Evaluating QBF Solvers: Quantifier Alternations Matter

Florian Lonsing and Uwe Egly

Research Division of Knowledge Based Systems
Institute of Logic and Computation, TU Wien, Vienna, Austria
http://www.kr.tuwien.ac.at/staff/lonsing/

24th International Conference on Principles and Practice of Constraint Programming (CP), August 27-31, Lille, France

This work is supported by the Austrian Science Fund (FWF) under grant S11409-N23.

Introduction (1)

Quantified Boolean Formulas (QBF):

- Existential ($\exists$) / universal ($\forall$) quantification of propositional variables.
- Checking QBF satisfiability: PSPACE-complete.
- QBF encodings: potentially more succinct than propositional logic.

Progress in QBF Reasoning:

- Theory: proof systems (foundations of solver implementations).
- Practice: solving, preprocessing.

Example

Syntax:

- QBF $\psi := \Pi.\phi$ in *prenex conjunctive normal form (PCNF)*.
- $\psi = \underbrace{\forall u \exists x.}_{\text{quantifier prefix}} \underbrace{(\bar{u} \vee x) \wedge (u \vee \bar{x})}_{\text{propositional CNF}}.$

Introduction (2)

Example

Syntax:

- QBF $\psi := \hat{Q}.\phi$ in *prenex conjunctive normal form (PCNF)*.
- $\psi = \underbrace{\forall u \exists x.}_{\text{quantifier prefix}} \underbrace{(\bar{u} \vee x) \wedge (u \vee \bar{x})}_{\text{propositional CNF}}$.

Semantics (recursive):

- Assign variables in prefix ordering, recurse on simplified formula $\psi[A]$ under current assignment A.
- Base cases: $\bot$ is unsatisfiable, $\top$ is satisfiable.
- $\forall u.\psi$ is satisfiable iff $\psi[u/\bot]$ and $\psi[u/\top]$ are satisfiable.
- $\exists x.\psi$ is satisfiable iff $\psi[x/\bot]$ or $\psi[x/\top]$ is satisfiable.

PCNF ψ above is satisfiable:

- $\psi[u/\bot] = \exists x.(\bar{x})$ is satisfiable by setting x to $\bot$.
- $\psi[u/\top] = \exists x.(x)$ is satisfiable by setting x to $\top$.

Introduction (2)

Example

Syntax:

- QBF $\psi := \hat{Q}.\phi$ in *prenex conjunctive normal form (PCNF)*.
- $\psi = \underbrace{\forall u \exists x.}_{\text{quantifier prefix}} \underbrace{(\bar{u} \vee x) \wedge (u \vee \bar{x})}_{\text{propositional CNF}}.$

Semantics (recursive):

- Assign variables in prefix ordering, recurse on simplified formula $\psi[A]$ under current assignment A.
- Base cases: $\bot$ is unsatisfiable, $\top$ is satisfiable.
- $\forall u.\psi$ is satisfiable iff $\psi[u/\bot]$ and $\psi[u/\top]$ are satisfiable.
- $\exists x.\psi$ is satisfiable iff $\psi[x/\bot]$ or $\psi[x/\top]$ is satisfiable.

PCNF ψ above is satisfiable:

- $\psi[u/\bot] = \exists x.(\bar{x})$ is satisfiable by setting x to $\bot$.
- $\psi[u/\top] = \exists x.(x)$ is satisfiable by setting x to $\top$.

Introduction (3)

Quantifier Alternations in PCNFs:

- A PCNF $Q_1 B_1 Q_2 B_2 \ldots Q_n B_n.\ \phi$ has $n \geq 1$ *quantifier blocks* $Q_i B_i$.
- $Q_i B_i$: sets B_i of variables, quantifiers $Q_i \in \{\forall, \exists\}$ with $Q_i \neq Q_{i+1}$.
- A PCNFs with n quantifier blocks has $n - 1$ *quantifier alternations*.

Polynomial Hierarchy (PH): cf. [MS72, Sto76, Wra76]

- Framework to describe the complexity of problems beyond NP.
- Satisfiability problem of a given PCNF is located in PH.

Proposition (cf. [BB09, MS72, Sto76, Wra76])

- *Let* $\psi := Q_1 B_1 \ldots Q_n B_n.\ \phi$ *be a PCNF with* $k \geq 0$ *alternations.*
- $Q_1 = \exists$: *satisfiability problem of* ψ *is* Σ^P_{k+1}*-complete.*
- $Q_1 = \forall$: *satisfiability problem of* ψ *is* Π^P_{k+1}*-complete.*

Introduction (4)

Class	*Prefix Pattern*	*Problems (e.g.)*
$\Sigma_1^P = NP$	$\exists B_1.\phi$	Checking prop. logic satisfiability
$\Pi_1^P = co\text{-}NP$	$\forall B_1.\phi$	Checking prop. logic validity
Σ_2^P	$\exists B_1 \forall B_2.\phi$	MUS membership testing [JS11, Lib05], encodings of conformant planning [Rin07], ASP-related problems [FR05], abstract argumentation [CDG+15]
Π_2^P	$\forall B_1 \exists B_2.\phi$	
$\vdots$		
PSPACE	$Q_1 B_1 \ldots Q_n B_n.\phi$ (*n* depending on problem instance)	LTL model checking [SC85], NFA language inclusion, games [Sch78]

Introduction (5): Solving Paradigms

1. **Expansion** [AB02, Bie04]:
   RAReQS [JKMSC16], Ijtihad [BBSH+18], Rev-Qfun [Jan18], DynQBF [CW17].
2. **QDPLL (backtracking search)** [CGS98]:
   GhostQ [JKMSC16, KSGC10].
3. **Nested SAT solving**:
   QSTS [BJT16a, BJT16b].
4. **Clause selection** and **clausal abstraction**:
   QESTO [JM15b], CAQE [RT15, Ten17].
5. **Backtracking search with learning (QCDCL)** [GNT06, Let02, ZM02b]:
   DepQBF [LE17], Qute [PSS17].
6. **Hybrid approach (expansion, QCDCL)**:
   Heretic [BBSH+18] (applies Ijtihad and DepQBF).

Theory of (orthogonal) proof systems, e.g.: [BCJ15, JM15a, Ten17].

Outline and Contributions

Progress in QBF Solving — Problems:

- Largely driven by empirical evaluation.
- Practically relevant problems: QBF encodings on low levels in PH.
- Risk of convergence of research to few alternations, cf. [Hoo95].
- Solver rankings by solved instances might not reflect diversity and strength of available paradigms.

Outline and Contributions

Our Contributions:

- Study impact of quantifier alternations on solver performance.
- Performance of paradigms varies wrt. alternations.
- More fine-grained analysis: highlighting diversity of paradigms.
- Motivation for combining orthogonal paradigms (proof complexity).

$\Rightarrow$ Improve QBF solving for encodings at higher levels up to PSPACE.

Solving Paradigm (1/2): Expansion

Example

$\psi = \exists x \forall u \exists y.\ (\bar{x} \vee y) \wedge (x \vee \bar{y}) \wedge (\bar{u} \vee y) \wedge (u \vee \bar{y})$

- Expand u: copy CNF and replace y by fresh y_d in copy of CNF.
- $\psi = \exists x, y, y_d.\ \underbrace{(\bar{x} \vee y) \wedge (x \vee \bar{y}) \wedge (\bar{y})}_{u \text{ replaced by } \bot} \wedge \underbrace{(\bar{x} \vee y_d) \wedge (x \vee \bar{y}_d) \wedge (y_d)}_{u \text{ replaced by } \top,\ y \text{ replaced by } y_d}$

Expansion of $\forall$-Variables: cf. [AB02, Bie04]

- Idea: eliminate all universal variables by *Shannon expansion* [Sha49].
- Replace $\hat{Q}\forall x.\phi$ by $\hat{Q}.(\phi[x/\bot] \wedge \phi[x/\top])$.
- Duplicate existential variables inner to x [Bie04, BK07].
- Finally, apply SAT solving to propositional formula.
- Modern: counter example guided abstraction refinement (CEGAR).

Solving Paradigm (2/2): Q-Resolution Calculus

Definition (Q-Resolution Rule)

$$\frac{C_1 \cup \{p\} \qquad C_2 \cup \{\bar{p}\}}{C_1 \cup C_2}$$

for all $x \in \hat{Q}$: $\{x, \bar{x}\} \not\subseteq (C_1 \cup C_2)$, $\bar{p} \notin C_1$, $p \notin C_2$, and $q(p) = \exists$

Example

$\psi = \exists x \forall u \exists y.(x) \wedge (\bar{x} \vee u \vee y) \wedge (\bar{x} \vee u \vee \bar{y})$

$$\frac{(\bar{x} \vee u \vee y) \qquad (\bar{x} \vee u \vee \bar{y})}{(\bar{x} \vee u)}$$

- Traditional Q-resolution [BKF95].
- Must resolve on $\exists$ pivots (cf. variant [VG12]).
- Cf. stronger variants [ZM02a, BJ12].

- PCNF ψ is unsatisfiable iff empty clause $\emptyset$ can be derived.
- Resolution-based QBF solvers: inspired by conflict-driven clause learning (CDCL) and DPLL algorithm for SAT solving.

Solving Paradigm (2/2): Q-Resolution Calculus

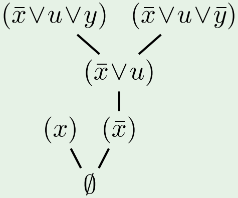
Definition (Reduction Rule)

$$\frac{C \cup \{l\}}{C} \qquad \text{for all } x \in \hat{Q}\text{: } \{x, \bar{x}\} \not\subseteq (C \cup \{l\}),\ q(l) = \forall, \text{ and } l' < l \text{ for all } l' \in C \text{ with } q(l') = \exists$$

Example

$\psi = \exists x \forall u \exists y.(x) \wedge (\bar{x} \vee u \vee y) \wedge (\bar{x} \vee u \vee \bar{y})$

- Reduction removes "trailing" $\forall$-literals.
- Local rule, applied to individual clauses.



- PCNF ψ is unsatisfiable iff empty clause $\emptyset$ can be derived.
- Resolution-based QBF solvers: inspired by conflict-driven clause learning (CDCL) and DPLL algorithm for SAT solving.

Experiments (1)

Benchmark Set and Solvers:

- QBFEVAL'17: 523 prenex CNF instances, 1800 CPU sec., 7 GB mem.
- Focus: instances not solved in preprocessing by HQSpre [WRMB17].
- Top-ranked solvers, based on orthogonal paradigms / proof systems.

Goals of Experimental Evaluation:

- Typical solver rankings: by total number of solved instances.
- Theory: numbers of alternations $\approx$ levels in polynomial hierarchy.
- Performance analysis wrt. instances and their numbers of alternations.
- How do different solving paradigms perform wrt. alternations?
- Is there a single best approach that dominates all the others?

Experiments (2): Alternation Bias

Table: Histograms of the benchmark sets illustrating the numbers of formulas (#f) in classes given by the number of qblocks (#q).

(a)

#q	#f
2	90
3	236
4–10	70
11–20	42
21–	85
2–3	326
4–	197

(b)

#q	#f
2	70
3	145
4–10	26
11–20	30
21–	41
2–3	215
4–	97

(c)

#q	#f
2	70
3	145
4–10	26
11–20	40
21–	31
2–3	215
4–	97

(a) 523 original benchmarks.

(b) 312 benchmarks filtered by HQSpre.

(c) 312 benchmarks preprocessed by HQSpre.

Experiments (3): Overall Rankings

Table: Solvers and corresponding paradigms (P), solved instances (S), unsatisfiable ($\bot$) and satisfiable ones ($\top$), and uniquely solved instances.

Solver	*P*	*S*	$\bot$	$\top$	*U*
GhostQ	2	112	61	51	**15**
Rev-Qfun	1	110	58	52	6
CAQE	4	68	42	26	6
DepQBF	5	64	41	23	4
QSTS	3	56	34	22	3
RAReQS	1	50	34	16	1
Heretic	6	49	34	15	0
Qute	5	47	25	22	0
DynQBF	1	46	24	22	9
QESTO	4	45	30	15	0
Ijtihad	1	36	27	9	1

(a) Filtered instances.

Solver	*P*	*S*	$\bot$	$\top$	*U*
CAQE	4	114	65	49	6
RAReQS	1	103	63	40	3
QESTO	4	97	63	34	1
Rev-Qfun	1	90	57	33	6
Heretic	6	87	55	32	0
QSTS	3	72	46	26	1
DepQBF	5	72	44	28	5
Qute	5	70	42	28	2
Ijtihad	1	58	43	15	1
GhostQ	2	58	33	25	0
DynQBF	1	45	24	21	**17**

(b) Preprocessed instances.

Experiments (4): Class-Based Analysis — Solvers

Table: Instances solved in classes by numbers of qblocks (#q) and numbers of formulas in each class (#f). Only class winners (bold face) are shown, paradigms (*P:*) are indicated in the first row.

P:		2	1	4	5
#q	#f	*GhostQ*	*Rev-Qfun*	*CAQE*	*DepQBF*
2	70	**36**	18	5	7
3	145	62	**71**	33	23
4–10	26	3	5	**7**	**7**
11–20	30	3	5	8	**16**
21–	41	8	11	**15**	11
2–3	215	**98**	89	38	30
4–	97	14	21	30	**34**

(a) Filtered instances.

P:		4	6	5	1
#q	#f	*CAQE*	*Heretic*	*DepQBF*	*DynQBF*
2	70	18	15	15	**24**
3	145	**67**	42	24	14
4–10	26	6	**10**	7	5
11–20	40	14	15	**20**	2
21–	31	**9**	5	6	0
2–3	215	**85**	57	39	38
4–	97	29	30	**33**	7

(b) Preprocessed instances.

Experiments (5): Class-Based Analysis — Paradigms

Table: Instances solved by solving paradigms 1 to 6 in classes by numbers of qblocks (#q).

#q	1	2	3	4	5	6
2	28	**36**	9	6	8	2
3	**85**	62	27	36	40	23
4–10	**9**	3	1	**9**	8	5
11-20	8	3	7	8	**16**	9
21–	**15**	8	12	**15**	11	10
2–3	**113**	98	36	42	48	25
4–	32	14	20	32	**35**	24
2–	**145**	112	56	74	83	49

(a) Filtered instances.

#q	1	2	3	4	5	6
2	**37**	7	17	18	21	15
3	**78**	40	35	71	40	42
4–10	10	1	2	**13**	7	10
11–20	17	6	13	15	**21**	15
21–	8	4	5	**10**	8	5
2–3	**115**	47	52	89	61	57
4–	35	11	20	**38**	36	30
2–	**150**	58	72	127	97	87

(b) Preprocessed instances.

Experiments (6): Class-Based VBS Analysis — Solvers

Table: Instances solved by the virtual best solver (VBS) in classes by number of qblocks (#q), number of formulas (#f) in each class, and relative contribution (%) of each solver to instances solved by the VBS.

#q	#f	VBS	GhostQ	Rev-Qfun	CAQE	DepQBF	QSTS	RAReQS	Heretic	Qute	DynQBF	QESTO	Ijtihad
2	70	46	**41.3**	6.5	6.5	6.5	6.5	0.0	0.0	0.0	30.4	2.1	0.0
3	145	89	12.3	**33.7**	2.2	2.2	15.7	22.4	0.0	3.3	2.2	4.4	1.1
4–10	26	19	5.2	0.0	**26.3**	**26.3**	0.0	0.0	0.0	15.7	10.5	10.5	5.2
11-20	30	18	0.0	0.0	11.1	**50.0**	27.7	5.5	0.0	0.0	5.5	0.0	0.0
21–	41	21	4.7	14.2	19.0	9.5	**28.5**	14.2	0.0	0.0	9.5	0.0	0.0
2–3	215	135	22.2	**24.4**	3.7	3.7	12.5	14.8	0.0	2.2	11.8	3.7	0.7
4–	97	58	3.4	5.1	18.9	**27.5**	18.9	6.8	0.0	5.1	8.6	3.4	1.7
2–	312	193	16.5	**18.6**	8.2	10.8	14.5	12.4	0.0	3.1	10.8	3.6	1.0

(a) Filtered instances.

Experiments (6): Class-Based VBS Analysis — Solvers

Table: Instances solved by the virtual best solver (VBS) in classes by number of qblocks (#q), number of formulas (#f) in each class, and relative contribution (%) of each solver to instances solved by the VBS.

#q	#f	VBS	CAQE	RAReQS	QESTO	Rev-Qfun	Heretic	QSTS	DepQBF	Qute	Ijtihad	GhostQ	DynQBF
2	70	40	7.5	17.5	2.5	7.5	2.5	10.0	10.0	0.0	0.0	2.5	**40.0**
3	145	87	9.1	**40.2**	8.0	12.6	1.1	6.8	0.0	8.0	3.4	4.5	5.7
4–10	26	20	**25.0**	10.0	15.0	5.0	0.0	0.0	**25.0**	5.0	5.0	0.0	10.0
11-20	40	26	3.8	19.2	7.6	0.0	7.6	26.9	**30.7**	0.0	0.0	0.0	3.8
21–	31	11	9.0	**27.2**	9.0	9.0	0.0	**27.2**	9.0	9.0	0.0	0.0	0.0
2–3	215	127	8.6	**33.0**	6.2	11.0	1.5	7.8	3.1	5.5	2.3	3.9	16.5
4–	97	57	12.2	17.5	10.5	3.5	3.5	17.5	**24.5**	3.5	1.7	0.0	5.2
2–	312	184	9.7	**28.2**	7.6	8.6	2.1	10.8	9.7	4.8	2.1	2.7	13.0

(a) Preprocessed instances.

Experiments (7): Class-Based VBS Analysis — Paradigms

Table: Instances solved by the virtual best solver (VBS) in classes by number of qblocks (#q), number of formulas (#f) in each class, and relative contribution (%) of solving paradigms to instances solved by the VBS.

#q	#f	VBS	1	2	3	4	5	6
2	70	46	36.9	**41.3**	6.5	8.6	6.5	0.0
3	145	89	**59.5**	12.3	15.7	6.7	5.6	0.0
4–10	26	19	15.7	5.2	0.0	36.8	**42.1**	0.0
11–20	30	18	11.1	0.0	27.7	11.1	**50.0**	0.0
21–	41	21	**38.0**	4.7	28.5	19.0	9.5	0.0
2–3	215	135	**51.8**	22.2	12.5	7.4	5.9	0.0
4–	97	58	22.4	3.4	18.9	22.4	**32.7**	0.0
2–	312	193	**43.0**	16.5	14.5	11.9	13.9	0.0

(a) Filtered instances.

Experiments (7): Class-Based VBS Analysis — Paradigms

Table: Instances solved by the virtual best solver (VBS) in classes by number of qblocks (#q), number of formulas (#f) in each class, and relative contribution (%) of solving paradigms to instances solved by the VBS.

#q	#f	VBS	1	2	3	4	5	6
2	70	40	**65.0**	2.5	10.0	10.0	10.0	2.5
3	145	87	**62.0**	4.5	6.8	17.2	8.0	1.1
4–10	26	20	30.0	0.0	0.0	**40.0**	30.0	0.0
11–20	40	26	23.0	0.0	26.9	11.5	**30.7**	7.6
21–	31	11	**36.3**	0.0	27.2	18.1	18.1	0.0
2–3	215	127	**62.9**	3.9	7.8	14.9	8.6	1.5
4–	97	57	**28.0**	0.0	17.5	22.8	**28.0**	3.5
2–	312	184	**52.1**	2.7	10.8	17.3	14.6	2.1

(a) Preprocessed instances.

Summary

QBF Solving:

- Different approaches, empirically-driven development of QBF tools.
- Power of different approaches often not reflected in overall rankings.
- Majority of available QBF benchmarks: problems from low PH levels.

Our Empirical Results:

- More fine-grained picture of solver performance.
- Highlighting different strengths in instance classes by alternations.
- VBS: large potential for combining different approaches.

Future Work and Open Problems:

- Risk of convergence of research to certain approaches / formulas.
- Proof complexity and alternations, cf. [BHP17, BBH18, Che16].

References

References I

[AB02] Abdelwaheb 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.

[BB09] Hans Kleine Büning and Uwe Bubeck.
Theory of Quantified Boolean Formulas.
In *Handbook of Satisfiability*, volume 185 of *FAIA*, pages 735–760. IOS Press, 2009.

[BBH18] Olaf Beyersdorff, Joshua Blinkhorn, and Luke Hinde.
Size, Cost and Capacity: A Semantic Technique for Hard Random QBFs.
In *ITCS*, volume 94 of *LIPIcs*, pages 9:1–9:18. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.

References II

[BBSH+18] Roderick Bloem, Nicolas Braud-Santoni, Vedad Hadzic, Uwe Egly, Florian Lonsing, and Martina Seidl.
Expansion-Based QBF Solving Without Recursion.
In *FMCAD*. IEEE, 2018.

[BCJ15] Olaf Beyersdorff, Leroy Chew, and Mikolás Janota.
Proof Complexity of Resolution-based QBF Calculi.
In *STACS*, volume 30 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 76–89. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015.

[BHP17] Olaf Beyersdorff, Luke Hinde, and Ján Pich.
Reasons for Hardness in QBF Proof Systems.
Electronic Colloquium on Computational Complexity (ECCC), 24:44, 2017.

[Bie04] Armin Biere.
Resolve and Expand.
In *SAT*, volume 3542 of *LNCS*, pages 59–70. Springer, 2004.

[BJ12] Valeriy Balabanov and Jie-Hong R. Jiang.
Unified QBF certification and its applications.
Formal Methods in System Design, 41(1):45–65, 2012.

[BJT16a] 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.

[BJT16b] Bart Bogaerts, Tomi Janhunen, and Shahab Tasharrofi.
Solving QBF Instances with Nested SAT Solvers.
In *Beyond NP Workshop 2016 at AAAI-16*, 2016.

References IV

[BK07] Uwe Bubeck and Hans Kleine Büning.
Bounded Universal Expansion for Preprocessing QBF.
In *SAT*, volume 4501 of *LNCS*, pages 244–257. Springer, 2007.

[BKF95] Hans Kleine Büning, Marek Karpinski, and Andreas Flögel.
Resolution for Quantified Boolean Formulas.
Inf. Comput., 117(1):12–18, 1995.

[CDG+15] Günther Charwat, Wolfgang Dvorák, Sarah Alice Gaggl, Johannes Peter Wallner, and Stefan Woltran.
Methods for solving reasoning problems in abstract argumentation - A survey.
Artif. Intell., 220:28–63, 2015.

References V

[CGS98] 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.

[Che16] Hubie Chen.
Proof Complexity Modulo the Polynomial Hierarchy: Understanding Alternation as a Source of Hardness.
In *ICALP*, volume 55 of *LIPIcs*, pages 94:1–94:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

[CW17] Günther Charwat and Stefan Woltran.
Expansion-based QBF Solving on Tree Decompositions.
In *RCRA Workshop*, volume 2011 of *CEUR Workshop Proceedings*, pages 16–26. CEUR-WS.org, 2017.

References VI

[FR05] Wolfgang Faber and Francesco Ricca.
Solving Hard ASP Programs Efficiently.
In *LPNMR*, volume 3662 of *LNCS*, pages 240–252. Springer, 2005.

[GNT06] Enrico Giunchiglia, Massimo Narizzano, and Armando Tacchella.
Clause/Term Resolution and Learning in the Evaluation of Quantified Boolean Formulas.
JAIR, 26:371–416, 2006.

[Hoo95] John N. Hooker.
Testing heuristics: We have it all wrong.
J. Heuristics, 1(1):33–42, 1995.

References VII

[Jan18] Mikolás Janota.
Towards Generalization in QBF Solving via Machine Learning.
In *AAAI*. AAAI Press, 2018.

[JKMSC16] Mikoláš Janota, William Klieber, Joao Marques-Silva, and Edmund Clarke.
Solving QBF with counterexample guided refinement.
Artificial Intelligence, 234:1–25, 2016.

[JM15a] Mikolás Janota and Joao Marques-Silva.
Expansion-based QBF solving versus Q-resolution.
Theor. Comput. Sci., 577:25–42, 2015.

[JM15b] Mikolás Janota and Joao Marques-Silva.
Solving QBF by Clause Selection.
In *IJCAI*, pages 325–331. AAAI Press, 2015.

References VIII

[JS11] Mikolás Janota and João P. Marques Silva.
On Deciding MUS Membership with QBF.
In *CP*, volume 6876 of *LNCS*, pages 414–428. Springer, 2011.

[KSGC10] 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.

[LE17] Florian Lonsing and Uwe Egly.
DepQBF 6.0: A Search-Based QBF Solver Beyond Traditional QCDCL.
In *CADE*, volume 10395 of *LNCS*, pages 371–384. Springer, 2017.

References IX

[Let02] Reinhold Letz.
Lemma and Model Caching in Decision Procedures for Quantified Boolean Formulas.
In *TABLEAUX*, volume 2381 of *LNCS*, pages 160–175. Springer, 2002.

[Lib05] Paolo Liberatore.
Redundancy in logic I: CNF propositional formulae.
Artif. Intell., 163(2):203–232, 2005.

[MS72] 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.

References X

[PSS17] Tomás Peitl, Friedrich Slivovsky, and Stefan Szeider.
Dependency Learning for QBF.
In *SAT*, volume 10491 of *LNCS*, pages 298–313. Springer, 2017.

[Rin07] Jussi Rintanen.
Asymptotically Optimal Encodings of Conformant Planning in QBF.
In *AAAI*, pages 1045–1050. AAAI Press, 2007.

[RT15] Markus N. Rabe and Leander Tentrup.
CAQE: A Certifying QBF Solver.
In *FMCAD*, pages 136–143. IEEE, 2015.

[SC85] A. Prasad Sistla and Edmund M. Clarke.
The Complexity of Propositional Linear Temporal Logics.
J. ACM, 32(3):733–749, 1985.

References XI

[Sch78] Thomas J Schaefer.
On the Complexity of Some Two-Person Perfect-Information Games.
Journal of Computer and System Sciences, 16(2):185–225, 1978.

[Sha49] Claude Elwood Shannon.
The Synthesis of Two-Terminal Switching Circuits.
Bell System Technical Journal, 28(1):59–98, 1949.

[Sto76] Larry J. Stockmeyer.
The Polynomial-Time Hierarchy.
Theor. Comput. Sci., 3(1):1–22, 1976.

References XII

[Ten17] Leander Tentrup.
On Expansion and Resolution in CEGAR Based QBF Solving.
In *CAV*, volume 10427 of *LNCS*, pages 475–494. Springer, 2017.

[VG12] Allen Van Gelder.
Contributions to the Theory of Practical Quantified Boolean Formula Solving.
In *CP*, volume 7514 of *LNCS*, pages 647–663. Springer, 2012.

[Wra76] Celia Wrathall.
Complete Sets and the Polynomial-Time Hierarchy.
Theor. Comput. Sci., 3(1):23–33, 1976.

References XIII

[WRMB17] Ralf Wimmer, Sven Reimer, Paolo Marin, and Bernd Becker.
HQSpre - An Effective Preprocessor for QBF and DQBF.
In *TACAS*, volume 10205 of *LNCS*, pages 373–390. Springer, 2017.

[ZM02a] Lintao Zhang and Sharad Malik.
Conflict Driven Learning in a Quantified Boolean Satisfiability Solver.
In *ICCAD*, pages 442–449. ACM / IEEE Computer Society, 2002.

[ZM02b] 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.