal{D} be a (k_0, p) -near-uniform graph class for some $k_0, p \in \mathbb{N}$. Then the FO model checking problem in \mathcal{D} is fixed-parameter tractable when parameterized by the formula size, i.e., solvable in time $f(|\varphi|) \cdot |V(G)|^{\mathcal{O}(1)}$ for a computable function f and input G, φ .*

This algorithm is based upon the above idea of interpretation; briefly, given a graph H we use the near- k -twin relation for a suitable value of k to partition the vertex set of H and to find a bounded degree graph G , such that H is interpreted in G using a universal formula ψ depending only on the class in question. Then we employ the aforementioned algorithm of Seese [89].

In addition to the algorithm of Theorem 6.20, it turns out that the concept of near-uniform graph classes is robust and sufficiently rich in content: the near-uniform graph classes are exactly the classes which are FO interpretable in graphs of bounded degree.

Theorem 6.21 ([35]). *Let \mathcal{G}_d be the class of (finite) graphs with maximum degree at most d and let $\psi(x, y)$ be an FO formula with two free variables. Then there exist k_0 and p , depending on d and ψ , such that for every $H \in I_\psi(\mathcal{G}_d)$ there exists $k \leq k_0$ for which the near- k -twin relation of H is an equivalence of index at most p .*

Note that, for different graphs H , we may need different values of k (in particular, there may not be a universal value of k which would work for the whole class $I_\psi(\mathcal{G}_d)$).

Articles in the collection

[41] R. Ganian, P. Hliněný, D. Král, J. Obdržálek, J. Schwartz, and J. Teska. “FO Model Checking of Interval Graphs”. In: *Log. Methods Comput. Sci.* 11.4 (2015). doi: 10.2168/LMCS-11(4:11)2015

Section 6.1 – the results for interval graphs.

[36] J. Gajarský, P. Hliněný, J. Obdržálek, and S. Ordyniak. “Faster Existential FO Model Checking on Posets”. In: *Log. Methods Comput. Sci.* 11.4 (2015). doi: 10.2168/LMCS-11(4:8)2015

Section 6.2 – efficient existential FO model checking for posets.

[34] J. Gajarský, P. Hliněný, D. Lokshtanov, J. Obdržálek, S. Ordyniak, M. S. Ramanujan, and S. Saurabh. “FO Model Checking on Posets of Bounded Width”. In: *FOCS’15*. IEEE Computer Society, 2015, pp. 963–974. doi: 10.1109/FOCS.2015.63

Section 6.3 – efficient (full) FO model checking for posets.

[35] J. Gajarský, P. Hliněný, J. Obdržálek, D. Lokshtanov, and M. S. Ramanujan. “A New Perspective on FO Model Checking of Dense Graph Classes”. In: *LICS’16*. ACM, 2016, pp. 176–184. doi: 10.1145/2933575.2935314

Section 6.4 – interpreting dense graphs.

Bibliography

- [1] S. Amiri, L. Kaiser, S. Kreutzer, R. Rabinovich, and S. Siebertz. “Graph Searching Games and Width Measures for Directed Graphs”. In: *STACS’15*. Vol. 30. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015, pp. 34–47. doi: 10.4230/LIPIcs.STACS.2015.34.
- [2] S. Arnborg, J. Lagergren, and D. Seese. “Easy Problems for Tree-Decomposable Graphs”. In: *J. Algorithms* 12.2 (1991), pp. 308–340.
- [3] J. Barát. “Directed path-width and monotonicity in digraph searching”. In: *Graphs and Combin.* 22.2 (2006), pp. 161–172.
- [4] D. Berwanger, A. Dawar, P. Hunter, and S. Kreutzer. “DAG-Width and Parity Games”. In: *STACS’06*. Vol. 3884. LNCS. Springer, 2006, pp. 524–536.
- [5] D. Berwanger, A. Dawar, P. Hunter, S. Kreutzer, and J. Obdržálek. “The DAG-width of directed graphs”. In: *J. Comb. Theory, Ser. B* 102.4 (2012), pp. 900–923. doi: 10.1016/j.jctb.2012.04.004.
- [6] D. Berwanger and E. Grädel. “Entanglement – A Measure for the Complexity of Directed Graphs with Applications to Logic and Games”. In: *LPAR’04*. Vol. 3452. LNCS. Springer, 2005, pp. 209–223.
- [7] H. Björklund, S. Sandberg, and S. Vorobyov. “A discrete subexponential algorithm for parity games”. In: *STACS 2003*. Vol. 2607. LNCS. Springer, 2003, pp. 663–674.
- [8] H. Bodlaender. “A tourist guide through treewidth”. In: *Acta Cybernet.* 11 (1993), pp. 1–21.
- [9] H. Bodlaender. “A linear time algorithm for finding tree-decompositions of small treewidth”. In: *SIAM J. Comput.* 25 (6 1996), pp. 1305–1317.
- [10] H. Bodlaender. “A partial k -arboretum of graphs with bounded treewidth”. In: *Theor. Comput. Sci.* 209 (1998), pp. 1–45.
- [11] H. Bodlaender and A. Koster. “Combinatorial Optimization on Graphs of Bounded Treewidth”. In: *Comput. J.* 51.3 (2008), pp. 255–269.
- [12] O. Borůvka. “O jistém problému minimálním” [About a certain minimal problem]. In: *Práce mor. přírodověd. spol. v Brně III (in Czech and German)* 3 (1926), pp. 37–58.
- [13] S. Bova, R. Ganian, and S. Szeider. “Model Checking Existential Logic on Partially Ordered Sets”. In: *CSL-LICS’14*. Article No. 21. ACM, 2014, p. 10. doi: 10.1145/2603088.2603110.

BIBLIOGRAPHY

- [14] S. Bova, R. Ganian, and S. Szeider. “Quantified Conjunctive Queries on Partially Ordered Sets”. In: *IPEC’14*. Vol. 8894. LNCS. Springer, 2014, pp. 122–134. ISBN: 978-3-319-13523-6. DOI: 10.1007/978-3-319-13524-3_11.
- [15] C. Calude, S. Jain, B. Khoussainov, W. Li, and F. Stephan. “Deciding Parity Games in Quasipolynomial Time”. In: *STOC’17*. to appear. ACM, 2017.
- [16] X. Chen, X. Hu, and W. Zang. “A Min-Max Theorem on Tournaments”. In: *SIAM J. Comput.* 37 (3 2007), pp. 923–937. DOI: 10.1137/060649987.
- [17] E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking*. The MIT Press, 1999.
- [18] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, Third Edition*. 3rd. The MIT Press, 2009. ISBN: 9780262033848.
- [19] B. Courcelle. “The monadic Second order Logic of graphs I: Recognizable sets of finite graphs”. In: *Inform. and Comput.* 85 (1990), pp. 12–75.
- [20] B. Courcelle, J. Engelfriet, and G. Rozenberg. “Handle-Rewriting Hypergraph Grammars”. In: *J. Comput. Syst. Sci.* 46.2 (1993), pp. 218–270. DOI: 10.1016/0022-0000(93)90004-G.
- [21] B. Courcelle, J. A. Makowsky, and U. Rotics. “Linear Time Solvable Optimization Problems on Graphs of Bounded Clique-Width.” In: *Theory Comput. Syst.* 33.2 (2000), pp. 125–150.
- [22] B. Courcelle and S. Olariu. “Upper bounds to the clique width of graphs”. In: *Discrete Appl. Math.* 101.1-3 (2000), pp. 77–114.
- [23] A. Dawar, M. Grohe, and S. Kreutzer. “Locally Excluding a Minor”. In: *LICS’07*. IEEE Computer Society, 2007, pp. 270–279. DOI: 10.1109/LICS.2007.31.
- [24] A. Dawar and S. Kreutzer. “Parameterized Complexity of First-Order Logic”. In: *Electronic Colloquium on Computational Complexity (ECCC) TR09-131* (2009).
- [25] R. Diestel. *Graph Theory*. 4th. Heidelberg: Springer-Verlag, 2010.
- [26] R. Downey and M. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013.
- [27] Z. Dvořák, D. Král, and R. Thomas. “Deciding First-Order Properties for Sparse Graphs”. In: *FOCS’10*. IEEE Computer Society, 2010, pp. 133–142. DOI: 10.1109/FOCS.2010.20.
- [28] H.-D. Ebbinghaus and J. Flum. *Finite Model Theory*. Springer, 1999.
- [29] E. A. Emerson and C. S. Jutla. “The complexity of tree automata and logics of programs”. In: *Proceedings of FoCS’88*. 1988, pp. 328–337.

- [30] J. Fearnley and S. Schewe. “Time and Parallelizability Results for Parity Games with Bounded Treewidth”. In: *ICALP 2012, Part II*. Vol. 7392. LNCS. Springer, 2012, pp. 189–200. doi: 10.1007/978-3-642-31585-5_20.
- [31] J. Flum and M. Grohe. “Fixed-Parameter Tractability, Definability, and Model-Checking”. In: *SIAM J. Comput.* 31.1 (2001), pp. 113–145.
- [32] M. Frick and M. Grohe. “Deciding first-order properties of locally tree-decomposable structures”. In: *J. ACM* 48.6 (2001), pp. 1184–1206.
- [33] M. Frick and M. Grohe. “The complexity of first-order and monadic second-order logic revisited”. In: *Ann. Pure Appl. Logic* 130.1-3 (2004), pp. 3–31. doi: 10.1016/j.apal.2004.01.007.
- [34] J. Gajarský, P. Hliněný, D. Lokshtanov, J. Obdržálek, S. Ordyniak, M. S. Ramanujan, and S. Saurabh. “FO Model Checking on Posets of Bounded Width”. In: *FOCS’15*. IEEE Computer Society, 2015, pp. 963–974. doi: 10.1109/FOCS.2015.63.
- [35] J. Gajarský, P. Hliněný, J. Obdržálek, D. Lokshtanov, and M. S. Ramanujan. “A New Perspective on FO Model Checking of Dense Graph Classes”. In: *LICS’16*. ACM, 2016, pp. 176–184. doi: 10.1145/2933575.2935314.
- [36] J. Gajarský, P. Hliněný, J. Obdržálek, and S. Ordyniak. “Faster Existential FO Model Checking on Posets”. In: *Log. Methods Comput. Sci.* 11.4 (2015). doi: 10.2168/LMCS-11(4:8)2015.
- [37] R. Ganian and P. Hliněný. “On Parse Trees and Myhill–Nerode–type Tools for handling Graphs of Bounded Rank-width”. In: *Discrete Appl. Math.* 158 (2010), pp. 851–867.
- [38] R. Ganian, P. Hliněný, J. Kneis, A. Langer, J. Obdržálek, and P. Rossmanith. “Digraph width measures in parameterized algorithmics”. In: *Discrete Appl. Math.* 168 (2014), pp. 88–107. doi: 10.1016/j.dam.2013.10.038.
- [39] R. Ganian, P. Hliněný, J. Kneis, D. Meister, J. Obdržálek, P. Rossmanith, and S. Sikdar. “Are there any good digraph width measures?” In: *J. Comb. Theory, Ser. B* 116 (2016), pp. 250–286. doi: 10.1016/j.jctb.2015.09.001.
- [40] R. Ganian, P. Hliněný, D. Král, J. Obdržálek, J. Schwartz, and J. Teska. “FO Model Checking of Interval Graphs”. In: *ICALP 2013, Part II*. Vol. 7966. LNCS. Springer, 2013, pp. 250–262. doi: 10.1007/978-3-642-39212-2_24.
- [41] R. Ganian, P. Hliněný, D. Král, J. Obdržálek, J. Schwartz, and J. Teska. “FO Model Checking of Interval Graphs”. In: *Log. Methods Comput. Sci.* 11.4 (2015). doi: 10.2168/LMCS-11(4:11)2015.

BIBLIOGRAPHY

- [42] R. Ganian, P. Hliněný, A. Langer, J. Obdržálek, P. Rossmanith, and S. Sikdar. “Lower bounds on the complexity of MSO_1 model-checking”. In: *J. Comput. Syst. Sci.* 80.1 (2014), pp. 180–194. doi: 10.1016/j.jcss.2013.07.005.
- [43] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. Freeman, San Francisco, 1979.
- [44] M. Golumbic and U. Rotics. “On the Clique-Width of Some Perfect Graph Classes”. In: *Int. J. Found. Comput. Sci.* 11.3 (2000), pp. 423–443. doi: 10.1142/S0129054100000260.
- [45] E. Grädel, P. Kolaitis, L. Libkin, M. Marx, J. Spencer, M. Vardi, Y. Venema, and S. Weinstein. *Finite Model Theory and Its Applications (Texts in Theoretical Computer Science. An EATCS Series)*. Springer, 2005. ISBN: 3540004289.
- [46] E. Grädel, W. Thomas, and T. Wilke, eds. *Automata, Logics, and Infinite Games*. Vol. 2500. LNCS. Springer, 2002.
- [47] M. Grohe. “Logic, Graphs, and Algorithms”. In: *Electronic Colloquium on Computational Complexity (ECCC)* 14.091 (2007), p. 44.
- [48] M. Grohe and S. Kreutzer. “Methods for Algorithmic Meta Theorems”. In: *Model Theoretic Methods in Finite Combinatorics: AMS-ASL Special Session, January 5-8, 2009*. Contemporary Mathematics. AMS, 2011, pp. 181–206. ISBN: 9780821849439.
- [49] M. Grohe, S. Kreutzer, and S. Siebertz. “Deciding first-order properties of nowhere dense graphs”. In: *STOC’14*. ACM, 2014, pp. 89–98. doi: 10.1145/2591796.2591851.
- [50] P. Hliněný and S. Oum. “Finding Branch-decomposition and Rank-decomposition”. In: *SIAM J. Comput.* 38 (2008), pp. 1012–1032.
- [51] P. Hliněný, S. Oum, D. Seese, and G. Gottlob. “Width Parameters Beyond Tree-width and their Applications”. In: *The Computer Journal* 51.3 (2008), pp. 326–362.
- [52] P. Hunter and S. Kreutzer. “Digraph measures: Kelly decompositions, games, and orderings”. In: *Theoret. Comput. Sci.* 399.3 (2008), pp. 206–219.
- [53] R. Impagliazzo, R. Paturi, and F. Zane. “Which Problems Have Strongly Exponential Complexity?” In: *J. Comput. System Sci.* 63.4 (2001), pp. 512–530.
- [54] D. Janin and I. Walukiewicz. “On the expressive completeness of the propositional mu-calculus with respect to monadic second order logic”. In: *CONCUR’96*. Springer, 1996, pp. 263–277. doi: 10.1007/3-540-61604-7_60.

-
- [55] T. Johnson, N. Robertson, P. D. Seymour, and R. Thomas. “Directed Tree-Width”. In: *J. Combin. Theory Ser. B* 82.1 (2001), pp. 138–154.
- [56] M. Jurdziński. “Deciding the Winner in Parity Games Is in $UP \cap co-UP$ ”. In: *Information Processing Letters* 68.3 (1998), pp. 119–124.
- [57] M. Jurdziński. “Small Progress Measures for Solving Parity Games”. In: *STACS 2000*. Vol. 1770. LNCS. Springer, 2000, pp. 290–301.
- [58] M. Jurdziński, M. Paterson, and U. Zwick. “A deterministic subexponential algorithm for solving parity games”. In: *SODA’06*. ACM-SIAM, 2006, pp. 117–123.
- [59] M. Kanté. *The Rank-Width of Directed Graphs*. arXiv:0709.1433v3. Mar. 2008.
- [60] M. Kanté and M. Rao. “The Rank-Width of Edge-Coloured Graphs”. In: *Theory Comput. Syst.* 52.4 (2013), pp. 599–644. doi: 10.1007/s00224-012-9399-y.
- [61] R. M. Karp and R. J. Lipton. “Some connections between nonuniform and uniform complexity classes”. In: *STOC’80*. ACM, 1980, pp. 302–309. doi: 10.1145/800141.804678.
- [62] J. Kneis, A. Langer, and P. Rossmanith. “Courcelle’s theorem - A game-theoretic approach”. In: *Discrete Optim.* 8.4 (2011), pp. 568–594. doi: 10.1016/j.disopt.2011.06.001.
- [63] D. Kozen. “Results on the propositional μ -calculus”. In: *Theoretical Computer Science* 27 (1983), pp. 333–354.
- [64] S. Kreutzer. “Algorithmic Meta-Theorems”. In: *Electronic Colloquium on Computational Complexity (ECCC)* TR09-147 (2009).
- [65] S. Kreutzer. “On the Parameterised Intractability of Monadic Second-Order Logic”. In: *CSL’09*. Vol. 5771. LNCS. Springer, 2009, pp. 348–363. doi: 10.1007/978-3-642-04027-6_26.
- [66] S. Kreutzer and S. Tazari. “Lower Bounds for the Complexity of Monadic Second-Order Logic”. In: *LICS’10*. IEEE, 2010, pp. 189–198. doi: 10.1109/LICS.2010.39.
- [67] S. Kreutzer and S. Tazari. “On Brambles, Grid-Like Minors, and Parameterized Intractability of Monadic Second-Order Logic”. In: *SODA’10*. SIAM, 2010, pp. 354–364.
- [68] M. Lampis. “Algorithmic Meta-theorems for Restrictions of Treewidth”. In: *ESA’10*. Vol. 6346. LNCS. Springer, 2010, pp. 549–560. doi: 10.1007/978-3-642-15775-2_47.
- [69] M. Lampis. “Model Checking Lower Bounds for Simple Graphs”. In: *ICALP’13*. Vol. 7965. LNCS. Springer, 2013, pp. 673–683.

BIBLIOGRAPHY

- [70] A. Langer, F. Reidl, P. Rossmanith, and S. Sikdar. “Practical algorithms for MSO model-checking on tree-decomposable graphs”. In: *Computer Science Review* 13-14 (2014), pp. 39–74. doi: 10.1016/j.cosrev.2014.08.001.
- [71] D. Lokshtanov, D. Marx, and S. Saurabh. “Lower bounds based on the Exponential Time Hypothesis”. In: *Bulletin of the EATCS* 105 (2011), pp. 41–72.
- [72] V. Lozin. “From Tree-Width to Clique-Width: Excluding a Unit Interval Graph”. In: *ISAAC’08*. Vol. 5369. LNCS. Springer, 2008, pp. 871–882.
- [73] J. A. Makowsky and J. Mariño. “Tree-width and the monadic quantifier hierarchy”. In: *Theoret. Comput. Sci.* 303.1 (2003), pp. 157–170.
- [74] R. McNaughton. “Infinite games played on finite graphs”. In: *Annals of Pure and Applied Logic* 65 (1993), pp. 149–184.
- [75] R. Motwani and M. Sudan. “Computing Roots of Graphs Is Hard”. In: *Discrete Appl. Math.* 54.1 (1994), pp. 81–88.
- [76] J. Nešetřil and P. O. de Mendez. “Tree-depth, subgraph coloring and homomorphism bounds”. In: *European J. Combin.* 27.6 (2006), pp. 1024–1041.
- [77] J. Nešetřil and P. Ossona de Mendez. *Sparsity (Graphs, Structures, and Algorithms)*. Vol. 28. Algorithms and Combinatorics. 465 pages. Springer, 2012.
- [78] J. Nešetřil and P. Ossona de Mendez. *Sparsity: Graphs, Structures, and Algorithms*. Vol. 28. Algorithms and Combinatorics. Springer, 2012.
- [79] J. Obdržálek. “Fast Mu-calculus Model Checking when Tree-width is Bounded”. In: *CAV 2003*. Vol. 2725. LNCS. Springer, 2003, pp. 80–92. doi: 10.1007/978-3-540-45069-6_7.
- [80] J. Obdržálek. “Algorithmic Analysis of Parity Games”. Submitted: January 31, 2006. Examined: May 29, 2006. PhD thesis. University of Edinburgh, 2006.
- [81] J. Obdržálek. “DAG-width – Connectivity Measure for Directed Graphs”. In: *SODA’06*. ACM-SIAM, 2006, pp. 814–821. doi: 10.1145/1109557.1109647.
- [82] J. Obdržálek. “Clique-Width and Parity Games”. In: *CSL’07*. Vol. 4646. LNCS. Springer, 2007, pp. 54–68. doi: 10.1007/978-3-540-74915-8_8.
- [83] S. Oum and P. D. Seymour. “Approximating clique-width and branch-width”. In: *J. Combin. Theory Ser. B* 96.4 (2006), pp. 514–528.

-
- [84] A. Proskurowski and J. A. Telle. “Classes of graphs with restricted interval models”. In: *Discrete Math. Theor. Comput. Sci.* 3.4 (1999), pp. 167–176.
- [85] R. Rabinovich. *Complexity Measures for Directed Graphs*. Diploma thesis, RWTH Aachen. 2008.
- [86] N. Robertson and P. D. Seymour. “Graph minors. II. Algorithmic aspects of tree-width”. In: *J. Algorithms* 7.3 (1986), pp. 309–322.
- [87] N. Robertson and P. D. Seymour. “Graph minors. X. Obstructions to tree-decomposition”. In: *J. Combin. Theory Ser. B* 52.2 (1991), pp. 153–190.
- [88] M. Safari. “D-width: A more natural measure for directed tree-width”. In: *MFCS’05*. Vol. 3618. LNCS. Springer, 2005, pp. 745–756.
- [89] D. Seese. “Linear Time Computable Problems and First-Order Descriptions”. In: *Math. Structures Comput. Sci.* 6.6 (1996), pp. 505–526.
- [90] P. D. Seymour and R. Thomas. “Graph searching and a min-max theorem for tree-width”. In: *J. Combin. Theory Ser. B* 58.1 (1993), pp. 22–33.
- [91] E. Wanke. “ k -NLC graphs and polynomial algorithms”. In: *Discrete Appl. Math.* 54 (1994), pp. 251–266.

PART II
COLLECTION OF ARTICLES

Articles in the collection

Bellow is the list (in the chronological order) of 10 research articles that were selected as the representatives of my contribution to the studied research field, together with a brief characterization of my contribution. The full texts of the articles are inserted into the corresponding appendices of the printed version of this thesis².

Paper A [79] J. Obdržálek. “Fast Mu-calculus Model Checking when Tree-width is Bounded”. In: *CAV 2003*. Vol. 2725. LNCS. Springer, 2003, pp. 80–92. DOI: 10.1007/978-3-540-45069-6_7

Contribution: Sole author.

Paper B [82] J. Obdržálek. “Clique-Width and Parity Games”. In: *CSL’07*. Vol. 4646. LNCS. Springer, 2007, pp. 54–68. DOI: 10.1007/978-3-540-74915-8_8

Contribution: Sole author.

Paper C [5] D. Berwanger, A. Dawar, P. Hunter, S. Kreutzer, and J. Obdržálek. “The DAG-width of directed graphs”. In: *J. Comb. Theory, Ser. B* 102.4 (2012), pp. 900–923. DOI: 10.1016/j.jctb.2012.04.004

Contribution: Joint journal version of my paper [81] (of which I am the sole author) and [4] of Berwanger, Dawar, Hunter and Kreutzer, published only a few months later. Both papers arrived at basically the same definition of DAG-width, and therefore we decided to produce a joint journal version. Significant contribution to editing the joint paper.

Paper D [42] R. Ganian, P. Hliněný, A. Langer, J. Obdržálek, P. Rossmanith, and S. Sikdar. “Lower bounds on the complexity of MSO_1 model-checking”. In: *J. Comput. Syst. Sci.* 80.1 (2014), pp. 180–194. DOI: 10.1016/j.jcss.2013.07.005

Contribution: Development of the ideas, part of the write-up.

Paper E [38] R. Ganian, P. Hliněný, J. Kneis, A. Langer, J. Obdržálek, and P. Rossmanith. “Digraph width measures in parameterized algorithmics”. In:

2. The full texts of the articles are excluded from the publicly available electronic version of this text to avoid copyright violations.

Discrete Appl. Math. 168 (2014), pp. 88–107. doi: 10.1016/j.dam.2013.10.038

Contribution: Development of the ideas, significant part of the write-up.

Paper F [41] R. Ganian, P. Hliněný, D. Král, J. Obdržálek, J. Schwartz, and J. Teska. “FO Model Checking of Interval Graphs”. In: *Log. Methods Comput. Sci.* 11.4 (2015). doi: 10.2168/LMCS-11(4:11)2015

Contribution: Development of the ideas, minor part of the write-up, editing the final conference version [40] and journal submission.

Paper G [36] J. Gajarský, P. Hliněný, J. Obdržálek, and S. Ordyniak. “Faster Existential FO Model Checking on Posets”. In: *Log. Methods Comput. Sci.* 11.4 (2015). doi: 10.2168/LMCS-11(4:8)2015

Contribution: Development of the ideas, minor part of the write-up.

Paper H [34] J. Gajarský, P. Hliněný, D. Lokshtanov, J. Obdržálek, S. Ordyniak, M. S. Ramanujan, and S. Saurabh. “FO Model Checking on Posets of Bounded Width”. In: *FOCS’15*. IEEE Computer Society, 2015, pp. 963–974. doi: 10.1109/FOCS.2015.63

Contribution: Development of the ideas, part of the write-up, editing the journal version.

Paper I [39] R. Ganian, P. Hliněný, J. Kneis, D. Meister, J. Obdržálek, P. Rossmanith, and S. Sikdar. “Are there any good digraph width measures?”. In: *J. Comb. Theory, Ser. B* 116 (2016), pp. 250–286. doi: 10.1016/j.jctb.2015.09.001

Contribution: The idea and its development, part of the write-up.

Paper J [35] J. Gajarský, P. Hliněný, J. Obdržálek, D. Lokshtanov, and M. S. Ramanujan. “A New Perspective on FO Model Checking of Dense Graph Classes”. In: *LICS’16*. ACM, 2016, pp. 176–184. doi: 10.1145/2933575.2935314

Contribution: Development of the ideas, minor part of the write-up.