Formal Methods in System Design (2021) 57:157–177
https://doi.org/10.1007/s10703-021-00371-7

Two SAT solvers for solving quantified Boolean formulas with an arbitrary number of quantifier alternations

Roderick Bloem[1] · Nicolas Braud-Santoni[1] · Vedad Hadzic[1] · Uwe Egly[2] · Florian Lonsing[3] · Martina Seidl[4]

Received: 16 October 2020 / Accepted: 26 March 2021 / Published online: 23 August 2021

Abstract
In recent years, expansion-based techniques have been shown to be very powerful in theory and practice for solving quantified Boolean formulas (QBF), the extension of propositional formulas with existential and universal quantifiers over Boolean variables. Such approaches partially expand one type of variable (either existential or universal) for obtaining a propositional abstraction of the QBF. If this formula is false, the truth value of the QBF is decided, otherwise further refinement steps are necessary. Classically, expansion-based solvers process the given formula quantifier-block wise and use one SAT solver per quantifier block. In this paper, we present a novel algorithm for expansion-based QBF solving that deals with the whole quantifier prefix at once. Hence recursive applications of the expansion principle are avoided and only two incremental SAT solvers are required. While our algorithm is naturally based on the ∀Exp+Res calculus that is the formal foundation of expansion-based solving, it is conceptually simpler than present recursive approaches. Experiments indicate that the

This work has been supported by the Austrian Science Fund (FWF) under Projects W1255-N23, S11406-N23, S11408-N23, and S11409-N23, and the LIT AI Lab funded by the State of Upper Austria.

✉ Vedad Hadzic
vedad.hadzic@iaik.tugraz.at

Roderick Bloem
roderick.Bloem@iaik.tugraz.at

Nicolas Braud-Santoni
nicolas.braud-santoni@iaik.tugraz.at

Uwe Egly
egly@kr.tuwien.ac.at

Florian Lonsing
lonsing@cs.stanford.edu

Martina Seidl
martina.seidl@jku.at

1 TU Graz, Inffeldgasse 16a, 8010 Graz, Austria

2 TU Wien, Favoritenstrasse 9–11, 1040 Wien, Austria

3 Stanford University, 353 Jane Stanford Way, Stanford, CA 94305-9025, USA

4 Johannes Kepler University Linz, Altenbergerstr. 69, 4040 Linz, Austria

performance of our simple approach is comparable with the state of the art of QBF solving, especially in combination with other solving techniques.

Keywords Quantified Boolean formulas · Decision procedures · CEGAR

1 Introduction

Efficient tools for deciding the satisfiability of Boolean formulas (SAT solvers) are the core technology in many verification and synthesis approaches [49]. However, verification and synthesis problems are often beyond the complexity class NP as captured by SAT, requiring more powerful formalisms like *quantified Boolean formulas* (QBFs) [7]. QBFs extend propositional formulas by universal and existential quantifiers over Boolean variables [34] resulting in a decision problem that is PSPACE-complete. Applications from verification and synthesis [10,15,16,20,22,26], realizability checking [21], bounded model checking [18,52], and planning [19,44] motivate the quest for efficient QBF solvers (see [45] for a survey).

Unlike for SAT, where *conflict-driven clause learning* (CDCL) is the single dominant solving approach for practical problems, two dominant approaches exist for QBF solving. On one hand, CDCL has been successfully extended to QCDCL that enables clause and cube learning [23,37,51]. On the other hand, *variable expansion* has become very popular. In short, expansion-based solvers eliminate one kind of variables by assigning them truth values and solve the resulting propositional formula with a SAT solver. For QBFs with one quantifier alternation (2QBF), a natural approach is to use two SAT solvers: one that deals with the existentially quantified variables and another one that deals with the universally quantified variables. For generalizing this SAT-based approach to QBFs with an arbitrary number of quantifier alternations, expansion is recursively applied per quantifier block, requiring multiple SAT solvers realizing a counter-example guided extraction approach (CEGAR) [17]. As noted by Rabe and Tentrup [42], these CEGAR-based approaches show poor performance for formulas with many quantifier alternations in general.

We also propose an approach that is guided by counter-examples, but that deals with quantifier alternations in a different manner than available CEGAR approaches. Inspired by Counterexample-Guided Inductive Synthesis (CEGIS), we present a novel solving algorithm based on non-recursive expansion for QBFs with arbitrary quantifier prefixes using only two SAT solvers. In short, CEGIS is a generic framework initially devised in the context of syntax-guided synthesis [1]. It involves the interaction between two components:

- The *learner* generates candidate solutions that are consistent with all currently found counterexamples or—if it does not find such a candidate solution—it has shown that the problem does not have a solution, i.e., it is unsatisfiable.
- The *verifier* provides, given a candidate solution to a problem, a counterexample that disproves it, or it correctly proves that the candidate solution is indeed a valid solution.

We adopt the CEGIS paradigm for QBF solving as follows. Our approach instantiates all variables of the same kind (either the universal variables or the existential variables) at once with a candidate solution/counterexample and passes the resulting propositional abstraction of the QBF to a SAT solver. If the SAT solver finds the formula to be unsatisfiable, the truth value of the original QBF is decided, otherwise the model returned by the SAT solver is used as candidate solution/counter example for refining the propositional abstraction. In theory (i.e., from a proof complexity perspective), our approach of non-recursive expansion is equivalent to approaches that apply recursive expansion since both non-recursive and

recursive expansion rely on the ∀Exp+Res proof system [6]. However, the non-recursive expansion has practical implications such as a modified search strategy. That is, the use of recursive or non-recursive expansion results in different search strategies for the proof. With respect to proof search, there is an analogy to, e.g., implementations of resolution-based CDCL SAT solvers that employ different search heuristics.

In addition to the new algorithm, we also implemented a hybrid approach that combines clause learning with non-recursive expansion-based solving for exploiting the power of QCDCL. Our experiments indicate that this hybrid approach performs very well, especially on formulas with multiple quantifier alternations.

This paper is structured as follows. After a review of related work in the next section, we introduce the necessary preliminaries in Sect. 3. After a short recapitulation of expansion in Sect. 4, our novel non-recursive expansion-based algorithm is presented in Sect. 5. The relation between our solving approach and ∀Exp+Res is explained in Sect. 6. Implementation details are discussed in Sect. 7 together with a short discussion of the hybrid approach. In Sect. 8 we compare our approach to state-of-the-art solvers.

This paper is an extended version of [9]. Besides a careful revision of the text, it contains more examples and illustrations, as well as an additional chapter relating our new solving approach to the ∀Exp+Res proof system. Furthermore, we added comprehensive experiments on the benchmark set used in QBFEval 2018.

2 Related work

Already the early QBF solvers Qubos [3] and Quantor [3] incorporate selective quantifier expansion for eliminating one kind of quantification to reduce the given QBF to a propositional formula. Qubos heuristically chooses which kind of quantifier to eliminate. If universal quantifiers are eliminated, subformulas of the form $\forall x.\phi$ are replaced by $\phi[x/\top] \wedge \phi[x/\bot]$. Dually, subformulas of the form $\exists x.\phi$ are replaced by $\phi[x/\top] \vee \phi[x/\bot]$. For handling the blow-up, Qubos implements several simplification techniques. Qubos does not require the input QBF to be in prenex conjunctive normal form (PCNF), but it is able to process formulas of arbitrary structure. Even more, the expansion of existential variables destroys any PCNF structure. Quantor, in contrast, preserves the PCNF structure by expanding universal variables only. In both cases, the resulting propositional formula is then solved by calling a SAT solver once. Over 15 years ago, Qubos and Quantor impressively demonstrated the power of expanding universal variables but also showed its enormous memory consumption. As a pragmatic compromise, bounded universal expansion was introduced for efficient preprocessing [13,24,25,50].

The first approach which uses two alternating SAT solvers A and B for solving 2QBF, i.e., QBFs of the form $\forall U \exists E.\phi$, was presented in [43]. Solver A is initialized with ϕ, B with the empty formula. Both propositional formulas are incrementally refined with satisfying assignments found by the other solver. If A finds its formula unsatisfiable, then the QBF is false. Otherwise, the negation of the universal part of the satisfying assignment is passed to solver B. If solver B finds its formula unsatisfiable, then the QBF is true. Otherwise, the existential part of the satisfying assignment is passed to solver A. Janota and Marques-Silva generalized the idea of alternating SAT solvers [33] such that one solver deals with the existentially quantified variables and one solver deals with the universally quantified variables exclusively. Solver A gets *instantiations* of ϕ in which the universal variables are assigned, and solver B gets instantiations of $\neg\phi$ in which the existential variables are assigned. The

satisfying assignment found by one solver is used to obtain a new instantiation for the other. This loop is repeated until one solver returns unsatisfiable. This approach realizes a natural application of the counter-example guided abstraction refinement (CEGAR) paradigm [17]. A detailed survey on 2QBF solving is given in [4].

A significant advancement of expansion-based solving for QBF with an arbitrary number of quantifier alternations was made with the solver RAReQS [28,29], which recursively applies the previously discussed 2QBF approach [33] for each quantifier alternation. The approach turned out to be highly competitive.[1] For formalizing this solving approach the calculus ∀Exp+Res was introduced [6], and proof-theoretical investigations revealed the orthogonal strength of ∀Exp+Res and Q-resolution [35], the QBF variant of the resolution calculus that forms the basis for QCDCL-based solvers. Research on the proof complexity of QBF has identified an exponential separation between Q-resolution and the ∀Exp+Res system. There are families of QBFs for which any Q-resolution proof has exponential size, in contrast to ∀Exp+Res proofs of polynomial size, and vice versa. Hence these two systems have orthogonal strength.

Recent work successfully combines machine learning with this CEGAR approach [27]. Motivated by the success of expansion-based QBF solving, several other approaches [12,32, 42,46–48] have been presented that are based on levelised SAT solving, i.e., one SAT solver is responsible for the variables of one quantifier block. In this paper, we also introduce a solving approach that is based upon propositional abstraction but considers the whole quantifier prefix at once.

3 Preliminaries

The QBFs considered in this paper are in prenex normal form $\Pi.\phi$ where Π is a quantifier prefix $Q_1x_1Q_2x_2\ldots Q_nx_n$ over the set of variables $X = \{x_1, \ldots, x_n\}$ with $Q_i \in \{\forall, \exists\}$ and $x_i \neq x_j$ for $i \neq j$. The propositional formula ϕ contains only variables from X. Unless stated otherwise, we do not make any assumptions on the structure of ϕ. Sometimes $\Pi.\phi$ is in *prenex conjunctive normal form* (PCNF), i.e., Π is a prefix as introduced before and ϕ is a conjunction of clauses. A clause is a disjunction of literals, and a literal is a variable or the negation of a variable. The prefix imposes the order $<_\Pi$ on the elements of X such that $x_i <_\Pi x_j$ if $i < j$. By U_Π (E_Π) we denote the set of universally (existentially) quantified variables of the prefix Π. If clear from the context we omit the subscript Π. We assume the standard semantics of QBF. A QBF consisting of only the syntactic truth constant $\bot$ ($\top$) is false (true). A QBF $\forall x\Pi.\phi$ is true if $\Pi.\phi[x \leftarrow \top]$ and $\Pi.\phi[x \leftarrow \bot]$ are both true, where $\phi[x \leftarrow t]$ is the substitution of each occurrence of x by t in ϕ. A QBF $\exists x\Pi.\phi$ is true if $\Pi.\phi[x \leftarrow \top]$ or $\Pi.\phi[x \leftarrow \bot]$ is true.

Given a set X of variables, we call a total function $\sigma : X \to \{\top, \bot, \epsilon\}$ an assignment for X. If there is an $x \in X$ with $\sigma(x) = \epsilon$ then σ is a *partial* assignment, otherwise σ is a *full* assignment of X. Informally, $\sigma(x) = \epsilon$ means that σ does not assign a truth value to variable x. A restriction $\sigma|_Y : Y \to \{\top, \bot, \epsilon\}$ of assignment $\sigma : X \to \{\top, \bot, \epsilon\}$ to $Y \subseteq X$ is defined by $\sigma|_Y(x) = \sigma(x)$ if $x \in Y$, otherwise $\sigma|_Y(x) = \epsilon$. By Σ_X we denote the set of all full assignments $\sigma : X \to \{\top, \bot\}$. Let ϕ be a propositional formula over X. By $\sigma(\phi)$ we denote the application of assignment $\sigma : X \to \{\top, \bot, \epsilon\}$ on ϕ, i.e., $\sigma(\phi)$ is the formula obtained by replacing variables $x \in X$ by $\sigma(x)$ if $\sigma(x) \in \{\top, \bot\}$ and performing standard propositional simplifications. Let ϕ, ψ be propositional formulas over the set of variables

[1] http://www.qbflib.org.

X. If for every full assignment $\sigma \in \Sigma_X$, $\sigma(\phi) = \sigma(\psi)$ then ϕ and ψ are equivalent. Let $\tau\colon X \to \{\top, \bot, \epsilon\}$ and $\sigma\colon Y \to \{\top, \bot, \epsilon\}$ be assignments such that for every $x \in X \cap Y$, $\tau(x) = \sigma(x)$ if $\tau(x) \neq \epsilon$ and $\sigma(x) \neq \epsilon$. Then the composite assignment of σ and τ is denoted by $\sigma\tau\colon X \cup Y \to \{\top, \bot, \epsilon\}$ and for every propositional formula ϕ over $X \cup Y$, it holds that $\sigma\tau(\phi) = \tau\sigma(\phi) = \sigma(\tau(\phi)) = \tau(\sigma(\phi))$. Furthermore, we use the equality $\sigma\sigma = \sigma$ for any assignment σ.

Example 1 Let $\sigma\colon X \to \{\top, \bot, \epsilon\}$ be an assignment over variables $\{a, b, x, y\}$ defined by $\sigma(a) = \top$, $\sigma(b) = \epsilon$, $\sigma(x) = \top$, and $\sigma(y) = \epsilon$. The restriction $\tau = \sigma|_Y$ of σ to $Y = \{x, y\}$ is given by $\tau(a) = \epsilon$, $\tau(b) = \epsilon$, $\tau(x) = \top$, $\tau(y) = \epsilon$. For the propositional formula $\phi = (x \vee a \vee y) \wedge (\neg x \vee \neg a \vee y) \wedge (\neg y \vee b)$, the application of σ and τ on ϕ gives us $\sigma(\phi) = y \wedge (\neg y \vee b)$ and $\tau(\phi) = (\neg a \vee y) \wedge (\neg y \vee b)$.

4 Expansion

In the following, we introduce the notation and terminology used for describing expansion-based QBF solving in general, and the algorithm introduced in the next section in particular. We first define the notion of *instantiation* that is inspired by the axiom rule of the calculus ∀Exp+Res [31] which is introduced in Sect. 6.

Definition 1 Let $\Pi.\phi$ be a QBF with prefix $\Pi = Q_1 x_1 \ldots Q_n x_n$ over the set of variables $X = \{x_1, \ldots, x_n\}$ and $\sigma\colon Y \to \{\top, \bot, \epsilon\}$ with $Y \subseteq X$ an assignment. If $Y \subset X$, we extend the domain of σ to X by setting $\sigma(x) = \epsilon$ if $x \in X$ but $x \notin Y$. The *instantiation of ϕ by σ*, denoted by ϕ^σ, is obtained from ϕ as follows:

1. All variables $x \in X$ with $\sigma(x) \neq \epsilon$ are set to $\sigma(x)$;
2. All variables $x \in X$ with $\sigma(x) = \epsilon$ are replaced by x^ω where annotation ω is uniquely defined by the sequence $\sigma(x_{k_1})\sigma(x_{k_2}) \ldots \sigma(x_{k_m})$ such that the set formed from the variables x_{k_i} contains all variables of X with $x_{k_i} <_\Pi x$ and $\sigma(x_{k_i}) \neq \epsilon$. Furthermore, $x_{k_i} <_\Pi x_{k_j}$ if $k_i < k_j$;
3. All truth constants occurring in the formula (not in the annotations) are eliminated by standard simplification rules.

If we instantiate a QBF $\Pi.\phi$ with the full assignment $\sigma\colon U_\Pi \to \{\top, \bot\}$ of the universal variables, we obtain a propositional formula that contains only (possibly annotated) variables from E_Π. The dual holds for the instantiation by a full assignment $\sigma\colon E_\Pi \to \{\top, \bot\}$ of the existential variables.

Example 2 Given the QBF $\forall a \exists x \forall b \exists y.\phi$ with $\phi = ((x \vee a \vee y) \wedge (\neg x \vee \neg a \vee y) \wedge (\neg y \vee b))$. Then $U = \{a, b\}$ and $E = \{x, y\}$. Let $\sigma\colon U \to \{\top, \bot, \epsilon\}$ be defined by $\sigma(a) = \top$ and $\sigma(b) = \bot$. Then $\phi^\sigma = (\neg x^\top \vee y^{\top\bot}) \wedge \neg y^{\top\bot}$. Further, let $\tau\colon E \to \{\top, \bot, \epsilon\}$ with $\tau(x) = \bot$ and $\tau(y) = \bot$. Then $\phi^\tau = a$. Note that a is not annotated because it occurs in the first quantifier block.

Sometimes we want to remove the annotations from an assignment or an instantiated formula. Therefore, we introduce the following notation. Let ϕ^σ be an instantiation by assignment $\sigma\colon X \to \{\top, \bot, \epsilon\}$ and X^σ the set of annotated variables. If we have an assignment $\tau\colon X^\sigma \to \{\top, \bot, \epsilon\}$, then we define $\tau^{-\sigma}\colon X \to \{\top, \bot, \epsilon\}$ by $\tau^{-\sigma}(x) = \tau(x^\sigma)$ for $x^\sigma \in X^\sigma$. If we have an instantiated formula ϕ^σ, then $(\phi^\sigma)^{-\sigma}$ is the formula obtained by replacing every annotated variable $x^\sigma \in X^\sigma$ by x. In general, $(\phi^\sigma)^{-\sigma} \neq \phi$.

input : QBF $\Pi.\phi$ with universal variables U and existential variables E
output: truth value of $\Pi.\phi$

1 $A_0 := \{\alpha_0\}$, where $\alpha_0 : U \to \{\top, \bot\}$ is an arbitrary assignment
2 $S_0 := \emptyset$
3 $i := 1$
4 **while** *true* **do**
5 $\quad (isUnsat, \tau) := \mathrm{SAT}(\bigwedge_{\alpha \in A_{i-1}} \phi^\alpha)$
6 $\quad$ **if** $isUnsat$ **then** return false;
7 $\quad S_i := S_{i-1} \cup \{(\tau|_{E^\alpha})^{-\alpha} \mid \alpha \in A_{i-1}\}$
8 $\quad (isUnsat, \rho) := \mathrm{SAT}(\bigwedge_{\sigma \in S_i} \neg\phi^\sigma)$
9 $\quad$ **if** $isUnsat$ **then** return true;
10 $\quad A_i := A_{i-1} \cup \{(\rho|_{U^\sigma})^{-\sigma} \mid \sigma \in S_i\}$
11 $\quad i{+}{+}$
12 **end**

Fig. 1 Non-Recursive Expansion-Based Algorithm

Example 3 Reconsider the propositional formula ϕ and assignments σ, τ from Example 2 above. Then $(\phi^\sigma)^{-\sigma} = ((\neg x^\top \vee y^{\top\bot}) \wedge \neg y^{\top\bot})^{-\sigma} = (\neg x \vee y) \wedge \neg y$. Furthermore, $(\phi^\tau)^{-\tau} = (a)^{-\tau} = a$.

Lemma 1 *Let $\Pi.\phi$ be a QBF with variables X and $\sigma : X \to \{\top, \bot, \epsilon\}$ be a partial assignment. Then $(\phi^\sigma)^{-\sigma}$ and $\sigma(\phi)$ are equivalent.*

Proof By induction over the formula structure. For the base case let $\phi = x$ with $x \in X$. If $\sigma(x) = \epsilon$, then $\sigma(\phi) = x$, $\phi^\sigma = x^\omega$, and $(\phi^\sigma)^{-\sigma} = x$. Otherwise, $\phi^\sigma = \sigma(x)$. Obviously, $\sigma(\phi) = \sigma(x) = (\sigma(x))^{-\sigma} \in \{\top, \bot\}$. The induction step naturally follows from the semantics of the logical connectives. □

Finally, we specify the semantics of a QBF in terms of universal and existential expansion on which expansion-based QBF solving is founded.

Lemma 2 *Let $\Phi = \Pi.\phi$ be a QBF with universal variables U. There is a set of assignments $A \subseteq \Sigma_U$ with $\bigwedge_{\alpha \in A} \phi^\alpha$ is unsatisfiable if and only if Φ is false.*

The lemma above has a dual version for true QBFs. This duality plays a prominent role in our novel solving algorithm.

Lemma 3 *Let $\Phi = \Pi.\phi$ be a QBF with existential variables E. There is a set of assignments $S \subseteq \Sigma_E$ with $\bigvee_{\sigma \in S} \phi^\sigma$ is valid if and only if Φ is true.*

5 A non-recursive algorithm for expansion-based QBF solving

The pseudo-code in Fig. 1 summarizes the basic idea of our novel approach for solving the QBF $\Pi.\phi$ with universal variables U and existential variables E.

First, an arbitrary assignment α_0 for the universal variables is selected in Line 1. The instantiation ϕ^{α_0} is handed over to a SAT solver. If ϕ^{α_0} is unsatisfiable, then $\Pi.\phi$ is false and the algorithm returns. Otherwise, $\tau : E^{\alpha_0} \to \{\top, \bot\}$ is a satisfying assignment of ϕ^{α_0}. Let σ_1 denote the assignment $\tau^{-\alpha_0}$. Then $\alpha_0\sigma_1$ is a satisfying assignment of ϕ.

Next, the propositional formula $\neg\phi^{\sigma_1}$ is handed over to a SAT solver for checking the validity of ϕ^{σ_1}. If $\neg\phi^{\sigma_1}$ is unsatisfiable, then $\Pi.\phi$ is true and the algorithm returns. If $\neg\phi^{\sigma_1}$ is satisfiable, then $\rho\colon U^{\sigma_1} \to \{\top, \bot\}$ is a satisfying assignment of $\neg\phi^{\sigma_1}$. Let α_1 denote the assignment $\rho^{-\sigma_1}$. Then $\alpha_1\sigma_1$ is a satisfying assignment for $\neg\phi$. The following lemma shows that α_0 and α_1 are different.

Lemma 4 *Let $\Pi.\phi$ be a QBF with universal variables U and existential variables E. Further, let $\alpha\colon U \to \{\top, \bot\}$ be an assignment such that the instantiation ϕ^α is satisfiable and has the satisfying assignment $\tau\colon E^\alpha \to \{\top, \bot\}$. Let $\sigma\colon E \to \{\top, \bot\}$ with $\sigma = \tau^{-\alpha}$. Then α falsifies $(\neg\phi^\sigma)^{-\sigma}$.*

Proof Since ϕ^α is satisfied by τ, ϕ is satisfied by the composite assignment $\alpha\tau^{-\alpha} = \alpha\sigma$, and therefore $\neg\phi$ is falsified by $\alpha\sigma$. Then α falsifies $\sigma(\neg\phi)$. According to Lemma 1 $\sigma(\neg\phi)$ is equivalent to $(\neg\phi^\sigma)^{-\sigma}$. Then α also falsifies $(\neg\phi^\sigma)^{-\sigma}$. □

In the next round of the algorithm, the propositional formula $\phi^{\alpha_0} \wedge \phi^{\alpha_1}$ is handed over to a SAT solver. If this formula is unsatisfiable, $\Pi.\phi$ is false and the algorithm returns. Otherwise, it is satisfiable under some assignment $\tau\colon E^{\alpha_0} \cup E^{\alpha_1} \to \{\top, \bot\}$, then at least one new assignment $\sigma_2\colon E \to \{\top, \bot\}$ with $\sigma_2 \neq \sigma_1$ can be extracted from $\tau|_{E^{\alpha_i}}$ with $0 \leq i \leq 1$. This assignment is then used for obtaining a new propositional formula $\phi^{\sigma_1} \vee \phi^{\sigma_2}$. To show the validity of this formula, its negation is passed to a SAT solver. If this formula is unsatisfiable, $\Pi.\phi$ is true and the algorithm returns. Otherwise, it is satisfiable under the assignment $\rho\colon U^{\sigma_1} \cup U^{\sigma_2} \to \{\top, \bot\}$. A new assignment $\alpha_2\colon U \to \{\top, \bot\}$ with $\alpha_2 \neq \alpha_1 \neq \alpha_0$ is obtained from $\rho|_{A^{\sigma_i}}$ with $1 \leq i \leq 2$. This assignment is then used in the next round of the algorithm. In this way, the propositional formulas $\bigwedge_{\alpha\in\Sigma_U} \phi^\alpha$ and $\bigvee_{\sigma\in\Sigma_E} \phi^\sigma$ are generated. If $\bigwedge_{\alpha\in A} \phi^\alpha$ is unsatisfiable for some $A \subseteq \Sigma_U$, by Lemma 2 $\Pi.\phi$ is false. Dually, if $\bigvee_{\sigma\in S} \phi^\sigma$ is valid for some $S \subseteq \Sigma_E$, by Lemma 3 $\Pi.\phi$ is true. The algorithm iteratively extends the sets A and S by adding parts of satisfying assignments of ϕ to S and parts of falsifying assignments to A. In particular, A is extended by assignments of the universal variables and S is extended by assignments of the existential variables. The order in which assignments are considered depends on the used SAT solver.

Example 4 We show how to solve the QBF $\forall a \exists x \forall b \exists y.\phi$ with $E = \{x, y\}$, $U = \{a, b\}$, and $\phi = ((a \vee x \vee y) \wedge (\neg a \vee \neg x \vee y) \wedge (b \vee \neg y))$ with the algorithm presented above. This formula can be solved in two iterations:

Init: We start with some random assignment $\alpha_0\colon U \to \{\top, \bot\}$, for example with $\alpha_0(a) = \top$ and $\alpha_0(b) = \bot$.

Iteration 1: The formula $\phi^{\alpha_0} = (\neg x^\top \vee y^{\top\bot}) \wedge \neg y^{\top\bot}$ is passed to a SAT solver and found satisfiable under the assignment $\tau\colon E^{\alpha_0} \to \{\top, \bot\}$ with $\tau(x^\top) = \bot$ and $\tau(y^{\top\bot}) = \bot$. By removing the variable annotations we get assignment $\sigma_1 = (\tau|_{E^{\alpha_0}})^{-\alpha_0}$, where $\sigma_1\colon E \to \{\top, \bot\}$ with $\sigma_1(x) = \bot$ and $\sigma_1(y) = \bot$. Based on this assignment we obtain $\phi^{\sigma_1} = a$. The formula $\neg\phi^{\sigma_1}$ is passed to a SAT solver. It is satisfiable and has the satisfying assignment $\rho\colon U^{\sigma_1} \to \{\top, \bot\}$ with $\rho(a) = \bot$ and $\rho(b^\bot) = \top$, which we then reduce to $\alpha_1 = (\rho|_{U^{\sigma_1}})^{-\sigma_1}$ where $\alpha_1\colon U \to \{\top, \bot\}$ with $\alpha_1(a) = \bot$ and $\alpha_1(b) = \top$.

Iteration 2: The formula $\phi^{\alpha_0} \wedge \phi^{\alpha_1} = (\neg x^\top \vee y^{\top\bot}) \wedge \neg y^{\top\bot} \wedge (x^\bot \vee y^{\bot\top})$ is passed to a SAT solver in the second iteration. It is satisfiable and one satisfying assignment is $\tau\colon E^{\alpha_0} \cup E^{\alpha_1} \to \{\top, \bot\}$ with $\tau(x^\top) = \bot, \tau(x^\bot) = \top, \tau(y^{\top\bot}) = \bot, \tau(y^{\bot\top}) = \bot$. From τ, we can extract the assignment $\sigma_2 = (\tau|_{E^{\alpha_1}})^{-\alpha_1}$ where $\sigma_2\colon E \to \{\top, \bot\}$ with $\sigma_2(x) = \top$ and $\sigma_2(y) = \bot$. Note that for any choice of τ, $\sigma_2 \neq \sigma_1$. Next, we construct $\phi^{\sigma_1} \vee \phi^{\sigma_2} = a \vee \neg a$. This formula is a tautology, so its negation that is passed to a SAT solver is unsatisfiable, hence $\Pi.\phi$ is true.

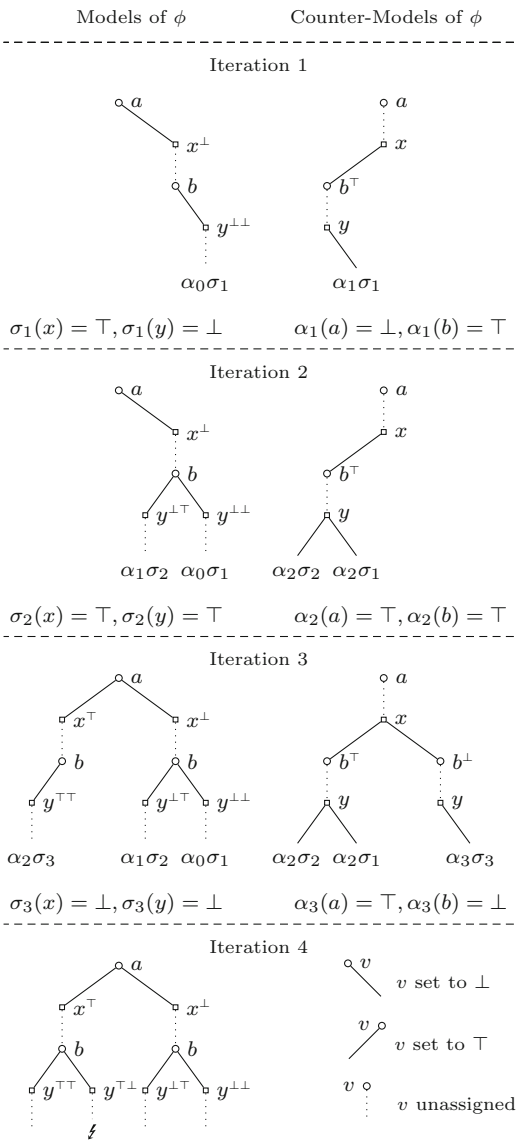


Fig. 2 Expansion trees relating the assignments found during solving the QBF $\forall a \exists x \forall b \exists y.\phi$ in Example 5, with initial assignment $\alpha_0(a) = \bot, \alpha_0(b) = \bot$. The assignments shown in the leaves of the trees satisfy (left trees) or falsify (right trees) ϕ

The soundness of our algorithm immediately follows from Lemmas 2 and 3 : the algorithm returns false (true) if, in some iteration i, it finds that the current partial expansion $\bigwedge_{\alpha \in A_{i-1}} \phi^{\alpha}$ (respectively $\bigwedge_{\sigma \in S_i} \neg \phi^{\sigma}$) is unsatisfiable.

Theorem 1 *The algorithm shown in Fig. 1 is sound.*

For showing that the algorithm also terminates, we argue that sets A_i and S_i increase in iteration $i+1$. To this end, we have to relate the variables of the QBF, the annotated variables as well as their assignments. Before we give the proof, we first consider another example in which we illustrate how the different assignments are related.

Example 5 We show one possible run of the algorithm presented above for the QBF $\Phi := \forall a \exists x \forall b \exists y . \phi$ with

$$\phi := (a \wedge b \wedge \neg x \wedge \neg y) \vee (\neg a \wedge x \wedge (b \leftrightarrow y))$$

and how it iteratively generates the sets Σ_U and Σ_E. Figure 2 shows the expansion trees that are implicitly built during the search. An expansion tree relates the variables of the partial expansion of Φ constructed from A_i (left column) and S_i (right column). Solid edges indicate that the variable on the top has been set by an assignment from A_i or S_i, and dotted edges indicate that the variable has to be assigned a value by the SAT solver. The order of the (annotated) variables in the expansion tree respects the order of the (original) variables in the prefix.

Init: For the initialization of A_0, an arbitrary assignment $\alpha_0 \colon U \to \{\top, \bot\}$ is chosen. Let $\alpha_0(a) = \bot$ and $\alpha_0(b) = \bot$.

Iteration 1: $\phi^{\alpha_0} := x^{\bot} \wedge \neg y^{\bot\bot}$ is satisfiable. Assignment $\sigma_1 \colon E \to \{\top, \bot\}$, with $\sigma_1(x) = \top$ and $\sigma_1(y) = \bot$, is extracted from model $\tau \colon E^{\alpha_1} \to \{\top, \bot\}$ and added to S_1. Now $\phi^{\sigma_1} := \neg a \wedge \neg b^{\top}$ is checked for validity. Assignment $\alpha_1 \colon U \to \{\top, \bot\}$, with $\alpha_1(a) = \bot$ and $\alpha_1(b) = \top$, obtained from counter-example $\rho \colon U^{\sigma_1} \to \{\top, \bot\}$ is added to A_1.

Iteration 2: Next, $\phi^{\alpha_0} \wedge \phi^{\alpha_1}$ with $\phi^{\alpha_1} := x^{\bot} \wedge y^{\bot\top}$ is checked. From model $\tau \colon E^{\alpha_0} \cup E^{\alpha_1} \to \{\top, \bot\}$, again σ_1 can be extracted for ϕ^{α_0}. For ϕ^{α_1} a new assignment σ_2 which is not in S_1 is found and added to S_2. In particular, we get $\sigma_2 \colon E \to \{\top, \bot\}$ with $\sigma_2(x) = \top$ and $\sigma_2(y) = \top$. When the validity of $\phi^{\sigma_1} \vee \phi^{\sigma_2}$ with $\phi^{\sigma_2} := \neg a \wedge b^{\top}$ is checked, we get a counter-example $\rho \colon U^{\sigma_1} \cup U^{\sigma_2} \to \{\top, \bot\}$, from which $\alpha_2 \colon U \to \{\top, \bot\}$, with $\alpha_2(a) = \top$ and $\alpha_2(b) = \top$, can be extracted. Assignment α_2 is added to A_2 leading to a new path in the left expansion tree (Iteration 3 in Fig. 2).

Iteration 3: Next, $\phi^{\alpha_0} \wedge \phi^{\alpha_1} \wedge \phi^{\alpha_2}$ with $\phi^{\alpha_2} := \neg x^{\top} \wedge \neg y^{\top\top}$ is checked. From model $\tau \colon E^{\alpha_0} \cup E^{\alpha_1} \cup E^{\alpha_2} \to \{\top, \bot\}, \sigma_3 \colon E \to \{\top, \bot\}$ is extracted, satisfying ϕ^{α_2}. This assignment is different from both σ_1 and σ_2: $\sigma_3(x) = \bot$ and $\sigma_3(y) = \bot$. This again results in a new branch of the expansion tree (see left expansion tree of Iteration 4 in Fig. 2). The resulting formula $\phi^{\sigma_1} \vee \phi^{\sigma_2} \vee \phi^{\sigma_3}$ with $\phi^{\sigma_3} := a \wedge b^{\bot}$ is not valid, and from the counter-example $\rho \colon U^{\sigma_1} \cup U^{\sigma_2} \cup U^{\sigma_3} \to \{\top, \bot\}$ we get $\alpha_3 \colon U \to \{\top, \bot\}$ with $\alpha_3(a) = \top$ and $\alpha_3(b) = \bot$.

Iteration 4: Finally, the full expansion $\phi^{\alpha_0} \wedge \phi^{\alpha_1} \wedge \phi^{\alpha_2} \wedge \phi^{\alpha_3}$ with $\phi^{\alpha_3} := \bot$ is not satisfiable, meaning that the original formula $\forall a \exists x \forall b \exists y . \phi$ is false.

In the example above we saw that new assignments are generated in each iteration because A_i and S_i build models and counter-models of ϕ. The following definition formalizes the relationship between A_i and S_i.

Definition 2 Let $\Pi . \phi$ be a QBF over universally quantified variables U and existentially quantified variables E. Further, let $A \subseteq \{\alpha \mid \alpha \colon U \mapsto \{\top, \bot\}\Delta\}$ and $S \subseteq \{\sigma \mid \sigma \colon E \mapsto \{\top, \bot\}\Delta\}$. If for every assignment $\sigma \in S$, there exists an assignment $\alpha \in A$ such that $\alpha\sigma(\neg\phi)$ is true, then we say that A *completes* S. If for every assignment $\alpha \in A$, there exists an assignment $\sigma \in S$ such that $\alpha\sigma(\phi)$ is true, then we say that S *completes* A.

We now show that S_i completes A_{i-1} and A_i completes S_i if the algorithm does not terminate in iteration i because of the unsatisfiability of the respective expansion.

Lemma 5 *Let $\Pi . \phi$ be a QBF over universally quantified variables U and existentially quantified variables E. Further, let A_{i-1} and A_i with $A_{i-1} \subseteq A_i$ be two sets of full assignments to the universal variables and let S_i be a set of full assignments to the existential variables obtained by iteration i during an execution of the algorithm shown in Fig. 1.*

(1) If $\bigwedge_{\alpha \in A_{i-1}} \phi^\alpha$ is satisfiable, then S_i completes A_{i-1}, i.e., for every $\mu \in A_{i-1}$, there is an assignment $\nu \in S_i$ such that $\mu\nu(\phi)$ is true.

(2) If $\bigwedge_{\sigma \in S_i} \neg\phi^\sigma$ is satisfiable, then A_i completes S_i, i.e., for every $\nu \in S_i$, there is an assignment $\mu \in A_i$ such that $\nu\mu(\neg\phi)$ is true.

Proof By contradiction. For (1), assume there is an assignment $\mu \in A_{i-1}$ such that there is no assignment $\nu \in S_i$ with $\mu\nu(\phi)$ is true. By assumption $\bigwedge_{\alpha \in A_{i-1}} \phi^\alpha$ is satisfiable, so there is a satisfying assignment τ with $\tau|_{E^\mu}(\phi^\mu)$ is true. Then also $\mu(\tau|_{E^\mu})^{-\mu}(\phi)$ is true. But $(\tau|_{E^\mu})^{-\mu} \in S_i$. For (2), assume that there is an assignment $\mu \in S_i$ such that there is no $\nu \in A_i$ with $\mu\nu(\neg\phi)$ is true. The rest of the argument is similar as in (1). □

Next, we show that the addition of new assignments A' to a set A of universal assignments forces a set S of existential assignments to increase if some completion criteria hold.

Lemma 6 *Let $\Phi = \Pi.\phi$ be a QBF over universally quantified variables U and existentially quantified variables E. Further, let $A \cup A'$ be a set of universal assignments such that $A \cap A' = \emptyset$ and $A' \neq \emptyset$. Let S be a set of existential assignments and assume that $\bigwedge_{\sigma \in S} \neg\phi^\sigma$ has the satisfying assignment ρ, $A' \subseteq \{(\rho|_{U^\sigma})^{-\sigma} \mid \sigma \in S\}$.*

If S completes A, and $A \cup A'$ completes S, and $\bigwedge_{\alpha \in A \cup A'} \phi^\alpha$ evaluates to true under assignment τ, then there exists an assignment $\nu \in \{(\tau|_{E^\alpha})^{-\alpha} \mid \alpha \in A \cup A'\}$ with $\nu \notin S$.

Proof By induction over the number of variables in Π.

Base Case. Assume that Φ has only one variable, i.e., $\Pi = Qx$. Note that $|A'| = 1$ because x is outermost in the prefix and A' is obtained from sub-assignments of ρ. If $Q = \forall$, then the elements of A are full assignments of ϕ, and S is either empty, or it contains the empty assignment $\omega\colon \emptyset \mapsto \{\top, \bot\}$. Let $A' = \{\mu\}$. If S is empty, so is A (because S has to complete A). If τ is a satisfying assignment of ϕ^μ, then $\nu = \tau = \omega$ is the empty assignment and $\nu \notin S$. Otherwise, $\omega \in S$. If there is an assignment $\alpha \in A$, then $\phi^\alpha \wedge \phi^\mu$ is a full expansion of Φ. If this full expansion is true, then $\neg\phi$ is unsatisfiable. Otherwise, $\phi^\alpha \wedge \phi^\mu$ is unsatisfiable. In both cases, the necessary preconditions for the lemma are not fulfilled. If $A = \emptyset$, then $\mu\omega(\neg\phi)$ is true. Then ϕ^μ is unsatisfiable, again violating a precondition. If $Q = \exists$, then $\mu = \omega$ and $A = \emptyset$. If $S = \emptyset$ and $\phi^\omega = \phi$ has the satisfying assignment τ, then $\nu = \tau$ and $\nu \notin S$. Otherwise, if there is an assignment $\sigma \in S$, then $\omega\sigma(\neg\phi)$ is true, because $A \cup \{\mu\} = \{\omega\}$ completes S. Hence, if assignment τ satisfies ϕ^μ, then $\nu = \tau$, so $\nu \notin S$.

Induction Step. Assume the lemma holds for QBFs with n variables. We show that it also holds for QBFs with $n + 1$ variables. Let $\Phi = Qx\Pi.\phi$ be a QBF over existential variables E and universal variables U with $\Pi = Q_1x_1 \ldots Q_nx_n$ and $A \cup A'$ and S be as required (S completes A, $A \cup A'$ completes S, $\bigwedge_{\alpha \in A \cup A'} \phi^\alpha$ has a satisfying assignment τ, and $\bigwedge_{\sigma \in S} \neg\phi^\sigma$ has a satisfying assignment ρ from which A' is obtained).

If $Q = \forall$, then all assignments $\alpha \in A'$ assign the same value t to x, i.e., $\alpha(x) = t$, because these assignments are extracted from assignment ρ and since x is the outermost variable of the prefix of Φ, $\rho(x) = t$. Further, let $A^t = \{\alpha \in A \mid \alpha(x) = t\}$. It is easy to argue that for $\Pi.\phi[x \leftarrow t]$ together with the assignment sets $A^t \cup A'$ and S the induction hypothesis applies, i.e., there is an assignment $\nu \notin S$ with $\nu \in \{(\tau'|_{E^\alpha})^{-\alpha} \mid \alpha \in A^t \cup A'\}$ where τ' is the part of τ that satisfies $\bigwedge_{\alpha \in A^t \cup A'}(\phi[x \leftarrow t])^\alpha$. Obviously, $\nu \in \{(\tau|_{E^\alpha})^{-\alpha} \mid \alpha \in A \cup A'\}$.

If $Q = \exists$, assume that $\tau(x) = t$. Let $\{\sigma \in S \mid \sigma(x) = t\} \subseteq S^t \subseteq S$, and let $A^t \subseteq A$ such that the induction hypothesis applies to $\Pi.\phi[x \leftarrow t]$, $A^t \cup A'$, and S^t. Let τ^t be those sub-assignments of τ that satisfy $\bigwedge_{\alpha \in A^t} \phi^\alpha$. Then there is an assignment ν that can be extracted from τ^t with $\nu \notin S^t$. Since $\nu(x) = t$, $\nu \notin S$. This concludes the proof. □

This property also holds in the other direction, i.e., adding a set S' of new assignments to S will force the set A to increase.

$$\frac{}{\{l^{[\tau]} \mid l \in C, l \text{ is existential}\} \cup \{\tau(l) \mid l \in C, l \text{ is universal}\}} \text{(Axiom)}$$

where

- C is a clause from the matrix of QBF $\Pi.\phi$
- τ is an assignment to all universal variables U_Π
- $[\tau]$ restricts τ to the universal variables that precede l in the prefix

$$\frac{C_1 \vee x^\tau \qquad C_2 \vee \neg x^\tau}{C_1 \vee C_2} \text{(Res)}$$

Fig. 3 The rules of the ∀Exp+Res [6,30,31]

Lemma 7 *Let $\Phi = \Pi.\phi$ be a QBF over universally quantified variables U and existentially quantified variables E. Further, let $S \cup S'$ be a set of existential assignments such that $S \cap S' = \emptyset$, $S' \neq \emptyset$, let A be a set of universal assignments, $\bigwedge_{\alpha \in A} \phi^\alpha$ has the satisfying assignment τ, $S' \subseteq \{(\tau|_{E^\alpha})^{-\alpha} \mid \alpha \in A\}$.*

If A completes S and $S \cup S'$ completes A and $\bigwedge_{\sigma \in S \cup S'} \neg\phi^\sigma$ evaluates to true under assignment ρ, then there exists an assignment $\nu \in \{(\rho|_{U^\sigma})^{-\sigma} \mid \sigma \in S \cup S'\}$ with $\nu \notin A$.

Proof The proof is analogous to the proof of Lemma 6. □

Now that we have identified the relations between the sets of universal and existential assignments, we use them to show that the algorithm from Fig. 1 terminates.

Theorem 2 *The algorithm shown in Fig. 1 terminates for any QBF $\Phi = \Pi.\phi$.*

Proof By induction over the number of iterations i, we argue that sets $A_{i-1} \subset A_i$ and $S_{i-1} \subset S_i$.

Base Case. Let $i = 1$ and $A_0 = \{\alpha_0\}$. $S_0 \subset S_1$, because $S_0 = \emptyset$ and $\sigma_1 \in S_1$ is a satisfying assignment of ϕ^{α_0} (if ϕ^{α_0} is unsatisfiable, the algorithm terminates). $A_0 \subset A_1$ directly follows from Lemma 4.

Induction Step. For $i > 1$, we argue that $S_i \subset S_{i+1}$. By induction hypothesis the theorem holds for iteration i, i.e., $A_i = A_{i-1} \cup A'$ with $A_{i-1} \cap A' = \emptyset$ and $A' \neq \emptyset$ and $S_i = S_{i-1} \cup S'$ with $S_{i-1} \cap S' = \emptyset$ and $S' \neq \emptyset$. Because of Lemma 5, S_i completes A_{i-1}, and A_i completes S_i. Furthermore, if $\bigwedge_{\sigma \in S_i} \neg\phi^\sigma$ is satisfiable under some assignment ρ (otherwise the algorithm would terminate), by construction $A' \subseteq \{(\rho|_{U^\sigma})^{-\sigma} \mid \sigma \in S_i\}$. Hence, Lemma 6 applies and if $\bigwedge_{\alpha \in A_i} \phi^\alpha$ is satisfiable under some assignment τ (otherwise the algorithm would immediately terminate), then there is an assignment $\nu \in \{(\tau|_{E^\alpha})^{-\alpha} \mid \alpha \in A_i\}$ with $\nu \notin S_i$.

The argument for $A_i \subset A_{i+1}$ is similar and uses the property shown in Lemma 7. □

Note that the algorithm presented above does not make any assumptions on the formula structure, i.e., for a QBF $\Pi.\phi$ it is not required that ϕ is in conjunctive normal form. Without any modification, our algorithm also works on formulas in PCNF—as SAT solvers typically process formulas in CNF only, we focus on this representation for the rest of the paper.

6 Relation to the ∀Exp+Res calculus

The ∀Exp+Res calculus [6,30,31] yields the theoretical foundation of our algorithm for refuting a formula $\Pi.\phi$ in PCNF with universal variables U and existential variables E.

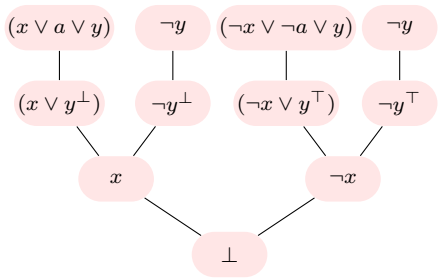


Fig. 4 ∀Exp+Res proof of $\exists x \forall a \exists y.((x \vee a \vee y) \wedge (\neg x \vee \neg y) \wedge (\neg x \vee \neg a \vee y) \wedge (\neg y))$

The ∀Exp+Res calculus consists of the two rules shown in Fig. 3. Given an assignment $\tau : U_\Pi \to \{\top, \bot\}$ of the universal variables U_Π and a clause occurring in a QBF $\Pi.\phi$, then the axiom rule instantiates C such that all universal literals u are assigned value $\tau(u)$ and the remaining existential literals l are annotated by $[\tau]$, i.e., by those universals that precede the variable of l in the prefix. In the notation introduced before, we can write the axiom rule by

$$\frac{}{C^{\tau}}$$

Note that only clauses that do not contain $\tau(l) = \top$ are of interest for a refutation proof. Further, any occurrences of $\bot$ are omitted in the proof.

The resolution rule corresponds exactly to propositional resolution, i.e., the annotated variables are seen as propositional variables. Resolution between two clauses is only possible, if one contains a literal x^{σ} and the other clause contains a literal $\neg x^{\tau}$ and $\sigma = \tau$, i.e., the pivot literals must have the same annotation. Note that we represent clauses as sets of literals.

A derivation in ∀Exp+Res is a sequence of clauses where each clause is either obtained by the axiom rule or derived from previously derived clauses by the application of the resolution rule. A refutation of a PCNF $\Pi.\phi$ is a derivation of the empty clause. The application of the axiom rule instantiates the universal variables of *one* clause of ϕ. If enough of these instantiations can be found in order to derive the empty clause by the application of the resolution rule, the QBF $\Pi.\phi$ is false.

Our algorithm presented in Fig. 1 does not instantiate selected clauses of the input formula, but all clauses of the matrix ϕ at once using a particular assignment of the universal variables. Hence, when the SAT solver finds $\psi_\forall = \bigwedge_{\alpha \in A_i} \phi^{\alpha}$ unsatisfiable for some A_i, not necessarily all clauses of $\psi_\forall$ are required to derive the empty clause via resolution, but only one minimal unsatisfiable core of $\psi_\forall$, i.e., a subset of the clauses such that the removal of any clause would make this formula satisfiable. This observation leads us to the following proposition.

Proposition 1 *Let $\Pi.\phi$ be a false QBF. Further, let $\psi_\forall = \bigwedge_{\alpha \in A_i} \phi^{\alpha}$ be obtained by the application of the algorithm in Fig. 1. Further, let $\psi'_\forall$ be a minimal unsatisfiable core of $\psi_\forall$. Then there is a ∀Exp+Res refutation such that all clauses that are introduced by the axiom rule occur in $\psi'_\forall$.*

Example 6 Consider the false QBF

$$\exists x \forall a \exists y.((x \vee a \vee y) \wedge (\neg x \vee \neg y) \wedge (\neg x \vee \neg a \vee y) \wedge (\neg y)).$$

When fully expanding universal a, we obtain the propositional formula

$$((\neg x \vee \neg y^{\top}) \wedge (\neg x \vee y^{\top}) \wedge (\neg y^{\top}) \wedge (x \vee y^{\bot}) \wedge (\neg x \vee \neg y^{\bot}) \wedge (\neg y^{\bot})).$$

For proving unsatisfiability of this formula, it is enough to consider the formula $((\neg x \vee y^{\top}) \wedge (\neg y^{\top}) \wedge (x \vee y^{\bot}) \wedge (\neg y^{\bot}))$. The corresponding ∀Exp+Res proof is shown in Fig. 4. Since

existential variable x occurs outermost in the prefix, it is not annotated during the applications of the axiom rule.

Currently our implementation supports the generation of refutation proofs for false formulas and checking them for correctness. In consequence, we are now able to efficiently check the correctness of the solving results for false formulas, because the correctness check is linear in the proof size. For such proofs, we designed a novel proof format, because to best of our knowledge recent QBF solvers implementing expansion-based approaches do not support any proof generation.

Conceptually, proof generation for true QBFs works dually: instead of refuting a set of clauses, a set of cubes (conjunctions of literals) is shown to be valid. For this purpose, the resolution rule has to be modified to operate on cubes instead of clauses. In practice, however, SAT solvers are used that operate on clauses, hence an extra transformation step introducing fresh variables is required. This is currently not supported by our checker and is subject to future work.

7 Implementation

The algorithm described in Sect. 5 is realized in the solver ljtihad[2] The most recent version of ljtihad is available at

https://extgit.iaik.tugraz.at/scos/ijtihad

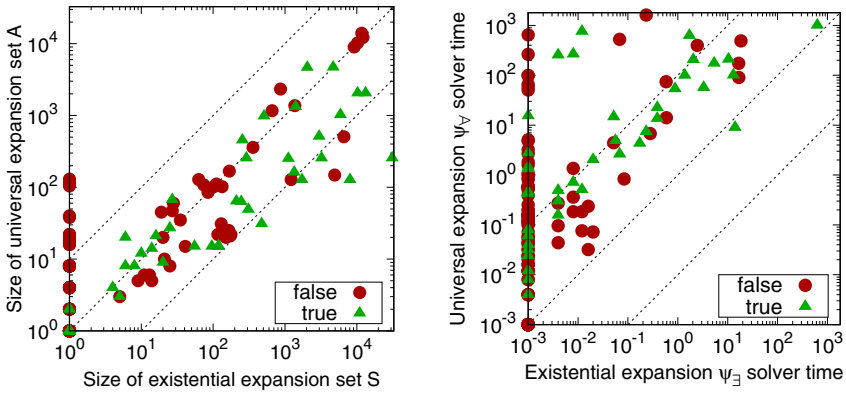
The solver is implemented in C++ and currently processes formulas in PCNF available in the QDIMACS format. For accessing SAT solvers, ljtihad uses the IPASIR interface [5], which makes changing the SAT solver very easy. The SAT solver used in all of our experiments is Glucose [2]. Although the base implementation does reasonably well, we have realized various optimizations to make ljtihad even more viable in practice. Some of them are discussed in the following.

For solving a QBF $\Pi.\phi$, the basic algorithm shown in Fig. 1 adds instantiations of ϕ to $\psi_\forall = \bigwedge_{\alpha \in A_{i-1}} \phi^\alpha$ and $\psi_\exists = \bigwedge_{\sigma \in S_i} \neg\phi^\sigma$ in each iteration i until the formula is decided. The calls to the SAT solver in Line 5 and Line 8 are done incrementally, i.e., we create two instances of the SAT solver and provide them with the clauses stemming from new instantiations of ϕ at each iteration. For simplicity, we omit indices of sets A and S and refer to an arbitrary iteration of the execution of the algorithm in the following discussion.

Figure 5 relates set sizes of A and S as well as the accumulated time that one SAT solver needs to solve $\psi_\forall$ with the time the other SAT solver needs to solve $\psi_\exists$ for the formulas of the PCNF track of QBFEVAL'17 (preprocessed with Bloqqer [8]). In this paper, we also distinguish between true and false formulas. In Fig. 5a we see that for true formulas, set S tends to be larger than A, while for false instances the picture is less clear. Figure 5b shows the overall time needed for solving $\psi_\forall$ (y-axis) and $\psi_\exists$ (x-axis). In almost all cases, the solver that handles $\psi_\forall$ needs more time than the solver that handles $\psi_\exists$. This may be founded on the observation that many QBFs have considerably more existential variables than universal variables [39], hence the instantiations added to $\psi_\forall$ are much larger than the instantiations added to $\psi_\exists$.

In Line 1 of Fig. 1, the set of universal assignments A is initialized with *one* arbitrary assignment α_0. Obviously, the set A may also be initialized with multiple assignments. In

[2] The name ljtihad refers to the effort of solving cases in Islamic law (for details see https://en.wikipedia.org/wiki/Ijtihad).



(a) Sizes of sets S and A

(b) Time used for solving $\psi_\exists$ and $\psi_\forall$

Fig. 5 Set sizes and time consumed during SAT calls for solved instances from QBFEVAL'17 preprocessed by Bloqqer

our current implementation, we initialize A with the assignments that set the variables of one universal quantifier block to $\bot$ and the variables of all other universal quantifier blocks to $\top$. The impact of various initialization heuristics remains to be investigated in future work.

In Line 7 and Line 10 our algorithm increases the size of S and A in each iteration of the main loop, as argued in Theorem 2. In the worst case, this leads to an exponential increase in space consumption. Although we detect shared clauses among the instantiations, that alone is not enough to significantly reduce the space consumption. However, some of the assignments found in an earlier iteration could become obsolete after better assignments were found. It is therefore beneficial to empty either S or A and then reconstruct them from $\psi_\forall$ and $\psi_\exists$, similarly to what is done in Line 7 and Line 10. We evaluated several heuristics for scheduling these set resets, and we found that resetting periodically and close to the memory limit works best. The regular resetting of one set has a similar effect as restarts in SAT solvers, and we observed a considerable improvement in performance, especially in terms of memory consumption. Our implementation periodically resets the set A, since experiments indicate that the resulting formula $\psi_\forall$ is much harder to solve than $\psi_\exists$ as seen in Fig. 5b. Besides the aforementioned imbalance between universal and existential variables, it is also likely due to the structure of $\psi_\exists$ which is a conjunction of formulas in disjunctive normal form. Note that this reset of A does not affect the termination argument presented in Theorem 2, since the sets A and S still complete each other.

Finally, we extended the presented approach with orthogonal reasoning techniques like QCDCL [23] for exploiting the different strengths of ∀Exp+Res and Q-resolution, yielding a hybrid solver that smoothly integrates both solving paradigms. To this end, we implemented the prototypical solver called Heretic which pursues the following idea: The main loop of the algorithm shown in Fig. 1 (Lines 4-12) is extended in a sequential portfolio style such that a QCDCL solver is periodically called. After each call, all clauses that were learned through QCDCL are added to $\Pi.\Phi$, making them available in further iterations. These new clauses potentially exclude assignments that would otherwise be possible and that could result in more iterations of the main loop.

The solver Heretic extends Ijtihad by additional invocations of the QCDCL solver DepQBF [38]. About every 30 seconds, DepQBF is called and run for about 30 seconds. The learned

clauses are obtained via the API of DepQBF. Leveraging learned cubes is subject to future work.

8 Evaluation

We evaluate non-recursive expansion as implemented in our solvers Ijtihad and its hybrid variant Heretic on the benchmarks from the PCNF track of the QBFEVAL'18 competition. All experiments were carried out on a cluster of Intel Xeon CPUs (E5-2650v4, 2.20 GHz) running Ubuntu 16.04.1 with a CPU time limit of 1800 seconds and a memory limit of 7 GB. We considered the following top-performing solvers from QBFEVAL'18: Qute [41], Rev-Qfun [27], RAReQS [28], CAQE [42,47], DynQBF [14], GhostQ [28,36], DepQBF [38], QESTO [32], and QSTS [11,12]. Our experiments are based on original benchmarks without preprocessing and benchmarks preprocessed using QRATPre+ [40], HQSpre [50], and Bloqqer [8,25] with a timeout of one hour.

The tables in the left column of Fig. 6 show the total numbers of solved instances (S), solved unsatisfiable ($\bot$) and satisfiable ones ($\top$), and total CPU time including timeouts. The plots in the right column of Fig. 6 visualize the runtimes of the respective solvers. In the first row, the results without any preprocessing are shown. Our solver Heretic is ranked third solving the most false formulas of all solvers. If the preprocessor QRATPre+ is applied, Heretic is ranked second. Only CAQE solves more formula instances. Also with the other two preprocessors, CAQE seems to be the solver that benefits most from the additional preprocessing step. In general, preprocessing has a considerable impact on the number of solved instances. With preprocessing enabled, Heretic solves up to 142 more formulas than without preprocessing. Also Ijtihad strongly benefits from preprocessing: alone it solves 151 formulas, and with a preprocessor it solves up to 229 formulas.

Notably Heretic, despite its simple design, significantly outperforms Ijtihad on the QBFEVAL'18 benchmark set. Moreover, Heretic is ranked third and second on preprocessed instances and thus is on par with state-of-the-art solvers. On the considered benchmark set, the gap in solved instances between RAReQS and Heretic is considerably smaller than the one between RAReQS and Ijtihad.

A direct comparison of our solver Heretic with RAReQS is shown in Fig. 7a. Unlike our solver, RAReQS is based on a recursive implementation of expansion. While the plot looks very balanced for the whole benchmark set, the picture changes for formulas with four or more quantifier blocks, i.e., three or more quantifier alternations (see Fig. 7b and below).

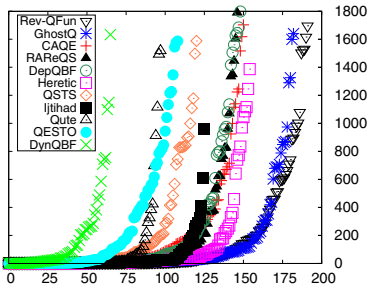
On such formulas with many quantifier blocks, the strength of Heretic becomes apparent, cf. [39]. As shown in Tables 8a, 8c and 8e, Heretic outperforms all other solvers on original instances and on instances with preprocessing by QRATPre+ and Bloqqer. The only exception are instances preprocessed by HQSpre (Table 8g).

Moreover, on entire benchmark sets without and with preprocessing (Tables 6a, 6c, 6e, and 6g), Heretic significantly outperforms both DepQBF and Ijtihad. These results indicate the potential of combining the orthogonal proof systems ∀Exp+Res as implemented in Ijtihad and Q-resolution as implemented in DepQBF in a hybrid solver like Heretic.

Although RAReQS outperforms both Ijtihad and Heretic on instances preprocessed by Bloqqer (Table 6e) RAReQS failed to solve certain instances that were solved by Ijtihad or Heretic. Table 1 shows related statistics. E.g., on instances preprocessed by HQSpre (row "HQ"), 258 instances were solved by both RAReQS and Heretic (column "R *vs.* H), 34 only by RAReQS, and 39 only by Heretic. Summing up these numbers yields a total of 331

Solver	*S*	⊥	⊤	*Time*
Rev-Qfun	192	112	80	515K
GhostQ	183	89	94	524K
Heretic	155	122	33	569K
CAQE	151	107	44	586K
DepQBF	149	87	62	592K
RAReQS	147	115	32	588K
Ijtihad	126	110	16	611K
QSTS	121	91	30	635K
QESTO	109	76	33	662K
Qute	98	79	19	665K
DynQBF	66	22	44	726K

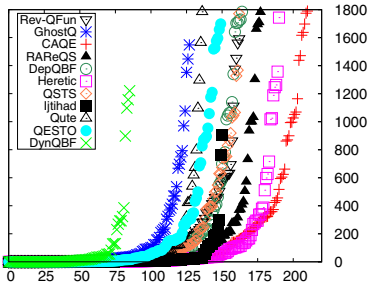
(a) Without preprocessing



(b) Plot related to Table 6a.

Solver	*S*	⊥	⊤	*Time*
CAQE	211	135	76	487K
Heretic	191	133	58	507K
RAReQS	178	120	58	533K
DepQBF	165	83	82	562K
QSTS	164	98	66	562K
Rev-Qfun	163	106	57	563K
Ijtihad	151	111	40	565K
QESTO	150	88	62	589K
Qute	137	92	45	598K
GhostQ	128	69	59	619K
DynQBF	86	24	62	685K

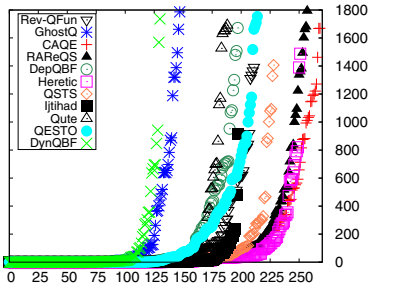
(c) Preprocessing with QRATPre+



(d) Plot related to Table 6c.

Solver	*S*	⊥	⊤	*Time*
CAQE	269	150	119	383K
RAReQS	258	159	99	399K
Heretic	252	166	86	394K
QSTS	229	135	94	435K
QESTO	215	115	100	480K
Rev-Qfun	212	123	89	473K
Ijtihad	198	128	70	480K
DepQBF	198	99	99	501K
Qute	189	114	75	512K
GhostQ	148	81	67	587K
DynQBF	131	62	69	608K

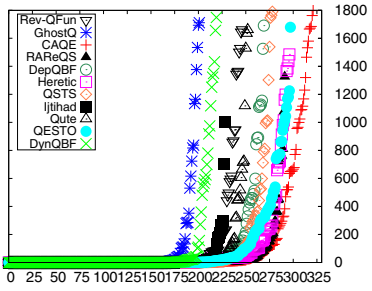
(e) Preprocessing with Bloqqer



(f) Plot related to Table 6e.

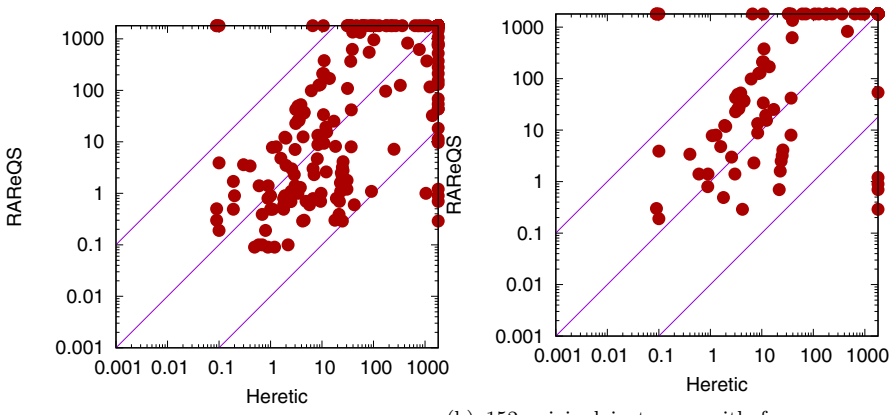
Solver	*S*	⊥	⊤	*Time*
CAQE	322	183	139	290K
QESTO	298	175	123	323K
Heretic	297	194	103	320K
RAReQS	292	180	112	317K
QSTS	279	169	110	351K
DepQBF	270	160	110	364K
Qute	253	161	92	390K
Rev-Qfun	247	162	85	408K
Ijtihad	229	162	67	424K
DynQBF	220	129	91	454K
GhostQ	202	130	72	484K

(g) Preprocessing with HQSpre



(h) Plot related to Table 6g.

Fig. 6 Results for the full QBFEval 2018 benchmark set with the application of different preprocessors



(a) All original instances.

(b) 152 original instances with four or more quantifier blocks.

Fig. 7 Scatter plots of the run times of Heretic and RAReQS on original instances (related to Tablexxxxx 6a) and on instances having four or more quantifier blocks (related to Table 8a)

Table 1 Statistics related to Tables 6a, 6c, 6e, and 6g: pairwise comparison of RAReQS (R), Ijtihad (I), Heretic (H), and DepQBF (D) by instances without (N) and with preprocessing by Bloqqer (B), HQSpre (HQ), and QRATPre+ (Q) that were solved by only one solver of the considered pair ($<$, $>$) or by both ($=$)

	R $<$	*vs.* $=$	I $>$	*R* $<$	*vs.* $=$	*H* $>$	*I* $<$	vs. $=$	*H* $>$	*D* $<$	*vs.* $=$	*H* $>$
N	32	115	11	23	124	31	3	123	32	63	86	69
B	69	189	9	40	218	34	1	197	55	32	166	86
HQ	70	222	7	34	258	39	1	228	69	29	241	56
Q	36	142	6	29	149	42	1	150	41	52	113	78

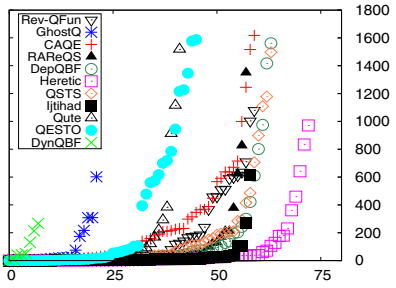
solved instances (more than any individual solver in Table 6g) that could have been solved by a *hypothetical solver* combining RAReQS and Heretic. This observation underlines the strength of expansion in general and, in particular, of the hybrid approach implemented in Heretic. Heretic solved a significant amount of instances not solved by RAReQS, and it clearly outperformed Ijtihad and DepQBF on all benchmarks (columns "*I vs. H*" and "*D vs. H*").

9 Conclusion

We presented a novel non-recursive algorithm for expansion-based QBF solving that uses only two SAT solvers for incrementally refining the propositional abstraction and the negated propositional abstraction of a QBF. We gave a concise proof of termination and soundness and demonstrated with several experiments that our prototype compares well with the state of the art. In addition to non-recursive expansion, we also studied the impact of combining Q-resolution and ∀Exp+Res in a hybrid approach. To this end, we coupled a QCDCL solver and non-recursive expansion to make clauses derived by the QCDCL solver available to the expansion solver. Experimental results indicated that the hybrid approach significantly outperforms our implementation of non-recursive expansion indicating the potential of combining expansion-based approaches with Q-resolution which gives rise to an exciting direction of future work. Further, our current implementation supports only formulas in conjunctive

Solver	*S*	⊥	⊤	*Time*
Heretic	73	57	16	147K
QSTS	64	47	17	167K
DepQBF	64	35	29	166K
Rev-Qfun	60	55	5	174K
CAQE	60	41	19	180K
Ijtihad	59	51	8	168K
RAReQS	58	52	6	173K
QESTO	46	30	16	204K
Qute	42	32	10	203K
GhostQ	22	17	5	235K
DynQBF	8	4	4	259K

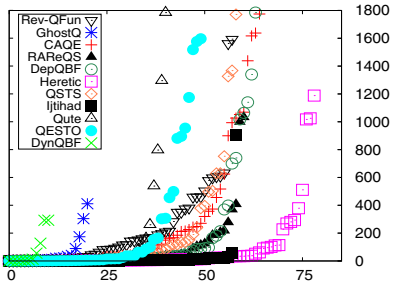
(a) Without preprocessing (152 instances)



(b) Plot related to Table 8a.

Solver	*S*	⊥	⊤	*Time*
Heretic	79	61	18	137K
CAQE	65	45	20	173K
DepQBF	64	35	29	168K
RAReQS	61	55	6	168K
QSTS	59	42	17	177K
Ijtihad	59	51	8	168K
Rev-Qfun	58	53	5	181K
QESTO	50	36	14	194K
Qute	41	32	9	204K
GhostQ	21	16	5	236K
DynQBF	11	6	5	254K

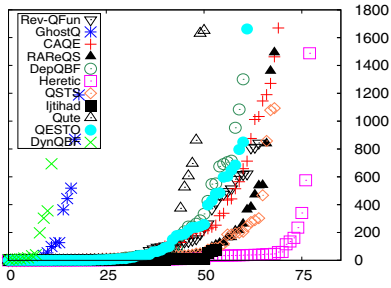
(c) Preprocessing with QRATPre+ (152 instances)



(d) Plot related to Table 8c.

Solver	*S*	⊥	⊤	*Time*
Heretic	78	63	15	95K
CAQE	70	44	26	122K
RAReQS	69	59	10	115K
QSTS	69	52	17	114K
Rev-Qfun	65	49	16	125K
QESTO	62	39	23	130K
DepQBF	61	32	29	132K
Ijtihad	54	46	8	135K
Qute	51	39	12	147K
GhostQ	19	13	6	201K
DynQBF	12	7	5	212K

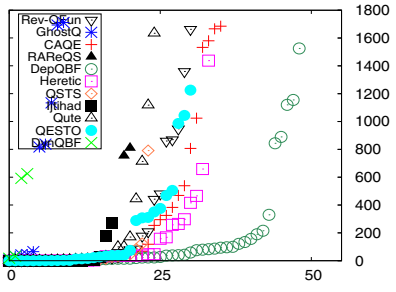
(e) Preprocessing with Bloqqer (129 instances)



(f) Plot related to Table 8e.

Solver	*S*	⊥	⊤	*Time*
DepQBF	49	23	26	76K
CAQE	36	15	21	102K
Heretic	34	24	10	99K
Rev-Qfun	31	23	8	108K
QESTO	31	18	13	106K
Qute	25	19	6	116K
QSTS	24	16	8	114K
RAReQS	21	18	3	120K
Ijtihad	18	17	1	124K
GhostQ	10	6	4	144K
DynQBF	4	2	2	150K

(g) Preprocessing with HQSpre (87 instances)



(h) Plot related to Table 8g.

Fig. 8 Results for the full QBFEval 2018 benchmark set with the application of different preprocessors such that the formulas have at least four quantifier blocks

normal form while in theory, our approach does not make any assumptions on the structure of the propositional part of the QBF. We also plan to investigate how this formula structure can be exploited for efficiently processing the negation of the formula.

Funding Open access funding provided by Graz University of Technology.

References

1. Alur R, Bodík R, Dallal E, Fisman D, Garg P, Juniwal G, Kress-Gazit H, Madhusudan P, Martin MMK, Raghothaman M, Saha S, Seshia SA, Singh R, Solar-Lezama A, Torlak E, Udupa A (2015) Syntax-guided synthesis. In: Dependable Software Systems Engineering, volume 40 of NATO Science for Peace and Security Series, D: Information and Communication Security. IOS Press, pp 1–25
2. Audemard G, Simon L (2009) Predicting learnt clauses quality in modern SAT solvers. In: IJCAI, pp 399–404
3. Ayari A, Basin DA (2002) QUBOS: deciding quantified Boolean logic using propositional satisfiability solvers. In: FMCAD, volume 2517 of LNCS. Springer, pp 187–201
4. Balabanov V, Jiang J-HR, Scholl C, Mishchenko A, Brayton RK (2016) 2QBF: challenges and Solutions. In: SAT, volume 9710 of LNCS. Springer, pp 453–469
5. Balyo T, Biere A, Iser M, Sinz C (2016) SAT Race 2015. Artif Intell 241:45–65
6. Beyersdorff O, Chew L, Janota M (2014) On unification of QBF resolution-based calculi. In: MFCS, volume 8635 of LNCS. Springer, pp 81–93
7. Beyersdorff O, Janota M, Lonsing F, Seidl M (2021) Quantified Boolean formulas. In: Handbook of satisfiability, 2nd edn. Frontiers in artificial intelligence and applications. IOS Press
8. Biere A, Lonsing F, Seidl M (2011) Blocked clause elimination for QBF. In: CADE, volume 6803 of LNCS. Springer, pp 101–115
9. Bloem R, Braud-Santoni N, Hadzic V, Egly U, Lonsing F, Seidl M (2018)Expansion-based QBF solving without recursion. In: FMCAD. IEEE, pp 1–10
10. Bloem R, Könighofer R, Seidl M (2014) SAT-based synthesis methods for safety specs. In: VMCAI, volume 8318 of LNCS. Springer, pp 1–20
11. Bogaerts B, Janhunen T, Tasharrofi S (2016) SAT-to-SAT in QBFEval 2016. In: QBF workshop, volume 1719 of CEUR workshop proceedings. CEUR-WS.org, pp 63–70
12. Bogaerts B, Janhunen T, Tasharrofi S (2016) Solving QBF instances with nested SAT solvers. In: Beyond NP, volume WS-16-05 of AAAI Workshops. AAAI Press
13. Bubeck U, Kleine Büning H (2007) Bounded universal expansion for preprocessing QBF. In: SAT, volume 4501 of LNCS. Springer, pp 244–257
14. Charwat G, Woltran S (2016) Dynamic programming-based QBF solving. In: QBF workshop, volume 1719 of CEUR workshop proceedings. CEUR-WS.org, pp 27–40
15. Cheng C-H, Hamza Y, Ruess H (2016) Structural synthesis for GXW specifications. In: CAV, volume 9779 of LNCS. Springer, pp 95–117
16. Cheng C-H, Lee EA, Ruess H (2017) autoCode4: structural controller synthesis. In TACAS, volume 10205 of LNCS. Springer, pp 398–404
17. Clarke EM, Grumberg O, Jha S, Yuan L, Veith H (2003) Counterexample-guided abstraction refinement for symbolic model checking. J ACM 50(5):752–794
18. Dershowitz N, Hanna Z, Katz J (2005) Bounded model checking with QBF. In: SAT, volume 3569 of LNCS. Springer, pp 408–414
19. Egly U, Kronegger M, Lonsing F, Pfandler A (2017) Conformant planning as a case study of incremental QBF solving. Ann Math Artif Intell 80(1):21–45

20. Faymonville P, Finkbeiner B, Rabe MN, Tentrup L (2017) Encodings of bounded synthesis. In: TACAS, volume 10205 of LNCS. Springer, pp 354–370
21. Bernd F, Leander T (2015) Detecting unrealizability of distributed fault-tolerant systems. Logic Methods Comput Sci 11(3):1–31
22. Gascón A, Tiwari A (2014) A synthesized algorithm for interactive consistency. In: NASA formal methods, volume 8430 of LNCS. Springer, pp 270–284
23. Giunchiglia E, Marin P, Narizzano M (2009) Reasoning with quantified Boolean formulas. In: Handbook of satisfiability, volume 185 of FAIA. IOS Press, pp 761–780
24. Giunchiglia E, Marin P, Narizzano M (2010) sQueezeBF: an effective preprocessor for QBFs based on equivalence reasoning. In: SAT, volume 6175 of LNCS. Springer, pp 85–98
25. Heule M, Järvisalo M, Lonsing F, Seidl M, Biere A (2015) Clause elimination for SAT and QSAT. JAIR 53:127–168
26. Heyman T, Smith D, Mahajan Y, Leong L, Abu-Haimed H (2014) Dominant controllability check using QBF-solver and netlist optimizer. In: SAT, volume 8561 of LNCS. Springer, pp 227–242
27. Janota M (2018) Towards generalization in QBF solving via machine learning. AAAI Press, In AAAI
28. Janota M, Klieber W, Marques-Silva J, Clarke E (2016) Solving QBF with counterexample guided refinement. Artif Intell 234:1–25
29. Janota M, Klieber W, Marques-Silva J, Clarke EM (2012) Solving QBF with counterexample guided refinement. In: SAT, volume 7317 of LNCS. Springer, pp 114–128
30. Janota M, Marques-Silva J (2013) On propositional QBF expansions and Q-resolution. In: SAT, volume 7962 of LNCS. Springer, pp 67–82
31. Janota M, Marques-Silva J (2015) Expansion-based QBF solving versus Q-resolution. Theor Comput Sci 577:25–42
32. Janota M, Marques-Silva J (2015) Solving QBF by clause selection. In: IJCAI. AAAI Press, pp 325–331
33. Janota M, Silva JPM (2011) Abstraction-based algorithm for 2QBF. In: SAT, volume 6695 of LNCS. Springer, pp 230–244
34. Büning HK, Bubeck U (2009) Theory of quantified Boolean formulas. In: Handbook of satisfiability, volume 185 of FAIA. IOS Press, pp 735–760
35. Büning HK, Karpinski M, Flögel A (1995) Resolution for quantified Boolean formulas. Inf Comput 117(1):12–18
36. Klieber W, Sapra S, Gao S, Clarke EM (2010) A non-prenex, non-clausal QBF Solver with game-state learning. In: SAT, volume 6175 of LNCS. Springer, pp 128–142
37. Letz R (2002) Lemma and model caching in decision procedures for quantified Boolean formulas. In: TABLEAUX, volume 2381 of LNCS. Springer, pp 160–175
38. Lonsing F, Egly U (2017) DepQBF 6.0: a search-based QBF solver beyond traditional QCDCL. In: CADE, volume 10395 of LNCS. Springer, pp 371–384
39. Lonsing F, Egly U (2018) Evaluating QBF solvers: quantifier alternations matter. In: CP, volume 11008 of LNCS. Springer, pp 276–294
40. Lonsing F, Egly U (2019) Qratpre+: effective QBF preprocessing via strong redundancy properties. In: SAT, volume 11628 of LNCS. Springer, pp 203–210
41. Peitl T, Slivovsky F, Szeider S (2017) Dependency Learning for QBF. In: SAT, volume 10491 of LNCS. Springer, pp 298–313
42. Rabe MN, Tentrup L (2015) CAQE: a certifying QBF solver. In: FMCAD. IEEE, pp 136–143
43. Ranjan DP, Tang D, Malik S (2004) A comparative study of 2QBF algorithms. In: SAT
44. Rintanen J (2007) Asymptotically optimal encodings of conformant planning in QBF. In: AAAI. AAAI Press, pp 1045–1050
45. Shukla A, Biere A, Pulina L, Seidl M (2019) A survey on applications of quantified Boolean formulas. In: ICTAI. IEEE, pp 78–84
46. Tentrup L (2016) Non-prenex QBF solving using abstraction. In: SAT, volume 9710 of LNCS. Springer, pp 393–401
47. Tentrup L (2017) On expansion and resolution in CEGAR based QBF solving. In: CAV, volume 10427 of LNCS. Springer, pp 475–494
48. Tu K-H, Hsu T-C, Jiang J-HR (2015) QELL: QBF reasoning with extended clause learning and levelized SAT solving. In: SAT, volume 9340 of LNCS. Springer, pp 343–359
49. Vizel Y, Weissenbacher G, Malik S (2015) Boolean satisfiability solvers and their applications in model checking. Proc IEEE 103(11):2021–2035
50. Wimmer R, Reimer S, Marin P, Becker B (2017) Hqspre—an effective preprocessor for QBF and DQBF. In: TACAS, volume 10205 of LNCS, pp 373–390
51. Zhang L, Malik S (2002) Towards a symmetric treatment of satisfaction and conflicts in quantified Boolean formula evaluation. In: CP, volume 2470 of LNCS. Springer, pp 200–215

52. Zhang W (2014) QBF encoding of temporal properties and QBF-based verification. In: IJCAR, volume 8562 of LNCS. Springer, pp 224–239