Structure-Guided Automated Reasoning

Max Bannach
European Space Agency, Advanced Concepts Team, Noordwijk, The Netherlands

Markus Hecher
Univ. Artois, CNRS UMR 8188, Centre de Recherche en Informatique de Lens (CRIL), 6230, France
Computer Science and Artificial Intelligence Lab, Massachusetts Institute of Technology, Cambridge, MA, USA

Abstract

Algorithmic meta-theorems state that problems definable in a fixed logic can be solved efficiently on structures with certain properties. An example is Courcelle's Theorem, which states that all problems expressible in monadic second-order logic can be solved efficiently on structures of small treewidth. Such theorems are usually proven by algorithms for the model-checking problem of the logic, which is often complex and rarely leads to highly efficient solutions. Alternatively, we can solve the model-checking problem by grounding the given logic to propositional logic, for which dedicated solvers are available. Such encodings will, however, usually not preserve the input's treewidth.

This paper investigates whether all problems definable in monadic second-order logic can efficiently be encoded into SAT such that the input's treewidth bounds the treewidth of the resulting formula. We answer this in the affirmative and, hence, provide an alternative proof of Courcelle's Theorem. Our technique can naturally be extended: There are treewidth-aware reductions from the optimization version of Courcelle's Theorem to MAXSAT and from the counting version of the theorem to #SAT. By using encodings to SAT, we obtain, ignoring polynomial factors, the same running time for the model-checking problem as we would with dedicated algorithms. Another immediate consequence is a treewidth-preserving reduction from the model-checking problem of monadic second-order logic to integer linear programming (ILP). We complement our upper bounds with new lower bounds based on ETH; and we show that the block size of the input's formula and the treewidth of the input's structure are tightly linked.

Finally, we present various side results needed to prove the main theorems: A treewidth-preserving cardinality constraints, treewidth-preserving encodings from CNFs into DNFs, and a treewidth-aware quantifier elimination scheme for QBF implying a treewidth-preserving reduction from QSAT to SAT. We also present a reduction from projected model counting to #SAT that increases the treewidth by at most a factor of $2^{k+3.59}$, yielding a algorithm for projected model counting that beats the currently best running time of $2^{2^{k+4}} \cdot \mathrm{poly}(|\psi|)$.

2012 ACM Subject Classification Theory of computation → Constraint and logic programming; Theory of computation → Finite Model Theory; Theory of computation → Fixed parameter tractability

Keywords and phrases automated reasoning, treewidth, satisfiability, max-sat, sharp-sat, monadic second-order logic, fixed-parameter tractability

Digital Object Identifier 10.4230/LIPIcs.STACS.2025.15

Related Version *Technical Report*: `https://arxiv.org/abs/2312.14620`

Funding This research was funded by the Austrian Science Fund (FWF), grants J 4656 and P 32830, the Society for Research Funding in Lower Austria (GFF, Gesellschaft für Forschungsförderung NÖ) grant ExzF-0004, as well as the Vienna Science and Technology Fund (WWTF) grant ICT19-065.

Acknowledgements Part of the research was carried out while Hecher was visiting the Simons Institute for the Theory of Computing at UC Berkeley. Author names are stated alphabetically.

42nd International Symposium on Theoretical Aspects of Computer Science (STACS 2025).
Editors: Olaf Beyersdorff, Michał Pilipczuk, Elaine Pimentel, and Nguyễn Kim Thắng;
Article No. 15; pp. 15:1–15:18
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Many tools from the automated reasoning quiver can be implemented efficiently if a graphical representation of the given formula with good structural properties is given. The textbook example is the *satisfiability problem* (SAT), which can be solved in time $O\big(2^k\mathrm{poly}(|\psi|)\big)$ on formulas ψ whose *primal graph* G_ψ has *treewidth* k. (The primal graph contains a vertex for every variable of the formula and connects them if they appear together in a clause. Its treewidth intuitively measures how close it is to being a tree.) The result extends to the *maximum satisfiability problem* (MAXSAT), in which the clauses of the formula have weights and the goal is to minimize the weights of falsified clauses, and to the *model counting problem* (#SAT), in which the goal is to compute the *number* of satisfying assignments. In this article, we will use the notation $\mathrm{tower}(h,t)$ to describe a tower of twos of height h with t at the top, and $\mathrm{tower}^*(h,t)$ as shorthand to hide polynomial factors, e.g., $O\big(2^k\mathrm{poly}(|\psi|)\big) = \mathrm{tower}^*(1,k)$:

▶ **Fact 1** (folklore, see for instance [1, 4, 5, 14, 24, 29, 30])**.** *One can solve* SAT*,* MAXSAT*, and* #SAT *in time* $\mathrm{tower}^*(1,k)$ *if a width-k tree decomposition is given.*

It is worth to take some time to inspect the details of Fact 1. The hidden polynomial factor is not the subject of this paper (as indicated by the notation), but can be made as small as $O(|\varphi|)$ [10, 26]. Our focus will be the value on top of the tower, which in Fact 1 is simply "k". Under the *exponential-time hypothesis* (ETH), this is best possible.

The natural extension of the satisfiability problem to higher logic is the validity problem of fully *quantified Boolean formulas* (QSAT). While it is well-known that QSAT is fixed-parameter tractable (i.e., it is in FPT) with respect to treewidth [11], the dependencies on the treewidth is less sharp than in Fact 1. The height of the tower depends on the *quantifier alternation* $\mathrm{qa}(\psi)$ of the formula, while the top value has the form $O(k + \log k + \log\log k + \dots)$ due to the management of nested tables in the involved dynamic program.

▶ **Fact 2** ([11, 10])**.** *One can solve* QSAT *in time* $\mathrm{tower}^*\big(\mathrm{qa}(\psi)+1, O(k)\big)$ *if a width-k tree decomposition is given.*

In contrast to Fact 1, there is a big-oh on top of the tower in Fact 2. The higher order version of the model counting problem is the *projected model counting problem* (PMC), in which we need to count the number of models that are not identical on a given set of variables.

▶ **Fact 3** ([15])**.** *One can solve* PMC *in time* $\mathrm{tower}^*\big(2, k+4\big)$ *if a width-k tree decomposition is given.*

The fine art of automated reasoning is *descriptive complexity*, which studies the complexity of problems in terms of the complexity of a description of these problems; independent of any abstract machine model [21, 25]. A prominent example is *Courcelle's Theorem* that states that the problems that can be expressed in *monadic second-order logic* can be solved efficiently on instances of bounded treewidth [12]. Differently phrased, the theorem states that the *model-checking problem* for MSO logic (MC(MSO)) is fixed-parameter tractable (the parameter is the size of the formula and the treewidth of the structure):

▶ **Fact 4** ([12])**.** *One can solve* MC(MSO) *in time* $\mathrm{tower}^*\big(\mathrm{qa}(\varphi)+1, O(k+|\varphi|)\big)$ *if a width-k tree decomposition is given.*

For instance, the 3-coloring problem (Can we color the vertices of a graph with three colors such that adjacent vertices obtain different colors?) can be described by the sentence:

$$\begin{aligned}\varphi_{3\mathrm{col}} = {} & \exists R\exists G\exists B\, \forall x\forall y\,.\, (Rx \lor Gx \lor Bx) \\ & \land Exy \to \neg\big((Rx \land Ry) \lor (Gx \land Gy) \lor (Bx \land By)\big).\end{aligned}$$

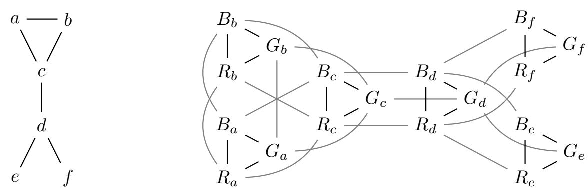
The sentence can be read aloud as: There are three colors red, blue, and green ($\exists R \exists G \exists B$) such that for all vertices x and y ($\forall x \forall y$) we have that (*i*) each vertex has at least one color ($Rx \vee Gx \vee Bx$), and (*ii*), if x and y are connected by an edge (Exy) then they do not have the same color ($\neg((Rx \wedge Ry) \vee (Gx \wedge Gy) \vee (Bx \wedge By))$). The model-checking problem MC(MSO) obtains as input a relational structure $\mathcal{S}$ (say a graph like  or ) and an MSO sentence φ (as the one from above) and asks whether $\mathcal{S}$ is a model of φ, denoted by $\mathcal{S} \models \varphi$. In our example we have  $\models \varphi_{3\mathrm{col}}$ and  $\not\models \varphi_{3\mathrm{col}}$. Using Fact 4, we can conclude from $\varphi_{3\mathrm{col}}$ that the 3-coloring problem parameterized by the treewidth lies in FPT.

Instead of utilizing Fact 4, another reasonable approach is to *ground* the MSO sentence to a propositional formula and to then apply Fact 1. Formally, this means to reduce the model checking problem MC(MSO) to SAT, i.e., given a relational structure $\mathcal{S}$ and an MSO sentence φ, we need to produce, in polynomial time, a propositional formula ψ such that $\mathcal{S} \models \varphi$ iff $\psi \in$ SAT. The naïve way of doing so is by generating an indicator variable X_u for every set variable X and every element u in the universe of $\mathcal{S}$. Then we replace every first-order $\exists$-quantifier by a "big-or" and $\forall$-quantifier by a "big-and":

$$\psi_{3\mathrm{col}} = \overbrace{\bigwedge_{u \in V(G)} \bigwedge_{v \in V(G)}}^{\forall x \forall y} \overbrace{(R_u \vee G_u \vee B_u)}^{Rx \vee Gx \vee Bx} \wedge \overbrace{\bigwedge_{\{u,v\} \in E(G)}}^{Exy \to} \neg((R_u \wedge R_v) \vee (G_u \wedge G_v) \vee (B_u \wedge B_v)).$$

propositional variables

The emerging question now is whether an automated translation such as the one we just sketched preserves treewidth in the following sense: Given a relational structure $\mathcal{S}$ of treewidth $\mathrm{tw}(\mathcal{S})$ and an MSO sentence φ, can we mechanically derive a propositional formula ψ with $\mathcal{S} \models \varphi$ iff $\psi \in$ SAT and $\mathrm{tw}(\psi) \leq f\big(\mathrm{tw}(\mathcal{S})\big)$ for some function computable $f \colon \mathbb{N} \to \mathbb{N}$? Consider for instance the following graph shown on the left (it is "almost a tree" and has treewidth 2) and the *primal graph* of $\psi_{3\mathrm{col}}$ obtained using the just sketched transformation on the right. In this example, the tree-like structure is preserved, as the treewidth gets increased by a factor of 3 and is at most 6:

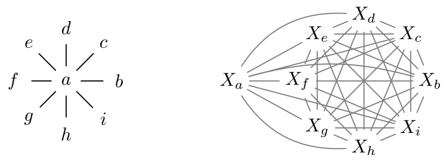


We recap this finding as the following observation: The automated grounding process from MC(MSO) to SAT *implies* a reduction from the 3-coloring problem parameterized by the input's treewidth to SAT. We can, thus, derive that the 3-coloring problem can be solved in time $\mathrm{tower}^*(1, 3k)$ using Fact 1 – without actually utilizing Courcelle's Theorem!

For a second example consider the optimization and counting version of the dominating set problem: Given a graph G, the task is either to find a minimum-size set $S \subseteq V(G)$ of vertices such that every vertex is in S or adjacent to vertex in S, or to count the number of such sets. Optimization and counting problems can be modeled in descriptive complexity by "moving" an existential second-order quantifier ("guessing" the solution) out of the sentence and making it a free variable. The task is either to find a set of minimum size such that the given structure together with this set is a model of the formula, or to count the number of such sets. For instance, the following formula describes that X is a dominating set:

$$\varphi_{\mathrm{ds}}(X) = \forall x \exists y \,\textbf{.}\, Xx \vee (Exy \wedge Xy).$$

We will also say that the formula *Fagin-defines* the property that X is a dominating set. The problem #FD(MSO) asks, given a relational structure $\mathcal{S}$ and an MSO formula with a free-set variable X, how many subsets S of the universe of $\mathcal{S}$ satisfy $\mathcal{S} \models \varphi(S)$. The optimization problem FD(MSO) gets as additional input an integer t and asks whether there is such a S with $|S| \leq t$. The reduction from MC(MSO) to SAT can be extended to a reduction from FD(MSO) to MAXSAT and from #FD(MSO) to PMC. In order to ground FD(MSO), we add new indicator variables X_u for the free-variable X and every element u of $\mathcal{S}$ (as we did for the second-order quantifiers). For FD(MSO), we additionally add a soft clause $(\neg X_u)$ for each of these variables – implying that we seek a model that minimizes $|X|$. We may now again ask: If we mechanically ground $\varphi_{\mathrm{ds}}(X)$ on a structure of bounded treewidth to a propositional formula ψ_{ds}, what can we say about the treewidth of ψ_{ds}? Unfortunately, not so much. Even if the input has treewidth 1, the primal graph of ψ_{ds} may become a clique (of treewidth n):



It follows that we can*not* derive an fpt-algorithm for the dominating set problem or its counting version by reasoning about ψ_{ds}, while we can conclude the fact from φ_{ds} using appropriate versions of Courcelle's Theorem. To summarize, we can naturally describe model-checking, optimization, and counting problems using monadic second-order logic. Using Courcelle's Theorem, we can solve all of these problems in fpt-time on structures of bounded treewidth. Alternatively, we may ground the MSO formulas to propositional logic and solve the problems using Fact 1. The produced encodings sometimes preserve the input's structure (as for 3-coloring) and, thus, themselves serve as proof that the problems lie in FPT. However, the input's structure can also get eradicated, as we observed for the dominating set problem. The present paper is concerned with the question whether there is a unifying grounding procedure that maps Fagin-defined MSO properties to propositional logic while preserving the input's treewidth.

Contribution I: Faster Structure-guided Reasoning. Before we develop a unifying, structure-aware grounding process from the model-checking problem of monadic second order logic to propositional logic, we first improve both of the underlying results. In particular, we remove the logarithmic dependencies on k in top of the tower of Fact 2 and, thus, provide the first major improvement on QBF upper bounds with respect to treewidth since 20 years:

▶ **Theorem 1** (QBF Theorem). *One can solve* QSAT *in time* $\mathrm{tower}^*\big(\mathrm{qa}(\psi)+1, k+3.92\big)$ *if a width-k tree decomposition is given.*

This bound matches the ETH lower bound for QSAT:

▶ **Fact 5** ([16]). *Unless* ETH *fails,* QSAT *cannot be solved in time* $\mathrm{tower}^*(\mathrm{qa}(\psi)+1, o(\mathrm{tw}(\psi)))$.

We will prove Theorem 1 fully in the spirit of an automated reasoning paper by an encoding into SAT. In particular, we will not need any pre-requirements other than Fact 1. With a similar encoding scheme, we will also slightly improve on Fact 3:

▶ **Theorem 2** (PMC Theorem). *One can solve* PMC *in time* $\mathrm{tower}^*\big(2, k+3.59\big)$ *if a width-k tree decomposition is given.*

▶ **Fact 6** ([15]). *Unless* ETH *fails,* PMC *cannot be solved in time* $\mathrm{tower}^*(2, o(\mathrm{tw}(\psi)))$.

Contribution II: A SAT Version of Courcelle's Theorem. We answer the main question of the introduction in the affirmative and provide a unifying, structure-aware encoding scheme from properties Fagin-defined with monadic second-order logic to variants of SAT:

▶ **Theorem 3** (A SAT Version of Courcelle's Theorem). *Assuming that the* MSO *formulas on the left side are in prenex normal form and that a width-k tree decomposition is given, there are encodings from ...*
1. MC(MSO) *to* SAT *of size* $\mathrm{tower}^*(\mathrm{qa}(\varphi), (k+9)|\varphi| + 3.92)$*;*
2. FD(MSO) *to* MAXSAT *of size* $\mathrm{tower}^*(\mathrm{qa}(\varphi)+1, (k+9)|\varphi| + 3.92)$*;*
3. #FD(MSO) *to* #SAT *of size* $\mathrm{tower}^*(\mathrm{qa}(\varphi)+1, (k+9)|\varphi| + 3.92)$*.*

All encodings of size $\mathrm{tower}^*(s,t)$ *have a treewidth of* $\mathrm{tower}(s,t)$ *and can be computed in linear time with respect to their size.*

In conjunction with Fact 1, the theorem implies Courcelle's Theorem with sharp bounds on the values on top of the tower:

▶ **Corollary 4.** *One can solve* MC(MSO) *in time* $\mathrm{tower}^*(\mathrm{qa}(\varphi)+1, (k+9)|\varphi| + 3.92)$*, and* FD(MSO) *and* #FD(MSO) *in time* $\mathrm{tower}^*(\mathrm{qa}(\varphi)+2, (k+9)|\varphi| + 3.92)$ *if a width-k tree decomposition is given.*

Since the reduction [27] from SAT to integer linear programming (ILP) is treewidth-preserving and results in an instance of bounded domain, another consequence of Theorem 3 is an "ILP Version of Courcelle's Theorem" via the dynamic program for ILP [22].

Contribution III: ETH Lower Bounds for the Encoding Size. Given that we can encode MSO definable properties into SAT while preserving the input's treewidth, we may ask next whether we can improve on the *size* of the encodings. While it is well-known that incarnations of Courcelle's Theorem have to depend on the input's treewidth and the formula's size in a non-elementary way [3] (and hence, the encodings have to be huge at some point as well), these insights do not give us precise bounds on achievable encoding sizes.

▶ **Theorem 5** (ETH Lower Bound). *Under* ETH*, there is no* SAT *encoding for* MC(MSO) *of size* $\mathrm{tower}^*(\mathrm{qa}(\varphi)-2, o(\mathrm{tw}(\mathcal{S})))$ *that can be computed in this time.*

We can make the lower bound a bit more precise in the following sense: The value at the top of the tower actually does not just depend on the treewidth $\mathrm{tw}(\mathcal{S})$, but on the product of the treewidth and the *block size* $\mathrm{bs}(\varphi)$ of the sentence φ. The block size of a formula is the maximum number of consecutive quantifiers of the same type.

▶ **Theorem 6** (Trade-off Theorem). *Under* ETH*, there is no* SAT *encoding for* MC(MSO) *of size* $\mathrm{tower}^*(\mathrm{qa}(\varphi)-2, o(\mathrm{tw}(\mathcal{S})\,\mathrm{bs}(\varphi)))$ *that can be computed within this time.*

1.1 Related Work

The concept of treewidth was discovered multiple times. The name was coined in the work by Robertson and Seymour [28], while the concept was studied by Arnborg and Proskurowski [2] under the name partial k-trees simultaneously. However, treewidth was discovered even earlier by Bertelè and Brioschi [6], and independently by Halin [19]. Courcelle's Theorem was proven in a series of articles by Bruno Courcelle [12], see also the textbook by Courcelle and Engelfriet for a detailed introduction [13]. The expressive power of monadic second-order logic was studied before, prominently by Büchi who showed that MSO over strings characterizes the regular languages [9]. Related to our treewidth-aware reduction from MC(MSO) to SAT is the work by Gottlob, Pichler, and Wei, who solve MC(MSO) using monadic Datalog [18]; and the work of Bliem, Pichler, and Woltran, who solve it using ASP [8].

1.2 Structure of this Article

We provide preliminaries in the next section, prove Theorem 1 and 2 in Section 3, and establish a SAT version of Courcelle's Theorem in Section 4. The technical details of the latter can be found in the technical report version of this article. We extend the result to Fagin-definable properties in Section 5 and provide corresponding ETH lower bounds in Section 6. We conclude and provide pointers for further research in the last section, which also contains an overview table of this article's results. Due to lack of space, most proofs are only avilable in the technical report and are replaced by a proof sketch within the main text. The corresponding positions are clearly marked with a "▼".

2 Preliminaries: Background in Logic and Structural Graph Theory

We use the notation of Knuth [23] and consider *propositional formulas* in conjunctive normal form (CNFs) like $\psi = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4 \vee \neg x_5) \wedge (x_2) \wedge (x_6)$ as *set of sets* $\{\{x_1, \neg x_2, \neg x_3\}, \{\neg x_1, x_4, \neg x_5\}, \{x_2\}, \{x_6\}\}$. We denote the sets of variables, literals, and clauses of ψ as $\mathrm{vars}(\psi)$, $\mathrm{lits}(\psi)$, and $\mathrm{clauses}(\psi)$. A *(partial) assignment* is a subset $\beta \subseteq \mathrm{lits}(\psi)$ such that $|\{x, \neg x\} \cap \beta| \leq 1$ for all $x \in \mathrm{vars}(\psi)$, that is, a set of literals that does not contain both polarities of any variable. We use $\beta \sqsubseteq \mathrm{vars}(\psi)$ to denote partial assignments. The formula *conditioned under a partial assignment* β is denoted by $\psi|\beta$ and obtained by removing all clauses from ψ that contain a literal $l \in \beta$ and by removing all literals l' with $\neg l' \in \beta$ from the remaining clauses. A assignment is *satisfying* for a CNF ψ if $\psi|\beta = \emptyset$, and it is *contradicting* if $\emptyset \in \psi|\beta$. A DNF is a disjunction of conjunctions, i.e., a set of terms. We use the same notations as for CNFs, however, in $\psi|\beta$ we delete terms that contain a literal that appears negated in β and remove the literals in β from the remaining terms. Hence, β is *satisfying* if $\emptyset \in \psi|\beta$, and *contradicting* if $\psi|\beta = \emptyset$.

The *model counting problem* asks to compute the number of satisfying assignments of a CNF and is denoted by #SAT. In *projected model counting* (PMC) we count the number of models that are not identical on a given set of variables. In the *maximum satisfiability problem* (MAXSAT) we partition the clauses of ψ into a set $\mathrm{hard}(\psi)$ of *hard clauses* and a set $\mathrm{soft}(\psi)$ of weighted *soft clauses*, i.e., every clause $C \in \mathrm{soft}(\psi)$ comes with a *weight* $w(C) \in \mathbb{Q}$. The formula is then called a WCNF and the goal is to find under all assignments $\beta \sqsubseteq \mathrm{vars}(\psi)$ with $\mathrm{hard}(\psi)|\beta = \emptyset$ the one that maximizes $\sum_{C \in \mathrm{soft}(\psi), \{C\}|\beta = \emptyset} w(c)$. In a fully *quantified Boolean formula* (a QBF, also called a *second-order propositional sentence*) all variables are bounded by existential or universal quantifiers. Throughout the paper we assume that QBFs are in prenex normal form, meaning that all quantifers appear in the front of a quantifier-free formula called the *matrix*. As is customary, we assume that the matrix is a CNF if the last (i.e., most inner) quantifier is existential, and a DNF otherwise. A QBF is *valid* if it evaluates to true (see Chapter 29–31 in [7]). Define QSAT to be the problem of deciding whether a given QBF is valid.

2.1 Descriptive Complexity

A *vocabulary* is a finite set $\tau = \{R_1^{a_1}, R_2^{a_2}, \dots, R_\ell^{a_\ell}\}$ of relational symbols R_i of arity a_i. A (finite, relational) *τ-structure* $\mathcal{S}$ is a tuple $\big(U(\mathcal{S}), R_1^{\mathcal{S}}, R_2^{\mathcal{S}}, \dots, R_\ell^{\mathcal{S}}\big)$ with *universe* $U(\mathcal{S})$ and *interpretations* $R_i^{\mathcal{S}} \subseteq U(\mathcal{S})^{a_i}$. The *size* of $\mathcal{S}$ is $|\mathcal{S}| = |U(\mathcal{S})| + \sum_{i=1}^{\ell} a_i \cdot |R_i^{\mathcal{S}}|$. We denote the *set of all τ-structures* by $\mathrm{STRUC}[\tau]$ – e.g., $\mathrm{STRUC}[\{E^2\}]$ is the set of directed graphs.

Let τ be a vocabulary and $x_0, x_1, x_2, \dots$ be an infinite repertoire of first-order variables. The *first-order language* $\mathcal{L}(\tau)$ is inductively defined, where the *atomic formulas* are the strings $x_i = x_j$ and $R_i(x_1, \dots, x_{a_i})$ for relational symbols $R_i \in \tau$. If $\alpha, \beta \in \mathcal{L}(\tau)$ then so are

$\neg(\alpha)$, $(\alpha \wedge \beta)$, and $\exists x_i(\alpha)$. A variable that appears next to $\exists$ is called *quantified* and *free* otherwise. We denote a formula $\varphi \in \mathcal{L}(\tau)$ with $\varphi(x_{i_1}, \dots, x_{i_q})$ if $x_{i_1}, \dots, x_{i_q}$ are precisely the free variables in φ. A formula without free variables is called a *sentence.* As customary, we extend the language of first-order logic by the usual abbreviations, e.g., $\alpha \rightarrow \beta \equiv \neg\alpha \vee \beta$ and $\forall x_i(\alpha) \equiv \neg\exists x_i(\neg\alpha)$. To increase readability, we will use other lowercase Latin letters for variables and drop unnecessary braces by using the usual operator precedence instead. Furthermore, we use the *dot notation* in which we place a "**.**" instead of an opening brace and silently close it at the latest syntactically correct position. A τ-structure $\mathcal{S}$ is a *model* of a sentence $\varphi \in \mathcal{L}(\tau)$, denoted by $\mathcal{S} \models \varphi$, if it evaluates to true under the semantics of quantified propositional logic while interpreting equality and relational symbols as specified by the structure. For instance, $\varphi_{\text{undir}} = \forall x \forall y \,\mathbf{.}\, Exy \rightarrow Eyx$ over $\tau = \{E^2\}$ describes the set of undirected graphs, and we have  $\models \varphi_{\text{undir}}$ and  $\not\models \varphi_{\text{undir}}$.

We obtain the language of *second-order logic* by allowing quantification over relational variables of arbitrary arity, which we will denote by uppercase Latin letters. A relational variable is said to be *monadic* if its arity is one. A *monadic second-order* formula is one in which all quantified relational variables are monadic. The set of all such formulas is denoted by MSO. The *model checking problem* for a vocabulary τ is the set $\text{MC}_\tau(\text{MSO})$ that contains all pairs $(\mathcal{S}, \varphi)$ of τ-structures $\mathcal{S}$ and MSO sentences φ with $\mathcal{S} \models \varphi$. Whenever τ is not relevant (meaning that a result holds for all fixed τ), we will refer to the problem as MC(MSO). We note that in the literature there is often a distinction between MSO_1- and MSO_2-logic, which describes the way the input is encoded [20]. Since we allow arbitrary relations, we do not have to make this distinction.

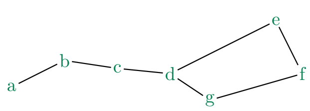
2.2 Treewidth and Tree Decompositions

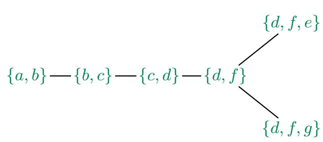
While we consider graphs G as relational structures $\mathcal{G}$ as discussed in the previous section, we also use common graph-theoretic terminology and denote with $V(G) = U(\mathcal{G})$ and $E(G) = E^{\mathcal{G}}$ the vertex and edge set of G. Unless stated otherwise, graphs in this paper are *undirected* and we use the natural set notations and write, for instance, $\{v, w\} \in E(G)$. The *degree* of a vertex is the number of its neighbors. A *tree decomposition* of G is a pair (T, χ) in which T is a tree (a connected graph without cycles) and $\chi\colon V(T) \rightarrow 2^{V(G)}$ a function with the following two properties:

1. for every $v \in V(G)$ the set $\{\, x \mid v \in \chi(x) \,\}$ is non-empty and connected in T;
2. for every $\{u, v\} \in E(G)$ there is at least one node $x \in V(T)$ with $\{u, v\} \subseteq \chi(x)$.

The *width* of a tree decomposition is the maximum size of its bags minus one, i.e., $\text{width}(T, \chi) = \max_{x \in V(T)} |\chi(x)| - 1$. The *treewidth* $\text{tw}(G)$ of a graph G is the minimum width any tree decomposition of G must have. We do not require additional properties of tree decompositions, but we assume that T is rooted at a $\text{root}(T) \in V(T)$ and, thus, that nodes $t \in V(T)$ may have a $\text{parent}(t) \in V(T)$ and $\text{children}(t) \subseteq V(T)$. Without loss of generality, we may also assume $|\text{children}(t)| \leq 2$.

▶ **Example 7.** The treewidth of the Big Dipper constellation (as graph shown on the left) is at most two, as proven by the tree decomposition on the right:





2.3 Treewidth of Propositional Formulas and Relational Structures

The definition of treewidth can be lifted to other objects by associating a graph to them. The most common graph for CNFs (or DNFs) ψ is the *primal graph* G_ψ, which is the graph on vertex set $V(G_\psi) = \text{vars}(\psi)$ that connects two vertices by an edge if the corresponding variables appear together in a clause. We then define $\text{tw}(\psi) \coloneqq \text{tw}(G_\psi)$ and refer to a tree decomposition of G_ψ as one of ψ. Note that other graphical representations lead to other definitions of the treewidth of propositional formulas. A comprehensive listing can be found in the *Handbook of Satisfiability* [7, Chapter 17]. A *labeled tree decomposition* (T, χ, λ) extends a tree decomposition with a mapping $\lambda\colon V(T) \to 2^\psi$ (i.e., a mapping from the nodes of T to a subset of the clauses (or terms) of ψ) such that for every clause (or term) C there is exactly one $t \in V(T)$ with $C \in \lambda(t)$ that contains all variables appearing in C. It is easy to transform a tree decomposition (T, χ) into a labeled one (T, χ, λ) by traversing the tree once's and by duplicating some bags. Hence, we will assume throughout this article that all tree decompositions are labeled.

A similar approach can be used to define tree decompositions of arbitrary structures: The *primal graph* $G_\mathcal{S}$ of a structure $\mathcal{S}$, in this context also called the *Gaifman graph*, has as vertex set the universe of $\mathcal{S}$, i.e., $V(G_\mathcal{S}) = U(\mathcal{S})$, and contains an edge $\{u, v\} \in E(G_\mathcal{S})$ iff u and v appear together in some tuple of $\mathcal{S}$. As before, we define $\text{tw}(\mathcal{S}) \coloneqq \text{tw}(G_\mathcal{S})$. One can alternatively define the concept of tree decompositions directly over relational structures, which leads to the same definition [17].

3 New Upper Bounds for Second-Order Propositional Logic

Central to our reductions are treewidth-preserving encodings from QSAT to SAT and from PMC to #SAT. These encoding establishes new proofs of Chen's Theorem [11] and the theorem by Fichte et al. [15], and improve the dependencies on k in the tower of Fact 2 and 3.

3.1 Treewidth-Aware Encodings from QSAT to SAT

We use a quantifier elimination scheme that eliminates the most-inner quantifier block at the cost of introducing $O(2^k|\varphi|)$ new variables while increasing the treewidth by a factor of $12 \cdot 2^k$. Let first $\varphi = Q_1 S_1 \dots \forall_\ell S_\ell \,.\, \psi$ be the given QBF, in which ψ is a DNF. Let further (T, χ, λ) be the given labeled width-k tree decomposition of φ. We describe an encoding into a QBF, in which the last quantifier block $Q_\ell S_\ell$ gets replaced by new variables in $S_{\ell-1}$.

We have to encode the fact that for an assignment on $\bigcup_{i=1}^{\ell-1} S_i$ *all* assignments to S_ℓ satisfy ψ, i.e., at least *one* term in ψ. For that end, we introduce auxiliary variables for every term $d \in \text{terms}(\psi)$ and any partial assignment α of the variables in S_ℓ that also appear in the bag that contains d. More precisely, let $\lambda^{-1}(d)$ be the node in $V(T)$ with $d \in \lambda(t)$ and let $\alpha \sqsubseteq \chi(\lambda^{-1}(d)) \cap S_\ell$ be an assignment of the variables of the bag that are quantified by Q_ℓ. We introduce the variable sat_d^α that indicates that this assignment satisfies d:

$$\bigwedge_{d \in \text{terms}(\psi)} \bigwedge_{\substack{\alpha \sqsubseteq \chi(\lambda^{-1}(d)) \cap S_\ell \\ \{d\}|\alpha \neq \emptyset}} \Big[sat_d^\alpha \leftrightarrow \bigwedge_{x \in \text{lits}(\{d\}|\alpha)} x \Big], \qquad \text{// } \alpha \text{ may satisfy } d \tag{1}$$

$$\bigwedge_{d \in \text{terms}(\psi)} \bigwedge_{\substack{\alpha \sqsubseteq \chi(\lambda^{-1}(d)) \cap S_\ell \\ \{d\}|\alpha = \emptyset}} \Big[\neg sat_d^\alpha \Big]. \qquad \text{// } \alpha \text{ falsifies } d \tag{2}$$

We have to track whether ψ can be satisfied by a local assignment α. For every $t \in V(T)$ and every $\alpha \sqsubseteq \chi(t) \cap S_\ell$ we introduce a variable $sat_{\leq t}^\alpha$ that indicates that α can be extended to a satisfying assignment for the subtree rooted at t. Furthermore, we create variables

$sat^{\alpha}_{<t,t'}$ for $t' \in \text{children}(t)$ that propagate the information about satisfiability along the tree decomposition. That is, $sat^{\alpha}_{<t,t'}$ is set to true if there is an assignment $\beta \sqsubseteq \chi(t') \cap S_\ell$ that can be extended to a satisfying assignment and that is compatible with α:

$$\text{// Either there is a term satisfing the bag or we can propagate:}$$
$$\bigwedge_{t \in V(T)} \bigwedge_{\alpha \sqsubseteq \chi(t) \cap S_\ell} \Big[sat^{\alpha}_{\leq t} \leftrightarrow \bigvee_{d \in \lambda(t)} sat^{\alpha}_{d} \vee \bigvee_{t' \in \text{children}(t)} sat^{\alpha}_{<t,t'} \Big], \tag{3}$$

$$\text{// Propagate satisfiability:}$$
$$\bigwedge_{t \in V(T)} \bigwedge_{\alpha \sqsubseteq \chi(t) \cap S_\ell} \bigwedge_{t' \in \text{children}(t)} \Big[sat^{\alpha}_{<t,t'} \leftrightarrow \bigwedge_{\substack{\beta \sqsubseteq \chi(t') \cap S_\ell \\ \beta \cap \text{lits}(\chi(t)) = \alpha \cap \text{lits}(\chi(t'))}} sat^{\beta}_{\leq t'} \Big]. \tag{4}$$

Finally, since $Q_\ell = \forall$, we need to ensure that for all possible assignments of S_ℓ there is at least one term that gets satisfied. Since satisfiability gets propagated to the root of the tree decomposition by the aforementioned constraint, we can enforce this property with:

$$\bigwedge_{\alpha \sqsubseteq \chi(\text{root}(T)) \cap S_\ell} sat^{\alpha}_{\leq \text{root}(T)}. \tag{5}$$

The following lemma observes the correctness of the construction, and the subsequent lemma handles the case $Q_\ell = \exists$.

▶ **Lemma 8** (▼). *There is an algorithm that, given a* QBF $\varphi = Q_1 S_1 \dots \exists_{\ell-1} S_{\ell-1} \forall_\ell S_\ell \,.\, \psi$ *and a width-k tree decomposition of G_φ, outputs in time $O^*(2^k)$ a* QBF $\varphi' = Q_1 S_1 \dots \exists_{\ell-1} S'_{\ell-1} \,.\, \psi'$ *and a width-$(12 \cdot 2^k)$ tree decomposition of $G_{\varphi'}$ such that φ is valid iff φ' is valid.*

▶ **Lemma 9** (▼). *There is an algorithm that, given a* QBF $\varphi = Q_1 S_1 \dots \forall_{\ell-1} S_{\ell-1} \exists_\ell S_\ell \,.\, \psi$ *and a width-k tree decomposition of G_φ, outputs in time $O^*(2^k)$ a* QBF $\varphi' = Q_1 S_1 \dots \forall_{\ell-1} S'_{\ell-1} \,.\, \psi'$ *and a width-$(12 \cdot 2^k)$ tree decomposition of $G_{\varphi'}$ such that φ is valid iff φ' is valid.*

Sketch of Proof. The case $Q_\ell = \exists$ (in which ψ is a CNF) works similarly: The result follows by negating the inverse, where the roles of CNF and DNF are switched, and universal and existential quantification are switched as well. ◀

Proof of Theorem 1. The theorem follows by exhaustively applying Lemma 8 and Lemma 9 until a CNF is reached. The price for removing one alternation are $O(2^k|\varphi|)$ new variables and an increase of the treewidth by a factor of $12 \cdot 2^k$. Hence, after removing one quantifier block we have a treewidth of $12 \cdot 2^k \leq 2^{k+\log 12}$, after two we have $12 \cdot 2^{2^{k+\log 12}} \leq 2^{2^{k+\log 12}+\log 12}$, after three we then have $2^{2^{2^{k+\log 12}+\log 12}+\log 12}$; and so on. We can bound all the intermediate "$+\log 12$" by adding a "$+1$" on top of the tower, leading to a bound on the treewidth of $\text{tower}(\text{qa}(\varphi), k + \log 12 + 1) \leq \text{tower}(\text{qa}(\varphi), k + 4.59)$. In fact, we can bound the top of the tower even tighter by observing $\log 12 \leq 3.59$ and guessing 3.92 as a fix point. Inserting yields $3.59 + 2^{3.59+k} \leq 2^{3.92+k}$ and $2^{3.92+2^{3.59+k}} \leq 2^{2^{3.92+k}}$. Consequently, we can bound the treewidth of the encoding by $\text{tower}(\text{qa}(\varphi), k+3.92)$ and the size by $\text{tower}^*(\text{qa}(\varphi), k+3.92)$. ◀

3.2 Treewidth-Aware Encodings from PMC to #SAT

Recall that the input for PMC is a CNF ψ and a set $X \subseteq \text{vars}(\psi)$. The task is to count the assignments $\alpha \sqsubseteq X$ that can be extended to models $\alpha^* \sqsubseteq \text{vars}(\psi)$ of ψ. We can also think of a formula $\psi(X) = \exists Y \,.\, \psi'(X, Y)$ with free variables X and existential quantified variables Y

(ψ' is quantifier-free), for which we want to count the assignments to X that make the formula satisfiable. The idea is to rewrite $\psi(X) = \exists Y . \psi'(X,Y) \equiv \exists X \exists Y . \psi'(X,Y)$, and to use a similar encoding as in the proof of Lemma 9 to remove the second quantifier.

In detail, we add a variable sat_c^α for every clause $c \in \text{clauses}(\psi)$ and every assignment of the corresponding bag $\alpha \sqsubseteq \chi(\lambda^{-1}(c)) \cap Y$. The semantic of this variable is that the clause c is satisfiable under the partial assignment α. We further add the propagation variables $sat_{\leq t}^\alpha$ and $sat_{<t,t'}^\alpha$ for all $t \in V(T)$, $t' \in \text{children}(t)$, and $\alpha \sqsubseteq \chi(\lambda^{-1}(c)) \cap Y$. The former indicates that the assignment α can be extended to a satisfying assignment of the subtree rooted at t; the later propagates partial solutions from children to parents within the tree decomposition:

// α may satisfy c:

$$\bigwedge_{c \in \text{clauses}(\psi)} \bigwedge_{\substack{\alpha \sqsubseteq \chi(\lambda^{-1}(c)) \cap Y \\ \{c\}|\alpha \neq \emptyset}} \Big[sat_c^\alpha \leftrightarrow \bigvee_{\ell \in \text{lits}(\{c\}|\alpha)} \ell \Big], \tag{1}$$

// α satisfies c:

$$\bigwedge_{c \in \text{clauses}(\psi)} \bigwedge_{\substack{\alpha \sqsubseteq \chi(\lambda^{-1}(c)) \cap Y \\ \{c\}|\alpha = \emptyset}} \Big[sat_c^\alpha \Big]. \tag{2}$$

// Either there is a clause satisfying the bag or we can propagate:

$$\bigwedge_{t \in V(T)} \bigwedge_{\alpha \sqsubseteq \chi(t) \cap Y} \Big[sat_{\leq t}^\alpha \leftrightarrow \bigwedge_{c \in \lambda(t)} sat_c^\alpha \wedge \bigwedge_{t' \in \text{children}(t)} sat_{<t,t'}^\alpha \Big], \tag{3}$$

// Propagate satisfiability:

$$\bigwedge_{t \in V(T)} \bigwedge_{\alpha \sqsubseteq \chi(t) \cap Y} \bigwedge_{t' \in \text{children}(t)} \Big[sat_{<t,t'}^\alpha \leftrightarrow \bigwedge_{\substack{\beta \sqsubseteq \chi(t') \cap Y \\ \beta \cap \text{lits}(\chi(t)) = \alpha \cap \text{lits}(\chi(t'))}} sat_{\leq t'}^\beta \Big]. \tag{4}$$

Observe that the constraints (1)–(4) contain no variable from Y (we removed them by locally speaking about α) and, furthermore, constraints (1), (3), and (4) are pure propagations, which leave no degree of freedom on the auxiliary variables. Hence, models of these constraint only have freedom in the variables in X within constraint (2). We are left with the task to count only models that actually satisfy the input formula, which we achieve with:

$$\bigvee_{\alpha \sqsubseteq \chi(\text{root}(T)) \cap Y} sat_{\leq \text{root}(T)}^\alpha. \tag{5}$$

▶ **Lemma 10** (▼)**.** *There is an algorithm that, given a* CNF ψ*, a set* $X \subseteq \text{vars}(\psi)$*, and a width-*$k$ *tree decomposition of* G_ψ*, outputs in time* $O^*(2^k)$ *a* CNF ψ' *and a width-*$(12 \cdot 2^k)$ *tree decomposition of* $G_{\psi'}$ *such that the projected model count of* ψ *on* X *equals* $\#(\psi')$*.*

Proof of Theorem 2. By applying Fact 1 to the formula generated by Lemma 10 we obtain an algorithm for PMC with running time $\text{tower}^*(2, k + 3.59)$. ◀

4 A SAT Version of Courcelle's Theorem

We demonstrate the power of treewidth-aware encodings by providing an alternative proof of Courcelle's theorem. We prove the main part of Theorem 3 in the following form:

▶ **Lemma 11.** *There is an algorithm that, given a relational structure $\mathcal{S}$, a width-k tree decomposition of $\mathcal{S}$, and an* MSO *sentence φ in prenex normal form, produces in time* $\mathrm{tower}^*(\mathrm{qa}(\varphi), (k+9)|\varphi| + 3.92)$ *a propositional formula ψ and tree decomposition of G_ψ of width* $\mathrm{tower}(\mathrm{qa}(\varphi), (k+9)|\varphi| + 3.92)$ *such that $\mathcal{S} \models \varphi \Leftrightarrow \psi \in$* SAT.

The lemma assumes that the sentence is in prenex normal form with a quantifier-free part ψ in CNF, i.e., $\varphi \equiv Q_1 S_1 \dots Q_{q-1} S_{q-1} Q_q s_q \dots Q_\ell s_\ell \,\text{.}\, \bigwedge_{i=1}^{p} \psi_i$ with $Q_i \in \{\exists, \forall\}$ and S_i (s_i) being second-order (first-order) variables. The requirement that the second-order quantifers appear before the first-order ones is for sake of presentation, the encoding works as is if the quantifiers are mixed. The main part of the proof is a treewidth-aware encoding from MC(MSO) into QSAT; which is then translated to SAT using Theorem 1.

4.1 Auxiliary Encodings

Let ψ be a propositional formula and $X \subseteq \mathrm{lits}(\psi)$ be an arbitrary set of literals. A *cardinality constraint* $\mathrm{card}_{\bowtie c}(X)$ with $\bowtie \in \{\leq, =, \geq\}$ ensures that { at most, exactly, at least } c literals of X get assigned to true. Classic encodings of cardinality constraints increase the treewidth of ψ by quite a lot. For instance, the naive encoding for $\mathrm{card}_{\leq 1}(X) \equiv \bigwedge_{u,v \in X; u \neq v} (\neg u \vee \neg v)$ completes X into a clique. We encode a cardinality constraint without increasing the treewidth by distributing a *sequential unary counter:*

▶ **Lemma 12 (▼).** *For every $c \geq 0$ we can, given a* CNF *ψ, a set $X \subseteq \mathrm{lits}(\psi)$, and a width-$k$ tree decomposition of ψ, encode $\mathrm{card}_{\bowtie c}(X)$ such that $\mathrm{tw}(\psi \wedge \mathrm{card}_{\bowtie c}(X)) \leq k + 3c + 3$.*

Sketch of Proof. We add $c+1$ variables to every bag t of the tree decomposition, which count the number of literals set to true in the subtree rooted at t. The semantics of the sequential counter encoding [31] is then implemented along the edges of the decomposition. To cover the new constraints, we can add the auxiliary variables of the (at most two) children of t to the bag of t as well, resulting in an overall increase of the treewidth by $3c+3$. ◀

The second auxiliary encoding is a treewidth-preserving conversion from CNFs to DNFs.

▶ **Lemma 13.** *There is a polynomial-time algorithm that, given a* CNF *ψ and a width-k tree decomposition of G_ψ, produce a* DNF *ψ' and a width-$(k+4)$ tree decomposition of $G_{\psi'}$ such that for any $\alpha \sqsubseteq \mathrm{vars}(\psi)$, $\psi|\alpha = \emptyset$ iff $\psi'|\alpha$ is a tautology ($\neg(\psi'|\alpha)$ is unsatisfiable).*

Sketch of Proof. For every clause C we add a variable f_C that is true iff C is satisfied. Satisfiability is encoded along the tree by variables $f_{\leq t}$ indicating that ψ is satisfied in the subtree rooted at t via $\bigvee_{t \in V(T)} \neg \Big[f_{\leq t} \leftrightarrow \bigwedge_{C \in \lambda(t)} f_C \wedge \bigwedge_{t' \in \mathrm{children}(t)} f_{\leq t'} \Big]$. ◀

4.2 Indicator Variables for the Quantifiers

To prove Lemma 11 we construct a QBF for a given MSO sentence φ, structure $\mathcal{S}$, and tree decomposition of $\mathcal{S}$. We first define the primary variables of ψ, i.e., the prefix of ψ (primary here refers to the fact that we will also need some auxiliary variables later). For every second-order quantifier $\exists X$ or $\forall X$ we introduce, as we did in the introduction, an indicator variable X_u for every element $u \in U(\mathcal{S})$ with the semantic that X_u is true iff $u \in X$. These variables are either existentially or universally quantified, depending on the second-order quantifier. If there are multiple quantifiers (say $\exists X \forall Y$), the order in which the variables are quantified is the same as the order of the second-order quantifiers. For first-order quantifiers $\exists x$ or $\forall x$ we do the same construction, i.e., we add variables x_u for all $u \in U(\mathcal{S})$ with the semantics that x_u is true iff x was assigned to u. Of course, of these variables we have to set *exactly one* to true, which we enforce by adding $\mathrm{card}_{=1}(\{x_u \mid u \in U(\mathcal{S})\})$ using Lemma 12.

4.3 Evaluation of Atoms

The last ingredient of our QBF encoding is the evaluation of the atoms in the MSO sentence φ. An atom is $Rx_1,\dots,x_a$ for a relational symbol R from the vocabulary of arity a, containment in a second-order variable Xu, equality $x=y$, and the negation of the aforementioned. For every atom ι that appears in φ we introduce variables p_t^ι and $p_{\leq t}^\iota$ for all $t\in V(T)$ that indicate that ι is true in bag t or somewhere in the subtree rooted at t, respectively. Note that the same atom can occur multiple times in φ, for instance in

$$\forall x\forall y\exists z\,\boldsymbol{.}(x=y\rightarrow x=z)\vee(x=y\rightarrow y=z)$$

there are two atoms $x=y$. However, since φ is in prenex normal form (and, thus, variables cannot be rebound), these always evaluate in exactly the same way. Hence, it is sufficient to consider the *set of atoms*, which we denote by $\operatorname{atoms}(\varphi)$. We can propagate information about the atoms along the tree decomposition with:

$$\bigwedge_{t\in V(T)}\ \bigwedge_{\iota\in\operatorname{atoms}(\varphi)}\Big[p_{\leq t}^\iota\leftrightarrow(p_t^\iota\vee\bigvee_{t'\in\operatorname{children}(t)}p_{\leq t'}^\iota)\Big].$$

This encoding introduces two variables per atom ι per bag t (namely p_t^ι and $p_{\leq t}^\iota$), which increases the treewidth by at most $2\cdot|\operatorname{atoms}(\varphi)|$. To synchronize with the two children t' and t'', we add $p_{\leq t'}^\iota$ and $p_{\leq t''}^\iota$ to $\chi(t)$, yielding a total treewidth of at most $4\cdot|\operatorname{atoms}(\varphi)|$.

An easy atom to evaluate is $x=y$, since if x and y are equal (i.e., they both got assigned to the same element $u\in U(\mathcal{S})$), we can conclude this fact within a bag that contains u:

$$\bigwedge_{t\in V(T)}\Big[p_t^{x=y}\leftrightarrow\bigvee_{u\in\chi(t)}(x_u\wedge y_u)\Big].$$

For every $u\in U(\mathcal{S})$ and every quantifier $\exists x$ (or $\forall x$), we add the propositional variable x_u to all bags containing u. We increase the treewidth by at most the quantifier rank and, in return, cover constraints as the above trivially. Similarly, if there is a second-order variable X and a first-order variable x, the atom Xx can be evaluated locally in every bag:

$$\bigwedge_{t\in V(T)}\Big[p_t^{X_x}\leftrightarrow\bigvee_{u\in\chi(t)}(X_u\wedge x_u)\Big].$$

We have to evaluate atoms corresponding to relational symbols R of the vocabulary. For each such symbol of arity a we encode:

$$\bigwedge_{t\in V(T)}\Big[p_t^{R(x_1,x_2,\dots,x_a)}\leftrightarrow\bigvee_{\substack{u_1,\dots,u_a\in\chi(t)\\(u_1,\dots,u_a)\in R^{\mathcal{S}}}}\big((x_1)_{u_1}\wedge(x_2)_{u_2}\wedge\cdots\wedge(x_a)_{u_a}\big)\Big].$$

Here "$R(x_1,x_2,\dots,x_a)$" is an atom in which R is a relational symbol and $x_1,x_2,\dots,x_a$ are quantified first-order variables. In the inner "big-or" we consider all $u_1,\dots,u_a$ in $\chi(t)$, i.e., elements $u_1,\dots,u_a\in U(\mathcal{S})$ that are in the relation $(u_1,\dots,u_a)\in R^{\mathcal{S}}$. Then "$(x_i)_{u_i}$" is a variable that describes that x_i gets assigned to u_i. Note that all tuples in $R^{\mathcal{S}}$ appear together in at least one bag of the tree decomposition and, hence, there is at least one bag t for which $p_t^{R(x_1,x_2,\dots,x_a)}$ can be evaluated to true. The propagation ensures that, for every $\iota\in\operatorname{atoms}(\varphi)$, the variable $p_{\leq\operatorname{root}(T)}^\iota$ will be true iff ι is true. Since the quantifier-free part of φ is a CNF $\bigwedge_{j=1}^p\psi_j$, we can encode it by replacing every occurrence of ι in ψ_j with $p_{\leq\operatorname{root}(T)}^\iota$.

4.4 The Full Encoding in one Figure

For the readers convenience, we compiled the encoding into Figure 1. Combining the insights of the last sections proves Lemma 11, but if the inner-most quantifier is universal, existentially projecting the encoding variables would produce a QBF with one more block. This can, however, be circumvent using Lemma 13. We formally prove that "combining the insights" indeed leads to a sound proof of Lemma 11 in the technical report.

Cardinality Propagation

$$c^x_{\leq t} \leftrightarrow \bigvee_{u\in\chi(t)\setminus\chi(\mathrm{parent}(t))} x_u \;\vee \bigvee_{t'\in\mathrm{children}(t)} c^x_{\leq t'} \qquad \text{for every } t \text{ in } T, x \in \{s_q,\dots,s_\ell\} \quad (1)$$

At-Least-One Constraint

$$c^x_{\leq \mathrm{root}(T)} \qquad \text{for every } x \in \{s_q,\dots,s_\ell\} \quad (2)$$

At-Most-One Constraint

$$\neg x_u \vee \neg x_{u'} \qquad \text{for every } t \text{ in } T, u,u' \in \chi(t), u \neq u', x \in \{s_q,\dots,s_\ell\} \quad (3)$$

$$\neg x_u \vee \neg c^x_{\leq t'} \qquad \text{for every } t \text{ in } T, t' \in \mathrm{children}(t), u \in \chi(t)\setminus\chi(\mathrm{parent}(t)), x\in\{s_q,\dots,s_\ell\} \quad (4)$$

$$\neg c^x_{\leq t'} \vee \neg c^x_{\leq t''} \qquad \text{for every } t \text{ in } T, t',t'' \in \mathrm{children}(t), t' \neq t'', x \in \{s_q,\dots,s_\ell\} \quad (5)$$

Proofs of MSO Atoms

$$p^{x=y}_t \leftrightarrow \bigvee_{u\in\chi(t)} (x_u \wedge y_u) \qquad \text{for every } t \text{ in } T, x,y \in \{s_q,\dots,s_\ell\}, (x=y)\in \mathrm{atoms}(\varphi) \quad (6)$$

$$p^{X(x)}_t \leftrightarrow \bigvee_{u\in\chi(t)} (X_u \wedge x_u) \qquad \text{for every } t \text{ in } T, X\in\{S_1,\dots,S_{q-1}\}, x\in\{s_q,\dots,s_\ell\}, X(x)\in \mathrm{atoms}(\varphi) \quad (7)$$

$$p^{R(x_1,\dots,x_a)}_t \leftrightarrow \bigvee_{\substack{u_1,\dots,u_a\in\chi(t)\\(u_1,\dots,u_a)\in R^{\mathcal{S}}}} ((x_1)_{u_1} \wedge \dots \wedge (x_a)_{u_a}) \qquad \text{for every } t \text{ in } T, \{x_1,\dots,x_a\} \subseteq \{s_q,\dots,s_\ell\}, R \in \mathcal{S}, R(x_1,\dots,x_r) \in \mathrm{atoms}(\varphi) \quad (8)$$

$$p^{\iota}_{\leq t} \leftrightarrow p^{\iota}_t \vee \bigvee_{t'\in\mathrm{children}(t)} p^{\iota}_{\leq t'} \qquad \text{for every } t \text{ in } T, \iota \in \mathrm{atoms}(\varphi) \quad (9)$$

Deriving MSO Atoms requires Proof

$$\iota \leftrightarrow p^{\iota}_{\leq \mathrm{root}(T)} \qquad \text{for every } t \text{ in } T, \iota \in \mathrm{atoms}(\varphi) \quad (10)$$

Verify MSO Formula

$$\psi \qquad (11)$$

Figure 1 The reduction $\mathcal{R}_{\text{MSO}\to\text{QSAT}}(\varphi,\mathcal{S},\mathcal{T})$ that takes as input an MSO formula in prenex normal form $\varphi = Q_1S_1\dots Q_{q-1}S_{q-1}Q_qs_q\dots Q_\ell s_\ell \,.\, \psi$ and a structure $\mathcal{S}$ with a TD $\mathcal{T}=(T,\chi)$ of $\mathcal{S}$ of width k. It obtains a QBF $\varphi' = Q_1S'_1\dots Q_\ell S'_\ell \exists E' \,.\, \psi'$, where ψ' is the conjunction of Equations (1)–(11), $S'_i = \{(S_i)_u \mid u \in U(\mathcal{S})\}$ and $E' = \mathrm{vars}(\psi') \setminus (\bigcup_{i=1}^{\ell} S'_i)$. Formula ψ' can be easily converted into CNF of width linear in k (for constant-size MSO formulas φ).

5 Fagin Definability via Automated Reasoning

In this section we prove the remaining two items of Theorem 3, i.e., a treewidth-aware encoding of the optimization version of Courcelle's Theorem to MAXSAT; and a #SAT encoding of the counting version of the theorem. The general approach is as follows: We obtain a MSO formula

$\varphi(X)$ with a free set variable X as input (rather than a MSO sentence as in Lemma 11). The objective of the model-checking problems adds requirements to this variable (for FD(MSO) we seek a $S \subseteq U(\mathcal{S})$ of minimum size such that $\mathcal{S} \models \varphi(S)$; for #FD(MSO) we want to count the number of sets $S \subseteq U(\mathcal{S})$ with $\mathcal{S} \models \varphi(S)$). The "trick" is to rewrite $\varphi(X) = \xi$ as $\varphi' = \exists X \xi$ and apply Lemma 11 to φ' in order to obtain a propositional formula ψ. Observe that the quantifier alternation of φ' may be one larger than the one of φ.

▶ **Lemma 14 (▼).** *There is an algorithm that, given a structure $\mathcal{S}$ with weights $w_i \colon U(\mathcal{S}) \to \mathbb{Q}$ for $i \in \{1, \dots, \ell\}$, a width-k tree decomposition of $\mathcal{S}$, and an* MSO *formula $\varphi(X_1, \dots, X_\ell)$ in prenex normal form, produces in time $\text{tower}^*(\text{qa}(\varphi) + 1, (k+9)|\varphi| + 3.92)$ a* WCNF *ψ and a tree decomposition of width $\text{tower}(\text{qa}(\varphi) + 1, (k+9)|\varphi| + 3.92)$ of G_ψ such that the maximum weight of any model of ψ equals the maximum value of $\sum_{i=1}^{\ell} \sum_{s \in S_i} w_i(s)$ under $S_1, \dots, S_\ell \subseteq U(\mathcal{S})$ with $\mathcal{S} \models \varphi(S_1, \dots, S_\ell)$.*

Sketch of Proof. Consider $\psi \wedge \bigwedge_{u \in U(\mathcal{S})} (\neg X_u)$ such that the clauses in ψ are *hard* and the added clauses are *soft*. A model maximizing the soft clauses will minimize the number of X_u variables set to true, i.e., corresponds to a minimum-size set S with $\mathcal{S} \models \varphi(S)$. ◀

▶ **Lemma 15 (▼).** *There is an algorithm that, given a relational structure $\mathcal{S}$, a width-k tree decomposition of $\mathcal{S}$, and an* MSO *formula $\varphi(X_1, \dots, X_\ell)$ in prenex normal form, produces in time $\text{tower}^*(\text{qa}(\varphi) + 1, (k+9)|\varphi| + 3.92)$ a* CNF *ψ and a tree decomposition of width $\text{tower}(\text{qa}(\varphi) + 1, (k+9)|\varphi| + 3.92)$ of G_ψ such that the number of models of ψ equals the number of sets $S_1, \dots, S_\ell \subseteq U(\mathcal{S})$ with $\mathcal{S} \models \varphi(S_1, \dots, S_\ell)$.*

Sketch of Proof. We need to compute the number of models of ψ *projected* to the X_u variables. In other words, it is sufficient to solve the *projected model counting problem* on the instance generated with Lemma 11 using Lemma 10. ◀

6 Lower Bounds for the Encoding Size of Model Checking Problems

We companion our SAT encodings for MC(MSO) with lower bounds on the achievable encoding size under ETH. The first lower bound (Theorem 5) is obtained by an encoding from QSAT into MC(MSO) that implies that SAT encodings of MC(MSO) lead to faster QSAT algorithms.

▶ **Lemma 16 (▼).** *There is a polynomial-time algorithm that, given a* QSAT *sentence ψ, outputs a structure $\mathcal{S}$ and an* MSO *sentence φ with $\text{tw}(\mathcal{S}) \leq \text{tw}(\psi) + 1$ and $\text{qa}(\varphi) \leq \text{qa}(\psi) + 2$ such that $\mathcal{S} \models \varphi$ iff ψ evaluates to true.*

Sketch of Proof. The structure $\mathcal{S}$ is the *incidence* graph of ψ (the graph containing a node for every variable and every clause that connects variables to the clauses containing them) with some additional labels. The sentence φ uses $\text{qa}(\psi)$ second-order quantifier to guess the assignment of ψ, and one additional $\forall x \exists y$-block to evaluate it. ◀

Proof of Theorem 5. Combine Lemma 16 with Fact 5. ◀

6.1 An Encoding for Compressing Treewidth

For QSAT one can "move" complexity from the quantifier rank of the formula to its treewidth and *vice versa* [16]. By Lemma 16, this means that any reduction from QSAT to MC(MSO) may produce an instance with small treewidth or quantifier alternation while increasing the other. We show that one can also decrease the treewidth by increasing the *block size*.

▶ **Lemma 17** (▼). *For every $c > 0$ there is a polynomial-time algorithm that, on input of a* CNF ψ *and a width-k tree decomposition of G_ψ, outputs a constant-size* MSO *sentence φ with $\mathrm{qa}(\varphi) = 2$ and $\mathrm{bs}(\varphi) = c$, and a structure $\mathcal{S}$ with $\mathrm{tw}(\mathcal{S}) \leq \lceil \frac{k+1}{c} \rceil$ such that $\psi \in$* SAT *$\Leftrightarrow \mathcal{S} \models \varphi$.*

Sketch of Proof. The idea of the proof is to *(i)* encode the input's formula as an incidence graph over which we reason with an MSO sentence; we then *(ii)* replace this structure by its tree decomposition with additional "sync edges"; and finally we *(iii)* contract vertices in the tree decomposition to lower the treewidth, while we encode statements like $x \in S$ (for a set variable S) by defining new set variables $S_1, \dots, S_c$ and by interpreting $y \in S_i$ as "the ith vertex contracted to y is in S". ◀

Proof of Theorem 6. We obtain the Trade-off Theorem by combining the proof strategy of Lemma 17 with the reduction from QSAT to MC(MSO) of Lemma 16. The result is a polynomial-time algorithm for every $c > 0$ that, on input of a QBF ψ and a width-k tree decomposition of G_ψ, outputs a constant-size MSO sentence φ with $\mathrm{qa}(\varphi) \leq \mathrm{qa}(\psi) + 2$ and $\mathrm{bs}(\varphi) = c$, and a structure $\mathcal{S}$ with $\mathrm{tw}(\mathcal{S}) \leq \lceil \frac{k+1}{c} \rceil$ such that ψ is valid iff $\mathcal{S} \models \varphi$. ◀

It is out of the scope of this article, but worth mentioning, that the proofs of Lemma 17 and Theorem 6 can be generalized to the following finite-model theoretic result:

▶ **Proposition 18.** *For every $c > 0$ there is a polynomial-time algorithm that, given a relational structure $\mathcal{S}$, a width-k tree decomposition of $\mathcal{S}$, and an* MSO *sentence φ, outputs a structure $\mathcal{S}'$ and a sentence φ' such that:*

1. $\mathcal{S} \models \varphi \iff \mathcal{S}' \models \varphi'$;
2. $\mathrm{tw}(\mathcal{S}') \leq \lceil \frac{k+1}{c} \rceil$;
3. $\mathrm{bs}(\varphi') \leq c \cdot \mathrm{bs}(\varphi)$.

7 Conclusion and Further Research

We studied *structure-guided automated reasoning*, where we utilize the input's structure in propositional encodings. The scientific question we asked was whether we can encode every MSO definable problem on structures of bounded treewidth into SAT formulas of bounded treewidth. We proved this in the affirmative, implying an alternative proof of Courcelle's Theorem. The most valuable aspects are, in our opinion, the simplicity of the proof (it is "just" an encoding into propositional logic) and the potential advantages in practice for formulas of small quantifier alternation (SAT solvers are known to perform well on instances of small treewidth, even if they do not actively apply techniques such as dynamic programming). Another advantage is the surprisingly simple generalization to the optimization and counting version of Courcelle's Theorem – we can directly "plug in" MAXSAT or #SAT and obtain the corresponding results. As a byproduct, we also obtain new proofs showing (purely as encodings into propositional logic) that QSAT parameterized by the input's treewidth plus quantifier alternation is fixed-parameter tractable (improving a complex dynamic program with nested tables) and that PMC parameterized by treewidth is fixed-parameter tractable (improving a multi-pass dynamic program). Table 1 provides an overview of the encodings presented within this article.

Our encodings are exponentially smaller than the best known running time for MC(MSO), i.e., when we solve the instances using Fact 1, we obtain the same runtime. We complemented this finding with new ETH-based lower bounds. Further research will be concerned with closing the remaining gap in the height of the tower between the lower and upper bounds. We show in an upcoming paper that the terms "$\mathrm{qa}(\varphi)$" in Theorem 3 and "$\mathrm{qa}(\varphi) - 2$" in

Table 1 We summarize the encodings presented within this article. An encoding maps *from* one problem *to* another. The third and fourth columns define the treewidth and size of the encoding, whereby we assume that $c > 0$ is a constant, a width-k tree decomposition is given, ψ is a propositional formula, and φ is a fixed MSO formula.

Encoding... *From*	*To*	*Treewidth*	*Size*	*Reference*
QSAT	SAT	$\mathrm{tower}(\mathrm{qa}(\psi), k+3.92)$	$\mathrm{tower}^*(\mathrm{qa}(\psi), k+3.92)$	Theorem 1
PMC	#SAT	$\mathrm{tower}(1, k+3.59)$	$\mathrm{tower}^*(1, k+3.59)$	Theorem 2
$\mathrm{card}_{\bowtie c}(X)$	SAT	$k+3c+3$	$O(c\|\psi\|)$	Lemma 12
CNF	DNF	$k+4$	$O(\|\psi\|)$	Lemma 13
MC(MSO)	SAT	$\mathrm{tower}(\mathrm{qa}(\varphi), (9k+9)\|\varphi\|+3.92)$	$\mathrm{tower}^*(\mathrm{qa}(\varphi), (9k+9)\|\varphi\|+3.92)$	Lemma 11
FD(MSO)	MAXSAT	$\mathrm{tower}(\mathrm{qa}(\varphi)+1, (9k+9)\|\varphi\|+3.92)$	$\mathrm{tower}^*(\mathrm{qa}(\varphi)+1, (9k+9)\|\varphi\|+3.92)$	Lemma 14
#FD(MSO)	#SAT	$\mathrm{tower}(\mathrm{qa}(\varphi)+1, (9k+9)\|\varphi\|+3.92)$	$\mathrm{tower}^*(\mathrm{qa}(\varphi)+1, (9k+9)\|\varphi\|+3.92)$	Lemma 15
SAT	MC(MSO)	$\lceil \frac{k+1}{c} \rceil$	$O(k\|\psi\|)$	Lemma 17
QSAT	MC(MSO)	$\lceil \frac{k+1}{c} \rceil$	$O(k\|\psi\|)$	Theorem 6

Theorem 5 can be replaced by “$\mathrm{qa}_2(\varphi)$” on *guarded* formulas, i.e., formulas in which there are only two first-order quantifers that are only allowed to quantify edges. Here, $\mathrm{qa}_2(\varphi)$ refers to the quantifier alternation of the second-order quantifers only. Hence, on such guarded formulas (e.g., on all examples in the introduction), the bounds are tight. Another task that remains for further research is to evaluate the encodings in practice. This would also be interesting for the auxiliary encodings, e.g., can a treewidth-aware cardinality constraint compete with classical cardinality constraint?

References

1 Michael Alekhnovich and Alexander A. Razborov. Satisfiability, Branch-Width and Tseitin Tautologies. *Comput. Complex.*, 20(4):649–678, 2011. doi:10.1007/S00037-011-0033-1.

2 Stefan Arnborg and Andrzej Proskurowski. Problems on Graphs with Bounded Decomposability. *Bull. EATCS*, 25:7–10, 1985.

3 Albert Atserias and Sergi Oliva. Bounded-width QBF is PSPACE-complete. *Journal of Computer and System Sciences*, 80(7):1415–1429, 2014. doi:10.1016/j.jcss.2014.04.014.

4 Fahiem Bacchus, Shannon Dalmao, and Toniann Pitassi. Algorithms and Complexity Results for #SAT and Bayesian Inference. In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*, pages 340–351, 2003. doi:10.1109/SFCS.2003.1238208.

5 Max Bannach, Malte Skambath, and Till Tantau. On the Parallel Parameterized Complexity of MaxSAT Variants. In *25th International Conference on Theory and Applications of Satisfiability Testing, SAT 2022, August 2-5, 2022, Haifa, Israel*, pages 19:1–19:19, 2022. doi:10.4230/LIPIcs.SAT.2022.19.

6 Umberto Bertelè and Francesco Brioschi. On Non-serial Dynamic Programming. *J. Comb. Theory, Ser. A*, 14(2):137–148, 1973. doi:10.1016/0097-3165(73)90016-2.

7 Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors. *Handbook of Satisfiability, Second Edition*. IOS Press, 2021. doi:10.3233/FAIA336.

8 Bernhard Bliem, Reinhard Pichler, and Stefan Woltran. Declarative Dynamic Programming as an Alternative Realization of Courcelle’s Theorem. In *Parameterized and Exact Computation – 8th International Symposium, IPEC 2013, Sophia Antipolis, France, September 4-6, 2013, Revised Selected Papers*, pages 28–40, 2013. doi:10.1007/978-3-319-03898-8_4.

9 J. Richard Büchi. Weak Second-Order Arithmetic and Finite Automata. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 6(1-6):66–92, 1960.

10 Florent Capelli and Stefan Mengel. Tractable QBF by Knowledge Compilation. In *36th International Symposium on Theoretical Aspects of Computer Science, STACS 2019, March 13-16, 2019, Berlin, Germany*, pages 18:1–18:16, 2019. `doi:10.4230/LIPICS.STACS.2019.18`.

11 Hubie Chen. Quantified Constraint Satisfaction and Bounded Treewidth. In *Proceedings of the 16th Eureopean Conference on Artificial Intelligence, ECAI'2004, including Prestigious Applicants of Intelligent Systems, PAIS 2004, Valencia, Spain, August 22-27, 2004*, pages 161–165, 2004.

12 Bruno Courcelle. The Monadic Second-Order Logic of Graphs. I. Recognizable Sets of Finite Graphs. *Inf. Comput.*, 85(1):12–75, 1990. `doi:10.1016/0890-5401(90)90043-H`.

13 Bruno Courcelle and Joost Engelfriet. *Graph Structure and Monadic Second-Order Logic – A Language-Theoretic Approach*, volume 138 of *Encyclopedia of mathematics and its applications*. Cambridge University Press, 2012.

14 Adnan Darwiche. Decomposable Negation Normal Form. *J. ACM*, 48(4):608–647, 2001. `doi:10.1145/502090.502091`.

15 Johannes Klaus Fichte, Markus Hecher, Michael Morak, Patrick Thier, and Stefan Woltran. Solving Projected Model Counting by Utilizing Treewidth and its Limits. *Artif. Intell.*, 314:103810, 2023. `doi:10.1016/j.artint.2022.103810`.

16 Johannes Klaus Fichte, Markus Hecher, and Andreas Pfandler. Lower Bounds for QBFs of Bounded Treewidth. In *LICS '20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science, Saarbrücken, Germany, July 8-11, 2020*, pages 410–424, 2020. `doi:10.1145/3373718.3394756`.

17 Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006. `doi:10.1007/3-540-29953-X`.

18 Georg Gottlob, Reinhard Pichler, and Fang Wei. Monadic Datalog Over Finite Structures of Bounded Treewidth. *ACM Trans. Comput. Log.*, 12(1):3:1–3:48, 2010. `doi:10.1145/1838552.1838555`.

19 Rudolf Halin. S-functions For Graphs. *Journal of geometry*, 8(1):171–186, 1976.

20 Petr Hlinený, Sang-il Oum, Detlef Seese, and Georg Gottlob. Width Parameters Beyond Treewidth and their Applications. *Comput. J.*, 51(3):326–362, 2008. `doi:10.1093/comjnl/bxm052`.

21 Neil Immerman. *Descriptive Complexity*. Graduate texts in computer science. Springer, 1999. `doi:10.1007/978-1-4612-0539-5`.

22 Bart M. P. Jansen and Stefan Kratsch. A Structural Approach to Kernels for ILPs: Treewidth and Total Unimodularity. In *Algorithms – ESA 2015 – 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*, pages 779–791, 2015. `doi:10.1007/978-3-662-48350-3_65`.

23 Donald E. Knuth. *The Art of Computer Programming*, volume 4, Fascicle 6. Addison-Wesley, 2016.

24 Tuukka Korhonen and Matti Järvisalo. Integrating Tree Decompositions into Decision Heuristics of Propositional Model Counters (Short Paper). In *27th International Conference on Principles and Practice of Constraint Programming, CP 2021, Montpellier, France (Virtual Conference), October 25-29, 2021*, pages 8:1–8:11, 2021. `doi:10.4230/LIPICS.CP.2021.8`.

25 Stephan Kreutzer. Algorithmic meta-theorems. In *Finite and Algorithmic Model Theory*, pages 177–270. Cambridge University Press, 2011.

26 Michael Lampis, Stefan Mengel, and Valia Mitsou. QBF as an Alternative to Courcelle's Theorem. In *Theory and Applications of Satisfiability Testing – SAT 2018 – 21st International Conference, SAT 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 9-12, 2018, Proceedings*, pages 235–252, 2018. `doi:10.1007/978-3-319-94144-8_15`.

27 Ruiming Li, Dian Zhou, and Donglei Du. Satisfiability and Integer Pprogramming as Complementary Tools. In *Proceedings of the 2004 Conference on Asia South Pacific Design Automation: Electronic Design and Solution Fair 2004, Yokohama, Japan, January 27-30, 2004*, pages 879–882, 2004. `doi:10.1109/ASPDAC.2004.178`.

28 Neil Robertson and Paul D. Seymour. Graph Minors. II. Algorithmic Aspects of Tree-width. *Journal of Algorithms*, 7(3):309–322, 1986. doi:10.1016/0196-6774(86)90023-4.

29 Sigve Hortemo Sæther, Jan Arne Telle, and Martin Vatshelle. Solving #SAT and MAXSAT by Dynamic Programming. *J. Artif. Intell. Res.*, 54:59–82, 2015. doi:10.1613/jair.4831.

30 Marko Samer and Stefan Szeider. Algorithms for Propositional Model Counting. *J. Discrete Algorithms*, 8(1):50–64, 2010. doi:10.1016/j.jda.2009.06.002.

31 Carsten Sinz. Towards an Optimal CNF Encoding of Boolean Cardinality Constraints. In *Principles and Practice of Constraint Programming – CP 2005, 11th International Conference, CP 2005, Sitges, Spain, October 1-5, 2005, Proceedings*, pages 827–831, 2005. doi:10.1007/11564751_73.