The Ninth QBF Solvers Evaluation – Preliminary Report

Luca Pulina

POLCOMING, Università di Sassari, Italy
lpulina@uniss.it

Abstract. In this paper we report about the 2016 competitive evaluation of quantified Boolean formulas (QBFs) solvers (QBFEVAL'16), the last in a series of events established with the aim of assessing the advancements in reasoning about QBFs. Aim of this preliminary report is to show at a glance design and results of QBFEVAL'16.

1 Introduction

Competitive events in the field of Boolean reasoning have influenced related research agendas and shaped the course of tool developments. Nowadays, organized evaluations are popular for several subfields of Boolean reasoning, including propositional satisfiability (SAT) [1, 2], quantified Boolean formula (QBF), and satisfiability modulo theory (SMT) solving [3].

The 2016 competitive evaluation of solvers for QBFs (QBFEVAL'16) is the last in a series of events established with the aim of assessing the advancements in the field of QBF reasoning and related research. With respect to the previous event (the QBF Gallery 2014 [4]) we witnessed a noticeable increase in the number of submitted systems (44), observing a growth in the variety of techniques that are used by the solvers. This fact confirms the vitality of the research on QBF reasoning tools.

The paper is structured as follows. In Section 2 we briefly describe the design of QBFEVAL'16, the systems that participated in the evaluation, and the instances that we used to construct the testset. Section 3 announces the competition winners and presents the results of QBFEVAL'16 arranged solver-wise.

2 The setup: tracks, solvers, and formulas

QBFEVAL'16 is composed of 8 different tracks described in the following

1. Prenex CNF (PCNF): it is comprised of prenex CNF formulas obtained by encoding various automated reasoning tasks into QBF. Submitted solvers must read instances must using the QDIMACS 1.1[1] input format.

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

2. Prenex non-CNF (PNCNF) is a track devoted to evaluate solvers supporting the QCIR input format [5] for prenex non-CNF instances.
3. 2QBF is a track in which instances in QDIMACS 1.1 format with a single alternation ∀∃ are evaluated.
4. Incremental Solvers: a track aimed at the evaluation of incremental QBF solvers. This track has been canceled because only one solver has been submitted.
5. Evaluate & Certify (EC): aim of this non-competitive track is to assess the current state of the art about the certification of QBFs in QDIMACS 1.1.
6. Solver Portfolio (SP): in this track it has been evaluated performance of systems portfolios taking as input QBFs in QDIMACS 1.1 format.
7. Parallel QBF Solvers (PS): aim of this non-competitive track is to assess the current state of the art about parallel QBF solvers. In this track, solvers partecipated accepting formulas in QDIMACS and/or in QCIR. The number of cores available for each solver is 4.
8. Random QBFs (RQBF) is a track devoted to the evaluation of randomly generated QBFs instances in QDIMACS 1.1.

Table 2 summarizes the solvers submitted to QBFEVAL'16. The salient features of the participants are briefly described in the following.

AIGSolve [6] uses And-Inverter Graphs (AIGs) as the main data structure, and AIG-based operations to reason about the input formula. The solver includes preliminary phases devoted to simplification, structure extraction and early quantification of the input formula.

Aqua is a search-based QBF solver for prenex conjunctive normal form (PCNF) formulas. Literal propagation is performed through unit, pure, and don't care literal detection using lazy data structures [7]. For backtracking, a conflict and solution driven constraint learning approach [8, 9] is used. A restart strategy [10] and phase saving [11] are also implemented. Aqua comes in three versions, namely

- Aqua-F3V, with common first UIP (F-UIP) [12] learning, 3 literals watching, and VSIDS decision heuristic [13];
- Aqua-S2V, with first semantic UIP (S-UIP) learning, 2 literals watching, and VSIDS decision heuristic;
- Aqua-S3O, with S-UIP learning, 3 literals watching, and OCCS decision heuristic [13].

Finally, the solver is coupled with the QBF preprocessor sQueezeBF [14], which is given a timeout of 100 seconds.

aqme [15] is a multi-engine solver, i.e., a tool using machine learning techniques to select among its reasoning engines the one which is more likely to yield optimal results. The reasoning engines of aqme are a subset of those submitted to QBFEVAL'06, while engine selection is performed according to the adaptive strategy described in [15]. It has been also submitted a prototype version coupled with sQueezeBF (SqueezeBF+aqme).

areqs, an implementation of the 2QBF algorithm described in [16].

Solver	Track	Author(s)
AIGSolve	PCNF	C. Scholl, F. Pigorsch
Aqua-F3V, Aqua-S2V, Aqua-S3O	PCNF, RQBF	P. Marin
aqme*, SqueezeBF+aqme*	SP	L. Pulina, A. Tacchella
areqs	2QBF	M. Janota
aspq*	2QBF	G. Amendola, C. Dodaro, F. Ricca
qesto	PCNF, RQBF	M. Janota
qestos, rareqs	PCNF, 2QBF, RQBF	M. Janota
cadet	2QBF	M. N. Rabe
caqe-minisat, caqe-picosat	PCNF, RQBF	L. Tentrup, M. N. Rabe
caqe-minisat-cert, caqe-picosat-cert	EC	L. Tentrup, M. N. Rabe
caqe-minisat-par, caqe-picosat-par	PS	L. Tentrup, M. N. Rabe
caqe-portfolio	SP	L. Tentrup, M. N. Rabe
CheQ*	EC	M. Narizzano, C. Peschiera, L. Pulina, A. Tacchella
depqbf-v1, depqbf-v2, depqbf-v3	PCNF, 2QBF, RQBF	F. Lonsing
depqbf-cert-v1, depqbf-cert-v2	EC	F. Lonsing
dynQBF	2QBF	G. Charwat, S. Woltran
ghostQ-cegar, ghostQ-plain	PCNF, PNCNF, 2QBF	W. Klieber
hiqqer1, hiqqer3, hiqqer1LDSQ*	PCNF, 2QBF, RQBF	A. Van Gelder, S. Wood
hiqqerFork	PS	A. Van Gelder
HordeQBF	PS	T. Balyo, F. Lonsing
iProver-QBF, iProver-QBF-bloqqer	PCNF, 2QBF, RQBF	K. Korovin
MPIDepQBF	PS	C. Jordan, L. Kaiser, F. Lonsing, M. Seidl
par-pd-depqbf	PS	U. Egly, F. Lonsing and J. Oetsch
quabs-minisat, quabs-picosat	PNCNF	L. Tentrup
qsts, xb-qsts	PCNF, PNCNF, 2QBF, RQBF	B. Bogaerts, T. Janhunen, S. Tasharrofi
rareqs-nn	PNCNF	M. Janota
StruQS*, SqueezeBF+StruQS*	PCNF, 2QBF, RQBF	L. Pulina, A. Tacchella
xb-bid-qsts	PCNF, PNCNF, 2QBF, RQBF	B. Bogaerts, J. Devriendt, T. Janhunen, S. Tasharrofi

Table 1. The QBFEVAL'16 systems. The table is structured as follows. The first column ("Solver") reports the name of the solver, the second column ("Track") indicates the track in which the given solver is involved, while the last column ("Author(s)") reports solvers' authors. Systems denoted with a "*" are *hors-concours*.

aspq is a proof of concept of a 2QBF solver based on ASP solvers. The input formula is preprocessed with bloqqer [17], and then transformed in a ground ASP program according to the classical Eiter-Gottlob encoding of 2QBF in ASP [18], so that it can be evaluated by using an ASP solver.

cadet [19] is a solver for 2QBF formulas based on the incremental construction of the Skolem functions aimed to prove the satisfiability of the formula.

caqe [20] is a CEGAR-based algorithm for QBF. The algorithm builds on a decomposition of QBFs into a sequence of propositional formulas called clausal abstraction. Each of the propositional formulas contains the variables of just one quantifier level and additional variables describing the interaction with adjacent quantifier levels. Two versions of caqe have been submitted, namely CAQE-MINISAT and CAQE-PICOSAT, using MINISAT [21] and PICOSAT [22] as SAT solver, respectively. Both versions use BLOQQER as preprocessor. These system are also involved in the EC track – with their version for certification, namely CAQE-MINISAT-CERT and CAQE-PICOSAT-CERT –, SP track (CAQE-PORTFOLIO), and PS track (CAQE-MINISAT-PAR, CAQE-PICOSAT-PAR).

CHEQ [23] is a suite for QBF certification. It is comprised of QUBE-CERT – an extension of QUBE3.1 able to output certificates – and CHECKER, a tool aimed at check QUBE-CERT output.

DEPQBF [24] is a search-based solver with conflict-driven clause and solution-driven cube learning (QCDCL) [8, 25, 26]. The submitted variants of DEPQBF are based on version 5.0 [27], with an advanced technique for cube learning by tightly integrating blocked clause elimination [17, 28] into QCDCL. The submitted variants of DEPQBF, namely DEPQBF-V1, DEPQBF-V2, and DEPQBF-V3, extend version 5.0 by additionally integrating the SAT solver PICOSAT and the QBF preprocessor BLOQQER. PICOSAT and BLOQQER are applied dynamically during the run of QCDCL to derive clauses and cubes. Two versions of DEPQBF have been also submitted for the EC track, namely DEPQBF-CERT-V1, DEPQBF-CERT-V2. The former intended to certify only unsatisfiable QBFs, while the latter can certificate both satisfiable and unsatisfiable QBFs. Both versions leverage on the QBFCERT [29] framework.

DYNQBF [30] is a structure-aware QBF solver. It splits the QBF instance into sub-problems by constructing a tree decomposition. The QBF is then solved by dynamic programming over the tree decomposition.

GHOSTQ [31] is a non-prenex DPLL-based solver which makes use of auxiliary variables to force necessary assignments, i.e., to force a value to an existential (resp. universal) variable if the opposite value directly makes the formula evaluate to false (resp. true). Additionally, it features a counterexample guided abstraction refinement (CEGAR) based learning to further prune the search space when the last decision literal is existential (resp. universal) and a conflict (resp. solution) is detected. Two versions of GHOSTQ have been submitted, namely GHOSTQ-CEGAR and GHOSTQ-PLAIN.

HIQQER As reported in [4], the QBF solver HIQQER consists of a `csh` script that invokes two preprocessors, PLODDER and EQXBF, then passes the resulting file to the complete solver STEPQBF. Three versions have been submitted to the evaluation, namely HIQQER1, HIQQER3, and HIQQER1LDSQ. It has been also submitted a parallel version (HIQQERFORK), running in the PS track.

HORDEQBF [32] is an MPI-based parallel portfolio solver with clause and cube sharing. It is based on the framework of HORDESAT [33], a modular and massively parallel SAT solver. The authors integrated DEPQBF 5.0 in HORDESAT to obtain HORDEQBF.

IPROVER [34] is a general purpose theorem prover for first-order logic based an instantiation calculus Inst-Gen. It incorporates a QBF solving mode which is based on a translation of QBF into the effectively propositional fragment of first-order logic (EPR). The basic translation follows [35], and it is also implemented a dedicated Skolemization procedure with several optimization.

Two versions of iProver have been submitted, namely iProver-QBF and iProver-QBF-bloqqer (it uses bloqqer for preprocessing).

MPIDepQBF [36] dynamically creates budgeted subproblems by setting outermost variables. The subproblems are solved using depqbf and its support for assumptions. The changes to the version described in [36] entail updating the version of DepQBF used to version 5.0.

par-pd-depqbf. In this solver, the approach to solve quantified circuits in prenex-normal form relies on running two instances of a QBF solver on a primal and a dual version of the problem encoding in parallel – as described in [37]. par-pd-depqbf makes use of preprocessing by means of bloqqer, and it uses depqbf as back-end PCNF QBF solver.

qesto is an implementation of the QCNF algorithm presented in [38]. The submitted version includes bloqqer as preprocessor.

qestos is a prototype based on the ideas described in [39]. The submitted version includes bloqqer as preprocessor.

quabs [40] is a certifying QBF solver based on a CEGAR-based abstraction algorithm for Prenex non-CNF formulas in QCIR format. Two different versions have been submitted, namely quabs-minisat and quabs-picosat, using the SAT solvers minisat and picosat, respectively.

qsts is based on nested SAT solving and theory transformations. The main tools utilized for translation are:

– sat-to-sat [41], a (non-)prenex (non-)CNF QBF solver that is based on nested SAT solving, and it is able to do early propagation of information between nested solvers.
– qbf2sts [42], a translator from QDIMACS/QCIR input format to sat-to-sat input format with the ability to reverse engineer circuits and apply several theory transformations to simplify the representation of QBF formulas.

Three versions of qsts have been submitted to QBFEVAL'16, namely the plain version (qsts), one version using both qxbf [43] and bloqqer preprocessors (xb-qsts), and xb-bid-qsts, that extends xb-qsts with breakid [44], a SAT symmetry breaker that has been modified to detect a (limited) class of symmetries in QBF instances.

rareqs [45] is an abstraction based solver, which performs a kind of resolution and expansion procedure but in a depth-first way, i.e., by expanding first only one value of a variable, and learns abstractions of the local partial solutions to refine the global solution. The submitted version includes bloqqer as preprocessor. It has been also submitted a version for the PNCNF track (rareqs-nn [46]).

StruQS [47], a QBF solver that implements a dynamic combination of search – with solution- and conflict-backjumping – and variable-elimination. The key point in this approach is to implicitly leverage graph abstractions of QBFs to yield structural features which support an effective decision between search and variable elimination. It has been also submitted a version coupled with sQueezeBF (SqueezeBF+StruQS).

Further details about the systems can be found in the descriptions submitted by their authors and made available in QBFLIB[1].

Concerning formulas and generators, there were three submissions:

– `Generalized Tic-Tac-Toe` [48]: 180 formulas in QDIMACS 1.1, submitted by Diptarama, C. Jordan, and A. Shinohara.
– `Random QBFs`: 60 formulas in QDIMACS 1.1 and QCIR format, submitted by F. Ricca, G. Amendola and M. Truszczynski. Details are available at `http://www.qbflib.org`
– QBF generator for formulas related to the rewriting algorithm described in [49], implemented and submitted by T. Peitl.

About the testset of QBFEVAL'16[2], it has been composed considering the specificities of the tracks previously described. Of course, the selection must satisfy two competing requirements: (*i*) obtaining meaningful data and (*ii*) completing the evaluation in reasonable time. In order to do that, we prepare four different datasets, namely:

– **Dataset 1** (D1), for Prenex CNF, Evaluate & Certify, Solver Portfolio, and Parallel QBF Solvers Tracks.
– **Dataset 2** (D2), for the Prenex non-CNF Track.
– **Dataset 3** (D3), for the 2QBF Track.
– **Dataset 4** (D4), for Random QBFs Track.

D1 is composed of prenex CNF fixed structure formulas in QDIMACS 1.1 format. It is composed of (up to) 10 formulas selected from each family on QBFLIB, and is has been extended adding 10 formulas from `Generalized Tic-Tac-Toe`, other 10 generated by Peitl's generator, for a total amount of 825 QBFs.

D2 is composed of prenex non-CNF fixed structure formulas in QCIR (QBF-Gallery 14) format. It is composed of the non-prenex non-CNF dataset of QBF-EVAL'10 [50] (478 formulas converted to QCIR and prenexed), extended with the QCIR conversion of 50% of D1 (412 formulas), for a total amount of 890 formulas. Regarding D3, it is composed of prenex CNF $\forall\exists$ fixed structure formulas in QDIMACS 1.1 format. (Up to) 50 formulas have been randomly selected from each family of QBFLIB containing QBFs with $\forall\exists$ prefix, for a total amount of 305 formulas.

Finally, D4 is composed of prenex CNF probabilistic structure formulas in QDIMACS 1.1 format. D4 includes 580 formulas, 320 of which have been selected from QBFLIB, 200 have been generated by the tool BlocksQBF [51] – based on the model described in [52] –, and all submitted `Random QBFs` (60).

Table 2 summarizes the total amount of solvers and formulas involved in QBFEVAL'16.

Finally, concerning the infrastructure, with the exception of solvers submitted to the PS track, systems ran as a single process (or a batch of processes). The

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

[2] All datasets used in QBFEVAL'16 can be downloaded from `http://www.qbflib.org/eval16.html`.

Track	# Systems	# Formulas
Prenex CNF	24	825
Prenex non-CNF	8	890
2QBF	21	305
Evaluate & Certify	5	825
Solver Portfolio	3	825
Parallel QBF Solvers	6	825
Random QBFs	21	580

Table 2. QBFEVAL'16 at a glance.

CPU time limit was set to 600 seconds, while the memory limit was 4GB. All tracks excepting PS ran on the StarExec [53] cluster, while PS ran on a cluster of Dell Workstations with double Intel Xeon E3-1245 PCs at 3.30 GHz quad core processor, equipped with 64 bit Ubuntu 12.04.

3 Results at a glance

In Tables 3–9 section we report for each track a solver-centric view of the results of QBFEVAL'16. Details are available at the QBFEVAL Web portal[1] [54].

Considering the PCNF Track, looking at Table 3 we can see that RAREQS is the winner of the track, followed by XB-QSTS and DEPQBF-V2, which rank second and third, respectively. In QBFEVAL'16 there are also provided awards for distinguished contribution to the state-of-the-art (SOTA) solver, i.e., the ideal solver that always fares the best time among all the participants. In the case of PCNF Track, best SOTA contributors are AIGSOLVE and QESTO. Finally, we discarded from Table 3 the performance of HIQQER1, HIQQER3, QSTS, SQUEEZEBF+STRUQS, and XB-BID-QSTS because some discrepancies have been reported for these solvers. We refer the reader to the QBFEVAL Web portal for details.

Looking at the results of the PNCNF Track (Table 4), we report that the two version of GHOSTQ are the best performing systems. GHOSTQ-CEGAR and GHOSTQ-PLAIN rank first and second, respectively, while QUABS-PICOSAT ranks third; QUABS-MINISAT was the best SOTA contributor. Also in this case, we do not show in the table the system reporting discrepancies (all three versions of QSTS).

Table 5 shows the results of the 2QBF Track. The winner of the track is AREQS, closely followed by RAREQS, while DEPQBF-V2 ranks third. In this track we report discrepancies for QSTS and XB-BID-QSTS.

In Tables 6 and 7 we report the results related to EC and SP tracks, respectively. The best performing in EC was DEPQBF-CERT-V2, while in the SP track CAQE-PORTFOLIO ranks first. We have not assigned awards in both tracks be-

[1] `http://www.qbfeval.org`

Solver	Total		True		False	
	#	Time	#	Time	#	Time
RAREQS	640	14166.80	309	4598.66	331	9568.11
XB-QSTS	613	15296.70	299	5212.69	314	10084.00
DEPQBF-V2	603	14076.90	297	6256.31	306	7820.60
CAQE-PICOSAT	590	17178.80	294	6272.92	296	10905.90
AIGSOLVE	589	15981.30	293	7833.21	296	8148.14
GHOSTQ-CEGAR	585	14538.80	298	7739.39	287	6799.38
QESTO	582	15552.80	285	4394.35	297	11158.50
CAQE-MINISAT	576	15219.10	292	4878.17	284	10340.90
HIQQER1LDSQ*	574	10951.50	288	6319.74	286	4631.80
GHOSTQ-PLAIN	568	13727.80	282	7000.20	286	6727.60
QESTOS	527	4356.04	252	1848.00	275	2508.04
DEPQBF-V3	527	16186.70	261	8995.82	266	7190.88
AQUA-S2V	484	7869.78	229	3290.43	255	4579.35
AQUA-F3V	482	7947.80	229	3753.27	253	4194.53
AQUA-S3O	479	6774.68	225	3036.41	254	3738.27
DEPQBF-V1	456	9999.76	201	4319.97	255	5679.79
STRUQS*	358	12825.20	175	4595.09	183	8230.08
IPROVER-QBF	348	12922.00	158	5385.42	190	7536.62
IPROVER-QBF-BLOQQER	324	9369.12	243	3955.44	81	5413.68

Table 3. Results of PCNF Track. For each solver, the table shows the number of instances solved ("#") and the total CPU time (in seconds) spent to solve them ("Time"). Total number of formulas solved ("Total") is also split into true and false formulas ("True" and "False", respectively). A dash means that a solver did not solve any instance in the related group. Solvers are sorted according to the number of instances solved, and, in case of a tie, according to CPU time. Finally, systems denoted with a "*" participate *hors-concours*

Solver	Total		True		False	
	#	Time	#	Time	#	Time
GHOSTQ-CEGAR	524	9009.13	231	5391.70	293	3617.43
GHOSTQ-PLAIN	521	7739.63	229	2802.33	292	4937.30
QUABS-PICOSAT	509	4784.62	223	2047.07	286	2737.55
QUABS-MINISAT	503	4287.17	217	2608.65	286	1678.52
RAREQS-NN	403	7427.47	174	3161.98	229	4265.49

Table 4. Results of PNCNF Track. The table is organized as Table 3.

cause the former was non-competitive, while in the latter 2 out of 3 participating systems where hors-concours.

Considering the non-competitive PS Track, the best performing system was PAR-PD-DEPQBF, followed by HIQQERFORK and CAQE-PICOSAT-PAR. Finally, AQUA is the winner of the RQBF Track, where AQUA-S2V, AQUA-F3V, and

Solver	Total		True		False	
	#	Time	#	Time	#	Time
AREQS	235	2963.33	179	2136.52	56	826.81
RAREQS	232	5287.58	156	2084.94	76	3202.64
DEPQBF-V2	223	5135.23	142	1553.21	81	3582.02
XB-QSTS	206	5581.42	154	3354.41	52	2227.01
ASPQ*	188	741.09	141	275.41	47	465.68
HIQQER3	185	3236.84	150	2235.98	35	1000.86
QESTOS	184	3487.24	135	1194.36	49	2292.88
HIQQER1LDSQ*	183	2663.74	147	2195.58	36	468.16
HIQQER1	183	2703.59	147	2232.26	36	471.33
CADET	169	790.78	120	512.95	49	277.83
GHOSTQ-CEGAR	155	8135.25	108	6031.48	47	2103.77
DEPQBF-V3	138	4901.65	97	1799.51	41	3102.14
DEPQBF-V1	133	5466.70	68	1262.27	65	4204.43
IPROVER-QBF-BLOQQER	124	188.14	122	78.66	2	109.48
StruQS*	100	933.77	73	483.18	27	450.59
SqueezeBF+StruQS*	100	1169.84	73	720.19	27	449.65
GHOSTQ-PLAIN	87	7545.74	40	4115.46	47	3430.28
DYNQBF	72	489.44	70	489.29	2	0.15
IPROVER	32	1249.98	30	1142.63	2	107.35

Table 5. Results of 2QBF Track. The table is organized as Table 3.

Solver	Total		True		False	
	#	Time	#	Time	#	Time
DEPQBF-CERT-V2	309	4732.51	115	1981.28	194	2751.23
CAQE-PICOSAT-CERT	268	7598.09	107	2382.27	161	5215.82
CAQE-MINISAT-CERT	236	6831.53	90	3099.06	146	3732.47
DEPQBF-CERT-V1	217	2760.83	–	–	217	2760.83
CheQ*	217	6188.85	118	3523.34	99	2665.51

Table 6. Results of EC Track. The table is organized as Table 3.

Solver	Total		True		False	
	#	Time	#	Time	#	Time
CAQE-PORTFOLIO	580	8824.50	295	3305.74	285	5518.76
AQME*	530	9657.69	239	5351.65	291	4306.04
SqueezeBF+AQME*	473	9599.09	220	4863.89	253	4735.20

Table 7. Results of SP Track. The table is organized as Table 3.

Aqua-S3O rank first, second, and third, respectively. Additionally, RAREQS has been awarded as the best SOTA contributor.

Solver	Total		True		False	
	#	Time	#	Time	#	Time
PAR-PD-DEPQBF	606	12269.10	305	2946.11	301	9323.03
HIQQERFORK	598	14624.00	300	7766.69	298	6857.33
CAQE-PICOSAT-PAR	585	13337.40	289	3560.56	296	9776.87
CAQE-MINISAT-PAR	570	12304.00	293	4941.32	277	7362.72
HORDEQBF	443	8434.86	191	3338.99	252	5095.87

Table 8. Results of PS Track. The table is organized as Table 3.

Solver	Total		True		False	
	#	Time	#	Time	#	Time
AQUA-S2V	306	10976.20	127	4952.36	179	6023.87
AQUA-F3V	306	11419.70	127	5031.04	179	6388.62
AQUA-S3O	300	10360.10	127	5085.79	173	5274.28
CAQE-PICOSAT	298	7324.49	128	2579.82	170	4744.67
RAREQS	295	4305.78	127	1699.99	168	2605.79
XB-QSTS	294	7963.03	132	2295.17	162	5667.86
QESTO	291	9398.91	131	3720.64	160	5678.27
DEPQBF-V2	287	8977.03	113	1613.30	174	7363.73
HIQQER1	267	6712.26	111	1709.68	156	5002.58
HIQQER1LDSQ*	267	7013.05	111	1681.42	156	5331.63
HIQQER3	261	4763.45	111	1560.89	150	3202.56
DEPQBF-V1	257	7572.00	104	2130.39	153	5441.61
DEPQBF-V3	257	7772.75	108	2726.88	149	5045.87
QESTOS	246	6904.90	107	2340.14	139	4564.76
QSTS	239	5231.98	106	2339.49	133	2892.49
CAQE-MINISAT	212	7360.30	107	3758.12	105	3602.18
IPROVER-QBF-BLOQQER	59	264.61	58	75.49	1	189.12
SQUEEZEBF+STRUQS*	41	5851.10	27	3814.07	14	2037.03
STRUQS*	35	3623.79	21	1602.65	14	2021.14
IPROVER-QBF	25	4341.95	25	4341.95	–	–

Table 9. Results of RQBF Track. The table is organized as Table 3.

In conclusion, the final balance of QBFEVAL'16 can be summarized as follows:

- 44 systems (7 hors-concours) submitted by 20 different teams participated.
- 300 new formulas and 1 benchmark generator have been submitted.
- State-of-the-art solvers for each track have been identified.

All the information contained in this paper can be retrieved at the QBFEVAL'16 web portal, to which we refer the reader for details.

Acknowledgments The author would like to thank all the participants to QBFEVAL'16 for submitting their work. The author wishes also to thank the judges of QBFEVAL'16, namely Hubie Chen, Martina Seidl, and Christoph Wintersteiger. Last but not least, the author would like to especially thank Aaron Stump and the StarExec team.

References

1. Balyo, T., Biere, A., Iser, M., Sinz, C.: SAT Race 2015. Artificial Intelligence **241** (2016) 45–65
2. Balyo, T., Heule, M.J., Järvisalo, M.: SAT Competition 2016. (2016)
3. Cok, D.R., Déharbe, D., Weber, T.: The 2014 SMT competition. Journal on Satisfiability, Boolean Modeling and Computation **9** (2014) 207–242
4. Janota, M., Jordan, C., Klieber, W., Lonsing, F., Seidl, M., Van Gelder, A.: The QBFGallery 2014: The QBF competition at the FLoC olympic games. Journal on Satisfiability, Boolean Modeling and Computation **9** (2016) 187–206
5. Jordan, C., Seidl, M.: QCIR-G14: A non-prenex non-CNF format for quantified boolean formulas. Technical report, Technical report (2014)
6. Pigorsch, F., Scholl, C.: Exploiting structure in an AIG based QBF solver. In: Design, Automation and Test in Europe (DATE 2009), IEEE (2009) 1596–1601
7. Gent, I., Giunchiglia, E., Narizzano, M., Rowley, A., Tacchella, A.: Watched Data Structures for QBF Solvers. In: Sixth International Conference on Theory and Applications of Satisfiability Testing (SAT 2003). Volume 2919 of Lecture Notes in Computer Science., Springer Verlag (2003)
8. Giunchiglia, E., Narizzano, M., Tacchella, A.: Clause-Term Resolution and Learning in Quantified Boolean Logic Satisfiability. Artificial Intelligence Research **26** (2006) 371–416
9. Marin, P., Giunchiglia, E., Narizzano, M.: Conflict and solution driven constraint learning in QBF. In: Doctoral Program of Constraint Programming Conference. Volume 2010. (2010)
10. Gomes, C.P., Selman, B., Kautz, H., et al.: Boosting combinatorial search through randomization. AAAI/IAAI **98** (1998) 431–437
11. Pipatsrisawat, K., Darwiche, A.: A lightweight component caching scheme for satisfiability solvers. In: International conference on theory and applications of satisfiability testing, Springer (2007) 294–299
12. Zhang, L., Malik, S.: Conflict driven learning in a quantified boolean satisfiability solver. In: Proceedings of International Conference on Computer Aided Design (ICCAD'02). (2002)
13. Zhang, L., Madigan, C.F., Moskewicz, M.W., Malik, S.: Efficient conflict driven learning in a Boolean satisfiability solver. In: International Conference on Computer-Aided Design (ICCAD'01). (2001) 279–285
14. Giunchiglia, E., Marin, P., Narizzano, M.: sQueezeBF: An effective preprocessor for QBFs based on equivalence reasoning. In: International Conference on Theory and Applications of Satisfiability Testing, Springer (2010) 85–98
15. Pulina, L., Tacchella, A.: A self-adaptive multi-engine solver for quantified Boolean formulas. Constraints **14**(1) (2009) 80–116
16. Janota, M., Marques-Silva, J.: Abstraction-based algorithm for 2QBF. In: International Conference on Theory and Applications of Satisfiability Testing, Springer (2011) 230–244
17. Biere, A., Lonsing, F., Seidl, M.: Blocked clause elimination for QBF. In: International Conference on Automated Deduction, Springer (2011) 101–115

18. Eiter, T., Gottlob, G.: On the computational cost of disjunctive logic programming: Propositional case. Annals of Mathematics and Artificial Intelligence **15**(3-4) (1995) 289–323
19. Rabe, M.N., Seshia, S.A.: Incremental determinization. In: International Conference on Theory and Applications of Satisfiability Testing, Springer (2016) 375–392
20. Rabe, M.N., Tentrup, L.: CAQE: A Certifying QBF Solver. In: Proceedings of the 15th Conference on Formal Methods in Computer-aided Design (FMCAD'15). (2015) 136–143
21. Eén, N., Sörensson, N.: An extensible SAT-solver. In: International conference on theory and applications of satisfiability testing, Springer (2003) 502–518
22. Biere, A.: PicoSAT essentials. Journal on Satisfiability, Boolean Modeling and Computation **4** (2008) 75–97
23. Narizzano, M., Peschiera, C., Pulina, L., Tacchella, A.: Evaluating and certifying QBFs: A comparison of state-of-the-art tools. AI communications **22**(4) (2009) 191–210
24. Lonsing, F., Biere, A.: DepQBF: A Dependency-Aware QBF Solver. JSAT **7**(2-3) (2010) 71–76
25. Letz, R.: Lemma and Model Caching in Decision Procedures for Quantified Boolean Formulas. In: Proceedings of Tableaux 2002. LNAI 2381, Springer (2002) 160–175
26. Zhang, L., Malik, S.: Towards a symmetric treatment of satisfaction and conflicts in quantified boolean formula evaluation. In: Proceedings of the Eighth International Conference on Principles and Practice of Constraint Programming. (2002) 200–215
27. Lonsing, F., Bacchus, F., Biere, A., Egly, U., Seidl, M.: Enhancing search-based QBF solving by dynamic blocked clause elimination. In: Logic for Programming, Artificial Intelligence, and Reasoning, Springer (2015) 418–433
28. Heule, M., Järvisalo, M., Lonsing, F., Seidl, M., Biere, A.: Clause elimination for SAT and QSAT. JAIR **53** (2015) 127–168
29. Niemetz, A., Preiner, M., Lonsing, F., Seidl, M., Biere, A.: Resolution-based certificate extraction for qbf. In: International Conference on Theory and Applications of Satisfiability Testing, Springer (2012) 430–435
30. Gunther Charwat and Stefan Woltran: dynQBF: A Dynamic Programming-based QBF Solver (2016) `http://dbai.tuwien.ac.at/proj/decodyn/dynqbf`.
31. Klieber, W., Sapra, S., Gao, S., Clarke, E.: A non-prenex, non-clausal QBF solver with game-state learning. In: Theory and Applications of Satisfiability Testing–SAT 2010. Springer (2010) 128–142
32. Balyo, T., Lonsing, F.: HordeQBF: A Modular and Massively Parallel QBF Solver. In: International Conference on Theory and Applications of Satisfiability Testing, Springer (2016) 531–538
33. Balyo, T., Sanders, P., Sinz, C.: HordeSat: A massively parallel portfolio SAT solver. In: International Conference on Theory and Applications of Satisfiability Testing, Springer (2015) 156–172
34. Korovin, K.: Inst-gen – a modular approach to instantiation-based automated reasoning. In: Programming Logics. Springer (2013) 239–270
35. Seidl, M., Lonsing, F., Biere, A.: qbf2epr: A Tool for Generating EPR Formulas from QBF. PAAR@ IJCAR **21** (2012) 139–148
36. Jordan, C., Kaiser, L., Lonsing, F., Seidl, M.: MPIDepQBF: towards parallel QBF solving without knowledge sharing. In: International Conference on Theory and Applications of Satisfiability Testing, Springer (2014) 430–437
37. Van Gelder, A.: Primal and dual encoding from applications into quantified boolean formulas. In: International Conference on Principles and Practice of Constraint Programming, Springer (2013) 694–707

38. Janota, M., Marques-Silva, J.: Solving QBF by clause selection. In: Proceedings of IJCAI. (2015) 325–331
39. Bjørner, N., Janota, M., Klieber, W.: On conflicts and strategies in QBF. In: 20th International Conferences on Logic for Programming, Artificial Intelligence and Reasoning - Short Presentations, LPAR 2015, Suva, Fiji, November 24-28, 2015. Volume 35 of EPiC Series in Computing., EasyChair (2015) 28–41
40. Tentrup, L.: Non-prenex QBF Solving Using Abstraction. In: International Conference on Theory and Applications of Satisfiability Testing, Springer (2016) 393–401
41. Janhunen, T., Tasharrofi, S., Ternovska, E.: SAT-TO-SAT: Declarative extension of SAT solvers with new propagators. In: Proceedings of AAAI. (2016)
42. Bogaerts, B., Janhunen, T., Tasharrofi, S.: Solving QBF instances with nested SAT solvers. Proceedings of Beyond NP (2016)
43. Lonsing, F., Biere, A.: Failed literal detection for QBF. In: International Conference on Theory and Applications of Satisfiability Testing, Springer (2011) 259–272
44. Devriendt, J., Bogaerts, B., Bruynooghe, M.: BreakIDGlucose: On the importance of row symmetry in SAT. In: Proceedings 4th International Workshop on the Cross-Fertilization Between CSP and SAT. (2014) 1–17
45. Janota, M., Klieber, W., Marques-Silva, J., Clarke, E.: Solving qbf with counterexample guided refinement. In: Theory and Applications of Satisfiability Testing–SAT 2012, Springer Berlin Heidelberg (2012) 114–128
46. Janota, M., Klieber, W., Marques-Silva, J., Clarke, E.: Solving QBF with counterexample guided refinement. Artificial Intelligence **234** (2016) 1–25
47. Pulina, L., Tacchella, A.: A structural approach to reasoning with quantified Boolean formulas. In: 21st International Joint Conference on Artificial Intelligence (IJCAI 2009). (2009) 596–602
48. Diptarama, Narisawa, K., Shinohara, A.: Extension of generalized tic-tac-toe: p stones for one move. IPSJ Journal **55** (2014) 2344–2352
49. Slivovsky, F., Szeider, S.: Soundness of Q-resolution with dependency schemes. Theoretical Computer Science **612** (2016) 83–101
50. Peschiera, C., Pulina, L., Tacchella, A., Bubeck, U., Kullmann, O., Lynce, I.: The seventh QBF solvers evaluation (QBFEVAL'10). In: Theory and Applications of Satisfiability Testing–SAT 2010, Springer Berlin Heidelberg (2010) 237–250
51. Brummayer, R., Lonsing, F., Biere, A.: Automated testing and debugging of SAT and QBF solvers. In: International Conference on Theory and Applications of Satisfiability Testing, Springer (2010) 44–57
52. Chen, H., Interian, Y.: A Model for Generating Random Quantified Boolean Formulas. In: Proceedings of 19th International Joint Conference on Artificial Intelligence (IJCAI'05). (2005) 66–71
53. Stump, A., Sutcliffe, G., Tinelli, C.: StarExec: a cross-community infrastructure for logic solving. In: International Joint Conference on Automated Reasoning, Springer (2014) 367–373
54. Narizzano, M., Pulina, L., Tacchella, A.: The QBFEVAL Web Portal. In: 10th European Conference on Logics in Artificial Intelligence (JELIA 2006). Volume 4160 of Lecture Notes in Computer Science., Springer Verlag (2006) 494–497