Evaluating QBF Solvers: Quantifier Alternations Matter*

Florian Lonsing and Uwe Egly

Knowledge-Based Systems Group, Vienna University of Technology, Austria
firstname.lastname@tuwien.ac.at

Abstract

Competitions of quantified Boolean formula (QBF) solvers are an important driving force for solver development. We consider solvers and benchmarks in prenex conjunctive normal form (PCNF) that participated in the recent QBF competition (QBFEVAL’16) and take a fresh look at the number of solved instances as a measure of solver performance. Rather than ranking solvers by the total number of solved instances, which is common practice in competitions, we determine solver rankings with respect to instances that are solved in classes of instances having a particular number of quantifier alternations. We report experimental results which indicate that solver performance substantially varies depending on the number of alternations in the underlying instance classes. In particular, we observed that solvers implementing orthogonal solving paradigms, such as variable expansion or backtracking search with clause learning, perform better on instances having either few or many alternations, respectively. Consequently, a bias towards instances with a certain number of alternations in a benchmark set may result in a biased solver ranking. In order to avoid biased rankings, our observations motivate the development of alternative performance measures of QBF solvers and competition setups that are more robust with respect to alternations.

1 Introduction

The logic of quantified Boolean formulas (QBF) [21] extends propositional logic (SAT) by existential and universal quantification of propositional variables. The use of quantification allows for encodings of problems that are potentially exponentially more succinct than SAT encodings, but also turns the satisfiability problem of QBFs PSPACE-complete [39].

Similar to related fields QBF solver competitions have been an important driving force for the development of QBF solvers since the first edition of QBFEVAL2 in 2003 [24]. Competition benchmarks are selected from QBFLIB [14] or may contain new encodings of problems.

In general problems from any level of the polynomial hierarchy (PH) [30, 38, 41] can be encoded as a QBF. PH is a theoretical framework to describe the complexity of problems that are beyond the class NP. Examples of problems in PH are conformant planning [34] or two-player games [35]. The satisfiability problem of a QBF ψ in prenex conjunctive normal form (PCNF) with k ≥ 0 quantifier alternations is located at level k + 1 of PH [38, 41] and thus is either Σk+1-complete or Πk+1-complete, depending on the first quantifier in the prefix of ψ.

We consider quantifier alternations as a way towards a more fine grain measure of QBF solver performance. Instead of counting the total number of solved instances in a given benchmark set, which is common practice in solver competitions, we count instances that were solved in certain classes of instances defined by the number of alternations. To this end, we report

---

*This work was supported by the Austrian Science Fund (FWF) under grant S11409-N23. Parts of the results in this paper were presented by the first author in a talk at Dagstuhl seminar 16381 SAT and Interactions [5]

1E.g. competitions of SAT solvers (http://www.satcompetition.org/), satisfiability modulo theories (SMT) solvers (http://www.smtcomp.org/), and theorem provers (http://www.cs.miami.edu/~tptp/CASC/).

2QBFEVAL (QBF solver competitions): http://www.qbflib.org/index_eval.php
experimental results based on solvers and instances in PCNF from the recent QBF competition (QBFEVAL'16). We show that the benchmark set used in QBFEVAL'16 is heavily biased towards instances with no more than two alternations. This bias is preserved under state of the art preprocessing, which is employed by many solvers.

We point out that the performance of approaches implemented in solvers largely varies with respect to classes of instances. In particular, variable expansion [2, 6] based on counter example guided abstraction refinement (CEGAR) [12, 18] tends to perform well on instances with one or two alternations. In contrast to that, we identified that backtracking search with clause learning (QCDCL) [15, 16, 25, 42] based on Q-resolution [32] performs best on instances with many alternations. While it is a common observation in competitions that the benchmark selection might influence overall solver rankings, we observed different rankings in different classes.

In a related study [29], instances were classified using a set of syntactic features by machine learning. Based on the resulting classification, the correlation between solver performance and treewidth of instances was studied. Treewidth has been studied both as an empirical [32] and theoretical hardness measure of instances, e.g. [1]. In contrast to a set of syntactic features, in our experimental study the classification relies on the single feature of alternations, which are naturally related to the theoretical hardness of instances in PH. With our simple classification by alternations we observe a similar diversity of solver performance as with the more complex classification by several features [29]. Rather than correlating features like treewidth with solver performance, we aim at raising the awareness of biased benchmark sets and related consequences.

A competition benchmark set where instances with a certain number of alternations are overrepresented gives unjust advantage to solving approaches which perform well on overrepresented instances. However, such advantage hinders the evaluation of approaches that perform well on underrepresented instances. Competition benchmarks are routinely used to compare existing and novel solving approaches and to evaluate progress of the field. As a result from biased benchmarks, future research may inadvertently be narrowed down to exploring approaches that tend to perform well on instances with a certain number of alternations.

Therefore, to establish QBF solving as a technology to solve problems from any levels of PH it is necessary to improve QBF solvers on instances with any number of alternations. Our observations motivate the development of new methodologies to evaluate approaches implemented in QBF solvers and to devise competition setups that are robust with respect to alternations.

2 Experimental Setup

We consider QBFs $\psi := \Pi \phi$ in prenex conjunctive normal form (PCNF) consisting of a quantifier prefix $\Pi := Q_1B_1 \ldots Q_nB_n$ and a quantifier-free propositional formula $\phi$ in CNF. The prefix $\Pi$ is a linearly ordered sequence of quantifier blocks (qblocks) $Q_iB_i$, where $Q_i \in \{\forall, \exists\}$ is a quantifier and $B_i$ is a block (i.e. a set) of propositional variables. The notation $Q_iB_i$ is shorthand for $Q_i x_1 \ldots Q_i x_m$ for all $x_j \in B_i$. Formula $\phi$ is defined precisely over the variables that appear in $\Pi$ and does not contain variables that are not in $\Pi$. Adjacent qblocks in $\Pi$ are quantified differently, i.e. $Q_i \neq Q_{i+1}$. Otherwise, if $Q_i = Q_{i+1}$ then $B_i$ and $B_{i+1}$ are merged to obtain $Q_i(B_i \cup B_{i+1})$. The innermost quantifier $Q_n = \exists$ is existential. Otherwise, if $Q_n = \forall$ then all variables in $B_n$ can be eliminated from $\psi$ by universal reduction [22]. A PCNF with $n$ qblocks has $n-1$ quantifier alternations. We focus on the number of qblocks.

The semantics of PCNFs are defined recursively. The PCNF consisting only of the syntactic truth constant $\top (\bot)$ is satisfiable (unsatisfiable). A PCNF $\psi := Q_1B_1 \ldots Q_nB_n\phi$ with $Q_1 = \exists (Q_1 = \forall)$ is satisfiable iff, for $x \in B_1$, $\psi[x]$ or (and) $\psi[\neg x]$ is satisfiable, where $\psi[x]$ (or $\psi[\neg x]$) is the PCNF obtained from $\psi$ by replacing all occurrences of $x$ ($\neg x$) by $\top (\bot)$ and deleting $x$ from $B_1$. 
For our experimental study, we use the set $S_{825}$ of 825 instances from the PCNF track of QBFEVAL’16 [31]. Partitioning $S_{825}$ by numbers of qblocks results in 70 classes. Table 1 summarizes the numbers of formulas (#f) in $S_{825}$ that appear in classes given by the number of qblocks (#q). Four instances contain only a single block of existentially quantified variables and thus are purely propositional. One instance has the maximum of 1061 qblocks. Instances with up to three qblocks (row “1–3”) amount to 56% of all instances and hence are overrepresented in $S_{825}$. To generate $S_{825}$ for QBFEVAL’16, at most ten instances were selected at random from every category of instances in QBFLIB [31] to avoid a bias towards a particular category. The categories contain instances from various application domains. Therefore, we conclude that the bias towards instances with few qblocks in $S_{825}$ is present already in QBFLIB and does not result from the way how $S_{825}$ was generated.

In order to evaluate the impact of qblocks on solver performance, we consider the following ten (variants of) solvers that participated in QBFEVAL’16 [31] and were top-ranked. The solvers implement the following five different solving paradigms:

1. **Variable expansion** [2, 6] eliminates variables from a PCNF $\psi$ until the formula reduces to either $\top$ or $\bot$. The solver AIGSolve [36] combines expansion with simplification techniques and an and-inverter graph (AIG) representation of $\psi$. In contrast to AIGSolve, RAReQS [18] applies expansion based on counter example guided abstraction refinement (CEGAR) [12].

2. **QDPLL** [11] is a backtracking search procedure that extends the DPLL algorithm [13]. The solver GhostQ [18, 23] combines QDPLL with clause and cube learning. Additionally, it reconstructs the structure of PCNFs encoded by Tseitin translation [40] and applies CEGAR based learning techniques. In variant GhostQ-nd, we disabled structure reconstruction.

3. **Nested SAT solving** uses one SAT solver for each qblock in a PCNF, where universal quantification is handled as negated existential quantification. The solver QSTS-dsb [8, 9] combines nested SAT solving with symmetry breaking and structure reconstruction. In the variant QSTS-ndsb we disabled symmetry breaking and structure reconstruction.

4. **Clause selection/clausal abstraction** decomposes the given PCNF into a sequence of propositional formulas and is based on a CEGAR approach. The solvers CAQE [33] and QESTO [20] implement clausal abstraction and clause selection, respectively.

5. **Backtracking search with clause and cube learning (QCDCL)** [15, 16, 25, 42] based on Q-resolution [22] extends the CDCL approach for SAT solving [37] to QBF. The solver DepQBF [27] implements QCDCL with generalized Q-resolution axioms allowing for a stronger calculus to derive learned clauses and cubes. The variant DepQBF-n [26] applies a restricted set of axioms.

### Table 1

| #q | #f  |
|----|-----|
| 1  | 4   |
| 2  | 71  |
| 3  | 391 |
| 4–6| 47  |
| 7–9| 58  |
| 10–15| 41 |
| 16–20| 57 |
| 21–156| 156 |
| 1–3| 466 |
| 4–  | 359 |

### 3 Experimental Results

In the following we report experimental results illustrating a diversity of solver performance with respect to the numbers of alternations in the instances. To this end, we also take the effects of preprocessing into account. In particular, we show that the considered benchmark set is biased before and after the application of preprocessing. On both original and preprocessed instances, we observe different solver rankings in different classes of instances given by their numbers of

---

3QBFEVAL’16 results: [http://www.qbflib.org/solver_view_domain.php?year=2016&idStruct=2&track=1](http://www.qbflib.org/solver_view_domain.php?year=2016&idStruct=2&track=1)

4A cube (clause) is a conjunction (disjunction) of literals.
Evaluating QBF Solvers: Quantifier Alternations Matter
Lonsing and Egly

| Solver    | $S$ | $\bot$ | $\top$ | Time |
|-----------|-----|--------|--------|------|
| AIGSolve  | 603 | 301    | 302    | 440K |
| GhostQ    | 593 | 292    | 301    | 457K |
| QSTS-dsb  | 578 | 294    | 284    | 469K |
| DepQBF    | 457 | 255    | 202    | 689K |
| DepQBF-n  | 433 | 246    | 187    | 734K |
| GhostQ-nd | 395 | 222    | 173    | 807K |
| CAQE      | 378 | 202    | 176    | 831K |
| QESTO     | 369 | 210    | 159    | 864K |
| RAReQS    | 341 | 211    | 130    | 891K |
| QSTS-ndsb | 307 | 170    | 137    | 952K |

| Solver   | $S$ | $\bot$ | $\top$ | Time |
|----------|-----|--------|--------|------|
| QSTS-dsb | 633 | 330    | 303    | 365K |
| RAReQS   | 633 | 334    | 299    | 375K |
| QESTO    | 620 | 321    | 299    | 395K |
| DepQBF   | 597 | 301    | 296    | 439K |
| DepQBF-n | 596 | 300    | 296    | 437K |
| CAQE     | 596 | 301    | 295    | 451K |
| QSTS-ndsb| 594 | 303    | 291    | 442K |
| GhostQ   | 570 | 282    | 288    | 485K |
| GhostQ-nd| 514 | 264    | 250    | 572K |

Table 2: Solved instances ($S$), solved unsatisfiable ($\bot$) and satisfiable ones ($\top$), and total wall clock time including time outs on set $S_{825}$.

alternations. All experiments reported were run on an AMD Opteron 6238 processor (2.6 GHz) under 64-bit Ubuntu Linux 12.04 with time and memory limits of 1800 seconds and seven GB. If a solver exceeds the memory limit, then the respective instance is counted as a time out.

In QBF EVAL’16 some solvers were submitted as a bundle including the preprocessor Bloqqer \[7,17\]. Bloqqer implements several techniques to eliminate variables and clauses from a given PCNF $\psi$ and literals from clauses in $\psi$. By applying these techniques in a resource bounded way, Bloqqer aims at transforming $\psi$ into a smaller PCNF $\psi'$ which is supposed to be solved faster than the original PCNF $\psi$.

However, it is well known that Bloqqer may have positive effects on the performance of certain solvers while negative effects on others (cf. \[28,29\]). Therefore, in order to compensate for the observed diverse effects of Bloqqer, in our experimental study we consider both original instances and instances that have been preprocessed. To this end, we ran Bloqqer (version 37) on the set $S_{825}$ to obtain the set $S'_{825}$ containing 825 preprocessed PCNFs. We included 13 instances where Bloqqer did not terminate within a two hour wall clock time limit in their original form in $S'_{825}$.

The ranking of the solvers by total solved instances with respect to sets $S_{825}$ and $S'_{825}$ in Tables 2 and 3, respectively, reveals a large diversity of solver performance. While Bloqqer is harmful to AIGSolve and GhostQ, which solve more instances on $S_{825}$ than on $S'_{825}$, it improves the performance of all other solvers. In particular, RAReQS benefits from Bloqqer as it solves 85% more instances on $S'_{825}$ than on $S_{825}$. Techniques like structure reconstruction and symmetry breaking are crucial for the variants of GhostQ and QSTS-dsb, respectively.

Bloqqer already solves 354 instances (177 satisfiable and 177 unsatisfiable ones) from $S_{825}$. On these 354 original instances, solver performance largely varies as shown by numbers of solved instances (column $S$) in Table 4.\(^6\) Hence these instances are non-trivial. The observation made for Tables 2 and 3 that Bloqqer improves the performance of RAReQS while it is harmful to AIGSolve conforms to Table 4. There, AIGSolve

\(^5\)http://fmv.jku.at/bloqqer/
\(^6\)Table 11 in the appendix shows detailed statistics.

| Solver    | $S$ |
|-----------|-----|
| AIGSolve  | 350 |
| QSTS-dsb  | 314 |
| GhostQ    | 303 |
| GhostQ-nd | 264 |
| DepQBF    | 252 |
| CAQE      | 240 |
| DepQBF-n  | 236 |
| QESTO     | 198 |
| QSTS-ndsb | 166 |
| RAReQS    | 164 |

Table 4
solves more than twice as many instances as RAREQS. On set $S_{825}'$ (Table 3) the 354 instances solved by Bloqqer are counted as solved by every solver, hence AlgSolve does no longer have an advantage on these instances compared to solvers that fail to solve many of them.

Furthermore, Bloqqer eliminated all universal variables from 69 instances in $S_{825}$, thus making them purely propositional. As these propositional instances could be forwarded to a SAT solver, they are of minor relevance when evaluating QBF solvers (all solvers we considered except the variants of GhostQ and DepQBF employ a SAT solver as a backend). To focus the evaluation on the actual QBF-specific approaches implemented in the solvers, we discarded the 354 instances solved by Bloqqer and the 69 propositional ones from $S_{825}$ to obtain the sets $S_{402}$ and $S_{402}'$ containing 402 instances each. Set $S_{402}$ contains instances in their original unprocessed form whereas $S_{402}'$ contains preprocessed ones (including the 13 original instances where Bloqqer timed out). In the following, we focus our experimental analysis on sets $S_{402}$ and $S_{402}'$.

|  | No Pre. | Prec. |
|---|---|---|
| min | 63 | 97 |
| max | 117,736 | 90,640 |
| avg | 11,787 | 5,408 |
| med | 5,488 | 1,435 |

Table 5

|  | No Pre. | Prec. |
|---|---|---|
| min | 1 | 1 |
| max | 22,696 | 22,411 |
| avg | 959 | 903 |
| med | 128 | 94 |

|  | No Pre. | Prec. |
|---|---|---|
| min | 2 | 2 |
| max | 1,061 | 179 |
| avg | 16 | 7 |
| med | 3 | 3 |

|  | No Pre. | Prec. |
|---|---|---|
| min | 446 | 2,130 |
| max | 2,304,186 | 2,299,186 |
| avg | 74,976 | 49,451 |
| med | 22,192 | 9,248 |

As illustrated by Tables 6 and 7, solver performance strongly varies on sets $S_{402}$ and $S_{402}'$ with respect to the use of preprocessing. This observation is similar to sets $S_{625}$ and $S_{625}'$ (Tables 2 and 3). However, whereas the filtering of instances that we applied to sets $S_{825}$ and $S_{825}'$ to obtain $S_{402}$ and $S_{402}'$, respectively, resulted in different solver rankings in Tables 2 and 6 and in Tables 3 and 7, it did not affect the respective sets of the five top performing solvers. On $S_{825}$ and $S_{402}$ AlgSolve, GhostQ, QSTS-dsb, and the variants of DepQBF perform best, while on $S_{825}'$ and $S_{402}'$ QSTS-dsb, RAREQS, QESTO, and the variants of DepQBF perform best. Hence empirically the filtered sets $S_{402}$ and $S_{402}'$ have characteristics with respect to the best performing solvers that are similar to their unfiltered counterparts $S_{825}$ and $S_{825}'$. As identified above, the qblocks bias observed in $S_{825}$ is also present in $S_{402}$ and $S_{402}'$. Due to these properties, we analyze solver performance in classes of instances in $S_{402}$ and $S_{402}'$ given by numbers of qblocks.
3.1 Class Based Performance Analysis

Tables 8 and 9 show the numbers of instances that were solved in a class of instances in $S_{402}$ and $S'_{402}$ defined by the numbers of qblocks. Only class winners are shown, i.e., solvers that solved the largest number of instances in at least one class (bold face). Ties are broken by wall clock time including time outs. The two rows at the bottom of the tables show statistics for instances with up to three (row “2–3”) and more than three qblocks (row “4–”). These class based statistics provide a more detailed view of solver performance than the solver rankings by total solved instances in Tables 6 and 7.

For set $S_{402}$ there are four different class winners (Table 8), which correspond to the four best performing solvers according to the overall ranking in Table 6. Interestingly, these class

Footnote 7: We refer to the appendix, where Tables 12 to 17 show detailed statistics related to classes “2–3”, “4–”, and statistics for all solvers for the sets $S_{402}$ and $S'_{402}$, respectively.
winners implement the following four different solving paradigms (as listed in Section 2): 2 (GhostQ), 3 (QSTS-dsb), 1 (AIGSolve), and 5 (DepQBF).

In contrast to $S_{402}$, the five different class winners for set $S'_{402}$ (Table 9) do not correspond to the five top ranked solvers in Table 7. QESTO is ranked third but does not win in a class. However, AIGSolve is ranked ninth but wins class “4–6”. The three different solving paradigms implemented by the class winners on $S_{402}$ are 1 (RAReQ5 and AIGSolve), 3 (QSTS-dsb), and 5 (DepQBF and DepQBF-n).

For sets $S_{402}$ and $S'_{402}$, GhostQ and RAReQ5, which employ CEGAR techniques, win on instances with up to three (rows “2–3”) qblocks. On both $S_{402}$ and $S'_{402}$, however, the QCDCL solvers DepQBF and DepQBF-n win in the class of instances with more than three qblocks (“4–”).

Tables 8 and 9 illustrate the different performance of the solving paradigms implemented in solvers that win in a particular class. Furthermore the results underpin the problem of benchmark sets that are biased towards instances with a certain number of qblocks. For example, GhostQ and RAReQ5 win in classes “2–3”, which are overrepresented in $S_{402}$ and $S'_{402}$ as they contain 64% and 67% of all instances, and are also the overall winners on the entire sets. The other class winners shown in Tables 8 and 9 outperform GhostQ and RAReQ5 on all classes with more than three qblocks. However, due to the bias their superior performance on instances with more than three qblocks does not compensate for their performance lack on class “2–3”.

### 3.2 Virtual Best Solver Analysis

Our above observation that the performance of solvers varies with respect to classes given by the numbers of qblocks is further substantiated by considering statistics related to the virtual best solver (VBS)—also called state of the art (SOTA) solver—on a particular benchmark set. The VBS is an ideal solver portfolio where the wall clock time of the fastest solver on an instance is attributed to the VBS. Thus the VBS reflects the best performance that can be achieved when running a set of solvers in parallel on an instance. VBS analysis is common in QBF [29] and SAT solver competitions (cf. [3]).

Table 10 shows the number of instances solved by the VBS in each class in set $S'_{402}$ and the relative contribution of each solver (percentage) to the VBS. The solvers have different relative contributions to the VBS depending on the number of qblocks of the considered instances. For example, while RAReQ5 and QSTS-dsb have higher contributions on class “2–3” than on class “4–”, DepQBF and DepQBF-n show the opposite tendency. Six different solvers have the highest contribution (bold face) to the VBS in particular classes. This observation conforms to the fact that there are five different class winners in Table 9. We made similar observations for set $S_{402}$.

Similar to solver performance in classes (Table 9), the relative contributions of a solver to the VBS with respect to classes provide a more fine grain picture of the strengths of the approaches implemented in the solvers than the contributions with respect to the entire set (row “2–”).

### 4 Conclusions

Regular competitions of QBF solvers are important to assess the state of the art in QBF solving. The benchmark sets used in competitions routinely become a standard set to compare novel solving approaches against existing ones.

---

8Due to limited space, we omit VBS statistics for set $S_{402}$ (cf. Table 8). These statistics can be found in Table 18 in the appendix.
We have pointed out that the benchmark set from the recent QBF competition QBF-EVAL’16 is biased towards instances with no more than two quantifier alternations as these instances amount to 56% of the set. This bias may result from an overrepresentation of instances with few alternations in QBFLIB, the benchmark repository of the QBF research community.

Our experimental results with top-performing solvers from QBF-EVAL’16 illustrated that solvers implementing different solving paradigms—rather than variations of a single paradigm—show diverse performance on instances with a different number of alternations. We observed different solver rankings on classes of instances given by their numbers of alternations. In contrast to a related study [29], our classification relies on alternations as the single syntactic feature.

By our experimental study we aim at raising the awareness of alternation bias in benchmark sets and the potential consequences. In order to prevent the field of QBF solving from potentially converging to solvers that implement a particular solving paradigm, alternation bias in benchmarks sets must be prevented. This is particularly important given the variety of solving paradigms that are currently implemented in QBF solvers. For example, regarding QBF proof complexity, there is an exponential separation between the proof systems that underlie CEGAR based expansion and QCDCL [4, 19] as implemented in the solvers RAReQS and DepQBF. Thus on a biased benchmark set, the strengths of a particular proof system applied by a solver may be obscured in the results. As a consequence, research on such an approach may be abandoned, which may leave behind a gap no other solving paradigm (or proof system) may be able to fill.

To further improve QBF solving as a technology to solve problems from any level of the polynomial hierarchy and from the class PSPACE, it is necessary to leverage the strengths of different solving paradigms. Our observations motivate the development of new methodologies to evaluate solving paradigms that compensate for alternation related performance differences.

Table 10: Number of instances from set $S'_{402}$ (preprocessed by Bloqqer, cf. Table 9) solved by the virtual best solver (VBS) in classes by number of qblocks (#q) and number of formulas (#f), and relative contribution (percentage) of each solver to instances solved by the VBS.

| #q | #f | VBS | RAReQS | QSTS-obs | QESTO | DepQBF | DepQBF-n | CAQE | QSTS-ndsQ | GhostQ | AIGSolve | GhostQ-nd |
|----|----|-----|--------|----------|--------|--------|----------|------|-----------|--------|----------|----------|
| 2  | 34 | 18  | 11.1   | 0.0      | 5.5    | 33.3   | 11.1     | 5.5  | 16.6      | 0.0    | 16.6     | 0.0      |
| 3  | 236| 164 | 33.5   | 35.9     | 3.6    | 4.8    | 1.2      | 0.0  | 11.5      | 0.0    | 9.1      | 0.0      |
| 4–6| 24 | 16  | 18.7   | 0.0      | 6.2    | 18.7   | 12.5     | 0.0  | 18.7      | 0.0    | 25.0     | 0.0      |
| 7–9| 31 | 31  | 16.1   | 6.4      | 3.2    | 22.5   | 22.5     | 0.0  | 19.3      | 0.0    | 9.6      | 0.0      |
| 10–15| 28 | 20  | 0.0    | 20.0     | 0.0    | 15.0   | 0.0      | 10.0 | 5.0       | 0.0    | 0.0      | 5.0      |
| 16–20| 30 | 24  | 8.3    | 25.0     | 4.1    | 16.6   | 8.3      | 0.0  | 29.1      | 0.0    | 4.1      | 4.1      |
| 21– | 19 | 10  | 30.0   | 20.0     | 0.0    | 30.0   | 0.0      | 0.0  | 20.0      | 0.0    | 0.0      | 0.0      |

2–3  | 270| 182 | 31.3   | 32.4     | 3.8    | 7.6    | 2.1      | 0.5  | 12.0      | 0.0    | 9.8      | 0.0      |
| 4–  | 132| 101 | 16.8   | 14.8     | 2.9    | 20.7   | 13.8     | 0.0  | 19.8      | 0.9    | 7.9      | 1.9      |
| 2–  | 402| 283 | 26.1   | 26.1     | 3.5    | 12.3   | 6.3      | 0.3  | 14.8      | 0.3    | 9.1      | 0.7      |
References

[1] Albert Atserias and Sergi Oliva. Bounded-width QBF is PSPACE-complete. In STACS, volume 20 of LIPIcs, pages 44–54. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2013.

[2] Abdelwahab Ayari and David A. Basin. QUBOS: Deciding Quantified Boolean Logic Using Propositional Satisfiability Solvers. In FMCAD, volume 2517 of LNCS, pages 187–201. Springer, 2002.

[3] Tomas Balys, Armin Biere, Markus Iser, and Carsten Sinz. SAT Race 2015. Artif. Intell., 241:45–65, 2016.

[4] Olaf Beyersdorff, Leroy Chew, and Mikoláš Janota. Proof Complexity of Resolution-based QBF Calculi. In STACS, volume 30 of LIPIcs, pages 76–89. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2015.

[5] Olaf Beyersdorff, Nadia Creignou, Uwe Egly, and Heribert Vollmer. SAT and Interactions (Dagstuhl Seminar 16381). Dagstuhl Reports, 6(9):74–93, 2017.

[6] Armin Biere. Resolve and Expand. In SAT, volume 3542 of LNCS, pages 59–70. Springer, 2004.

[7] Armin Biere, Florian Lonsing, and Martina Seidl. Blocked Clause Elimination for QBF. In CADE, volume 6803 of LNCs, pages 101–115. Springer, 2011.

[8] Bart Bogaerts, Tomi Janhunen, and Shahab Tasharrofi. SAT-to-SAT in QBFEval 2016. In QBF Workshop, volume 1719 of CEUR Workshop Proceedings, pages 63–70. CEUR-WS.org, 2016.

[9] Bart Bogaerts, Tomi Janhunen, and Shahab Tasharrofi. Solving QBF Instances with Nested SAT Solvers. In Beyond NP Workshop 2016 at AAAI-16, 2016.

[10] Uwe Bubeck and Hans Kleine Bünning. Bounded Universal Expansion for Preprocessing QBF. In SAT, volume 4501 of LNCS, pages 244–257. Springer, 2007.

[11] Marco Cadoli, Andrea Giovanardi, and Marco Schaerf. An Algorithm to Evaluate Quantified Boolean Formulae. In AAAI, pages 262–267. AAAI Press / The MIT Press, 1998.

[12] Edmund M. Clarke, Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith. Counterexample-guided abstraction refinement for symbolic model checking. J. ACM, 50(5):752–794, 2003.

[13] Martin Davis, George Logemann, and Donald W. Loveland. A Machine Program for Theorem-Proving. Commun. ACM, 5(7):394–397, 1962.

[14] E. Giunchiglia, M. Narizzano, L. Pulina, and A. Tacchella. Quantified Boolean Formulas satisfiability library (QBFLIB), 2005. www.qbflib.org.

[15] Enrico Giunchiglia, Massimo Narizzano, and Armando Tacchella. Learning for Quantified Boolean Logic Satisfiability. In AAAI, pages 649–654. AAAI Press / The MIT Press, 2002.

[16] Enrico Giunchiglia, Massimo Narizzano, and Armando Tacchella. Clause/Term Resolution and Learning in the Evaluation of Quantified Boolean Formulas. JAIR, 26:371–416, 2006.

[17] Marijn Heule, Matti Järvisalo, Florian Lonsing, Martina Seidl, and Armin Biere. Clause Elimination for SAT and QSAT. JAIR, 53:127–168, 2015.

[18] Mikoláš Janota, William Klieber, João Marques-Silva, and Edmund Clarke. Solving QBF with counterexample guided refinement. Artif. Intell., 234:1–25, 2016.

[19] Mikoláš Janota and Joao Marques-Silva. Expansion-based QBF solving versus Q-resolution. Theor. Comput. Sci., 577:25–42, 2015.

[20] Mikoláš Janota and Joao Marques-Silva. Solving QBF by Clause Selection. In IJCAI, pages 325–331. AAAI Press, 2015.

[21] Hans Kleine Bünning and Uwe Bubeck. Theory of Quantified Boolean Formulas. In Handbook of Satisfiability, volume 185 of FAIA, pages 735–760. IOS Press, 2009.

[22] Hans Kleine Bünning, Marek Karpinski, and Andreas Flögel. Resolution for Quantified Boolean Formulas. Inf. Comput., 117(1):12–18, 1995.

[23] William Klieber, Samir Sapra, Sicun Gao, and Edmund M. Clarke. A Non-prenex, Non-clausal QBF Solver with Game-State Learning. In SAT, volume 6175 of LNCS, pages 128–142. Springer, 2010.
[24] Daniel Le Berre, Laurent Simon, and Armando Tacchella. Challenges in the QBF Arena: the SAT’03 Evaluation of QBF Solvers. In SAT, volume 2019 of LNCS, pages 468–485. Springer, 2003.

[25] Reinhold Letz. Lemma and Model Caching in Decision Procedures for Quantified Boolean Formulas. In TABLEAUX, volume 2381 of LNCS, pages 160–175. Springer, 2002.

[26] Florian Lonsing, Fahiem Bacchus, Armin Biere, Uwe Egly, and Martina Seidl. Enhancing Search-Based QBF Solving by Dynamic Blocked Clause Elimination. In LPAR, volume 9450 of LNCS, pages 418–433. Springer, 2015.

[27] Florian Lonsing, Uwe Egly, and Martina Seidl. Q-Resolution with Generalized Axioms. In SAT, volume 9710 of LNCS, pages 435–452. Springer, 2016.

[28] Florian Lonsing, Martina Seidl, and Allen Van Gelder. The QBF Gallery: Behind the scenes. Artif. Intell., 237:92–114, 2016.

[29] Paolo Marin, Massimo Narizzano, Luca Pulina, Armando Tacchella, and Enrico Giunchiglia. Twelve Years of QBF Evaluations: QSAT Is PSPACE-Hard and It Shows. Fundam. Inform., 149(1-2):133–158, 2016.

[30] Albert R. Meyer and Larry J. Stockmeyer. The Equivalence Problem for Regular Expressions with Squaring Requires Exponential Space. In 13th Annual Symposium on Switching and Automata Theory, pages 125–129. IEEE Computer Society, 1972.

[31] Luca Pulina. The Ninth QBF Solvers Evaluation - Preliminary Report. In Proceedings of the 4th International Workshop on Quantified Boolean Formulas QBF 2016, volume 1719 of CEUR Workshop Proceedings, pages 1–13. CEUR-WS.org, 2016.

[32] Luca Pulina and Armando Tacchella. Treewidth: A Useful Marker of Empirical Hardness in Quantified Boolean Logic Encodings. In LPAR, volume 5330 of LNCS, pages 528–542. Springer, 2008.

[33] Markus N. Rabe and Leander Tentrup. CAQE: A Certifying QBF Solver. In FMCA, pages 136–143. IEEE, 2015.

[34] Jussi Rintanen. Asymptotically Optimal Encodings of Conformant Planning in QBF. In AAAI, pages 1045–1050. AAAI Press, 2007.

[35] Thomas J Schaefer. On the Complexity of Some Two-Person Perfect-Information Games. Journal of Computer and System Sciences, 16(2):185–225, 1978.

[36] Christoph Scholl and Florian Pigorsch. The QBF Solver AI{GSolve}. In QBF Workshop, volume 1719 of CEUR Workshop Proceedings, pages 55–62. CEUR-WS.org, 2016.

[37] João P. Marques Silva, Inês Lynce, and Sharad Malik. Conflict-driven clause learning SAT solvers. In Handbook of Satisfiability, volume 185 of FALIA, pages 131–153. IOS Press, 2009.

[38] Larry J. Stockmeyer. The Polynomial-Time Hierarchy. Theor. Comput. Sci., 3(1):1–22, 1976.

[39] Larry J. Stockmeyer and Albert R. Meyer. Word Problems Requiring Exponential Time: Preliminary Report. In STOC, pages 1–9. ACM, 1973.

[40] G. S. Tseitin. On the Complexity of Derivation in Propositional Calculus. Studies in Constructive Mathematics and Mathematical Logic, 1968.

[41] Celia Wrathall. Complete Sets and the Polynomial-Time Hierarchy. Theor. Comput. Sci., 3(1):23–33, 1976.

[42] Lintao Zhang and Sharad Malik. Towards a Symmetric Treatment of Satisfaction and Conflicts in Quantified Boolean Formula Evaluation. In CP, volume 2470 of LNCS, pages 200–215. Springer, 2002.
A Appendix

A.1 Additional Experimental Data

| Solver    | S  | ⊥  | ⊤  | Time |
|-----------|----|----|----|------|
| AIGSolve  | 350| 175| 175| 12K  |
| QSTS-dsb  | 314| 161| 153| 79K  |
| GhostQ    | 303| 154| 149| 101K |
| GhostQ-nd | 264| 144| 120| 176K |
| DepQBF    | 252| 141| 111| 189K |
| CAQE      | 240| 130| 110| 214K |
| DepQBF-n  | 236| 133| 103| 221K |
| QESTO     | 198| 107| 91 | 299K |
| QSTS-nds  | 166| 92 | 74 | 346K |
| RAReQS    | 164| 100| 64 | 350K |

Table 11: Solved instances (S), solved unsatisfiable (⊥) and satisfiable ones (⊤), and total wall clock time including time outs on those 354 original instances from set $S_{825}$ that are solved by Bloqer (related to Table 4).

| Solver    | S  | ⊥  | ⊤  | Time |
|-----------|----|----|----|------|
| GhostQ    | 176| 75 | 101| 171K |
| AIGSolve  | 138| 66 | 72 | 250K |
| QSTS-dsb  | 136| 58 | 78 | 232K |
| RARaQS    | 76 | 43 | 33 | 340K |
| DepQBF    | 70 | 35 | 35 | 351K |
| DepQBF-n  | 67 | 34 | 33 | 354K |
| QESTO     | 66 | 37 | 29 | 359K |
| QSTS-nds  | 51 | 24 | 27 | 386K |
| GhostQ-nd | 49 | 29 | 20 | 389K |
| CAQE      | 43 | 17 | 26 | 397K |

Table 12: Solved instances (S), solved unsatisfiable (⊥) and satisfiable ones (⊤), and total wall clock time including time outs on instances in the class “2–3” of set $S_{402}$ (not preprocessed by Bloqer) as shown in Table 8.
| Solver      | $S$ | $\perp$ | $\top$ | Time |
|------------|-----|---------|--------|------|
| DepQBF     | 78  | 47      | 31     | 125K |
| DepQBF-n   | 74  | 48      | 26     | 134K |
| QSTS-dsb   | 72  | 44      | 28     | 132K |
| GhostQ     | 56  | 31      | 25     | 160K |
| AIGSolve   | 54  | 25      | 29     | 161K |
| QESTO      | 49  | 33      | 16     | 179K |
| CAQE       | 46  | 29      | 17     | 182K |
| RAReQS     | 43  | 33      | 10     | 180K |
| QSTS-nds   | 42  | 28      | 14     | 179K |
| GhostQ-nd  | 31  | 20      | 11     | 205K |

Table 13: Solved instances ($S$), solved unsatisfiable ($\perp$) and satisfiable ones ($\top$), and total wall clock time including time outs on instances in the class “4–” of set $S_{402}$ (not preprocessed by Bloqqer) as shown in Table 8.

| #q | #f | GhostQ | QSTS-dsb | AIGSolve | DepQBF | DepQBF-n | RAReQS | QESTO | QSTS-nds | CAQE | GhostQ-nd |
|----|----|--------|----------|----------|--------|----------|--------|-------|----------|------|-----------|
| 2  | 30 | 16     | 3        | 24       | 8      | 8        | 2      | 4     | 3        | 2    | 0         |
| 3  | 231| 160    | 133      | 114      | 62     | 59       | 74     | 62    | 48       | 41   | 49        |
| 4-6| 25 | 7      | 7        | 14       | 8      | 9        | 7      | 6     | 5        | 7    | 5         |
| 7-9| 31 | 14     | 15       | 16       | 22     | 18       | 12     | 6     | 2        | 9    | 5         |
| 10-15| 24 | 9      | 13       | 6        | 13     | 13       | 7      | 7     | 6        | 7    | 8         |
| 16-20| 26 | 9      | 16       | 7        | 15     | 14       | 5      | 14    | 15       | 13   | 2         |
| 21-| 35 | 18     | 21       | 11       | 20     | 20       | 12     | 16    | 14       | 10   | 11        |
| 2-3| 261| 176    | 136      | 138      | 70     | 67       | 76     | 66    | 51       | 43   | 49        |
| 4-| 141| 56     | 72       | 54       | 78     | 74       | 43     | 49    | 42       | 46   | 31        |

Table 14: Complete version of Table 8 (set $S_{402}$ not preprocessed by Bloqqer, cf. Table 6).
Table 15: Solved instances ($S$), solved unsatisfiable ($\bot$) and satisfiable ones ($\top$), and total wall clock time including time outs on instances in the class “2–3” of set $S'_402$ (preprocessed by Bloqger) as shown in Table 9.

| Solver    | $S$ | $\bot$ | $\top$ | Time |
|-----------|-----|--------|--------|------|
| RAReQS    | 157 | 79     | 78     | 227K |
| QESTO     | 138 | 66     | 72     | 255K |
| QSTS-dsb  | 136 | 62     | 74     | 255K |
| CAQE      | 118 | 49     | 69     | 298K |
| QSTS-ndsb | 117 | 49     | 68     | 297K |
| GhostQ    | 111 | 46     | 65     | 304K |
| DepQBF    | 106 | 42     | 64     | 310K |
| DepQBF-n  | 105 | 42     | 63     | 312K |
| AIGSolve  | 102 | 49     | 53     | 313K |
| GhostQ-nd | 65  | 33     | 32     | 374K |

Table 16: Solved instances ($S$), solved unsatisfiable ($\bot$) and satisfiable ones ($\top$), and total wall clock time including time outs on instances in the class “4–” of set $S'_402$ (preprocessed by Bloqger) as shown in Table 9.

| Solver    | $S$ | $\bot$ | $\top$ | Time |
|-----------|-----|--------|--------|------|
| DepQBF-n  | 79  | 50     | 29     | 107K |
| DepQBF    | 77  | 49     | 28     | 107K |
| QSTS-dsb  | 75  | 50     | 25     | 107K |
| QESTO     | 69  | 45     | 24     | 120K |
| CAQE      | 64  | 42     | 22     | 136K |
| QSTS-ndsb | 62  | 43     | 19     | 127K |
| RAReQS    | 62  | 45     | 17     | 131K |
| AIGSolve  | 51  | 27     | 24     | 151K |
| GhostQ    | 46  | 26     | 20     | 162K |
| GhostQ-nd | 36  | 21     | 15     | 178K |
Table 17: Complete version of Table 9 (set $S'_{102}$ preprocessed by Bloqqer, cf. Table 7).

| #q | #f | VBS | GhostQ | QSTSDsbl | GhostSolve | DepQBF | DepQBF-n | CAQE | QSTSDsbl | GhostQ | AIGSolve | GhostQ-n |
|----|----|-----|--------|----------|------------|--------|----------|------|----------|--------|----------|----------|
| 2  | 34 | 14  | 11     | 14       | 13         | 13     | 11       | 11   | 5        | 11     | 4        |          |
| 3  | 236| 143 | 125    | 124      | 92         | 93     | 107      | 106  | 106      | 91     | 61       |          |
| 4-6| 24 | 10  | 8      | 10       | 11         | 10     | 10       | 10   | 8        | 14     | 8        |          |
| 7-9| 31 | 21  | 20     | 21       | 21         | 23     | 18       | 13   | 8        | 17     | 5        |          |
| 10-15| 28 | 14  | 16     | 10       | 19         | 19     | 11       | 9    | 11       | 5      | 10       |          |
| 16-20| 30 | 9   | 21     | 18       | 20         | 20     | 17       | 17   | 20       | 13     | 10       | 7        |
| 21-| 19 | 10  | 8      | 10       | 8          | 8      | 10       | 6    | 5        | 6      |          |          |
| 2-3| 270| 157 | 136    | 138      | 105        | 106    | 118      | 117  | 111      | 102    | 65       |          |
| 4- | 132| 62  | 75     | 69       | 79         | 77     | 62       | 46   | 51       | 36     |          |          |

Table 18: Number of instances from set $S'_{102}$ (not preprocessed by Bloqqer, cf. Table 8) solved by the virtual best solver (VBS) in classes by number of qblocks (#q) and number of formulas (#f), and relative contribution (percentage) of each solver to instances solved by the VBS. See also Table 10.