Expansion-Based QBF Solving Without Recursion

Roderick Bloem, Nicolas Braud-Santoni, Vedad Hadzic, TU Graz
Uwe Egly, Florian Lonsing, TU Wien
Martina Seidl, JKU Linz

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) and pass the obtained formula to a SAT solver for deciding the QBF. State-of-the-art expansion-based solvers process the given formula quantifier-block wise and recursively apply expansion until a solution is found.

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

I. INTRODUCTION

Efficient tools for deciding the satisfiability of Boolean formulas (SAT solvers) are the core technology in many verification and synthesis approaches [44]. 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). QBFs extend propositional formulas by universal and existential quantifiers over Boolean variables [31] resulting in a decision problem that is PSPACE-complete. Applications from verification and synthesis [7], [12], [13], [17], [19], [23], realizability checking [18], bounded model checking [15], [47], and planning [16], [40] motivate the quest for efficient QBF solvers.

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 [20], [34], [46]. 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 generalising this SAT-based approach to QBFs with an arbitrary number of quantifier alternations, expansion is recursively applied per quantifier block, requiring multiple SAT solvers. As noted by Rabe and Tentrup [38], these CEGAR-based approaches show poor performance for formulas with many quantifier alternations in general.

In this paper, we present a novel solving algorithm based on non-recursive expansion for QBFs with arbitrary quantifier prefixes using only two SAT solvers. Our approach of non-recursive expansion is theoretically (i.e., from a proof complexity perspective) equivalent to approaches that apply recursive expansion since both non-recursive and recursive expansion rely on the \textit{Exp+Res} proof system [5]. 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, we 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 work has been supported by the Austrian Science Fund (FWF) under projects W1255-N23, S11406-N23, S11408-N23, and S11409-N23. The paper will appear in the proceedings of FMCAD 2018, IEEE.
solvers [30] 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 $\phi$ 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 realises a natural application of the counter-example guided abstraction refinement (CEGAR) paradigm [14]. A detailed survey on 2QBF solving is given in [3].

A significant advancement of expansion-based solving for QBF with an arbitrary number of quantifier alternations was made with the solver RAReQS [25], [26], which recursively applies the previously discussed 2QBF approach [30] for each quantifier alternation. The approach turned out to be highly competitive [11]. For formalising this solving approach the calculus $\forall \exists$Exp+Res was introduced [5], and proof-theoretical investigations revealed the orthogonal strengths of $\forall \exists$Exp+Res and Q-resolution [32], the QBF variant of the resolution calculus that forms the basis for QCDCL-based solvers. Research and Q-resolution [32], the QBF variant of the resolution calculus [30] for which every full assignment $\sigma \in \Sigma_X$, $\phi(\sigma) = \phi(\psi)$ then $\phi$ and $\psi$ are equivalent. Let $\tau: X \rightarrow \{\top, \bot\}$ and $\sigma: X \rightarrow \{\top, \bot\}$ be assignments such that for every $x \in X \land Y$, $\tau(x) = \sigma(x)$ if $\tau(x) \neq \sigma(x)$. Then the composite assignment of $\sigma$ and $\tau$ is denoted by $\sigma \tau: X \cup Y \rightarrow \{\top, \bot\}$. Let $\sigma \tau(\phi) = \sigma(\tau(\phi)) = \tau(\sigma(\phi))$.

Furthermore, $\sigma \sigma = \sigma$ for any assignment $\sigma$.

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

**IV. 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 axiomatic rule of the calculus $\forall \exists$Exp+Res [29].

**Definition 1.** Let $\Pi \phi$ be a QBF with prefix $\Pi = Q_1x_1 \ldots Q_nx_n$ over the set of variables $X = \{x_1, \ldots, x_n\}$ and $\sigma: Y \rightarrow \{\top, \bot\}$ with $Y \subseteq X$ an assignment. If $Y \subseteq X$, we extend the domain of $\sigma$ to $X$ by setting $\sigma(x) = \bot$ if $x \notin Y$.

The instantiation of $\phi$ by $\sigma$, denoted by $\phi^\sigma$, is obtained from $\phi$ as follows:

1) all variables $x \in X$ with $\sigma(x) \neq \bot$ are set to $\sigma(x)$; 2) all variables $x \in X$ with $\sigma(x) = \bot$ are replaced by $x^\omega$ where annotation $\omega$ 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 $x_{k_i} < \bot x$. Furthermore, $x_{k_i} < \bot x_{k_j}$ if $k_i < k_j$.

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

**Example 2.** Given the QBF $\forall a \exists x \forall b \forall y. \phi$ with $\phi = ((x \lor a \lor y) \land \neg (x \lor a \lor y) \land (\neg y \lor y \lor b))$ Then $U = \{a, b\}$ and $E = \{x, y\}$. Let $\sigma: U \rightarrow \{\top, \bot\}$ be defined by $\sigma(a) = \top$.
and \( \sigma(b) = \bot \). Then \( \phi^\sigma = (\neg x^T \lor y^T) \land \neg y^T \). Further, let \( \tau : E \to \{\top, \bot, \epsilon\} \) 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 again from an assignment or an instantiated formula. Therefore, we introduce the following notation. Let \( \phi^\sigma \) be an instantiation by assignment \( \sigma : X \to \{\top, \bot, \epsilon\} \) and \( X^\sigma \) the set of annotated variables. If we have an assignment \( \tau : X^\sigma \to \{\top, \bot, \epsilon\} \), then we define \( \tau^\sigma : X \to \{\top, \bot\} \) by \( \tau^\sigma(x) = \tau(x^\sigma) \) for \( x^\sigma \in X^\sigma \). If we have an instantiated formula \( \phi^\sigma \), the \( (\phi^\sigma)^{-\sigma} \) is the formula obtained by replacing every annotated variable \( x^\sigma \in X^\sigma \) by \( x \). In general, \( (\phi^\sigma)^{-\sigma} \neq \phi \).

**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 \( \sigma(x) = \epsilon \), \( \sigma(\phi) = x \) and \( \phi^\sigma = x \). 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.

**Example 3.** Reconsider the propositional formula \( \phi \) and assignments \( \sigma, \tau \) from above (Example 2). Then \( (\phi^\sigma)^{-\sigma} = (\neg x^T \lor y^T) \land \neg y^T = \neg x \lor y \). Furthermore, \( (\phi^\tau)^{-\tau} = (a)^{-\tau} = a \).

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 \( \Phi \) 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 \( \Phi \) is true.

**V. A NON-RECURSIVE ALGORITHM FOR EXPANSION-BASED QBF SOLVING**

The pseudo-code in Figure 1 summarises 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 \( \alpha_0 \) for the universal variables is selected in Line 1. The instantiation \( \phi^{\alpha_0} \) is handed over to a SAT solver. If \( \phi^{\alpha_0} \) is unsatisfiable, then \( \Pi.\phi \) is false and the algorithm returns. Otherwise, \( \tau : E^{\alpha_0} \to \{\top, \bot\} \) is a satisfying assignment of \( \phi^{\alpha_0} \). Let \( \sigma_1 \) denote the assignment \( \tau^{-\alpha_0} \). Then \( \alpha_0 \sigma_1 \) is a satisfying assignment of \( \phi \).

Next, the propositional formula \( \neg \phi^{\sigma_1} \) is handed over to a SAT solver for checking the validity of \( \phi^{\sigma_1} \). If \( \neg \phi^{\sigma_1} \) is unsatisfiable, then \( \Pi.\phi \) is true and the algorithm returns. If \( \neg \phi^{\sigma_1} \) is satisfiable, then \( \rho : U^{\sigma_1} \to \{\top, \bot\} \) is a satisfying assignment of \( \neg \phi^{\sigma_1} \). Let \( \alpha_1 \) denote the assignment \( \rho^{-\sigma_1} \). Then \( \alpha_1 \sigma_1 \) is a satisfying assignment of \( \phi \).

**V. A NON-RECURSIVE ALGORITHM FOR EXPANSION-BASED QBF SOLVING**

The pseudo-code in Figure 1 summarises 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 \( \alpha_0 \) for the universal variables is selected in Line 1. The instantiation \( \phi^{\alpha_0} \) is handed over to a SAT solver. If \( \phi^{\alpha_0} \) is unsatisfiable, then \( \Pi.\phi \) is false and the algorithm returns. Otherwise, \( \tau : E^{\alpha_0} \to \{\top, \bot\} \) is a satisfying assignment of \( \phi^{\alpha_0} \). Let \( \sigma_1 \) denote the assignment \( \tau^{-\alpha_0} \). Then \( \alpha_0 \sigma_1 \) is a satisfying assignment of \( \phi \).

Next, the propositional formula \( \neg \phi^{\sigma_1} \) is handed over to a SAT solver for checking the validity of \( \phi^{\sigma_1} \). If \( \neg \phi^{\sigma_1} \) is unsatisfiable, then \( \Pi.\phi \) is true and the algorithm returns. If \( \neg \phi^{\sigma_1} \) is satisfiable, then \( \rho : U^{\sigma_1} \to \{\top, \bot\} \) is a satisfying assignment of \( \neg \phi^{\sigma_1} \). Let \( \alpha_1 \) denote the assignment \( \rho^{-\sigma_1} \). Then \( \alpha_1 \sigma_1 \) is a satisfying assignment of \( \phi \).

**Input:** QBF \( \Pi.\phi \) with universal variables \( U \) and existential variables \( E \)

**Output:** truth value of \( \Pi.\phi \)

1. Let \( \alpha_0 \) be an arbitrary assignment of \( U \). The set of annotated existential variables \( E \) is handed over to a SAT solver. If \( \neg \phi^{\alpha_0} \) is selected in Line 1. The instantiation \( \phi^{\alpha_0} \) is true.

2. If \( \neg \phi^{\alpha_0} \) is selected in Line 1. The instantiation \( \phi^{\alpha_0} \) is true.

3. If \( \neg \phi^{\alpha_0} \) is selected in Line 1. The instantiation \( \phi^{\alpha_0} \) is true.

4. **While** \( \neg \phi^{\alpha_0} \) is selected in Line 1. The instantiation \( \phi^{\alpha_0} \) is true.

5. If \( \neg \phi^{\alpha_0} \) is selected in Line 1. The instantiation \( \phi^{\alpha_0} \) is true.

6. **If** \( \neg \phi^{\alpha_0} \) is selected in Line 1. The instantiation \( \phi^{\alpha_0} \) is true.

7. **If** \( \neg \phi^{\alpha_0} \) is selected in Line 1. The instantiation \( \phi^{\alpha_0} \) is true.

8. **If** \( \neg \phi^{\alpha_0} \) is selected in Line 1. The instantiation \( \phi^{\alpha_0} \) is true.

9. **If** \( \neg \phi^{\alpha_0} \) is selected in Line 1. The instantiation \( \phi^{\alpha_0} \) is true.

10. **If** \( \neg \phi^{\alpha_0} \) is selected in Line 1. The instantiation \( \phi^{\alpha_0} \) is true.

11. **If** \( \neg \phi^{\alpha_0} \) is selected in Line 1. The instantiation \( \phi^{\alpha_0} \) is true.

12. **End**

Figure 1: Non-Recursive Expansion-Based Algorithm
in which assignments are considered depends on the used SAT solver.

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

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

**Iteration 1:** The formula \( \phi^{\sigma_0} = (\neg x \lor y) \land \neg y \lor (x \lor y) \) is passed to a SAT solver and found satisfiable under the assignment \( \tau : E^{\sigma_0} \to \{ \top, \bot \} \) with \( \tau(x) = \bot \) and \( \tau(y) = \bot \).

By removing the variable annotations we get assignment \( \sigma_1 = (\tau|_{E^{\sigma_0}})^{-\alpha_0} \) where \( \sigma_1 : E \to \{ \top, \bot \} \) 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 \( \rho : U^{\sigma_1} \to \{ \top, \bot \} \) with \( \rho(a) = \bot \) and \( \rho(b) = \top \), which we then reduce to \( \alpha_1 = (\rho|_{U^{\sigma_1}})^{-\sigma_1} \) where \( \alpha_1 : U \to \{ \top, \bot \} \) with \( \alpha_1(a) = \bot \) and \( \alpha_1(b) = \top \).

**Iteration 2:** The formula \( \phi^{\sigma_1 \land \phi^{\alpha_1}} = (\neg x \lor y) \land \neg y \lor x \) is passed to a SAT solver in the second iteration. It is satisfiable and one satisfying assignment is \( \tau : E^{\sigma_1} \cup E^{\alpha_1} \to \{ \top, \bot \} \) with \( \tau(x) = \bot \), \( \tau(x) = \top \), \( \tau(y) = \bot \), \( \tau(y) = \top \). From \( \tau \), we can extract the assignment \( \sigma_2 = (\tau|_{E^{\sigma_1}})^{-\alpha_1} \) where \( \sigma_2 : E \to \{ \top, \bot \} \) with \( \sigma_2(x) = \top \) and \( \sigma_2(y) = \bot \). Note that for any choice of \( \tau \), \( \sigma_2 \neq \sigma_1 \). Next, we construct \( \phi_1 \lor \phi_2 = a \lor \neg a \). This formula is a tautology, so its negation that is passed to a SAT solver is unsatisfiable, hence \( \Pi. \phi \) is true.

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 \( \bigwedge_{a \in A_{i-1}} \phi^a \) (respectively \( \bigwedge_{\sigma \in S_i} \neg \phi^\sigma \)) is unsatisfiable.

**Theorem 1.** The algorithm shown in Figure 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 \( \Phi := \forall a \exists x \forall y \exists z. \phi \) with

\[
\phi := (a \land b \land \neg x \land \neg y) \lor (\neg a \land x \land (b \leftrightarrow y))
\]

and how it iteratively generates the sets \( \Sigma_U \) and \( \Sigma_E \). Figure 1 shows the expansion trees that are implicitly built during the search. An expansion tree relates the variables of the partial expansion of \( \Phi \) 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 initialisation of \( A_0 \), an arbitrary assignment \( \alpha_0 : U \to \{ \top, \bot \} \) is chosen. Let \( \alpha_0(a) = \bot \) and \( \alpha_0(b) = \bot \).

**Iteration 1:** \( \phi^{\sigma_0} := x^+ \land \neg y^+ \) is satisfiable. Assignment \( \sigma_1 : E \to \{ \top, \bot \} \), with \( \sigma_1(x) = \top \) and \( \sigma_1(y) = \bot \), is extracted from model \( \tau : E^{\sigma_1} \to \{ \top, \bot \} \) and added to \( S_1 \). Now \( \phi^{\sigma_1} := \neg a \land \neg b^+ \) is checked for validity. Assignment \( \alpha_1 : U \to \{ \top, \bot \} \), with \( \alpha_1(a) = \bot \) and \( \alpha_1(b) = \top \), obtained from counter-example \( \rho : U^{\sigma_1} \to \{ \top, \bot \} \) is added to \( A_1 \).

**Iteration 2:** Next, \( \phi^{\sigma_1} \land \phi^{\alpha_1} \land \phi^{\alpha_2} \) with \( \phi^{\alpha_2} := x^+ \land y^+ \) is checked. From model \( \tau : E^{\sigma_1} \cup E^{\alpha_1} \to \{ \top, \bot \} \), again \( \sigma_1 \) can be extracted for \( \phi^{\alpha_1} \). For \( \phi^{\alpha_2} \) a new assignment \( \sigma_2 \) which is not in \( S_1 \) is found and added to \( S_1 \). In particular, we get \( \sigma_2 : E \to \{ \top, \bot \} \) with \( \sigma_2(x) = \top \) and \( \sigma_2(y) = \top \). When the validity of \( \phi^{\alpha_1} \lor \phi^{\alpha_2} \lor \phi^{\sigma_1} := a \lor b^+ \) is checked, we get a counter-example \( \rho : U^{\sigma_1} \cup U^{\alpha_1} \cup U^{\alpha_2} \to \{ \top, \bot \} \), from which \( \alpha_2 : U \to \{ \top, \bot \} \), with \( \alpha_2(a) = \top \) and \( \alpha_2(b) = \bot \), can be extracted. Assignment \( \alpha_2 \) is added to \( A_2 \) leading to a new path in the left expansion tree (Iteration 3 in Figure 1).

![Figure 1: Expansion trees relating the assignments found during solving the QBF \( \forall a \exists x \forall y \exists z. \phi \) 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) \( \phi \).](image-url)
Iteration 3: Next, $\phi'^0 \land \phi'^1 \land \phi'^2$ with $\phi'^2 := \neg x^T \land \neg y^T \land \neg z^T$ is checked. From model $\tau: E^0 \cup E^1 \cup E^2 \rightarrow \{\top, \bot\}$, $\sigma_3: E \rightarrow \{\top, \bot\}$ is extracted, satisfying $\phi'^2$. This assignment is different from both $\sigma_1$ and $\sigma_2$: $\sigma_3(x) = \bot$ and $\sigma_3(y) = \bot$. This again results in a new branch of the execution tree (see left expansion tree of Iteration 4 in Figure 2). The resulting formula $\phi'^2 \lor \phi'^3 \lor \phi'^3$ with $\phi'^3 := a \land b$ is not valid, and from the counter-example $\rho: U^3 \cup U^2 \cup U^3 \rightarrow \{\top, \bot\}$ we get $\sigma_3: U \rightarrow \{\top, \bot\}$ with $\sigma_3(a) = \top$ and $\sigma_3(b) = \bot$.

Iteration 4: Finally, the full expansion $\phi'^0 \land \phi'^1 \land \phi'^2 \land \phi'^3$ with $\phi'^3 := \bot$ is not satisfiable, meaning that the original formula $\forall a \exists x \forall y \exists \phi$. 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 $\phi$. The following definition formalises 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 $V$. Further, let $A \subseteq \{\alpha \mid \alpha: U \rightarrow \{\top, \bot\}\}$ and $S \subseteq \{\sigma \mid \sigma: E \rightarrow \{\top, \bot\}\}$. 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 $A$ complete $S$. If for every assignment $\alpha \in A$, there exists an assignment $\sigma \in S$ such that $\sigma \alpha (\phi)$ is true, then we say that $S$ complete $A$.

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 $V$. 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 existentially variables obtained by iteration $i$ during an execution of the algorithm shown in Figure 7.

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 $\tau$ with $\tau |_{E^0} (\phi^\alpha)$ is true. Then also $\mu (\tau |_{E^0}) (\phi)$ is true. But $(\tau |_{E^0}) (\phi)$ is 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 $\nu \mu (\neg \phi)$ 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 $S$ of universal assignments forces a set $T$ 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 $V$. 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 $\rho$, $A' \subseteq \{ (\rho |_{U'}) \sigma \mid \sigma \in S \}$.

If $S$ completes $A$, and $A \cup A'$ completes $S$, and $\bigwedge_{\alpha \in A \cup A'} \phi^\alpha$ evaluates to true under assignment $\tau$, then there exists an assignment $\nu \in \{ (\tau |_{E^0}) \alpha \mid \alpha \in A \cup A' \}$ with $\nu \notin S$.

Proof. By induction over the number of variables in $\Pi$.

Base Case. Assume that $\Phi$ 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 $\rho$. If $Q = \exists$, then the elements of $A$ are full assignments of $\phi$, and $S$ is either empty, or it contains the empty assignment $\omega: \emptyset \rightarrow \{\top, \bot\}$. Let $A' = \{ \mu \}$. If $S$ is empty, so is $A$ (because $S$ has to complete $A$). If $\tau$ is a satisfying assignment of $\phi^\alpha$, 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 $\phi^\alpha \land \phi^\beta$ is a full expansion of $\Phi$. If this full expansion is true, then $\neg \phi$ is unsatisfiable. Otherwise, $\phi^\alpha \land \phi^\beta$ 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 $\phi^\beta$ is unsatisfiable, again violating a precondition. If $Q = \forall$, then $\mu = \omega$ and $A = \emptyset$. If $S = \emptyset$ and $\phi^\alpha = \phi$ has the satisfying assignment $\tau$, then $\nu = \tau$ and $\nu \notin S$. Otherwise, there is an assignment $\sigma \in S$, then $\omega \sigma (\neg \phi)$ is true, because $A \cup \{ \mu \} = \{ \omega \}$ completes $S$. Hence, if assignment $\tau$ satisfies $\phi^\beta$, then $\nu = \tau$, so $\nu \notin S$.

Induction Step. Assume that 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 existentially variables $E$ and universal variables $U$ with $Q = Q_1(x_1) \ldots Q_n(x_n)$ 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 $\tau$, and $\bigwedge_{\sigma \in S} \neg \phi^\sigma$ has a satisfying assignment $\rho$ 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 $\rho$ and since $x$ is the outermost variable of the prefix of $\Phi$, $\rho(x) = t$. 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 $A' \cup A$ and $S$ the induction hypothesis applies, i.e., there is an assignment $\nu \notin S$ with $\nu \in \{ (\tau |_{E^0}) | \alpha \in A' \cup A \}$ where $\tau$ is the part of $\tau$ that satisfies $\bigwedge_{\alpha \in A' \cup A} (\phi[x \leftarrow t])^\alpha$. Obviously, $\nu \in \{ (\tau |_{E^0}) | \alpha \in A' \cup A \}$.

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

Lemma 7. Let $\Phi = \Pi.\phi$ be a QBF over universally quantified variables $U$ and existentially quantified variables $V$. 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 $\tau$, $S' \subseteq \{ (\tau |_{E^0}) | \alpha \in A \}$.

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 $\rho$, then there exists an
assignment \( \nu \in \{(\rho|_{U^v})^{-}\sigma \mid \sigma \in S \cup S'\} \) with \( \nu \notin A \).

**Proof.** The proof is analogous to the proof of Lemma 6. \( \square \)

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

**Theorem 2.** The algorithm shown in Figure 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 \( \phi^{\alpha_0} \) (if \( \phi^{\alpha_0} \) 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 \( \bigwedge_{\alpha \in S_i} \neg \phi^{\sigma} \) is satisfiable under some assignment \( \rho \) (otherwise the algorithm would terminate), by construction \( A' \subseteq \{(\rho|_{U^v})^{-}\sigma \mid \sigma \in S_i\} \). Hence, Lemma 6 applies and if \( \bigwedge_{\alpha \in A_i} \phi^{\sigma} \) is satisfiable under some assignment \( \tau \) (otherwise the algorithm would immediately terminate), then there is an assignment \( \nu \in \{(\tau|_{E^v})^{-}\sigma \mid \alpha \in A_i\} \) with \( \nu \notin S_i \).

The argument for \( A_i \subset A_i+1 \) is similar and uses the property shown in Lemma 7. \( \square \)

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 \( \phi \) is in conjunctive normal form. Without any modification, our algorithm also works on formulas in PCNF—where SAT solvers typically process formulas in CNF only, we focus on this representation for the rest of the paper.

We conclude this section by arguing that the \( \forall \text{Exp}+\text{Res} \) calculus yields the theoretical foundation of our algorithm for refuting a formula \( \Pi.\phi \) in PCNF with universal variables \( U \). The \( \forall \text{Exp}+\text{Res} \) calculus consists of two rules, the axiom rule

\[
\vdash_{\forall \text{Exp}+\text{Res}}
\]

where \( C \) is a clause of \( \phi \) and \( \alpha: U \to \{\top, \bot\} \) is a universal assignment as well as the resolution rule

\[
\frac{C_1 \lor \alpha^\omega \quad C_2 \lor \neg \alpha^\omega}{C_1 \lor \neg C_2}
\]

A derivation in \( \forall \text{Exp}+\text{Res} \) is a sequence of clauses where each clause is either obtained by the axiom or derived from previous 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 instantiates the universal variables of one clause of \( \phi \). 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 in Figure 1 does not instantiate individual clauses, but all clauses of \( \phi \) at once with 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 the 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.

**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 Figure 1. Further, let \( \psi'_\forall \) be an unsatisfiable core of \( \psi_\forall \). Then there is a \( \forall \text{Exp}+\text{Res} \) refutation such that all clauses that are introduced by the axiom rule occur in \( \psi'_\forall \).

VI. IMPLEMENTATION

The algorithm described in Section V is realised in the solver \texttt{Ijtihad}\footnote{The name \texttt{Ijtihad} refers to the effort of solving cases in Islamic law (for details see \url{https://en.wikipedia.org/wiki/Ijtihad}).}. The most recent version of \texttt{Ijtihad} is available at \url{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, \texttt{Ijtihad} uses the IPASIR interface [4], which makes changing the SAT solver very easy. The SAT solver used in all of our experiments is Glucose [1]. Although the base implementation does reasonably well, we have realised various optimizations to make \texttt{Ijtihad} even more viable in practice. Some of them are discussed in the following.

For solving a QBF \( \Pi.\phi \), the basic algorithm shown in Figure 1 adds instantiations of \( \phi \) to \( \psi_\forall = \bigwedge_{\alpha \in A_i} \phi^{\alpha} \) and \( \psi_3 = \bigwedge_{\sigma \in S_i} \neg \phi^{\sigma} \) 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 \( \phi \) 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 \( \psi_3 \) for the formulas of the PCNF track of QBFEVAL’17 (preprocessed with Bloqer [6]). We also distinguish between true and false formulas. In Figure 5 we see that for true formulas, set \( S \) tends to be larger than \( A \), while for false instances the picture is less clear. Figure 4 shows the overall time needed for solving \( \psi_\forall \) (y-axis) and \( \psi_3 \) (x-axis). In almost all cases, the solver that handles \( \psi_\forall \) needs more time than the solver that handles \( \psi_3 \). This may be founded on the observation that many QBFs have considerably more existential variables than universal variables [50], hence the instantiations added to \( \psi_\forall \) are much larger than the instantiations added to \( \psi_3 \).

In Line 1 of Figure 1 the set of universal assignments \( A \) is initialised with one arbitrary assignment \( \alpha_0 \). Obviously, the set \( A \) may also be initialized with multiple assignments. In 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 reset sets, 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 Figure 4. 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 [20] for exploiting the different strengths of $\forall$Exp+$\forall$Res and $\forall$-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 Figure 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 ljtihad by additional invocations of the QCDCL solver DepQBF [35]. About every 30 seconds, DepQBF is called and runs for about 30 seconds. The learned clauses are obtained via the API of DepQBF. Leveraging learned cubes is subject to future work.

VII. Evaluation

We evaluate non-recursive expansion as implemented in our solvers ljtihad and its hybrid variant Heretic on the benchmarks from the PCNF track of the QBFEVAL’17 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’17: Qute [37], Rev-Qfun [24], RAREQS [25], CAQE [38], [42], DynQBF [11], GhostQ [25], [33], DepQBF [35], QESTO [29], and QSTS [8], [9]. Our experiments are based on original benchmarks without preprocessing and benchmarks preprocessed using Bloqqer [6], [22] with a timeout of two hours. We included the 76 formulas already solved by Bloqqer in both benchmark sets.

Tables I and III show the total numbers of solved instances ($S$), solved unsatisfiable (⊥) and satisfiable ones (⊤), and total CPU time including timeouts. In the following, we focus on a comparison of our solvers ljtihad and Heretic with RAREQS (cf. Figure 6). Unlike our solvers, RAREQS is based on a recursive implementation of expansion.

In general, preprocessing has a considerable impact on the number of solved instances. The difference in solved instances between ljtihad and RAREQS is 17 on original instances (Table I), and becomes larger on preprocessed instances (Table III). Notably Heretic, despite its simple design, significantly outperforms ljtihad on the two benchmark sets. Moreover, Heretic is ranked third on preprocessed instances (Table III) and thus is on par with state-of-the-art solvers. On the two benchmark sets, the gap in solved instances between RAREQS and Heretic is considerably smaller than the one between RAREQS and ljtihad.

We report on memory consumption of expansion-based solvers. While RAREQS, ljtihad, and Heretic run out of...
memory on 42, 61, and 39 original instances (Table I), respectively, these numbers drop to 17, 41, and 24, respectively, with preprocessing (Table II). The average memory footprint is 1718 MB, 1836 MB, and 1842 MB for RAReQS, Ijtihad, and Heretic, respectively, and 1056 MB, 1311 MB, and 1187 MB on preprocessed instances. Interestingly, Ijtihad has a smaller median memory footprint than RAReQS without (792 MB vs. 802 MB) and with preprocessing (286 MB vs. 364 MB).

The strength of Heretic becomes obvious for formulas that have four or more quantifier blocks (i.e., three or more quantifier alternations), cf. [36]. As shown in Table III, Heretic outperforms all other solvers on these instances. We made a similar observation on preprocessed formulas.

Moreover, Heretic solves only four original instances less than DepQBF (Table I), and outperforms DepQBF on preprocessed instances (Table II). These results indicate the potential of combining the orthogonal proof systems \texttt{\texttt{Exp+Res}} as implemented in Ijtihad and Q-resolution as implemented in DepQBF in a hybrid solver such as Heretic.

Although RAReQS outperforms both Ijtihad and Heretic on the two given benchmark sets (Tables I and II), RAReQS failed to solve certain instances that were solved by Ijtihad and Heretic. Table IV shows related statistics. E.g., on preprocessed instances (row “B”), 218 instances were solved by both RAReQS and Heretic (column “R vs. H”), 38 only by RAReQS, and 27 only by Heretic. Summing up these numbers yields a total of 283 solved instances (more than any individual solver on preprocessed instances in Table II) 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, it clearly outperformed Ijtihad on all benchmarks (column “I vs. H”) and DepQBF on preprocessed ones (“D vs. H”).

### VIII. 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 \texttt{\texttt{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 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.
### APPENDIX A

**ADDITIONAL EXPERIMENTAL DATA**

**Preprocessing with HQSpre**

| Solver   | $S$ | $\perp$ | $\perp$ | Time |
|----------|-----|---------|---------|------|
| CAQE     | 306 | 197     | 109     | 415K |
| RAReQS   | 294 | 194     | 100     | 429K |
| QESTO    | 287 | 194     | 93      | 443K |
| Rev-Qfun | 281 | 190     | 91      | 453K |
| Heretic  | 279 | 188     | 91      | 460K |
| QSTS     | 264 | 179     | 85      | 484K |
| DepQBF   | 263 | 176     | 87      | 490K |
| Qute     | 255 | 171     | 84      | 497K |
| Ijtihad  | 250 | 176     | 74      | 500K |
| GhostQ   | 244 | 163     | 81      | 522K |
| DynQBF   | 233 | 156     | 77      | 528K |

**TABLE V:** Preprocessing by HQSpre.

![Figure 7](image.png)

**Preprocessing with Bloqqer**

| Solver   | $S$ | $\perp$ | $\perp$ | Time |
|----------|-----|---------|---------|------|
| CAQE     | 306 | 197     | 109     | 415K |
| RAReQS   | 294 | 194     | 100     | 429K |
| QESTO    | 287 | 194     | 93      | 443K |
| Rev-Qfun | 281 | 190     | 91      | 453K |
| Heretic  | 279 | 188     | 91      | 460K |
| QSTS     | 264 | 179     | 85      | 484K |
| DepQBF   | 263 | 176     | 87      | 490K |
| Qute     | 255 | 171     | 84      | 497K |
| Ijtihad  | 250 | 176     | 74      | 500K |
| GhostQ   | 244 | 163     | 81      | 522K |
| DynQBF   | 233 | 156     | 77      | 528K |

**TABLE VI:** Preprocessing by HQSpre: 97 instances with four or more quantifier blocks.

![Figure 8](image.png)

**TABLE VII:** Pairwise comparison of RAReQS ($R$), Ijtihad ($I$), and Heretic ($H$) by numbers of instances from QBFEVAL’17 with and without preprocessing that were solved by only one solver of the considered pair ($<$, $>$) or by both ($\simeq$).

| Solver   | $S$ | $\perp$ | $\perp$ | Time |
|----------|-----|---------|---------|------|
| No preprocessing | 27 140 | 10 | 26 141 | 22 | 5 145 | 18 |
| Bloqqer  | 56 200 | 17 | 38 218 | 27 | 7 210 | 35 |
| HQSpre   | 54 240 | 10 | 40 254 | 25 | 1 249 | 30 |

**TABLE VIII:** 154 preprocessed instances with four or more quantifier blocks.

![Figure 9](image.png)

**Additional plots**

![Figure 10](image.png)
Figure 11: Plot related to Table II.

Figure 12: Plot related to Table III.