

Solving Quantified Boolean Formulas with Few Existential Variables

LEIF ERIKSSON, Department of Computer and Information Science, Linköping University, Sweden

VICTOR LAGERKVIST, Department of Computer and Information Science, Linköping University, Sweden

GEORGE OSIPOV, Department of Computer and Information Science, Linköping University, Sweden

SEBASTIAN ORDYNIAK, School of Computing, University of Leeds, UK

FAHAD PANOLAN, School of Computing, University of Leeds, UK

MATEUSZ RYCHLICKI, School of Computing, University of Leeds, UK

The *quantified Boolean formula* (QBF) problem is an important decision problem generally viewed as the archetype for PSPACE-completeness. Many problems of central interest in AI are in general not included in NP, e.g., planning, model checking, and non-monotonic reasoning, and for such problems QBF has successfully been used as a modelling tool. However, solvers for QBF are not as advanced as state of the art SAT solvers, which has prevented QBF from becoming a universal modelling language for PSPACE-complete problems. A theoretical explanation is that QBF (as well as many other PSPACE-complete problems) lacks natural *parameters* guaranteeing *fixed-parameter tractability* (FPT).

In this paper we tackle this problem and consider a simple but overlooked parameter: the number of existentially quantified variables. This natural parameter is virtually unexplored in the literature which one might find surprising given the general scarcity of FPT algorithms for QBF. Via this parameterization we then develop a novel FPT algorithm applicable to QBF instances in conjunctive normal form (CNF) of bounded clause length. We complement this by a W[1]-hardness result for QBF in CNF of unbounded clause length as well as sharper lower bounds for the bounded arity case under the (strong) *exponential-time hypothesis*.

1 INTRODUCTION

The *quantified Boolean formula* (QBF) problem is the decision problem of verifying a formula $Q_1x_1 \dots Q_nx_n.\varphi(x_1, \dots, x_n)$, where $\varphi(x_1, \dots, x_n)$ is a propositional formula and $Q_i \in \{\forall, \exists\}$ for each i . Throughout, we write QBFSAT (respectively QBF-DNF) for the subproblem restricted to formulas in conjunctive normal form (respectively disjunctive normal form), and d -QBFSAT for the problem where clauses have maximum size $d \geq 1$. From a theoretical angle, the QBF problem serves as a foundational example of PSPACE-completeness, and by restricting the quantifier alternations, we get examples of complete problems for any class in the polynomial hierarchy. From a more practical point of view, the field of applied QBF solving has developed on the shoulders of SAT solving, which has seen tremendous advances in the last decade [17]. Arguably, the *raison d'être* behind SAT solving is not only to solve a specific NP-complete problem faster than exhaustive search, but to provide a combinatorial framework applicable to any problem which can be reduced to SAT. This naturally includes any problem in NP but if one considers stronger reductions than Karp reductions (e.g., by allowing a superpolynomial running time or by viewing the SAT solver as an oracle) more problems fall under the umbrella of SAT. However, this approach is not always optimal, and for many problems of importance in artificial intelligence, e.g., planning, model checking, and non-monotonic reasoning [26], this approach is not ideal since the best reductions in the literature incur an exponential overhead. These problems are instead more naturally formulated via QBF. However, this comes with the downside that QBF solvers, despite steady advances, are not nearly as advanced

Authors' addresses: Leif Eriksson, leif.eriksson@liu.se, Department of Computer and Information Science, Linköping University, Linköping, Sweden; Victor Lagerkvist, victor.lagerkvist@liu.se, Department of Computer and Information Science, Linköping University, Linköping, Sweden; George Osipov, george.osipov@liu.se, Department of Computer and Information Science, Linköping University, Linköping, Sweden; Sebastian Ordyniak, sordyniak@gmail.com, School of Computing, University of Leeds, Leeds, UK; Fahad Panolan, fahad.panolan@gmail.com, School of Computing, University of Leeds, Leeds, UK; Mateusz Rychlicki, mkrychlicki@gmail.com, School of Computing, University of Leeds, Leeds, UK.

as their SAT brethren. For more information about applied QBF solving we refer the reader to the handbook by Biere et al. [4].

To bridge the gap between SAT and QBF solving we need algorithmic breakthroughs for the latter. In this paper we analyze QBF from a *theoretical* perspective and are therefore interested in obtaining unconditionally improved algorithms. To analyze the complexity of QBF we use the influential paradigm of *parameterized complexity* where the goal is to identify structural properties of instances, represented by natural numbers called *parameters*, such that one can effectively solve instances with bounded parameter size. More formally, for every instance I of a computational problem we associate a parameter $k \in \mathbb{N}$ with it and then we are primarily interested in algorithms with a running time bounded by $f(k) \cdot \text{poly}(|I|)$ for a computable function $f: \mathbb{N} \rightarrow \mathbb{N}$ depends on k and a polynomial function poly depends on the length $|I|$ of the input I . Such algorithms are said to be *fixed-parameter tractable* (FPT). Thus, while f is generally going to be superpolynomial, an FPT algorithm may still be very competitive in practice if the parameter is sufficiently small. For problems in NP there exists a wealth of results [11], but for PSPACE-complete problems the landscape is rather scarce in comparison since there are fewer natural parameters to choose from. For example, the go-to parameter for NP problems is *tree-width*, which measures how close a graph is to being a tree, and which is typically sufficient to produce an FPT algorithm. But this fails for QBF where it is even known that QBF is PSPACE-complete for constant primal tree-width [1]. Here, the situation becomes more manageable if one simultaneously bounds e.g. the number of quantifier alternations [9] but for the general QBF problem the few FPT results that exists are primarily with respect to more exotic parameters such as prefix pathwidth [15] and respectful treewidth [1] which takes the ordering of the quantifiers into account. Two interesting counter examples are (1) the FPT algorithm parameterized by primal vertex cover number by [15], further optimized and simplified by [22], and (2) the backdoor approach in [23] which generalizes the classical tractable fragments of QBF in an FPT setting. Moreover, very recently, tractability has been shown for two parameters that fall between vertex cover number and treewidth [18]. Thus, while FPT results for QBF and related problems exist, they are comparably few in number and generally defined with respect to more complicated structural properties. Do simple parameters not exist, or have we been investigating the wrong ones?

In this paper we demonstrate that a natural (and previously overlooked) parameter *does* exist: the number of existentially quantified variables. Thus, we bound the number of existentially quantified variables but otherwise make no restrictions on the prefix. While quantifier elimination techniques have been an important tool ever since the early days of QBF solving [2, 3] (see also Janota & Marques-Silva [20] for a more recent discussion and a comparison to Q -resolution) the predominant focus has been to expand universal quantifiers since the removal of universal quantifiers produces instances that can be solved with classical SAT techniques (however, methods for expanding existential quantifiers have also been tried in practice [5]). This is, for example, made explicit by Szeider & de Haan [12] who prove that QBF-DNF parameterized by the number of universally quantified variables is FPT-reducible to SAT (parameterized by the number of all variables). However, they only prove para-NP-completeness of the problem which in the world of parameterized complexity is a far cry away from FPT. Conversely, we show that concentrating on existential variable elimination is much more lucrative since in this case one can construct an FPT algorithm once all existential variables have been removed. Let us also remark that if a QBFSAT instance contains k existential variables and $n - k$ universal variables, then the combination n is not a relevant parameter: it makes the problem technically FPT but there are very few applications where one would expect the total number of variables to be bounded. Moreover, while QBFSAT for arbitrary but constant quantifier depth admits a moderately improved algorithm with a running time of the form $2^{n-n^{\Omega(1)}}$ [25], not even 3-QBFSAT restricted to universal followed by existential quantifiers admits an exponentially improved $2^{\varepsilon n} \cdot \text{poly}(|\phi|)$ algorithm for $\varepsilon < 1$ under the so-called *strong exponential-time hypothesis* [8].

After having defined the basic notions (Section 2), we obtain our major results in Section 3 as follows. First, given an instance $Q_1x_1 \dots \exists x_i \dots Q_nx_n\phi(x_1, \dots, x_n)$ we can remove the existential quantifier for x_i by creating a disjunction of the two subinstances (with fresh variables) obtained by fixing x_i to 0 or 1. This extends to a quantifier elimination preprocessing scheme which reduces to the problem of checking whether a disjunction of k formulas in d -CNF is a tautology or not. We call this problem OR-CNF TAUTOLOGY and then construct an FPT algorithm by converting it to the problem of finding an independent set in a certain well-structured graph (CLAUSE-GRAPH INDEPENDENT SET). The latter problem has a strong combinatorial flair and we manage to construct a kernel with at most $kd!((k-1)d+1)^d$ vertices via the *sunflower lemma* of Erdős & Rado [16]. Put together, this results in an FPT algorithm with a running time of $2^{O(k2^k)} \cdot |\phi|$ for QBFSAT with constant clause size when parameterized by the number of existentially quantified variables. At a first glance, this might not look terribly impressive compared to the naive $2^n \cdot |\phi|$ algorithm obtained by branching on $n-k$ universal and k existential quantifiers, but we stress that the FPT algorithm is suitable for applications where the total number of existentially quantified variables is kept relatively small. Under this constraint our algorithm shows that one can effectively ignore the cost of universally quantified variables and solve the instance in polynomial time. Via an FPT reduction we also demonstrate (in Section 4) that our FPT result straightforwardly can be extended to the more general problem *quantified constraint satisfaction* where variables take values in arbitrary finite domains.

In Section 5 we show that the clause size dependency in our FPT algorithm is necessary under the conjecture that $\text{FPT} \neq \text{W}[1]$, which is widely believed conjecture in parameterized complexity. Specifically we show that QBFSAT, when parameterized by the number of existentially quantified variables (but not the arity), is $\text{W}[1]$ -hard by reducing from the MULTIPARTITE INDEPENDENT SET problem. Moreover, under the Exponential Time Hypothesis (ETH), the same reduction rules out algorithms with running time $f(k) \cdot 2^{o(2^k)}$ for every computable function $f: \mathbb{N} \rightarrow \mathbb{N}$. We then proceed to establish a sharper lower bound under the (Strong) ETH for the specific case of clause size 3. First, we have an easy $2^{o(k)}$ lower bound for 3-QBFSAT (under the exponential-time hypothesis) since 3-SAT can be viewed as a special case of 3-QBFSAT. However, under the strong exponential-time hypothesis we prove a markedly stronger bound: there is *no* constant c such that 3-QBFSAT (even for only two quantifier blocks) is solvable in c^k time, i.e. the problem is not solvable in $2^{O(k)}$ time. Thus, while our $2^{O(k2^k)}$ algorithm likely can be improved to some extent, we should not hope to obtain a $2^{O(k)}$ algorithm. This proof is based on an interesting observation: any instance with comparably few number of universally quantified variables can be solved by enumerating all possible assignments to the universal variables and then solving the remaining part with a 3-SAT algorithm. Thus, as also remarked by Calabro et al. [8], instances with few existential variables are in a certain sense harder, which makes our FPT algorithm in Section 3 all the more surprising.

We close the paper with a discussion in Section 6. Most importantly, our FPT algorithm shows tractability of new classes of previously hard QBFs, and it would be interesting to investigate whether similar parameters could be used to study other hard problems outside NP. A promising candidate is the NEXPTIME-complete problem obtained by extending Boolean formulas with *Henkin quantifiers*, resulting in the *dependency QBF* (DQBF) formalism.

2 PRELIMINARIES

In this section we briefly cover the necessary background on parameterized complexity and quantified Boolean formulas. We assume familiarity with the basics of graph theory, cf. [13]. We use the notation $[n]$ for the set $\{1, \dots, n\}$ for every $n \in \mathbb{N}$.

Computational Complexity. We follow [14] and [19] in our presentation. Let Σ be a finite alphabet. A parameterized problem L is a subset of $\Sigma^* \times \mathbb{N}$. The problem L is *fixed-parameter tractable* (or, in FPT) if there is an algorithm deciding whether an instance $(I, k) \in \Sigma^* \times \mathbb{N}$ is in L in time $f(k) \cdot |I|^c$, where f is some computable function and c is a constant independent of (I, k) .

Let $L, L' \subseteq \Sigma^* \times \mathbb{N}$ be two parameterized problems. A mapping $P : \Sigma^* \times \mathbb{N} \rightarrow \Sigma^* \times \mathbb{N}$ is a *fixed-parameter (FPT) reduction from L to L'* if there exist computable functions $f, p : \mathbb{N} \rightarrow \mathbb{N}$ and a constant c such that the following conditions hold:

- $(I, k) \in L$ if and only if $P(I, k) = (I', k') \in L'$,
- $k' \leq p(k)$, and
- $P(I, k)$ can be computed in $f(k) \cdot |I|^c$ time.

Let $L \subseteq \Sigma^* \times \mathbb{N}$ be a parameterized problem. A *kernelization (algorithm)* for L is an algorithm that takes $(I, k) \in \Sigma^* \times \mathbb{N}$ as input and in time polynomial in $|I, k|$, outputs $(I', k') \in \Sigma^* \times \mathbb{N}$ such that:

- $(I, k) \in L$ if and only if $(I', k') \in L$, and
- $|I'|, k' \leq h(k)$ for some computable function h .

The output (I', k') of the kernelization algorithm is called a *kernel*. Clearly, if L is decidable and admits a kernel, L is in FPT, and the converse also holds (see e.g. [19, Theorem 4]).

The class $W[1]$ contains all problems that admit FPT reductions from INDEPENDENT SET parameterized by the solution size, i.e. the number of vertices in the independent set. Under the standard assumption that $FPT \neq W[1]$, we can show that a problem is not fixed-parameter tractable by proving its $W[1]$ -hardness, i.e. by providing an FPT reduction from INDEPENDENT SET to the problem. The class XP contains all parameterized problems that can be solved in time $n^{f(k)}$ for instances with input size n , parameter k and some computable function f .

For sharper lower bounds, stronger assumptions are sometimes necessary. Here, we will primarily consider the *exponential-time hypothesis* which states that the 3-SAT problem is not solvable in *subexponential time* when parameterized by the number of variables n or the number of clauses m . To make this more precise, for $d \geq 3$ let c_d denote the infimum of all constants c such that d -SAT is solvable in 2^{cn} time by a deterministic algorithm; then the ETH states that $c_3 > 0$. The *strong exponential-time hypothesis* (SETH) additionally conjectures that the limit of the sequence c_3, c_4, \dots tends to 1, which in particular is known to imply that the satisfiability problem for clauses of arbitrary length (CNF-SAT) is not solvable in 2^{cn} time for *any* $c < 1$ [7].

Quantified Boolean Formulas. Boolean expressions, formulas, variables, literals, conjunctive normal form (CNF) and clauses are defined in the standard way (cf. [4]). We treat 1 and 0 as the truth-values “true” and “false”, respectively, and clauses in CNF as sets of literals. We will assume that no clause contains a literal twice or a literal and its negation, and no clause is repeated in any formula. A *quantified Boolean formula* is of the form $Q.\phi$, where $Q = Q_1x_1 \dots Q_nx_n$ with $Q_i \in \{\forall, \exists\}$ for all $1 \leq i \leq n$ is the (*quantifier*) *prefix*, x_1, \dots, x_n are variables, and ϕ is a propositional Boolean formula on these variables called the *matrix*. If $Q_i = \forall$, we say that x_i is a *universal variable*, and if $Q_i = \exists$, we say that x_i is an *existential variable*.

For a Boolean formula ϕ , a variable x and value $b \in \{0, 1\}$, define $\phi[x = b]$ to be the formulas obtained from ϕ by replacing every occurrence of x with b . The truth value of a formula $Q.\phi$ is defined recursively. Let $Q = Q_1x_1 \dots Q_nx_n$ and $Q' = Q_2x_2 \dots Q_nx_n$. Then $Q.\phi$ is *true* if

- $n = 0$ and ϕ is a true Boolean expression, or

- $Q_1 = \exists$ and $Q'.\phi[x_1 = 0]$ or $Q'.\phi[x_1 = 1]$ is true, or
- $Q_1 = \forall$ and $Q'.\phi[x_1 = 0]$ and $Q'.\phi[x_1 = 1]$ are true.

Otherwise, $Q.\phi$ is false.

A formula $Q.\phi$ is a QCNF if ϕ is in CNF, and a Qd-CNF if ϕ is in d -CNF, i.e. a CNF where every clause has size at most d . For a clause C in a QCNF, we use C^\exists and C^\forall to denote the restriction of C to existential and universal variables, respectively. If ϕ is in CNF, then $\phi[x = 0]$ can be simplified by removing every clause containing literal \bar{x} , and removing literal x from the remaining clauses. Analogously, $\phi[x = 1]$ is simplified by removing every clause containing the literal x , and removing the literal \bar{x} from the remaining clauses.

Let $Q.\phi$ be a formula such that $Q_1 = \dots = Q_i = \forall$ and $Q_{i+1} = \dots = Q_n = \exists$ for some $1 \leq i \leq n$. Then we write that $Q.\phi$ is a $\forall\exists$ BF, and replace BF by CNF and d -CNF if ϕ is in CNF and d -CNF, respectively.

Following the convention in the literature, we write QBFSAT for the problem of deciding whether a QCNF is true, and by d -QBFSAT, $\forall\exists$ QBFSAT and $\forall\exists d$ -QBFSAT we denote the same problem restricted to Qd-CNF, $\forall\exists$ CNF and $\forall\exists d$ -CNF formulas, respectively.

3 ALGORITHMS FOR QBFSAT

We show that QBFSAT parameterized by the number of existential variables and the maximum size of any clause is linear-time fixed-parameter tractable, i.e., with a running time linear in the size of formula. The algorithm is based on a chain of reductions involving two novel natural problems and for which we provide corresponding fixed-parameter algorithms along the way. We begin by defining the OR-CNF TAUTOLOGY problem.

OR-CNF TAUTOLOGY	
INSTANCE:	A set of variables X and d -CNF formulas ϕ_1, \dots, ϕ_k on X .
QUESTION:	Is $\phi_1 \vee \dots \vee \phi_k$ a tautology?

The following lemma follows via an application of quantifier elimination.

LEMMA 1. *There is an algorithm that takes a Qd-CNF $Q.\phi$ with k existentially quantified variables, and in time $O(2^k |\phi|)$ constructs an instance I of OR-CNF TAUTOLOGY with 2^k d -CNF formulas of size at most $|\phi|$ such that $Q.\phi$ holds if and only if I is a yes-instance.*

PROOF. We will define a procedure that takes as input a quantifier prefix Q on variables X and a set of d -CNF formulas D on X , and in time $O(|D| \cdot |\phi|)$ computes a new quantifier prefix Q' on a new set of variables variables X' and a new set of d -CNF formulas D' on X' such that

- (1) $|X'| \leq 2|X|$,
- (2) $|D'| \leq 2|D|$,
- (3) $|\phi'| \leq |\phi|$ for all $\phi' \in D'$ and $\phi \in D$,
- (4) there is one less existential quantifier in Q' than in Q , and
- (5) the formula $Q.\bigvee_{\phi \in D} \phi$ is equivalent to $Q'.\bigvee_{\phi' \in D'} \phi'$.

If the initial Qd-CNF is $Q.\phi$, we start with Q and $D = \{\phi\}$, and recursively apply the procedure k times until we obtain an equivalent formula $Q^*.\bigvee_{\phi^* \in D^*} \phi^*$, where Q^* contains only universal quantifiers, the number of variables $|X^*| \leq 2^k |X|$, and the number of formulas $|D^*| \leq 2^k |D| \leq 2^k$. The total running time sums up to $O(2^k |\phi|)$. Since all

quantifiers in Q^* are universal, $Q.\phi$ holds if and only if $\bigvee_{\phi^* \in D^*} \phi^*$ is a tautology, i.e. (X^*, D^*) is a yes-instance of OR-CNF TAUTOLOGY.

Now we define the procedure. Let $Q = Q_1 x_1 \dots Q_n x_n$ be the quantifier prefix, and D be a set of d -CNF formulas. Let i be the index of the last existential quantifier in Q , and observe that $Q_j = \forall$ for all $j > i$. Create a new quantifier prefix consisting of three parts $Q' = Q_{<i} Q_0 Q_1$, where $Q_{<i} = Q_1 x_1 \dots Q_{i-1} x_{i-1}$ is a copy of Q up to index i , and $Q_b = \forall x_{i+1}^b \dots \forall x_n^b$ for both $b = 0$ and $b = 1$. For a formula ϕ with variables x_1, \dots, x_n and $b \in \{0, 1\}$, let $\phi^{i,b}$ denote the formula obtained from ϕ by replacing every variable x_j with $j > i$ by x_j^b . For every $\phi \in D$, add $\phi[x_i = 0]^{i,0}$ and $\phi[x_i = 1]^{i,1}$ to D' .

Clearly, the procedure can be implemented in $O(|D'|)$ time, and the resulting Q' and D' satisfy conditions (1), (2), (3) and (4). It remains to show that (5) also holds, i.e. that the new formula is equivalent to the original one. To this end, let $\phi_D = \bigvee_{\phi \in D} \phi$ and observe that

$$\begin{array}{lll}
Q. & \bigvee_{\phi \in D} \phi & \iff \\
Q_{<i} & \exists x_i \forall x_{i+1} \dots x_n. \phi_D & \iff \\
Q_{<i}. & (\forall x_{i+1} \dots x_n. \phi_D[x_i = 0]) \vee & \\
& (\forall x_{i+1} \dots x_n. \phi_D[x_i = 1]) & \iff \\
Q_{<i}. & \forall x_{i+1}^0 \dots x_n^0. \phi_D[x_i = 0]^{i,0} \vee & \\
& \forall x_{i+1}^1 \dots x_n^1. \phi_D[x_i = 1]^{i,1} & \iff \\
Q_{<i} & \forall x_{i+1}^0 \dots x_n^0 \forall x_{i+1}^1 \dots x_n^1. & \\
& \phi_D^{i,0}[x_i = 0] \vee \phi_D^{i,1}[x_i = 1] & \iff \\
Q'. & \bigvee_{\phi' \in D'} \phi', &
\end{array}$$

where the last equivalence follows from the expansion $\phi_D^{i,b}[x_i = b] = \bigvee_{\phi \in D} \phi^{i,b}[x_i = b]$. \square

As the next step, we provide a polynomial-time reduction from OR-CNF TAUTOLOGY to the problem of finding an independent set of size k in a well-structured graph. For a set of variables X and integer d , let C_X^d be the set of all clauses with exactly d distinct literals over variables in X . A pair (G, λ) is a k -partite d -clause graph over X if G is an undirected k -partite graph with $V(G) = V_1 \uplus \dots \uplus V_k$, and $\lambda : V(G) \rightarrow C_X^d$ is a function, injective on V_i for every $i \in [k]$. Moreover, two vertices $u \in V_i$ and $v \in V_j$ in G are connected by an edge if and only if $i \neq j$ and $\lambda(u)$ and $\lambda(v)$ clash, i.e. they contain a pair of opposite literals. The problem of finding an independent set in such a graph can be formally defined as follows.

CLAUSE-GRAPH INDEPENDENT SET

INSTANCE: A k -partite d -clause graph (G, λ) over X .

QUESTION: Is there an independent set $S \subseteq V(G)$ such that $|S \cap V_i| = 1$ for all $1 \leq i \leq k$?

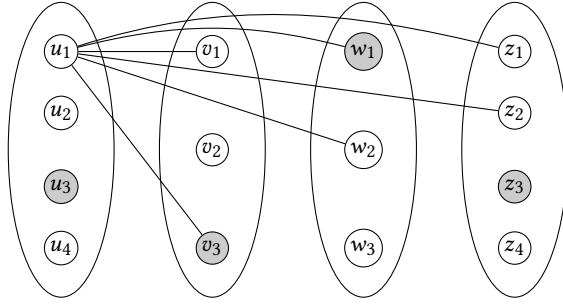


Fig. 1. Let $X = \{x_1, x_2, \dots, x_6\}$ be a set of variables. Let $\phi_1 = (x_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee x_2 \vee x_4) \wedge (\bar{x}_1 \vee x_5 \vee \bar{x}_6) \wedge (\bar{x}_3 \vee x_2 \vee x_5)$, $\phi_2 = (x_2 \vee \bar{x}_3 \vee \bar{x}_6) \wedge (\bar{x}_4 \vee \bar{x}_5 \vee x_6) \wedge (\bar{x}_2 \vee x_3 \vee \bar{x}_6)$, $\phi_3 = (\bar{x}_2 \vee x_3 \vee x_4) \wedge (\bar{x}_2 \vee \bar{x}_4 \vee x_5) \wedge (x_3 \vee x_4 \vee \bar{x}_5)$, and $\phi_4 = (x_1 \vee \bar{x}_2 \vee x_5) \wedge (\bar{x}_3 \vee x_4 \vee x_6) \wedge (x_3 \vee x_4 \vee \bar{x}_6) \wedge (\bar{x}_4 \vee \bar{x}_5 \vee x_6)$. The vertices u_1, \dots, u_4 corresponds to the first, second, third, and fourth clauses of ϕ_1 , respectively. Similarly, the vertices v_1, v_2, v_3 corresponds to the first second, and third clauses of ϕ_2 , respectively. Analogously, we have drawn vertices for clauses in ϕ_3 and ϕ_4 . All the edges incident on u_1 are drawn in the figure. The set $\{u_3, v_3, w_1, z_3\}$ is an independent set the graph and the assignment $\alpha(x_1) = \alpha(x_2) = \alpha(x_6) = 1$ and $\alpha(x_3) = \alpha(x_4) = \alpha(x_5) = 0$ implies that $\bigvee_{i=1}^4 \phi_i$ is not a tautology.

In the following we will use λ^{-1} as the inverse of λ , which is well-defined if the part V_i is clear from the context since λ is injective on every part V_i . We also use $\lambda(S)$ to denote the set $\{\lambda(v) : v \in S\}$ for every set $S \subseteq V$.

LEMMA 2. *There is a linear-time reduction that takes an instance $I = (X, \{\phi_1, \dots, \phi_k\})$ of OR-CNF TAUTOLOGY, where each ϕ_i is a d -CNF, and produces an instance $I' = (G, X, \lambda)$ of CLAUSE-GRAPH INDEPENDENT SET where (G, λ) is a k -partite d -clause graph over X with the i -th part having at most $|\phi_i|$ vertices such that I is a no-instance if and only if I' is a yes-instance.*

PROOF. Given an instance I of OR-CNF TAUTOLOGY, construct the k -partite d -clause graph (G, λ) over X by letting $V(G) = V_1 \uplus \dots \uplus V_k$, where V_i contains one vertex v for every clause C in ϕ_i , and setting $\lambda(v) = C$; by the assumption that all clauses in ϕ_i are distinct, this definition ensures that λ is injective on every part V_i . For every $1 \leq i < j \leq n$ and every $u \in V_i$ and $v \in V_j$, add an edge $\{u, v\}$ to G if clauses $\lambda(u)$ and $\lambda(v)$ clash. To ensure that every clause $\lambda(v)$ contains exactly d literals, we add $d - 1$ new variables to X and add $d - |\lambda(v)|$ of those positively to the clause $\lambda(v)$. Note that this does not modify the edge set of the graph because all new variables only occur positively. The ideas behind the reduction is illustrated in Figure 1. Clearly, this reduction requires polynomial time.

For correctness, first assume that I is a no-instance, i.e. $\phi_1 \vee \dots \vee \phi_k$ is not a tautology. Then there exists an assignment $\alpha : X \rightarrow \{0, 1\}$ that falsifies every formula ϕ_i . For each $1 \leq i \leq k$, let C_i be a clause in ϕ_i falsified by α , and let $v_i \in V_i$ be the vertex of G such that $\lambda(v_i) = C_i$. Observe that α satisfies $\bigwedge_{i=1}^k \neg C_i = \bigwedge_{\ell \in C_1 \cup \dots \cup C_k} \bar{\ell}$, hence no pair of clauses C_i and C_j clash. By construction of G , this implies that there is no edge between v_i and v_j for any i and j , i.e. $\{v_i : 1 \leq i \leq k\}$ is an independent set in G .

Now suppose I' is a yes-instance, i.e. G contains an independent set $S = \{v_1, \dots, v_k\}$, where $v_i \in V_i$ for all $1 \leq i \leq k$. By construction of (G, λ) , no two clauses $\lambda(v_i)$ and $\lambda(v_j)$ clash. Observe that $\phi' = \bigwedge_{i=1}^k \neg \lambda(v_i)$ is a conjunction of literals, and it contains no two opposite literals, hence ϕ' is satisfiable. Any assignment $\alpha' : X \rightarrow \{0, 1\}$ that satisfies ϕ' falsifies at least one clause in every formula ϕ_i , namely $\lambda(v_i)$. Thus, α' falsifies $\phi_1 \vee \dots \vee \phi_k$, proving that it is not a tautology. \square

The following theorem shows that CLAUSE-GRAPH INDEPENDENT SET is in XP parameterized by k only.

THEOREM 3. *CLAUSE-GRAPH INDEPENDENT SET can be solved in time $O((\max_{i=1}^k |V_i|)^k d \binom{k}{2})$ and is therefore in XP parameterized by k .*

PROOF. Let G be a k -partite d -clause graph G over X with k -partition $V_1 \uplus \dots \uplus V_k$. A simple brute-force algorithm that enumerates all possible k -tuples (v_1, \dots, v_k) such that $v_i \in V_i$ and for each tuple checks in polynomial-time whether $\{v_1, \dots, v_k\}$ is an independent set in G runs in $O((\max_i |V_i|)^k d \binom{k}{2})$ time. \square

Now we will show that CLAUSE-GRAPH INDEPENDENT SET is in FPT parameterized by k and d . To this end, we will use the sunflower lemma. For a family \mathcal{F} of sets over some universe U , we say that a subset $\mathcal{S} \subseteq \mathcal{F}$ is a *sunflower* if there is a subset $C \subseteq U$ such that $F \cap F' = C$ for every two distinct $F, F' \in \mathcal{S}$, i.e. all pairs of distinct sets in \mathcal{S} have a common intersection C , which we also call the *core* of the sunflower. If \mathcal{S} is a sunflower with core C and $F \in \mathcal{S}$, then we call $F \setminus C$ the *petal* of F . Observe that the petals $\{F \setminus C \mid F \in \mathcal{S}\}$ of a sunflower are pairwise disjoint.

LEMMA 4 ([16]). *Let \mathcal{F} be a family of subsets of a universe U , each of size exactly b , and let $a \in \mathbb{N}$. If $|\mathcal{F}| \geq b!(a-1)^b$, then \mathcal{F} contains a sunflower \mathcal{S} of size at least a . Moreover, \mathcal{S} can be computed in time $O(|\mathcal{F}|d)$.*

For a graph G and a vertex v , we write $G - v$ to denote the graph obtained from G by deleting the vertex v with all incident edges.

LEMMA 5. *Let (G, λ) be a k -partite d -clause graph, and V_i be one of the parts. If the family $\lambda(V_i)$ contains a sunflower \mathcal{S} of size at least $s = (k-1)d + 2$, then, for every v with $\lambda(v) \in \mathcal{S}$, instances (G, λ, X) and $(G - v, \lambda, X)$ of CLAUSE-GRAPH INDEPENDENT SET are equivalent.*

PROOF. Let \mathcal{S} be the sunflower in $\lambda(V_i)$ of size at least s , let $F \in \mathcal{S}$ be an arbitrary clause in \mathcal{S} , and $v = \lambda^{-1}(F)$ be the corresponding vertex in G . We claim that v satisfies the statement of the lemma, i.e. (G, λ) has an independent set S with $|S \cap V_i| = 1$ if and only if so does $(G - v, \lambda)$. The reverse direction of the claim is clear since $G - v$ is a subgraph of G .

Towards showing the forward direction, let S be a solution to (G, λ, X) . If $v \notin S$, then S is also a solution to $(G - v, \lambda, X)$. Now suppose that $v \in S$. We claim that there exists a clause $F' \in \mathcal{S}$ such that F' does not clash with any clause $\lambda(u)$ for $u \in S \setminus \{v\}$, so we can replace v with $v' = \lambda^{-1}(F')$ in the independent set S . To this end, let $S' = S \setminus v$ and let C be the core of \mathcal{S} . Note that every clause contains exactly d literals, therefore it can share variables with the petals of at most d clauses in \mathcal{S} . Thus, the clauses in $\lambda(S')$ share variables with at most $|S'|d = (k-1)d$ petals of \mathcal{S} in total. Since $|\mathcal{S} \setminus \{F\}| \geq s - 1 = (k-1)d + 1$, there is a clause $F' \in \mathcal{S} \setminus \{F\}$ whose petal $F' \setminus C$ shares no variables with any clause in $\lambda(S')$. Moreover, the core $C \subseteq F$ does not clash with any clause in $\lambda(S')$ since $v = \lambda^{-1}(F)$ is not adjacent to any vertex in S' . Therefore, $v' = \lambda^{-1}(F')$ is not adjacent to any vertex in S' and $S' \cup \{v'\}$ is an independent set in $G - v$. \square

THEOREM 6. *CLAUSE-GRAPH INDEPENDENT SET has a kernel with at most $d!(s-1)^d - 1$ vertices in every part V_i , where $s = (k-1)d + 2$. Note that this implies that CLAUSE-GRAPH INDEPENDENT SET has a kernel of size at most $dkd!(s-1)^d$. The kernel can be computed in time $O(\min\{d!(s-1)^d |V(G)|, d|V(G)|^2\})$.*

PROOF. Let (G, λ) be a k -partite d -clause graph with parts V_1, \dots, V_k over some variables X . If $|V_i| < d!(s-1)^d$ for every $i \in [k]$, where $s = (k-1)d + 2$, then the instance is already kernelized. So suppose that $|V_i| \geq d!(s-1)^d$. By Lemma 4, $\lambda(V_i)$ has a sunflower \mathcal{S} of size at least s , and \mathcal{S} can be found in polynomial time. Note that by taking any subset $V' \subseteq \lambda(V_i)$ of size at least $d!(s-1)^d$, \mathcal{S} can be found in time $O(\min\{d!(s-1)^d, d|V(G)|\})$. Then, by Lemma 5,

we can remove any vertex in $\lambda^{-1}(S)$ from V_i and obtain an equivalent but smaller instance. Therefore, applying this procedure exhaustively, we obtain in polynomial time an equivalent instance of CLAUSE-GRAPH INDEPENDENT SET such that $|V_i| < d!(s-1)^d$ for every $1 \leq i \leq k$. Notice that we need to apply the above procedure $\mathcal{O}(|V(G)|)$ times and hence the running time is at most $\mathcal{O}(\min\{d!(s-1)^d d|V(G)|, d|V(G)|^2\})$. \square

COROLLARY 7. *CLAUSE-GRAPH INDEPENDENT SET is fixed-parameter tractable parameterized by $k+d$. In particular, it can be solved in time $(dk)^{\mathcal{O}(dk)}|V(G)|$.*

PROOF. The statement that CLAUSE-GRAPH INDEPENDENT SET is fixed-parameter tractable parameterized by $k+d$ follows immediately from Theorem 6. Let (G, λ) be a k -partite d -clause graph with parts V_1, \dots, V_k over some variables X . First we can employ Theorem 6 to obtain the kernel (G', λ') with parts V'_1, \dots, V'_k in time $d!(s-1)^d d|V(G)|$, where $s = (k-1)d + 2$, that is equivalent to (G, λ) and satisfies $|V'_i| \leq d!(s-1)^d$. We can then use Theorem 3 to solve the instance (G', λ') of CLAUSE-GRAPH INDEPENDENT SET in time $\mathcal{O}((\max_{i=1}^k |V'_i|)^k d \binom{k}{2}) = \mathcal{O}((d!)^k (s-1)^{dk} d \binom{k}{2})$, which is bounded from above by $(dk)^{\mathcal{O}(dk)}$. Altogether, we therefore obtain $(dk)^{\mathcal{O}(dk)} + \mathcal{O}(d!(s-1)^d d|V(G)|)$, which is bounded by $(dk)^{\mathcal{O}(dk)}|V(G)|$ as the running time of our algorithm. \square

Combining Lemma 2 and Lemma 1, we obtain:

COROLLARY 8. *There is a reduction that takes a Qd -CNF $Q.\phi$ with k existentially quantified variables, and in time $\mathcal{O}(2^k|\phi|)$ produces an instance $I' = (G, \lambda, X)$ of CLAUSE-GRAPH INDEPENDENT SET where (G, λ) is a 2^k -partite d -clause graph over X with each part having at most $|\phi|$ vertices such that I is a no-instance if and only if I' is a yes-instance.*

We now show that QBFSAT with k existential variables and clauses of size d is in FPT parameterized by $k+d$.

THEOREM 9. *QBFSAT is fixed-parameter tractable parameterized by the number k of existential variables plus the maximum size d of any clause. In particular, there is an algorithm solving this problem in time $(2^k d)^{\mathcal{O}(2^k d)}|\phi|$ for a QCNF formula $Q.\phi$.*

PROOF. Let $\Phi = Q.\phi$ be the given QBFSAT formula with k existential variables having clauses of size at most d . We first use Corollary 8 to obtain in time $\mathcal{O}(2^k|\phi|)$ the 2^k -partite d -clause graph (G, λ) with parts V_1, \dots, V_{2^k} such that Φ is false if and only if (G, λ) has an independent set S with $|S \cap V_i| = 1$. We then use Corollary 7 to decide in time $(d2^k)^{\mathcal{O}(d2^k)}|V(G)| = (d2^k)^{\mathcal{O}(d2^k)}2^k|\phi| = (d2^k)^{\mathcal{O}(d2^k)}|\phi|$ whether (G, λ) has an independent set S with $|S \cap V_i| = 1$. If so, we return that Φ is false, otherwise we return that Φ is true. Altogether the total runtime of the algorithm is at most $(d2^k)^{\mathcal{O}(d2^k)}|\phi|$. \square

Last, via a straightforward algorithm we remark that QBFSAT is in XP parameterized by the number of existential variables. As we will see in Section 5 this problem is unlikely to admit an FPT algorithm unless $\text{FPT}=\text{W}[1]$.

THEOREM 10. *QBFSAT is in XP parameterized by the number k of existential variables. In particular, QBFSAT can be solved in time $\mathcal{O}(m^{2^k} d \binom{k}{2} + 2^k|\phi|)$ for a QCNF-formula $Q.\phi$ with m clauses.*

PROOF. Let $\Phi = Q.\phi$ be the given QBFSAT formula with k existential variables having clauses of size at most d . We first use Corollary 8 to obtain in time $\mathcal{O}(2^k|\phi|)$ the 2^k -partite d -clause graph (G, λ) with parts V_1, \dots, V_{2^k} such that Φ is not satisfiable if and only if (G, λ) has an independent set S with $|S \cap V_i| = 1$. We then use Theorem 3 to solve the instance (G, λ) of CLAUSE-GRAPH INDEPENDENT SET in time $\mathcal{O}(m^{2^k} d \binom{k}{2})$. Therefore, we obtain $\mathcal{O}(m^{2^k} d \binom{k}{2} + 2^k|\phi|)$ as the total runtime of the algorithm. \square

4 THE QCSP PROBLEM

We now consider the finite-domain generalization of QBFSAT known as the *quantified constraint satisfaction problem (QCSP)*. An instance of the *constraint satisfaction problem (CSP)* (without quantifiers) is (X, D, C) , where $X = \{x_1, \dots, x_n\}$ is a set of variables, $D = \{D_1, \dots, D_n\}$ is a set of *domains (of values)* for each variable, and $C = \{C_1, \dots, C_m\}$ is a set of constraints, where $C_j = R_j(x_{j_1}, \dots, x_{j_{\text{ar}(R_j)}})$, $R_j \subseteq D_{j_1} \times \dots \times D_{j_{\text{ar}(R_j)}}$ is a relation of arity $\text{ar}(R_j)$, and $1 \leq j_1, \dots, j_{\text{ar}(R_j)} \leq n$. The instance is *satisfiable* if there exists an assignment $\alpha : X \rightarrow \bigcup_{i=1}^n D_i$ of values to the variables such that $\alpha(x_i) \in D_i$ for all $i \in [n]$, and $(\alpha(x_{j_1}), \dots, \alpha(x_{j_{\text{ar}(R_j)}})) \in R_j$ for all constraints in $j \in [m]$. Let $d = \max_{i=1}^n |D_i|$ be the largest domain size and $r = \max_{j=1}^m \text{ar}(R_j)$ be the maximum arity of a constraint in C . For example, 3-SAT can be cast as CSP with $d = 2$ (every variable is assigned a Boolean value) and $r = 3$ (every clause is a ternary constraint).

Parameterized complexity of the CSP with respect to n , d and r is well-understood [24]: the problem is in FPT parameterized by $n + d$, W[1]-hard parameterized by $n + r$, in XP parameterized by n , and paraNP-hard parameterized by $d + r$.

QCSP is a generalization where the input additionally comes with a set of quantifiers $Q = (Q_1, \dots, Q_n)$. The basic notions in Section 2 easily extends to the CSP setting and we write QCSP for the (PSPACE-complete) decision problem of verifying whether an instance $Q.\phi$ is true or false, where ϕ is a CSP instance over the variables occurring in the quantifier prefix Q . Naturally, if a variable x_i has domain D_i then we in the context of a universal quantifier require that the subsequent formula is true for all values in D_i , and for at least one value in D_i if the quantifier is existential. We manage to generalize Theorem 9 to QCSP.

THEOREM 11. *QCSP is FPT when parameterized by the number of existentially quantified variables, the domain, and the maximum arity of any relation.*

PROOF. Let

$$\Phi = Q_1 x_1, \dots, Q_n x_n. \phi$$

where each $Q_i \in \{\forall, \exists\}$ be an instance of QCSP over variables $V = \{x_1, \dots, x_n\}$, domain values $D = \{D_1, \dots, D_n\}$, and constraints C . We write $\Gamma = \{R \mid R(\mathbf{x}) \in C\}$ for the set of relations in the instance and let

$$r = \max\{\text{ar}(R) \mid R \in \Gamma\}$$

be the maximum arity of any relation. Last, let $d = \lceil \log_2 |D| \rceil$. Define the surjective function $h: \{0, 1\}^d \rightarrow D$ such that there exists a unique element $m \in D$ where $|h^{-1}(m)| \geq 1$ and where we for every other $a \in D$ have $|h^{-1}(a)| = 1$. Hence, every domain value in D is represented by a Boolean d -ary tuple, and if $2^d > |D|$ then a unique value in D corresponds to the additional tuples in $\{0, 1\}^d$.

We will show an fpt reduction to QBFSAT parameterized by arity $q = r \cdot d$ and the number of existentially quantified variables. First, for an n -ary relation $R \in \Gamma$ we let $R_{\mathbb{B}} = \{(x_1^1, \dots, x_d^1, \dots, x_1^n, \dots, x_d^n) \mid (h(x_1^1, \dots, x_d^1), \dots, h(x_1^n, \dots, x_d^n))\}$ be the Boolean relation obtained by viewing each domain value in D as a d -ary Boolean tuple via the surjective function h . Importantly, it is not hard to see that $R_{\mathbb{B}}$ can be defined by a conjunction of $(d \cdot \text{ar}(R))$ -clauses over Boolean variables $x_1, \dots, x_{d \cdot \text{ar}(R)}$: for each $(b_1, \dots, b_{d \cdot \text{ar}(R)}) \notin R_{\mathbb{B}}$ simply add the clause $(\neg x_1 \vee \dots \vee \neg x_{d \cdot \text{ar}(R)})$. Furthermore, we observe that any clause of arity smaller than $d \cdot r$ can be simulated by a $(d \cdot r)$ -clause by repeating one of its arguments.

Now, let

$$\Phi = Q_1 x_1, \dots, Q_n x_n. \phi$$

where each $Q_i \in \{\forall, \exists\}$ be an instance of QCSP(Γ). Crucially, at most k variables are existentially quantified. For each x_i we introduce d fresh variables x_i^1, \dots, x_i^d and observe that this in total requires $n \cdot d$ fresh variables but that at most $k \cdot d$ of these correspond to existentially quantified variables. For each constraint $R(x_{i_1}, \dots, x_{i_{\text{ar}(R)}})$ occurring in ϕ we replace it by the conjunction of $(d \cdot \text{ar}(R))$ -clauses defining $R_{\mathbb{B}}(x_{i_1}^1, \dots, x_{i_{\text{ar}(R)}}^1, \dots, x_{i_1}^d, \dots, x_{i_{\text{ar}(R)}}^d)$. We let

$$\Phi = Q_1 x_1^1, \dots, x_1^d \dots Q_n x_n^1, \dots, x_n^d.$$

$$\phi_{\mathbb{B}}(x_1^1, \dots, x_1^d, x_n^1, \dots, x_n^d)$$

be the instance of $Q(d \cdot r)$ -CNF resulting from replacing each constraint by the corresponding conjunction of $(d \cdot r)$ -clauses, where we with a slight abuse of notation write $Q_i x_i^1, \dots, x_i^d$ with the meaning that all variables x_i^1, \dots, x_i^d have the same quantifier Q_i . For each variable domain $D_i \in D$ we first remark that D_i can be treated as a unary relation and that $D_{i_{\mathbb{B}}}$ is thus a well-defined d -ary Boolean relation. Hence, for every x_i we add the set of d -clauses defining $D_{i_{\mathbb{B}}}$ over the variables x_i^1, \dots, x_i^d . Last, for every Boolean variable x_i^j we simply use $\{0, 1\}$ as the allowed domain values. We let Φ' be the resulting $Q(d \cdot r)$ -CNF instance.

For correctness, assume that Φ is true and has a winning strategy which for every existential variable x_i is witnessed by a function $f_i: D^j \rightarrow D$ where j is the number of universally quantified variables preceding x_i . We construct a winning strategy for the Boolean instance Φ' as follows. For each existential variable x_i let $f_{i_{\mathbb{B}}}$ be the $(j \cdot d)$ -ary Boolean function defined as $f_{i_{\mathbb{B}}}(h^{-1}(a_1), \dots, h^{-1}(a_j)) = f_i(a_1, \dots, a_j)$ for all $a_1, \dots, a_j \in D$. We observe that this correctly defines a total Boolean function since h is a bijection and it is easy to see that it must be a winning strategy for Φ . The other direction can be proven with a similar argument. \square

It is worth remarking that by Samer & Szeider [24] and the forthcoming Theorem 15 each of the above three conditions are necessary in the sense that we obtain a $W[1]$ -hard problem if any condition is dropped.

5 LOWER BOUNDS

We proceed by complementing our positive FPT result by two strong lower bounds. We begin by ruling out an FPT algorithm for $\forall\exists$ QBF SAT parameterized by the number of existential variables for unbounded clause size via a reduction from the $W[1]$ -complete problem MULTIPARTITE INDEPENDENT SET. Using the following auxiliary result, we strengthen this result even further under the ETH.

THEOREM 12 ([10], cf. THEOREM 14.21 IN [11]). *Assuming the ETH, there is no algorithm that decides if a graph on n vertices has an independent set of size k in $f(k) \cdot n^{o(k)}$ time for any computable function f .*

For our purposes, it is more convenient to work with the following variant of the INDEPENDENT SET problem. In MULTIPARTITE INDEPENDENT SET, an instance is a graph with the vertex set partitioned into k parts, and the question is whether the graph contains an independent set with one vertex from each part. Using a well-known reduction (cf. Section 13.2 in [11]) that takes an instance (G, k) of INDEPENDENT SET and constructs in polynomial time an equivalent instance of MULTIPARTITE INDEPENDENT SET with $|V(G)|k$ vertices and the same parameter k , we obtain the following corollary.

COROLLARY 13. *Assuming the ETH, there is no algorithm that solves MULTIPARTITE INDEPENDENT SET in $f(k) \cdot n^{o(k)}$ time for any computable function f .*

PROOF. We provide a short proof for completeness. Let G be a graph with vertices v_1, \dots, v_n . Create a k -partite graph G' with vertices $V_1 \uplus \dots \uplus V_k$, where $V_i = \{(i, j) : j \in [n]\}$; add edges $\{(i, j), (i', j')\}$ to $E(G')$ for all $i \neq i'$ and $\{v_j, v_{j'}\}$ in $E(G)$. It is easy to see that G contains an independent set of size k if and only if G' contains an independent set with one vertex from each part V_i , i.e. (G', k) is a yes-instance of MULTIPARTITE INDEPENDENT SET (see Section 13.2 in [11] for a full proof).

Now, suppose there is an algorithm that solves MULTIPARTITE INDEPENDENT SET in time $f(k) \cdot |V(G')|^{o(k)}$. Since $|V(G')| = nk$, we can use it to decide whether G has an independent set of size k in $f(k) \cdot (nk)^{o(k)} = (f(k)k^{o(k)}) \cdot n^{o(k)}$ time plus the polynomial time of the reduction, which contradicts the ETH by Theorem 12. \square

LEMMA 14. *There is a polynomial-time reduction that takes an instance (G, k) of MULTIPARTITE INDEPENDENT SET and produces in polynomial time a $\forall\exists$ CNF formula with $\lceil \log_2(k) \rceil$ existential variables such that (G, k) is a yes-instance if and only if the formula is false.*

PROOF. Let G be a graph with vertex set $V_1 \uplus \dots \uplus V_k$. It will be convenient to assume that k is a power of two. To this end, let $\kappa = \lceil \log_2(k) \rceil$, and add $2^\kappa - k$ new parts to $V(G)$, each consisting of one isolated vertex. Clearly, the new instance is equivalent to the original one, so we assume from now on that $k = 2^\kappa$. Enumerate vertices in each part of the graph. For convenience, we refer to vertex j in part V_i as (i, j) .

We will construct a formula $\forall Y \exists X. \phi$ on variables $Y = \{y_v : v \in V(G)\}$ and $X = \{x_1, \dots, x_\kappa\}$ that is false if and only if (G, k) has an independent set with exactly one vertex from every part. To this end, enumerate all functions $\alpha_1, \dots, \alpha_\kappa$ from X to $\{0, 1\}$. For every vertex $v = (i, j) \in V(G')$, add a clause C_v to ϕ with the following literals:

- y_v and $\overline{y_u}$ for all $u \in V(G) \setminus V_i$ such that $\{u, v\} \in E(G)$,
- x_ℓ if $\alpha_i(x_\ell) = 0$ and $\overline{x_\ell}$ if $\alpha_i(x_\ell) = 1$ for all $\ell \in [k]$.

This completes the construction.

Towards correctness, first assume that S is an independent set in G with one vertex from each V_i . Consider the set of clauses $C_S = \{C_v^\forall : v \in S\}$; recall that C^\exists and C^\forall for a clause C denotes the restriction of C to existential and universal variables, respectively. We claim that there is an assignment that falsifies every clause in C_S . It suffices to show that no pair of clauses in C_S clashes, i.e. contain opposite literals. Consider two clauses $C_u^\forall, C_v^\forall \in C_S$. Since S is an independent set, u and v are not adjacent, so C_u^\forall does not contain y_v and C_v^\forall does not contain y_u . Furthermore, both $C_u^\forall \setminus \{y_u\}$ and $C_v^\forall \setminus \{y_v\}$ only contain negative literals, so they do not clash either. Now, let $\tau : Y \rightarrow \{0, 1\}$ be an assignment that falsifies all clauses in C_S . We claim that $\exists X. \phi[\tau]$ is false, i.e. $\phi[\tau]$ is not satisfiable. Indeed, for every assignment $\alpha_i : X \rightarrow \{0, 1\}$, there is a vertex $v = (i, j) \in S$ in G , and hence a clause C_v^\exists remains in $\phi[\tau]$, which excludes α_i as the satisfying assignment. Thus, all assignments are excluded, and $\exists X. \phi[\tau]$ is false.

For the other direction, suppose $\forall Y \exists X. \phi$ is false. Then there exists an assignment $\tau' : Y \rightarrow \{0, 1\}$ such that $\exists X. \phi[\tau']$ is false, i.e. $\phi[\tau']$ is not satisfiable. By construction, every clause of $\phi[\tau']$ is a κ -clause with literals over all variables x_1, \dots, x_κ . Each such clause excludes exactly one satisfying assignment, so $\phi[\tau']$ contains exactly $2^\kappa = k$ clauses. Thus, for every assignment $\alpha_i : X \rightarrow \{0, 1\}$, there exists a clause C_v where $v \in V_i$ and τ' falsifies C_v^\forall . Pick one such clause C_v for every i , and let vertices v form a set S . We claim that S is an independent set in G . Suppose towards contradiction that $u, v \in S$ and $\{u, v\} \in E(G)$. By construction, $y_u \in C_u^\forall$ and $\overline{y_u} \in C_v^\forall$, so τ' cannot falsify both of them, which contradicts our choice of u, v . \square

THEOREM 15. $\forall\exists\text{QBFSAT}$ is $W[1]$ -hard parameterized by the number of existential variables. Moreover, assuming the ETH, this problem cannot be solved in $f(k) \cdot |\phi|^{o(2^k)}$ time for any computable function $f: \mathbb{N} \rightarrow \mathbb{N}$, where $\mathcal{Q}.\phi$ is the $\forall\exists\text{CNF}$ with k existential variables.

PROOF. $W[1]$ -hardness is immediate from Lemma 14 and the fact that $\text{MULTIPARTITE INDEPENDENT SET}$ is $W[1]$ -hard. Moreover, if there is an algorithm deciding whether a $\forall\exists\text{CNF}$ with matrix ϕ and κ existential variables is true or false in $f(\kappa) \cdot |\phi|^{o(2^\kappa)}$ time for any computable function f , then it can be combined with the reduction of Lemma 13 to solve $\text{MULTIPARTITE INDEPENDENT SET}$ in $f(\lceil \log_2 k \rceil) \cdot 2^{o(k)}$ time, contradicting the ETH. \square

For lower bounds for $\forall\exists\text{-QBFSAT}$, we first observe that this problem cannot be solved in $2^{o(k)} \cdot |\phi|^{O(1)}$ time under the ETH (since we have a trivial reduction from 3-SAT). However, we can significantly sharpen this under the SETH and in fact rule out every $2^{O(k)}$ time algorithm, i.e. the problem is not solvable in $c^k \cdot |\phi|^{O(1)}$ time for any constant $c \geq 1$. We rely on the following result.

THEOREM 16 ([8]). Assuming the SETH, $\forall\exists\text{-QBFSAT}$ is not solvable in $O^*(c^n)$ time¹ for any $c < 2$.

THEOREM 17. Assuming the SETH, $\forall\exists\text{-QBFSAT}$ parameterized by the number of existentially quantified variables k is not solvable in $c^k \cdot |\phi|^{O(1)}$ time for any constant c .

PROOF. Let k and ℓ denote the number of existential and universal variables in an input formula, respectively, and let $n = k + \ell$. We consider two algorithms for $\forall\exists\text{-QBFSAT}$. The first one is the hypothetical FPT algorithm that solves $\forall\exists\text{-QBFSAT}$ in $O^*(c^k)$ time for constant c . The second algorithm enumerates assignments to universal variables, and solves the resulting 3-SAT formulas. The latter requires $O(2^\ell c_3^k)$ time, where c_3 the infimum of all b such that 3-SAT is solvable in $O^*(b^n)$ time. Observe that $c_3 < 2$. Now, let $\delta = k/n$ be the proportion of existential variables in the input formula. We claim that, depending on the value of δ , we can use either the first or the second algorithm to solve $\forall\exists\text{-QBFSAT}$ in $O^*(d^n)$ time for some $d < 2$, contradicting SETH by Theorem 16. To this end, define $T = 1 + \log_2(c) - \log_2(c_3)$, and observe that $T \geq 1$ since $\forall\exists\text{-QBFSAT}$ is more general than 3-SAT and $c \geq c_3$.

First, suppose $\delta > 1/T$, i.e. $k > n/T$. Then use the first algorithm, which runs in $O^*(c^{n/T}) = O^*(2^{(\log_2(c)/T)n})$ time. Now suppose $\delta \leq 1/T$, i.e. $k \leq n/T$. Then use the second algorithm, which runs in $O^*(2^{(1-\delta)n} c_3^{\delta n})$ time. Observe that $(1 - \delta) + \log_2(c_3)\delta \leq 1 - (1 - \log_2(c_3))/T \leq \log_2(c)/T$. In both cases, our algorithm runs in $O^*(2^{(\log_2(c)/T)n})$ time. Since $c_3 < 2$, we have $\log_2(c_3) < 1$ and $\log_2(c) < T$, which completes the proof. \square

6 DISCUSSION

In this paper we investigated a simple and overlooked parameter for QBFSAT and proved FPT with respect to the number of existentially quantified variables and the maximum arity of any clause. This parameterization is particularly noteworthy since applied QBF solving is frequently based on the idea of expanding universally quantified variables in order to get an instance that can be solved with SAT techniques. This strategy comes with the downside that (1) after removing universally quantified variables one still needs to solve an NP-hard problem, and (2) the strategy is inefficient for instances with many universally quantified variables. Our result is complementary in the sense that instances with many universal but few existential variables can now be handled efficiently with our novel FPT algorithm. While we in this paper concentrate on the theory it is natural to speculate whether these two approaches can be merged in actual QBF solvers to solve previously intractable instances faster.

¹The notation O^* hides factors polynomial in n .

For improvements, there is a gap between our $2^{O(k2^k)}$ FPT algorithm and our lower bound (under the SETH) which rules out any single-exponential $2^{O(k)}$ algorithm. It is not immediately which direction could be strengthened and new algorithmic ideas would likely be needed to bring down the running time to $2^{O(k^c)}$ for some fixed c . It would also be interesting to generalize our FPT algorithm to even broader classes of problems. A promising candidate is the *dependency quantified Boolean formula* (DQBF) formalism, i.e. Boolean formulas equipped with Henkin quantifiers. This problem is generally NEXPTIME-complete and has comparably few FPT results. Naturally, Lemma 1 would need to be modified to the DQBF setting, but besides that the main ideas should carry over.

From a purely theoretical perspective rather little is known about the logical fragment where we allow unrestricted universal quantification but only limited existential quantification. As a starting point one could define a closure operator on sets of Boolean relations induced by logical formulas allowing universal but no existential quantification over conjunctions of atoms from a predetermined structure. Such formulas would generalize *quantifier-free primitive positive* definitions (qfpp-definitions) which has been used to study fine-grained complexity aspects of CSPs [21], but be more restrictive than the formulas considered by Börner et al. [6] developed to study classical complexity of QCSPs. Could it, for example, be possible to give a classification akin to Post’s classification of Boolean clones, or find a reasonable notion of algebraic invariance? A reasonable guess extending Börner et al. [6] would be to consider algebras consisting of partial, surjective polymorphisms.

ACKNOWLEDGEMENTS

The second author is partially supported by the Swedish research council under grant VR-2022-03214. The fourth author was supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

REFERENCES

- [1] A. Atserias and S. Oliva. Bounded-width QBF is PSPACE-complete. *Journal of Computer and System Sciences*, 80(7):1415–1429, 2014.
- [2] A. Ayari and D. Basin. Qubos: Deciding quantified boolean logic using propositional satisfiability solvers. In *Proceedings of the International Conference on Formal Methods in Computer-Aided Design (FMCAD-2002)*, pages 187–201. Springer, 2002.
- [3] A. Biere. Resolve and expand. In *Proceedings of the International Conference on Theory and Applications of Satisfiability Testing (SAT-2005)*, pages 59–70. Springer, 2004.
- [4] A. Biere, M. Heule, and H. v. van Maaren. *Handbook of Satisfiability: Second Edition*. Frontiers in Artificial Intelligence and Applications. IOS Press, 2021.
- [5] R. Bloem, N. Braud-Santoni, V. Hadzic, U. Egly, F. Lonsing, and M. Seidl. Two SAT solvers for solving quantified boolean formulas with an arbitrary number of quantifier alternations. *Formal Methods in System Design*, 57(2):157–177, 2021.
- [6] F. Börner, A. Bulatov, P. Jeavons, and A. Krokhin. Quantified constraints: Algorithms and complexity. In *Proceedings of the 17th International Workshop on Computer Science Logic (CSL-2003)*, volume 2803, pages 58–70, 08 2003.
- [7] C. Calabro, R. Impagliazzo, and R. Paturi. The complexity of satisfiability of small depth circuits. In J. Chen and F. V. Fomin, editors, *Parameterized and Exact Computation*, volume 5917 of *Lecture Notes in Computer Science*, pages 75–85. Springer Berlin Heidelberg, 2009.
- [8] C. Calabro, R. Impagliazzo, and R. Paturi. On the exact complexity of evaluating quantified k -cnf. *Algorithmica*, 65(4):817–827, 2013.
- [9] H. Chen. Quantified constraint satisfaction and bounded treewidth. In *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI-2004)*, page 161–165, NLD, 2004. IOS Press.
- [10] J. Chen, X. Huang, I. A. Kanj, and G. Xia. Strong computational lower bounds via parameterized complexity. *Journal of Computer and System Sciences*, 72(8):1346–1367, 2006.
- [11] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer Publishing Company, Incorporated, 1st edition, 2015.
- [12] R. de Haan and S. Seider. Fixed-parameter tractable reductions to sat. In *Proceedings of Theory and Applications of Satisfiability Testing (SAT-2014)*, pages 85–102, Cham, 2014. Springer International Publishing.
- [13] R. Diestel. *Graph Theory, 5th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2016.
- [14] R. G. Downey, M. R. Fellows, et al. *Fundamentals of parameterized complexity*, volume 4. Springer, 2013.

- [15] E. Eiben, R. Ganian, and S. Ordyniak. Using decomposition-parameters for qbf: Mind the prefix! *Journal of Computer and System Sciences*, 110:1–21, 2020.
- [16] P. Erdős and R. Rado. Intersection theorems for systems of sets. *Journal of the London Mathematical Society*, 1(1):85–90, 1960.
- [17] J. K. Fichte, D. L. Berre, M. Hecher, and S. Szeider. The silent (r)evolution of SAT. *Commun. ACM*, 66(6):64–72, 2023.
- [18] J. K. Fichte, R. Ganian, M. Hecher, F. Slivovsky, and S. Ordyniak. Structure-aware lower bounds and broadening the horizon of tractability for QBF. In *Proceedings of the 38th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS-2023)*, pages 1–14, 2023.
- [19] F. V. Fomin, D. Lokshantov, S. Saurabh, and M. Zehavi. *Kernelization: theory of parameterized preprocessing*. Cambridge University Press, 2019.
- [20] M. Janota and J. Marques-Silva. Expansion-based QBF solving versus Q-resolution. *Theoretical Computer Science*, 577:25–42, 2015.
- [21] V. Lagerkvist and M. Wahlström. The (coarse) fine-grained structure of NP-hard SAT and CSP problems. *ACM Transactions on Computation Theory*, 14(1):2:1–2:54, 2022.
- [22] M. Lampis and V. Mitsou. Treewidth with a Quantifier Alternation Revisited. In *Proceedings of the 12th International Symposium on Parameterized and Exact Computation (IPEC-2017)*, volume 89 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 26:1–26:12, Dagstuhl, Germany, 2018. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.
- [23] M. Samer and S. Szeider. Backdoor sets of quantified boolean formulas. *Journal of Automated Reasoning*, 42(1):77–97, 2009.
- [24] M. Samer and S. Szeider. Constraint satisfaction with bounded treewidth revisited. *Journal of Computer and System Sciences*, 76(2):103–114, 2010.
- [25] R. Santhanam and R. Williams. New algorithms for QBF satisfiability and implications for circuit complexity. *Electronic Colloquium on Computational Complexity*, TR13-108, 2013.
- [26] A. Shukla, A. Biere, L. Pulina, and M. Seidl. A survey on applications of quantified boolean formulas. In *Proceedings of the 31st International Conference on Tools with Artificial Intelligence (ICTAI-2019)*, pages 78–84, 2019.