Skolem Function Continuation for Quantified Boolean Formulas*

Katalin Fazekas[1], Marijn J. H. Heule[2], Martina Seidl[1], Armin Biere[1]

[1] Institute for Formal Models and Verification, JKU Linz, Austria
[2] The University of Texas at Austin, Austin, USA

Abstract. Modern solvers for quantified Boolean formulas (QBF) not only decide the satisfiability of a formula, but also return a set of Skolem functions representing a model for a true QBF. Unfortunately, in combination with a preprocessor this ability is lost for many preprocessing techniques. A preprocessor rewrites the input formula to an equi-satisfiable formula which is often easier to solve than the original formula. Then the Skolem functions returned by the solver represent a solution for the preprocessed formula, but not necessarily for the original encoding.
Our solution to this problem is to combine Skolem functions obtained from a QRAT trace as produced by the widely-used preprocessor Bloqqer with Skolem functions for the preprocessed formula. This approach is agnostic of the concrete rewritings performed by the preprocessor and allows the combination of Bloqqer with any Skolem function producing solver, hence realizing a smooth integration into the solving tool chain.

1 Introduction

Quantified Boolean formulas (QBFs) [1] extend propositional logic with existential and universal quantifiers over the Boolean variables. This extension allows a compact formalization of PSPACE-hard problems, thus QBFs can be remarkably beneficial in applications of formal verification [2], synthesis [3], and artificial intelligence [4], driving the demand for efficient and reliable QBF solving tools.

Modern QBF solvers not only decide the satisfiability of a formula, but also produce *certificates* [5–14]. Such a certificate is either syntactical or semantical. A syntactical certificate is basically a trace of the individual steps taken by a solver to derive either a conflict in the case of unsatisfiability or to derive the empty formula in the case of satisfiability. The correctness of the individual steps has to be checkable efficiently, i.e., in polynomial time by an external tool that independently confirms the correctness of the solving result. Examples of such syntactical certificates are Q-resolution proofs [15], the QBF variant of resolution proofs produced by solvers based on conflict/solution-driven clause/cube learning (QCDCL) [16], and QRAT proofs [12]. *Quantified Asymmetric Tautologies*

* This work has been supported by the Austrian Science Fund (FWF) under projects W1255-N23 and S11408-N23, and by the National Science Foundation (NSF) under grant number CCF-1618574.

(QRAT) is a redundancy criterion allowing for the safe addition/deletion/update of clauses with this property. A semantic certificate is a model (counter-model) of a satisfiable (unsatisfiable) QBF. Usually such certificates are represented as so-called Skolem (Herbrand) functions, encoding a strategy to set the existential (universal) variables to satisfy (falsify) the QBF. Skolem functions are of particular interest when solving application problems, because they contain information about the solution. For example, if a verification approach based on bounded model checking reports that an error state is reachable by the program to be verified, then the Skolem function encodes a program trace showing the erroneous behavior. Checking the correctness of a Skolem (Herbrand) function is a coNP-complete (NP-complete) problem, because the validity (satisfiability) of a propositional formula has to be checked. Skolem functions are either efficiently extracted from syntactic certificates like Q-resolution and QRAT proofs [12, 7], or they are produced directly by a solver [17].

QBF solving tool chains often involve an additional *preprocessing* phase in which the original QBF is rewritten to an equi-satisfiable formula that is then passed to the solver. In many cases, solvers are only able to solve the preprocessed formula, but not the original one [18, 19]. Then, however, the found set of Skolem functions is only a solution and witness for the preprocessed formula, because in general, the preprocessing techniques are not model-preserving. To enable both preprocessing and certification at the same time, Janota at al. [20] showed how to obtain Skolem functions for a subset of the preprocessing techniques implemented in the widely-used preprocessor Bloqqer [21] by establishing a solution reconstruction rule for each of the considered preprocessing technique.

In this paper, we present a general approach to obtain Skolem functions with *all* preprocessing techniques implemented in Bloqqer enabled. Therefore, we use the QRAT trace produced by Bloqqer together with the Skolem function produced by state-of-the-art QBF solvers like DepQBF [22] or Caqe [17]. The QRAT trace is not a full syntactical certificate if the formula is not solved by Bloqqer, but it only justifies the rewriting steps from the original QBF to the preprocessed formula. We show that it is nevertheless sufficient to build an incomplete Skolem function which—when continued with the Skolem function provided by the solver—yields a complete Skolem function for the original problem.

This paper is structured as follows. First we review the basic concepts of QBFs and Skolem functions in Section 2, then in Section 3 we present the formal foundation of our procedure and illustrate the approach on a simple example. In Section 4 we describe the main steps of our general tool chain to construct valid Skolem functions for true QBFs in presence of preprocessing. Finally, the paper ends with the experimental evaluation in Section 5 and concludes in Section 6.

2 Preliminaries

In this section, we introduce concepts and terminology used in the rest of the paper. A *literal* is a variable (x) or the negation of a variable ($\bar{x}$). The negation of a literal l is denoted by $\bar{l}$ and $\mathsf{var}(l) := x$ if $l = x$ or $l = \bar{x}$. A *clause* is a disjunction of literals and (possibly negated) truth constants $\top$ (verum) and $\bot$ (falsum). A propositional formula in *conjunctive normal form* (CNF) is a conjunction of clauses. A QBF in *prenex conjunctive normal form* (PCNF) has the form $\Pi.\psi$ where ψ is a propositional CNF formula defined over the variables of prefix $\Pi := Q_1X_1 \dots Q_nX_n$ with $Q_i \in \{\forall, \exists\}$, $Q_i \neq Q_{i+1}$, and $X_i \cap X_j = \emptyset$. The set of all variables occurring in prefix Π is denoted by $vars(\Pi)$. The quantifier $\mathsf{quant}(\Pi, l)$ of literal l is Q_i if $\mathsf{var}(l) \in X_i$. If $\mathsf{quant}(\Pi, l) = Q_i$ and $\mathsf{quant}(\Pi, k) = Q_j$, then $l <_\Pi k$ if $i < j$. A QBF $\forall x\Pi.\psi$ is satisfiable iff both $\Pi.\psi[x/\top]$ and $\Pi.\psi[x/\bot]$ are satisfiable where $\psi[x/t]$ denotes the replacement of x by t in ψ. Dually, a QBF $\exists x\Pi.\psi$ is satisfiable iff $\Pi.\psi[x/\top]$ or $\Pi.\psi[x/\bot]$ is satisfiable. The truth constants $\top$ and $\bot$ as well as the Boolean connectives follow the standard propositional semantics. Two QBFs ϕ_1 and ϕ_2 are equivalent (written as $\phi_1 \sim \phi_2$) iff they have the same truth value. The expression $ite(c, b_1, b_2)$ stands for $(c \rightarrow b_1) \wedge (\bar{c} \rightarrow b_2)$.

The *Skolem function* of an existential variable x w.r.t. QBF $\phi = \Pi.\psi$ is a propositional formula $f(y_1, \dots, y_n)$ where $y_1, \dots, y_n$ are all the universal variables of ϕ with $y_i <_\Pi x$. A Skolem function f_x for variable x of QBF $\Pi.\psi$ is valid iff $\Pi.\psi \sim \Pi.\psi[x/f_x]$. A *Skolem set* F of QBF ϕ maps each existential variable x of ϕ to a Skolem function $\mathsf{F}(x)$ of x. A Skolem set is valid if it maps existential variables only to valid Skolem functions. By $\phi[\mathsf{F}]$ we denote $\phi[x_1/\mathsf{F}(x_1), \dots, x_n/\mathsf{F}(x_n)]$ where $x_1, \dots, x_n$ are the existential variables of ϕ. If clear from the context, we sometimes speak about Skolem functions when referring to a complete Skolem set.

3 Skolem Function Continuation

Skolem functions as introduced above yield a strategy for assigning truth values to the existential variables based on the truth values of the universal variables such that the QBF under consideration evaluates to true. Some solvers are able to directly produce Skolem functions [8, 17, 6] while for others it is possible to extract the Skolem functions from proofs produced during the search [7, 10, 12]. In [12] we showed how to extract a Skolem function if the preprocessor Bloqqer is able to solve a true formula. In that case the produced QRAT proof of Bloqqer provides all the necessary information to construct a Skolem function. Now we reuse this technique for the case that not Bloqqer solves the formula, but another solver finds the solution for the preprocessed formula produced by Bloqqer.

In particular, we consider the following scenario: Given a satisfiable QBF ϕ, a QRAT trace T produced by a preprocessor rewriting ϕ to a QBF ϕ', and a valid Skolem set F' of ϕ', we show how to obtain a valid Skolem set F for ϕ.

Definition 1. *A* QRAT *trace T of a QBF $\Pi.\psi$ is a sequence of clause additions and clause deletions in the form of $(p_1, C_1), \dots, (p_n, C_n)$, where prefix $p_i \in \{+, -\}$[3] indicates if clause C_i is added ($p_i = +$) or deleted ($p_i = -$) justified by the rules of the* QRAT *proof system [12].*

[3] In the QRAT format, clause deletion lines start with "d".

For the detailed definitions of the QRAT rules and soundness arguments, we kindly refer to [12]. Note that the QRAT proof system also contains rules for modifying clauses. We omit these rules here because for satisfiable formulas a modification rule always can be expressed by a clause addition and a clause deletion rule. Basically, a QRAT trace of a QBF $\Pi.\psi$ compactly describes the sequence $\psi_T^0, \ldots, \psi_T^n$ of propositional formulas as follows:

$$\psi_T^i := \begin{cases} \psi & \text{if } i = 0 \\ \psi_T^{i-1} \cup \{C_i\} & \text{if } p_i = + \\ \psi_T^{i-1} \setminus \{C_i\} & \text{if } p_i = - \end{cases}$$

A clause addition step may even introduce new variables. We follow the convention of the QRAT proof format that such variables are existentially quantified and that they are appended right-most to the quantifier prefix. By Π_T^i we therefore refer to the quantifier prefix of ψ_T^i. A QRAT trace T with $|T| = n$ of a QBF $\Pi.\psi$ is a satisfaction proof if $\psi_T^n = \emptyset$. To construct a valid Skolem set from a QRAT satisfaction proof, a randomly initialised Skolem set is refined by traversing the proof backwards until it is valid [12]. For the case that the formula has not been solved by the preprocessor, i.e., $\psi_T^n \neq \emptyset$, we use the Skolem functions of the preprocessed formula for this initialization.

Definition 2. *Let $\phi = \Pi.\psi$ be a satisfiable QBF that is transformed to an equi-satisfiable QBF $\phi' = \Pi'.\psi'$ with valid Skolem set F'. Further, let T be a QRAT trace, with $|T| = n$, that describes the transformation of ψ to ψ' by the sequence $(\psi_T^0, \ldots, \psi_T^n)$ of propositional formulas where $\psi = \psi_T^0$, $\psi' = \psi_T^n$, and $\Pi' = \Pi_T^n$ as above.*

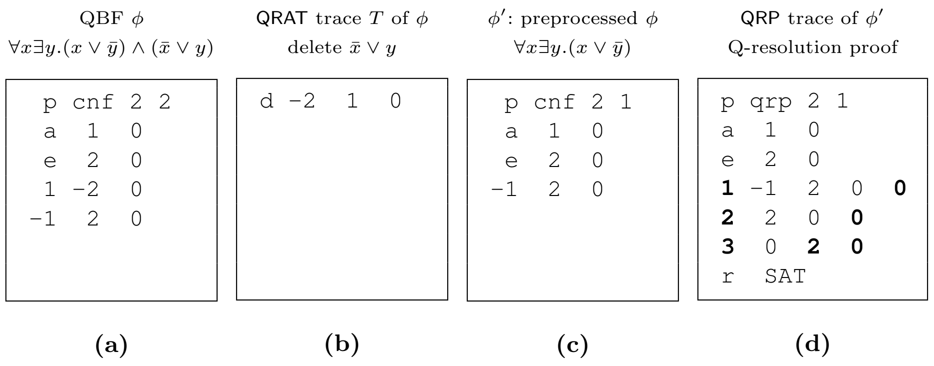
Then a sequence of Skolem sets $(\mathsf{F}_T^0, \ldots, \mathsf{F}_T^n)$ *for $(\psi_T^0, \ldots, \psi_T^n)$ is defined as follows. The Skolem sets F_T^i for $0 \leq i < n$ are constructed as in [12] and*

$$\mathsf{F}_T^n(x) := \begin{cases} \mathsf{F}'(x) & x \in vars(\Pi') \\ \bot & x \in \bigcup_{j=0}^{n-1} vars(\Pi_T^j) \setminus vars(\Pi'). \end{cases}$$

Variables not occurring in ϕ', but somewhere in the QRAT trace are assigned an arbitrary value in F_T^n ($\bot$ in our case). Next, we argue that each F_T^i is a valid Skolem set for $\Pi_T^i.\psi_T^i$ and so F_T^0 is a valid Skolem set for ϕ, the formula for which we want to construct the Skolem set.

Theorem 3. *Let $\phi = \Pi.\psi$ be a satisfiable QBF that is transformed to an equi-satisfiable QBF $\phi' = \Pi'.\psi'$ with valid Skolem set F'. Further, let T be a QRAT trace that describes the transformation of ψ to ψ'. Then the Skolem set $\mathsf{F}_T = \mathsf{F}_T^0$ obtained from $(\mathsf{F}_T^0, \ldots, \mathsf{F}_T^n)$ as described above, is valid on ϕ.*

Proof. We show by reverse induction that Skolem set F_T^i is valid on $\Pi_T^i.\psi_T^i$, i.e., $\psi_T^i[F_T^i] \sim \top$, for all $0 \leq i \leq |T|$. Since F' is a valid Skolem set on ϕ', the base case ($i = n = |T|$) trivially holds. The induction step is the same as in [12]. $\square$



QBF ϕ — $\forall x \exists y.(x \vee \bar{y}) \wedge (\bar{x} \vee y)$

```
 p cnf 2 2
 a  1  0
 e  2  0
 1 -2  0
-1  2  0
```

(a)

QRAT trace T of ϕ — delete $\bar{x} \vee y$

```
d -2  1  0
```

(b)

ϕ': preprocessed ϕ — $\forall x \exists y.(x \vee \bar{y})$

```
 p cnf 2 1
 a  1  0
 e  2  0
-1  2  0
```

(c)

QRP trace of ϕ' — Q-resolution proof

```
p qrp 2 1
a  1  0
e  2  0
1 -1  2  0  0
2  2  0  0
3  0  2  0
r  SAT
```

(d)

Fig. 1. (a) original QBF ϕ in QDIMACS format, (b) QRAT trace, (c) preprocessed formula ϕ' in QDIMACS format, (d) Q-resolution satisfaction proof in QRP format.

As a consequence of Theorem 3 we can reuse the Skolem function extraction algorithm of [12] and extract partial Skolem functions which we then continue with the Skolem functions of the preprocessed formula resulting in a valid Skolem set of the original formula. The approach is illustrated by the following example.

Example 4. Let ϕ be the true QBF $\forall x \, \exists y \, .(x \vee \bar{y}) \wedge (\bar{x} \vee y)$ (the QDIMACS representation is shown in Figure 1(a)). Assume that a simple preprocessor removes the first clause because it is a blocked clause [21] producing the QRAT trace T shown in Figure 1(b). The preprocessed formula $\phi' = \forall x \, \exists y \, .(\bar{x} \vee y)$ (see Figure 1(c)) is then passed to a QBF solver that decides its satisfiability. A solver like DepQBF also produces a Q-resolution proof in the QRP format [10] as shown in Figure 1(d). From this proof, a Skolem set F' can be extracted for ϕ' with $\mathsf{F}'(y) = f'_y(x) = \top$ [7]. Note that F' is not a valid Skolem set for ϕ because $\phi\,[y/\top] \sim \bot$. In order to get a valid Skolem set for ϕ, we use the extraction algorithm of [12] and get $\mathsf{F}^0_T(y) = f_y(x) = ite(\bar{x}, \bot, I)$, where we plug in f'_y for I. After simplifications, we get $f_y(x) = x$ which is a valid Skolem function for ϕ.

4 Architecture

We implemented the Skolem function continuation approach described above in a tool called `extract`, which is available on http://fmv.jku.at/sk-extract. The full tool chain is shown in Figure 2. The upper part shows the typical QBF evaluation process involving preprocessing. The lower part shows the extension with our new tool. Given a satisfiable QBF problem ϕ in PCNF, it is first simplified by the preprocessor Bloqqer, that employs different rewritings on the formula and produces a QRAT trace in order to ensure the correctness of these simplification steps. We modified the `qrat-trim` tool for checking QRAT traces (the original version only checks full QRAT proofs). The simplified formula ϕ' is then passed to a QBF solver that decides its truth value. The solver also generates (maybe with the help of further tools) a valid Skolem set F' on ϕ'.

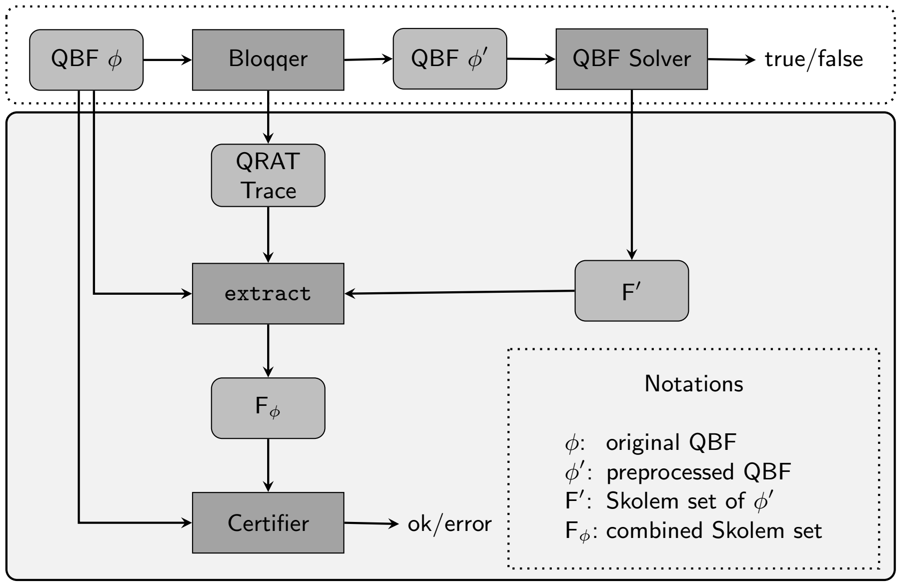


Fig. 2. Overview of our Skolem function continuation tool chain. The upper part of the figure shows a standard QBF evaluation process including preprocessing with Bloqqer, while the lower part presents the Skolem function combination steps.

This Skolem set is assumed to be represented as an And-Inverter-Graph (AIG) in the AIGER format (see http://fmv.jku.at/aiger/). Note that the AIG is the only interface to the QBF solver, hence the approach can be used with any solver that produces Skolem functions in AIGER format.

Given QBF ϕ, the QRAT trace, as well as the Skolem set of the preprocessed formula in AIGER format as input, `extract` constructs a valid Skolem function set F_ϕ of the original QBF ϕ. In a final step, we check if the produced Skolem set is valid. Therefore, the Certifier of Figure 2 (1) checks the structural correctness of the Skolem set with the tool `cheskol` and (2) builds $\phi\,[\mathsf{F}_\phi]$ that is a universally quantified formula. For evaluating it with a SAT solver, its negation is translated to a CNF formula that must be unsatisfiable if F is valid.

5 Experimental Evaluation

To evaluate our approach we consider 367 formulas of the QBF Eval 2016 main track that was claimed to be satisfiable by at least one QBF solver participating in the competition. All experiments were run on a cluster of computers with Intel Q9550 2.83GHz CPUs equipped with 8GB of memory. We set the memory limit to 7GB and the time limit to $900s$ for the full solving tool chain.

For preprocessing with QRAT tracing, we use the preprocessor Bloqqer version `v038` in configurations `FULL` (all options enabled), `noMS` (miniscoping for

Table 1. Comparison of different Skolem function extraction tool chains

solver	pre-#	sol-#	ext-#	che-#	MO-#	TO-#	che-t	tot-t
DepQBF	–	160	151	123	10	234	21	30
Caqe	–	148	148	111	186	70	31	40
Bloqqer-BP-DepQBF	114	273	268	251	12	104	14	48
Bloqqer-noCCE-QRAT-DepQBF	150	282	275	258	7	102	18	39
Bloqqer-FULL-QRAT-DepQBF	178	289	275	257	9	101	14	28
Bloqqer-noMS-QRAT-DepQBF	136	281	266	247	11	109	21	35
Bloqqer-noCCE-QRAT-Caqe	150	270	268	236	53	78	18	35

pre-#: formulas solved by preprocessor sol-#: formulas solved in total MO-#: memoryouts
ext-#: extracted Skolem functions che-#: checked Skolem functions TO-#: timeouts
ch-t: average checking time (s) tot-t: average total time without time/memoryouts (s)

universal expansion disabled), and `noCCE` (covered clause elimination disabled). Miniscoping is a syntactic-based technique relaxing the quantifier ordering and is the only technique currently not supported by the `qrat-trim` checker which verifies that all steps of the QRAT trace are correct. If we keep miniscoping enabled, we currently lose this additional check. CCE is a preprocessing technique that often considerably increases the size of the Skolem functions. For comparison, we also include the version of Bloqqer modified by Janota et al. [20] (called Bloqqer-BP in the following) that performs only a subset of preprocessing techniques for which solution reconstruction is implemented in a tool called backport. Checking the extracted Skolem functions is done with the checker `king-cc`. In all other tool chains, we use the SAT solver Lingeling version ayv for verifying that the extracted Skolem functions are valid. We further checked syntactical correctness of the Skolem functions generated by our approach with the tool `cheskol`. As complete QBF solvers, we integrated the two recent tools Caqe [17] and DepQBF [22] into the framework as shown in Figure 2. Both of them provide Skolem functions represented as AIG. The solver Caqe directly produces Skolem functions during solving, while DepQBF dumps Q-resolution proofs from which Skolem functions are extracted by `qbfcert` [10].

The results of our experiments are summarized in Table 1. Timeouts and memory outs are given in columns TO-# and MO-#. Column sol-# shows the number of solved formulas. Out of them pre-# formulas are directly solved by the preprocessor. The column che-# shows the number of formulas that passed the complete solving flow, i.e., for these formulas Skolem functions could be extracted that were successfully checked. We did not encounter any formulas where the check failed except for timeouts or memoryouts. With preprocessing enabled more than 100 further formulas pass the whole solving flow. We also observe that our general approach based on QRAT traces performs in the same order of magnitude as the specialized approach based on the traces produced by Bloqqer-BP. Detailed runtime comparisons between all solvers and a comparison of Skolem function sizes produced by the Bloqqer-BP-DepQBF and the Bloqqer-noCCE-QRAT-DepQBF configurations are shown in Figure 3. Scripts and log-files of the experiments are available on http://fmv.jku.at/sk-extract.

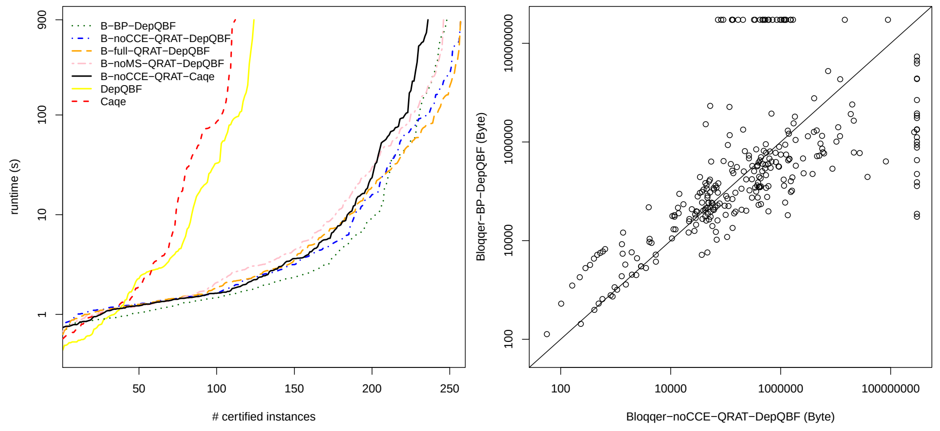


Fig. 3. Runtime comparison of full tool chains (left) and size comparison of Skolem functions by Bloqqer-BP-DepQBF and Bloqqer-noCCE-QRAT-DepQBF (right).

6 Conclusion

In this paper we described a general approach to obtain valid Skolem functions for a quantified Boolean formula that has been preprocessed by a QRAT trace producing preprocessor like Bloqqer. The described method reuses and modifies a previously presented approach for extracting Skolem functions from QRAT proofs [12]. We showed how to continue the incomplete Skolem functions extracted from the QRAT trace with the Skolem functions of the preprocessed formula. We implemented this method in a new Skolem function extraction tool and performed an extensive evaluation on formulas of the QBF Eval 2016 main track. We observed that our general method performs similarly well as the only available specialized approach for that purpose. Thus, our tool can be smoothly integrated into typical QBF solving tool chains in order to find semantic certificates of true QBFs. Such certificates are witnesses for the correctness of the solving results as solutions of the application problem encoded in QBF.

Potential future work is the extraction of Herbrand functions as witnesses of unsatisfiable QBFs as well as the optimization of extracted Skolem functions. For obtaining a tighter integration of preprocessing and solving, we consider to directly integrate proofs of different proof systems.

Acknowledgements

We would like to thank Luca Pulina for providing us with the list of satisfiable instances of the QBF Eval 2016.

References

1. Kleine Büning, H., Bubeck, U.: Theory of quantified Boolean formulas. In: Handbook of Satisfiability. IOS Press (2009) 735–760
2. Benedetti, M., Mangassarian, H.: QBF-based formal verification: Experience and perspectives. JSAT **5**(1-4) (2008) 133–191
3. Bloem, R., Könighofer, R., Seidl, M.: SAT-based synthesis methods for safety specs. In: VMCAI. Volume 8318 of LNCS., Springer (2014) 1–20
4. Egly, U., Kronegger, M., Lonsing, F., Pfandler, A.: Conformant planning as a case study of incremental QBF solving. In: AISC. Volume 8884 of LNCS., Springer (2014) 120–131
5. Balabanov, V., Jiang, J.R., Scholl, C.: Skolem functions computation for CEGAR based QBF solvers. In: QBF. (2015)
6. Rabe, M.N., Seshia, S.A.: Incremental determinization. In: SAT. Volume 9710 of LNCS., Springer (2016) 375–392
7. Balabanov, V., Jiang, J.R.: Unified QBF certification and its applications. Formal Methods in System Design **41**(1) (2012) 45–65
8. Benedetti, M.: Extracting certificates from quantified Boolean formulas. In: IJCAI, Professional Book Center (2005) 47–53
9. Narizzano, M., Peschiera, C., Pulina, L., Tacchella, A.: Evaluating and certifying QBFs: A comparison of state-of-the-art tools. AI Commun. **22**(4) (2009) 191–210
10. Niemetz, A., Preiner, M., Lonsing, F., Seidl, M., Biere, A.: Resolution-based certificate extraction for QBF - (tool presentation). In: SAT. (2012) 430–435
11. Jussila, T., Biere, A., Sinz, C., Kröning, D., Wintersteiger, C.M.: A first step towards a unified proof checker for QBF. In: SAT. Volume 4501 of LNCS., Springer (2007) 201–214
12. Heule, M.J.H., Seidl, M., Biere, A.: Solution validation and extraction for QBF preprocessing. J. Autom. Reasoning **58**(1) (2017) 97–125
13. Goultiaeva, A., Van Gelder, A., Bacchus, F.: A uniform approach for generating proofs and strategies for both true and false QBF formulas. In: IJCAI, IJCAI/AAAI (2011) 546–553
14. Van Gelder, A.: Certificate extraction from variable-elimination QBF preprocessors. In: QBF. (2013) 35–39
15. Kleine Büning, H., Karpinski, M., Flögel, A.: Resolution for quantified Boolean formulas. Inf. Comput. **117**(1) (1995) 12–18
16. Giunchiglia, E., Marin, P., Narizzano, M.: Reasoning with quantified Boolean formulas. In: Handbook of Satisfiability. IOS Press (2009) 761–780
17. Rabe, M.N., Tentrup, L.: CAQE: A certifying QBF solver. In: FMCAD. (2015) 136–143
18. Lonsing, F., Seidl, M., Van Gelder, A.: The QBF gallery: Behind the scenes. Artif. Intell. **237** (2016) 92–114
19. Marin, P., Narizzano, M., Pulina, L., Tacchella, A., Giunchiglia, E.: Twelve years of QBF evaluations: QSAT is PSPACE-hard and it shows. Fundam. Inform. **149**(1-2) (2016) 133–158
20. Janota, M., Grigore, R., Marques-Silva, J.: On QBF proofs and preprocessing. In: LPAR. (2013) 473–489
21. Biere, A., Lonsing, F., Seidl, M.: Blocked clause elimination for QBF. In: CADE. Volume 6803 of LNCS., Springer (2011) 101–115
22. Lonsing, F., Egly, U.: DepQBF: An incremental QBF solver based on clause groups. CoRR **abs/1502.02484** (2015)