DepQBF 6.0: A Search-Based QBF Solver Beyond Traditional QCDCL

Florian Lonsing and Uwe Egly

Knowledge-Based Systems Group, Vienna University of Technology, Austria
http://www.kr.tuwien.ac.at/staff/lonsing/

26th International Conference on Automated Deduction (CADE-26)
August 6-11, 2017, Gothenburg, Sweden

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

Introduction

Quantified Boolean Formulas (QBF):

- Existential ($\exists$) / universal ($\forall$) quantification of propositional variables.
- E.g., QBF in prenex CNF (PCNF): $\psi := \forall u \exists x.(\bar{u} \vee x) \wedge (u \vee \bar{x})$
- Checking QBF satisfiability: PSPACE-completeness.
- Potentially more succinct encodings than propositional logic.

$\Rightarrow$ motivation: combination of proof systems in QCDCL via extension of QRES calculus.

Introduction

DepQBF 6.0:

- QCDCL-based QBF solver [GNT06, Let02, ZM02].
- QCDCL: conflict-driven clause learning via *Q-resolution calculus (QRES)* [GNT06, KBKF95, Let02, ZM02].
- Extension of CDCL algorithm in SAT solving (propositional logic).

QBF Proof Complexity:

- Recent results: orthogonality of proof systems [BCJ15, JM15a].
- Expansion, e.g., [AB02, Bie04, JKMSC16, JM15b, RT15].
- E.g. expansion vs. Q-resolution: solvers have different strengths.

$\Rightarrow$ motivation: combination of proof systems in QCDCL via extension of QRES calculus.

Introduction

DepQBF 6.0:

- QCDCL-based QBF solver [GNT06, Let02, ZM02].
- QCDCL: conflict-driven clause learning via *Q-resolution calculus (QRES)* [GNT06, KBKF95, Let02, ZM02].
- Extension of CDCL algorithm in SAT solving (propositional logic).

QBF Proof Complexity:

- Recent results: orthogonality of proof systems [BCJ15, JM15a].
- Expansion, e.g., [AB02, Bie04, JKMSC16, JM15b, RT15].
- E.g. expansion vs. Q-resolution: solvers have different strengths.

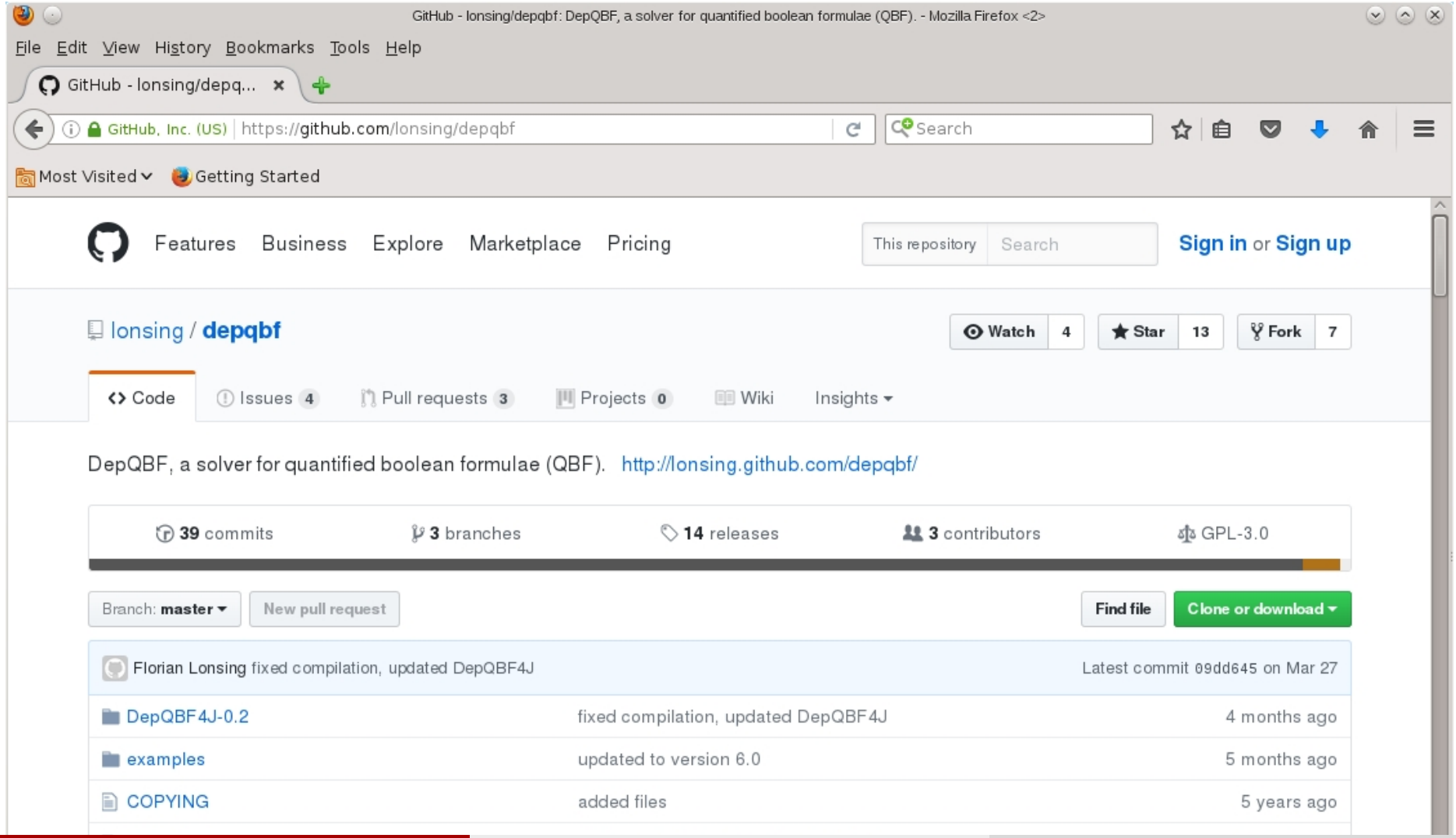
$\Rightarrow$ motivation: combination of proof systems in QCDCL via extension of QRES calculus.

DepQBF: Overview and Version History

- Approx. 20,000 lines of C code, APIs in C and JAVA.
- Free software (GPLv3): `http://lonsing.github.io/depqbf/`

DepQBF: Overview and Version History

- Approx. 20,000 lines of C code, APIs in C and JAVA.
- Free software (GPLv3): `http://lonsing.github.io/depqbf/`



DepQBF: Overview and Version History

Rel. Date	Version	Features	Publications
Mar 2010	0.1	initial release, integration of dependency schemes	[LB10a, LB10b]
Jul 2012	1.0	proof generation	[NPL+12]
Aug 2013	2.0	advanced QCDCL	[LEVG13]
Feb 2014	3.0	incremental solving	[LE14]
Feb 2015	4.0	incremental solving (new API)	[LE15]
Oct 2015	5.0	first step beyond QCDCL	[LBB+15]
Feb 2017	6.0	beyond QCDCL (framework)	CADE 2017
Aug 2017	6.03	latest maintenance release	—

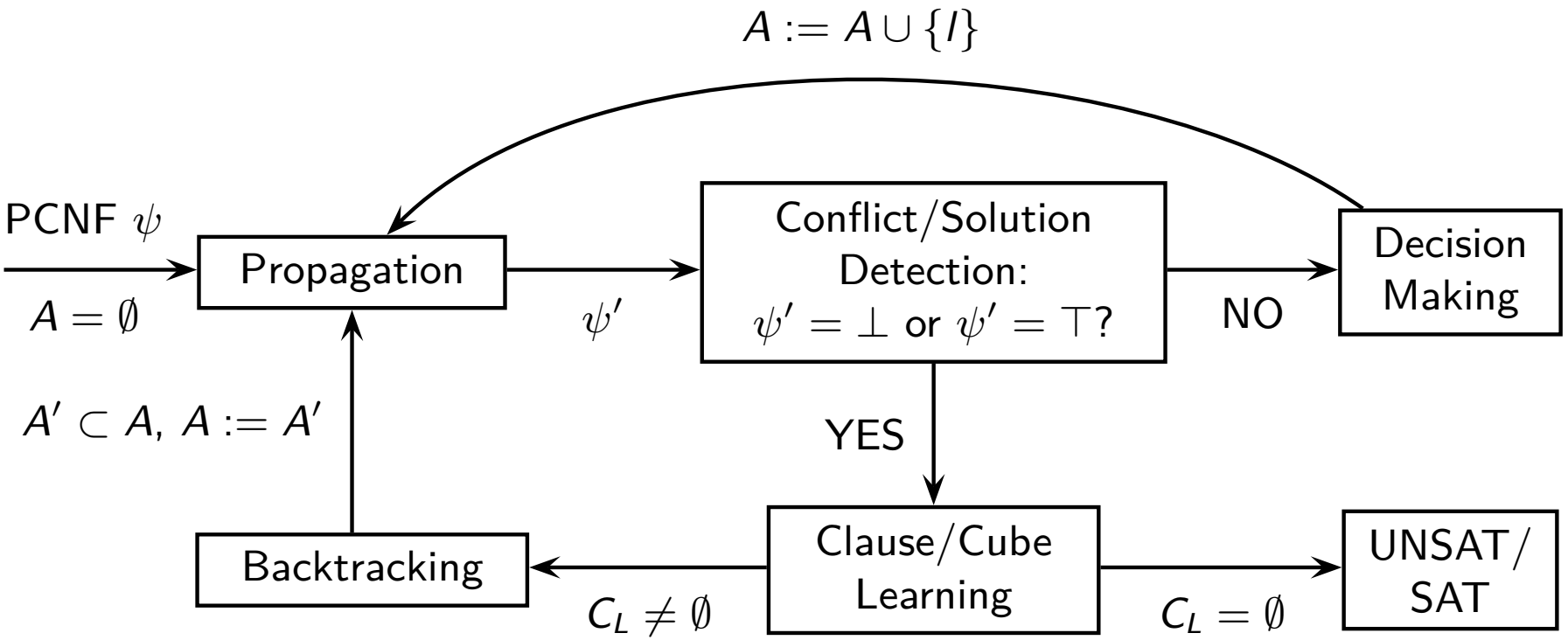
⇒ Version 6.0: framework to integrate *arbitrary* proof systems in QCDCL based on extension of QRES calculus.

DepQBF: Overview and Version History

Rel. Date	Version	Features	Publications
Mar 2010	0.1	initial release, integration of dependency schemes	[LB10a, LB10b]
Jul 2012	1.0	proof generation	[NPL+12]
Aug 2013	2.0	advanced QCDCL	[LEVG13]
Feb 2014	3.0	incremental solving	[LE14]
Feb 2015	4.0	incremental solving (new API)	[LE15]
Oct 2015	5.0	first step beyond QCDCL	[LBB+15]
Feb 2017	6.0	beyond QCDCL (framework)	CADE 2017
Aug 2017	6.03	latest maintenance release	—

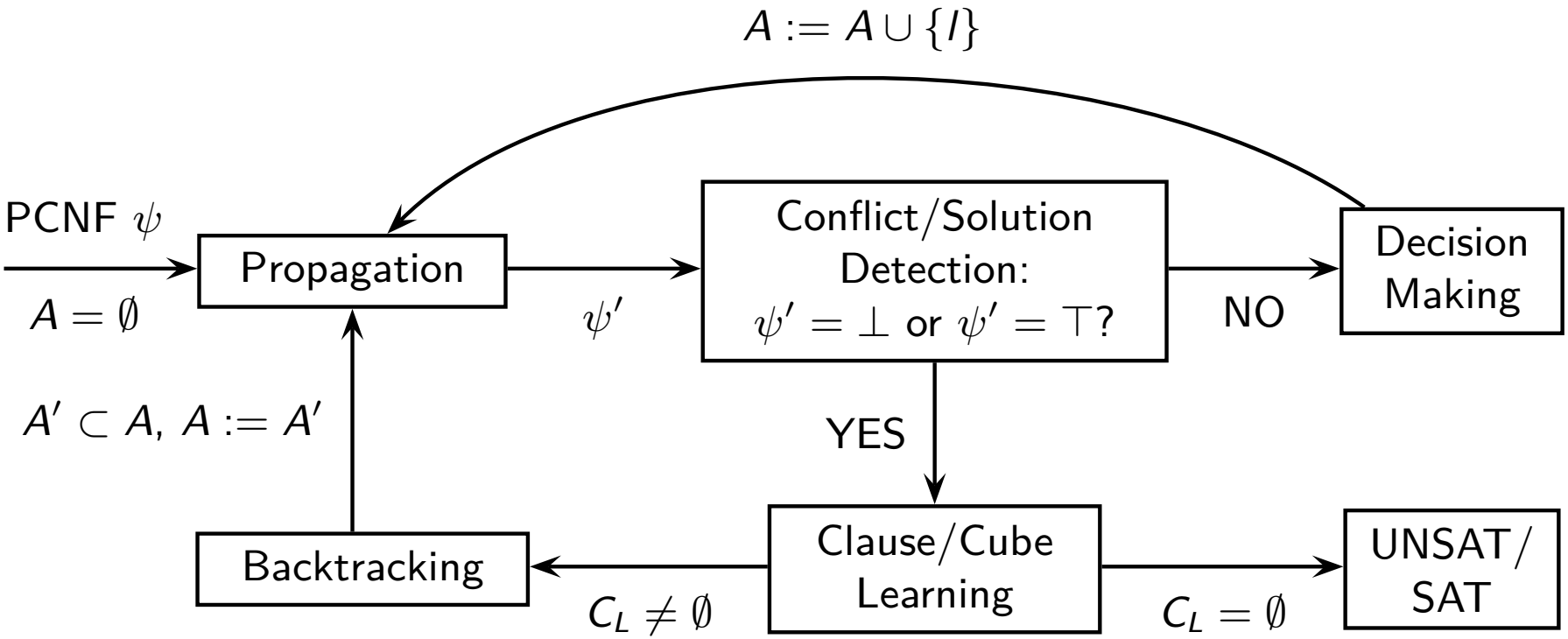
⇒ Version 6.0: framework to integrate *arbitrary* proof systems in QCDCL based on extension of QRES calculus.

Traditional QCDCL



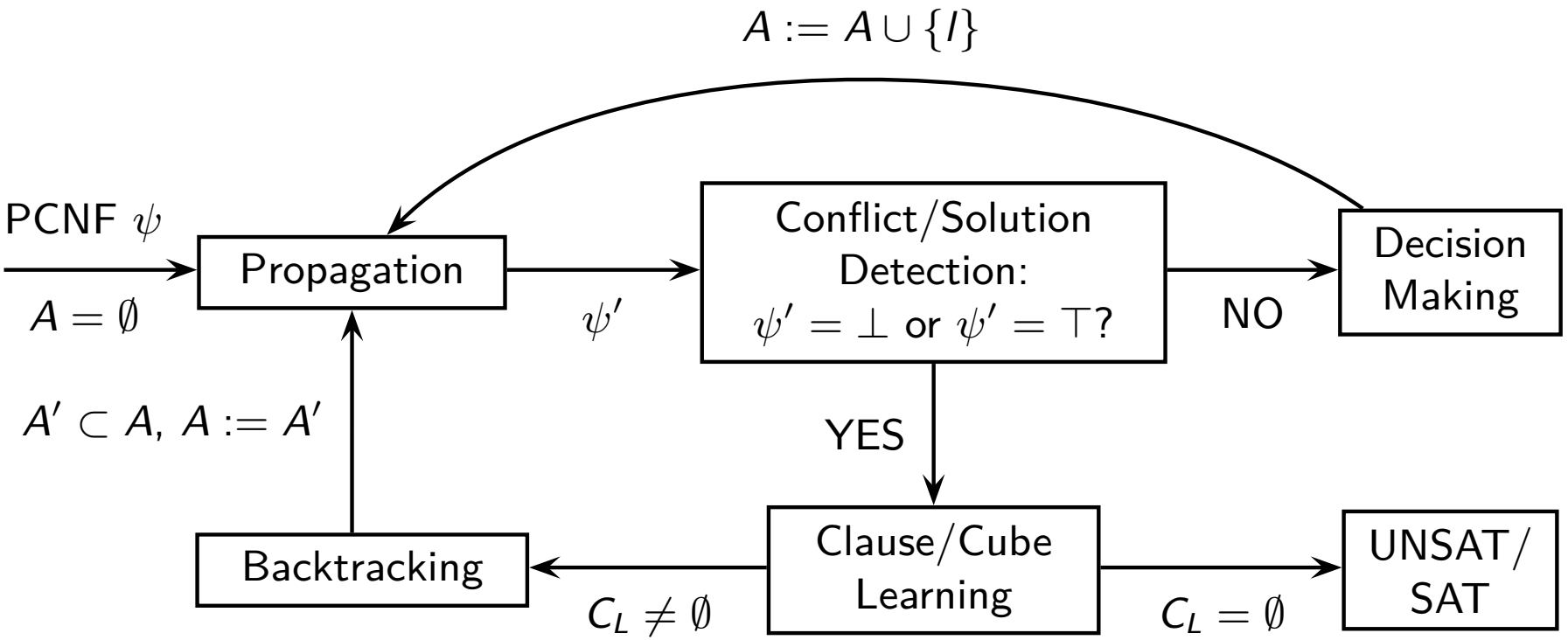
- Generate assignments A by decision making and (unit) propagation.
- Simplify ψ under A to obtain ψ'.
- Conflict: $\psi' = \bot$: ψ' contains a falsified clause.
- Solution: $\psi' = \top$: all clauses in ψ' satisfied (i.e., empty CNF).

Traditional QCDCL



- Generate learned clause (cube) C_L by Q-resolution, added to ψ.
- Empty clause (cube) $C_L = \emptyset$: formula proved UNSAT (SAT).
- Q-resolution proofs of (un)satisfiability by QRES.

Traditional QCDCL

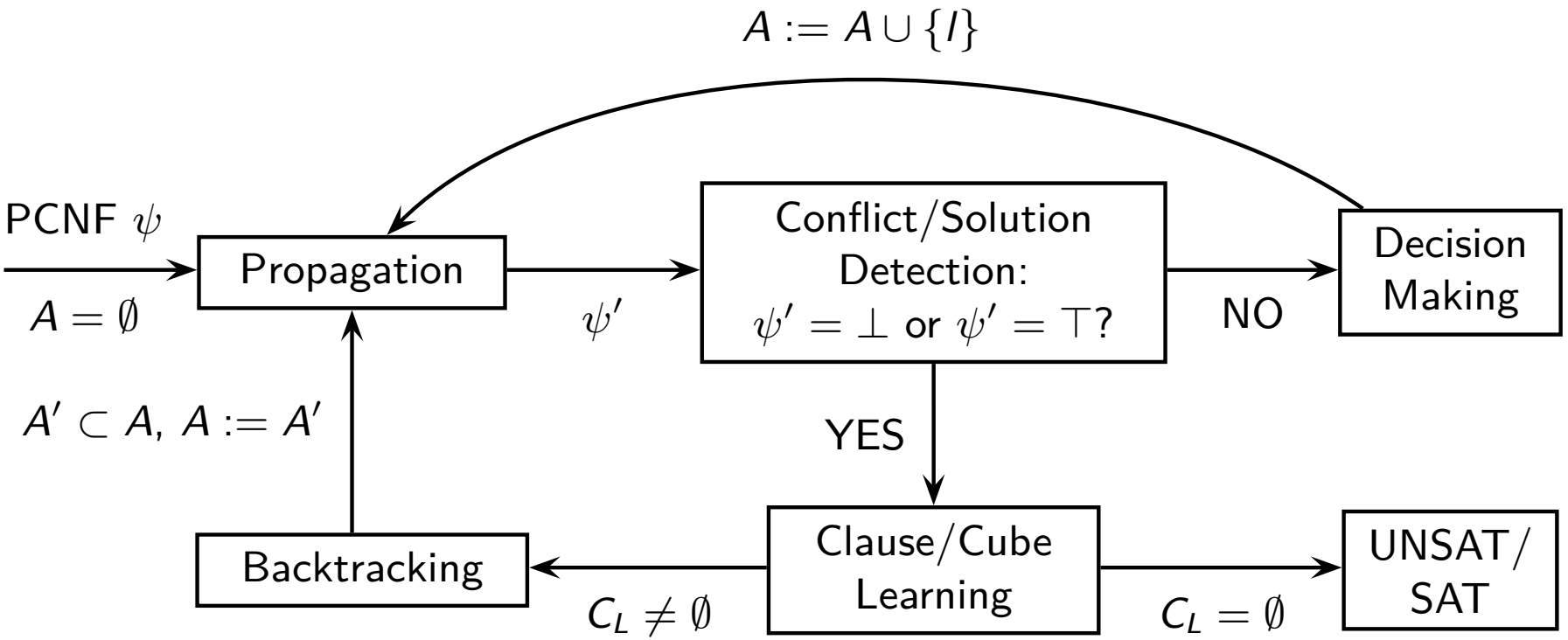


- Conflict detected: select clauses for Q-resolution.

Definition (Clause Axiom of QRES)

$\frac{}{C}$ Given a PCNF $\psi = \hat{Q}.\phi$, $C \in \phi$ is a clause.

Traditional QCDCL

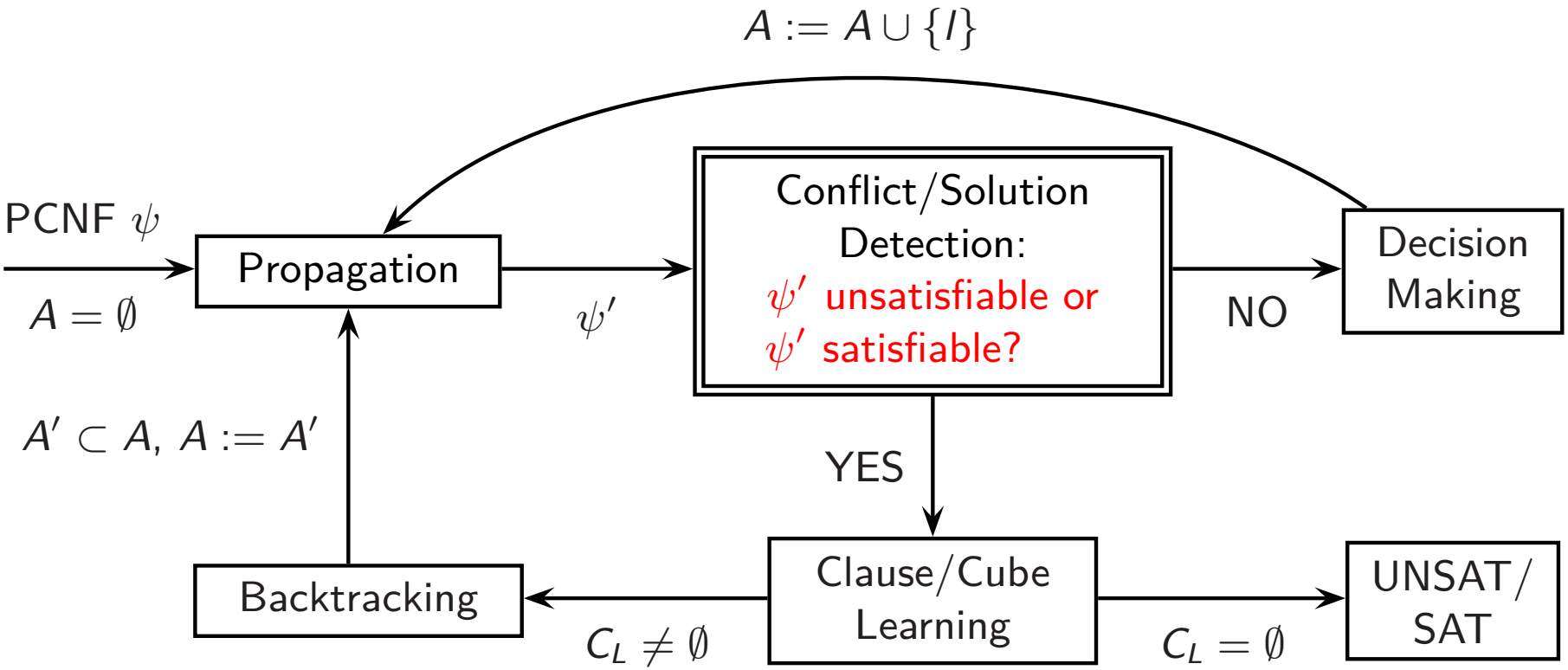


- Solution detected: select cubes for Q-resolution.

Definition (Cube Axiom of QRES)

$\frac{}{C}$ Given a PCNF $\psi = \hat{Q}.\phi$ and an assignment A with $\psi[A] = \top$, $C = (\bigwedge_{l \in A})$ is a cube.

DepQBF 6.0: Beyond Traditional QCDCL

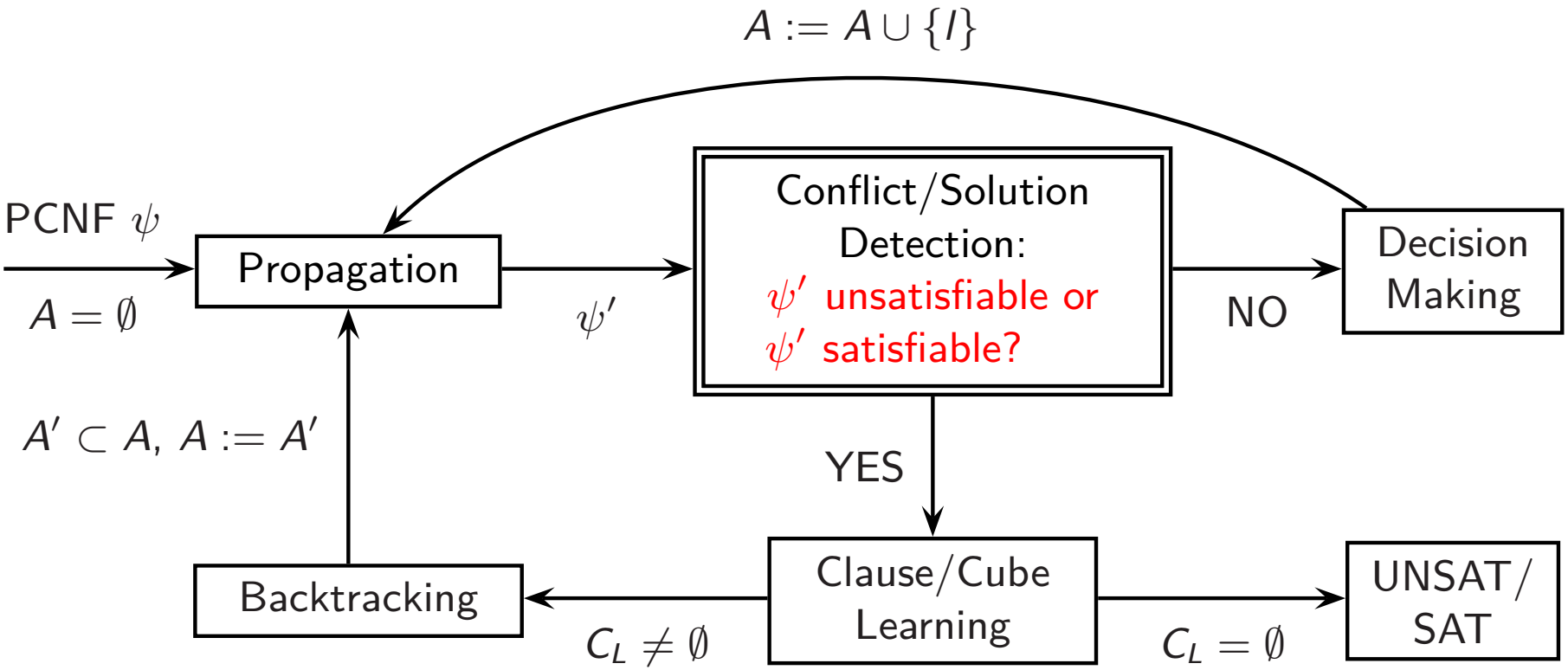


- Generalized conflict detection by incomplete satisfiability test of ψ'.

Definition (Generalized Clause Axiom of QRES [LES16])

$\frac{}{C}$ Given a PCNF $\psi = \hat{Q}.\phi$, A is an assignment generated in QCDCL, $\psi[A]$ is unsatisfiable, and $C = (\bigvee_{l \in A} \overline{l})$ is a clause.

DepQBF 6.0: Beyond Traditional QCDCL

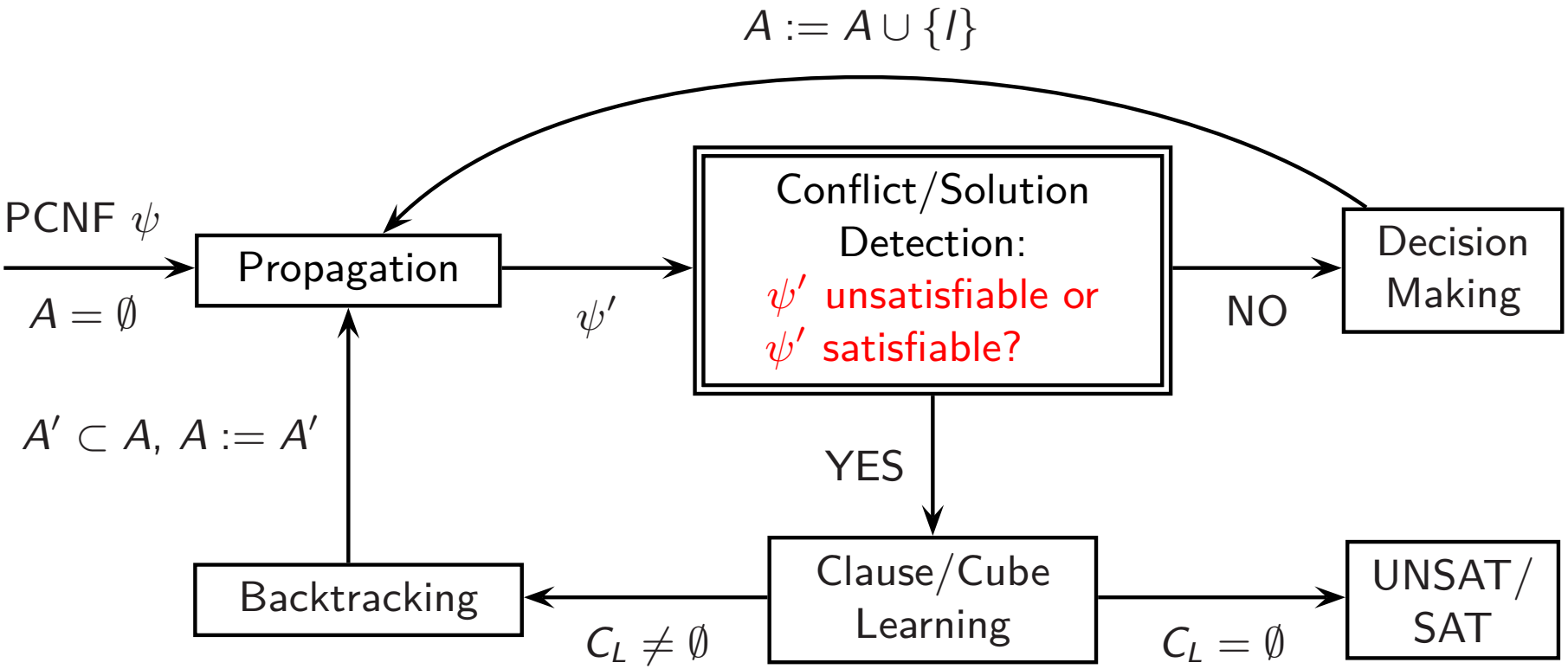


- Generalized solution detection by incomplete satisfiability test of ψ'.

Definition (Generalized Cube Axiom of QRES [LES16])

$\overline{\text{C}}$ Given a PCNF $\psi = \hat{Q}.\phi$, A is an assignment generated in QCDCL, $\psi[A]$ is satisfiable, and $C = (\bigwedge_{l \in A} l)$ is a cube.

DepQBF 6.0: Beyond Traditional QCDCL



- Semantic vs. syntactic check: ψ' (un)satisfiable? vs. $\psi' = \bot/\top$?
- Any QBF decision procedure / proof system can be applied.
- Generalized axioms enable integration of proof systems in QRES.
- In practice: resource-bounded applications.
- Expansion, clause elimination techniques. . . [BLS11, WRMB17].

Experiments 1/3

Benchmark Set from QBFEVAL'16:

- 825 prenex CNF instances, 1800 seconds, 7 GB memory limits.

QBF Solvers:

- Top ranked solvers from QBFEVAL'16.
- Five different solving paradigms and orthogonal proof systems.
- DQ: DepQBF 6.0 with generalized axioms.
- DQ-n: DepQBF 6.0 without generalized axioms (traditional QCDCL).

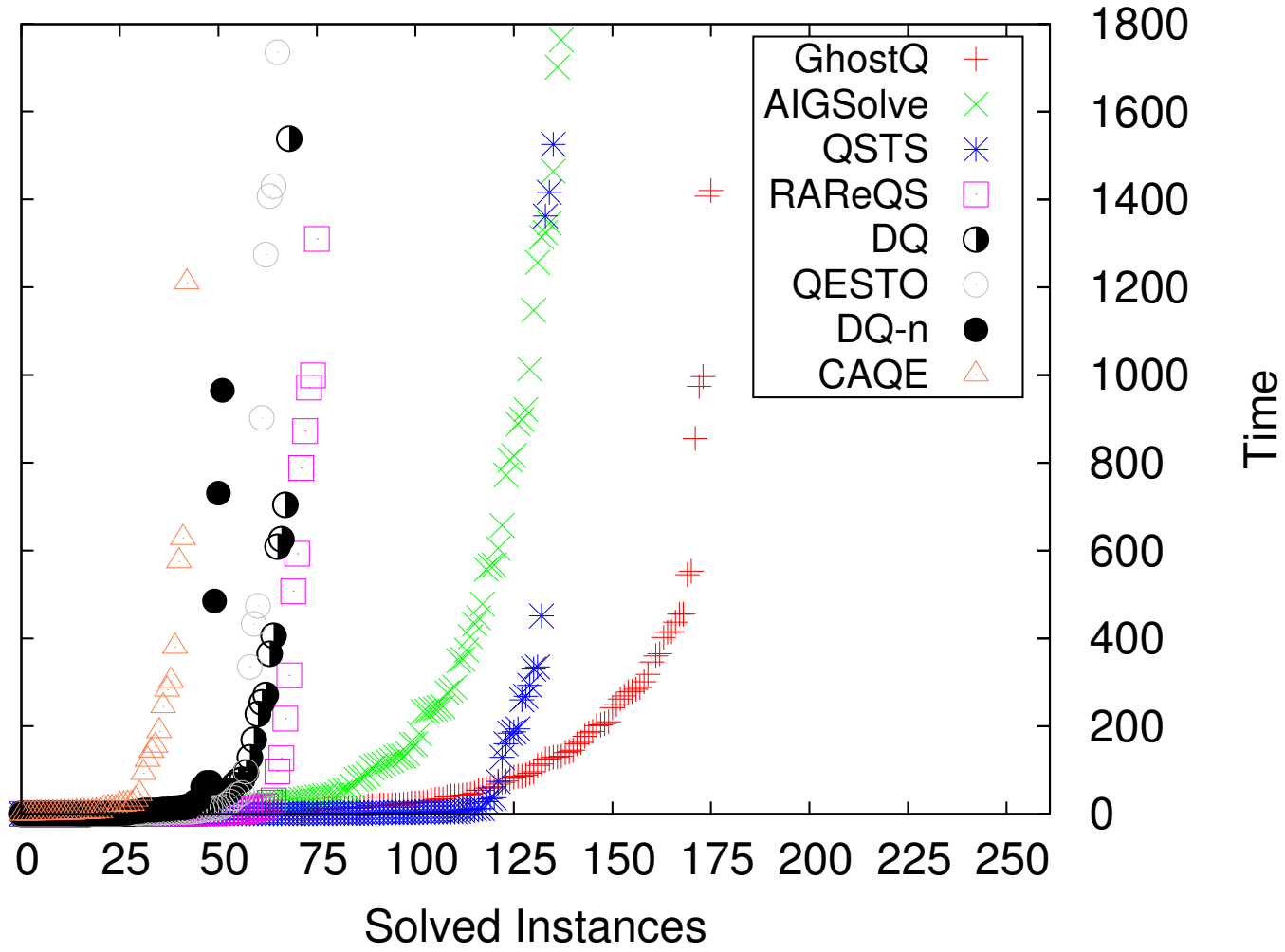
Experiments 1/3

Alternation Bias: cf. [LE17]

- 56% of the benchmarks have no more than two quantifier alternations.
- Theory: numbers of alternations $\approx$ levels in polynomial hierarchy.
- Focus: 402 instances not solved by preprocessing using Bloqqer [BLS11].
- Analysis wrt. instances having few/many alternations.

Experiments 2/3: 402 Filtered Instances

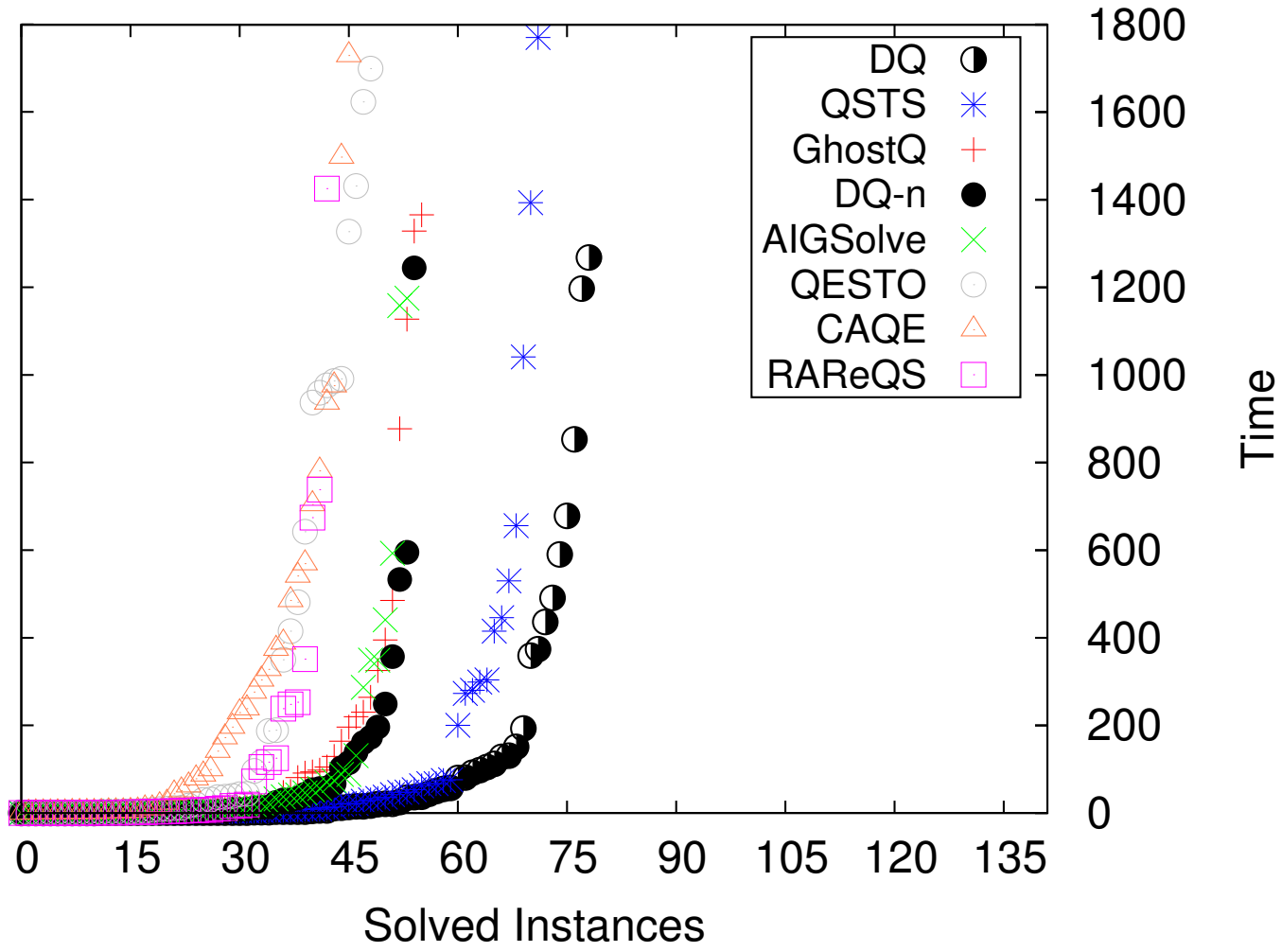
	Solved
GhostQ	176
AIGSolve	138
QSTS	136
RAReQS	76
DQ	**69**
QESTO	66
DQ-n	**52**
CAQE	43



- 261 instances (65%), ≤ 2 alternations, filtered but *not* preprocessed.

Experiments 2/3: 402 Filtered Instances

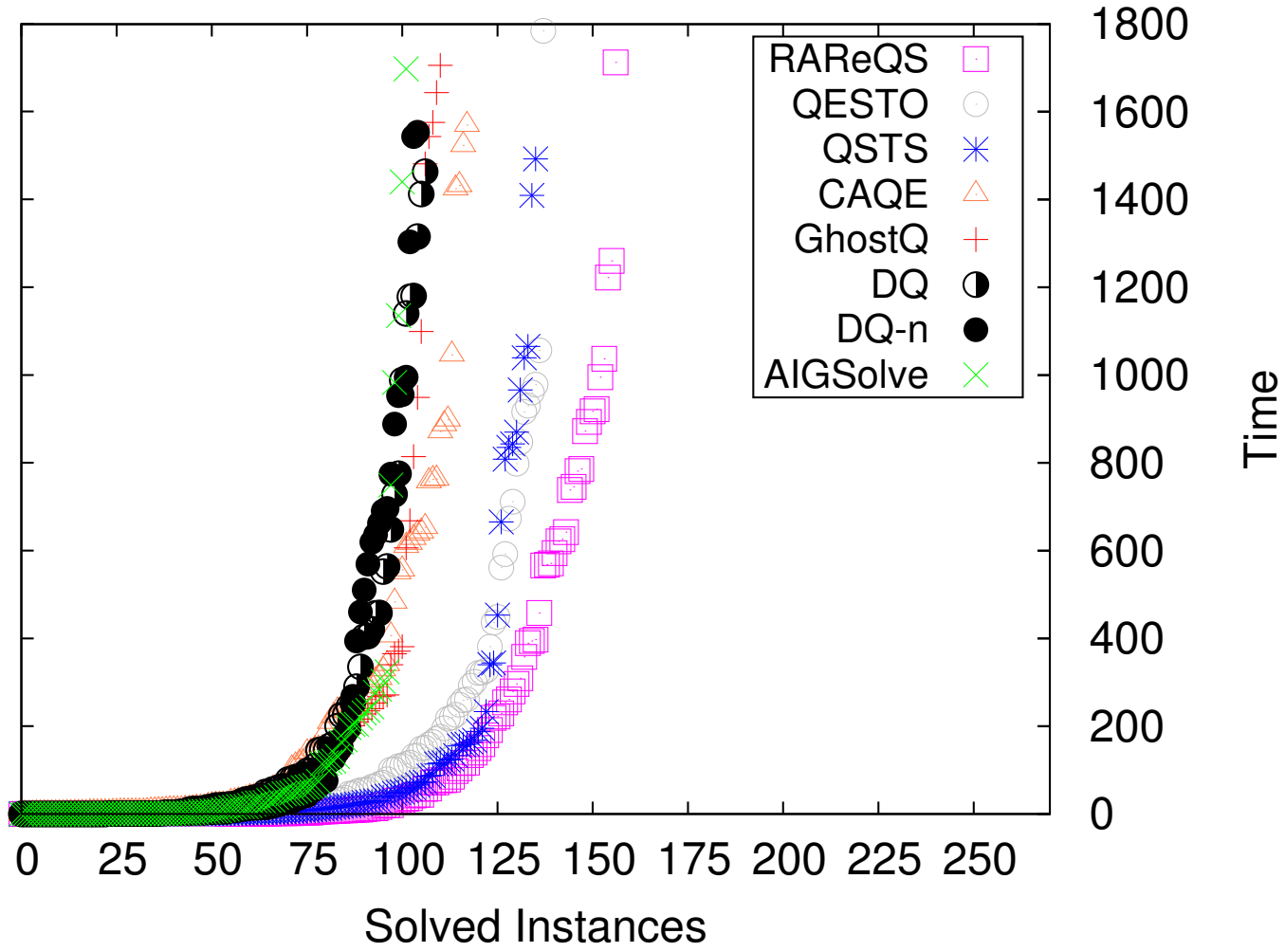
	Solved
DQ	**79**
QSTS	72
GhostQ	56
DQ-n	**55**
AIGSolve	54
QESTO	49
CAQE	46
RAReQS	43



- 141 instances (35%), $\geq$ 3 alternations, filtered but *not* preprocessed.
- DQ outperforms other solvers on instances with many alternations.

Experiments 3/3: 402 Filtered Instances

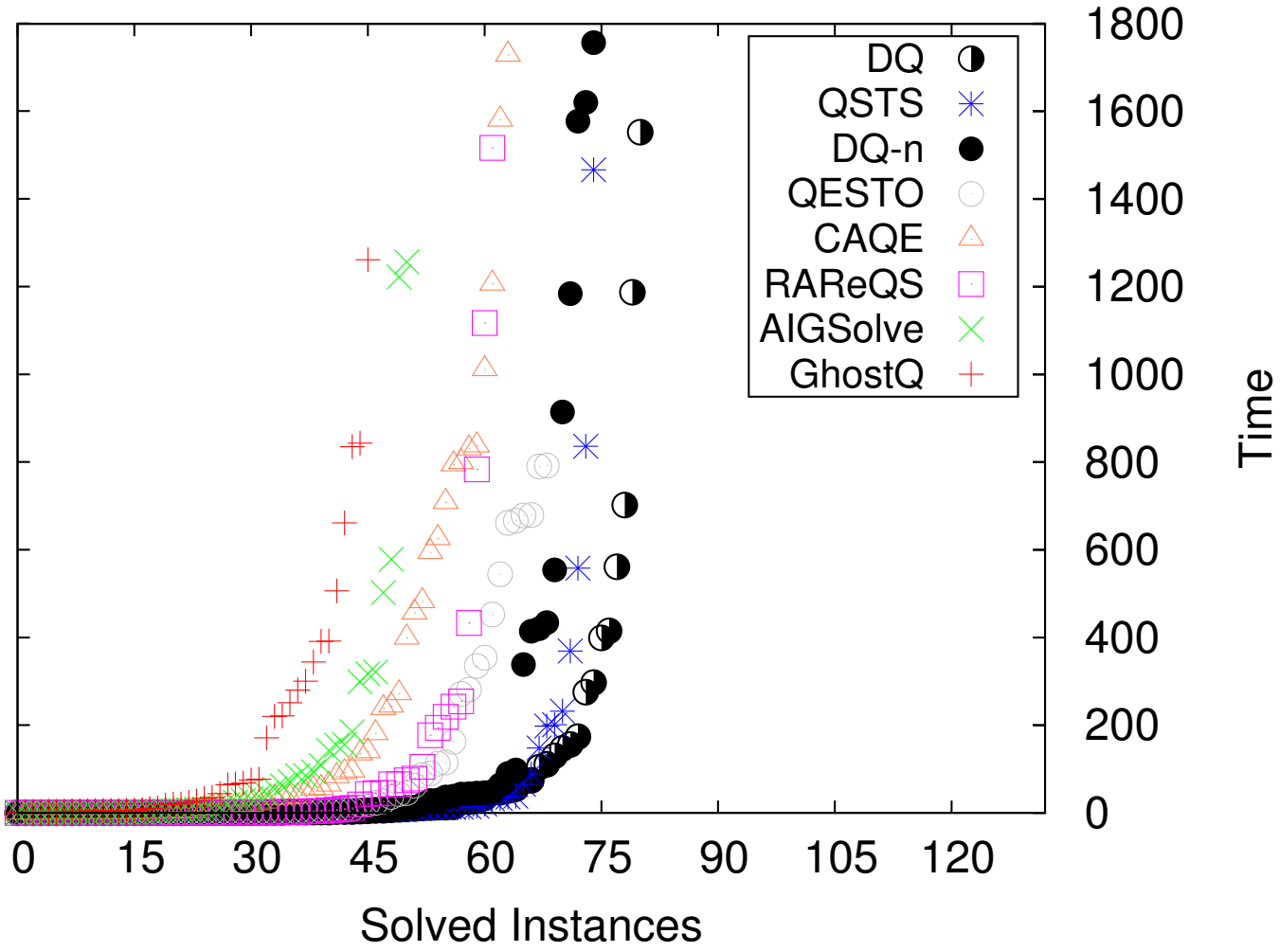
	Solved
RAReQS	157
QESTO	138
QSTS	136
CAQE	118
GhostQ	111
DQ	**107**
DQ-n	**105**
AIGSolve	102



- 270 instances (67%), ≤ 2 alternations, filtered *and* preprocessed.

Experiments 3/3: 402 Filtered Instances

	Solved
DQ	**81**
QSTS	75
DQ-n	**75**
QESTO	69
CAQE	64
RAReQS	62
AIGSolve	51
GhostQ	46



- 132 instances (33%), $\geq$ 3 alternations, filtered *and* preprocessed.
- DQ outperforms other solvers on instances with many alternations.

Summary

- QCDCL: QBF-specific variant of CDCL based on the Q-resolution calculus (QRES).
- Extension of QRES by additional axioms to derive clauses and cubes.
- Derivations by axioms rely on applications of incomplete QBF decision procedures.
- Axioms provide an interface to integrate arbitrary QBF proof systems.
- Theory: extension of QRES is exponentially stronger.
- DepQBF 6.0: implementation of QCDCL based on QRES extension.
- QCDCL framework unchanged: support of all Q-resolution variants.
- Experiments highlight potential, but fine-tuning required.

Free software (GPLv3): `http://lonsing.github.io/depqbf/`

APPENDIX

[APPENDIX] Experiments

Table : Solved instances (S), solved unsatisfiable ($\bot$) and satisfiable ones ($\top$), uniquely solved ones among all solvers (U), and total wall clock time including time outs on 825 PCNFs from QBFEVAL'16 without (1a) and with preprocessing by Bloqqer (1b).

Solver	S	$\bot$	$\top$	U	*Time*
AIGSolve	603	301	302	34	440K
GhostQ	593	292	301	7	457K
QSTS	578	294	284	3	469K
DQ	458	255	203	0	682K
DQ-linldq	458	257	201	2	686K
DQ-lin	456	255	201	0	686K
DQ-ncl	448	246	202	0	703K
DQ-nq	397	228	169	0	788K
DQ-ncu	393	229	164	0	796K
DQ-n	383	221	162	0	814K
CAQE	378	202	176	9	831K
QESTO	369	210	159	0	864K
RAReQS	341	211	130	2	891K

(a) Original instances.

Solver	S	$\bot$	$\top$	U	*Time*
QSTS	633	330	303	11	365K
RAReQS	633	334	299	8	375K
QESTO	620	321	299	0	395K
DQ-ncl	601	303	298	0	428K
DQ	601	301	300	0	429K
DQ-linldq	598	300	298	2	437K
DQ-lin	597	299	298	0	436K
CAQE	596	301	295	4	451K
DQ-n	593	296	297	0	444K
DQ-ncu	591	297	294	0	455K
DQ-nq	587	293	294	0	455K
GhostQ	570	282	288	0	485K
AIGSolve	567	286	281	14	481K

(b) Preprocessed by Bloqqer.

[APPENDIX] Experiments

Table : Related to Table 1a: solver performance on 402 filtered original (*not preprocessed*) instances partitioned into 261 instances with at most two (2a) and 141 with three or more quantifier alternations (2b).

Solver	*S*	⊥	⊤	*U*	*Time*
GhostQ	176	75	101	5	171K
AIGSolve	138	66	72	14	250K
QSTS	136	58	78	0	232K
RAReQS	76	43	33	1	340K
DQ-lin	69	35	34	0	351K
DQ	69	35	34	0	351K
DQ-ncl	68	35	33	0	354K
DQ-linldq	67	34	33	0	354K
QESTO	66	37	29	0	359K
DQ-ncu	53	24	29	0	378K
DQ-n	52	24	28	0	378K
DQ-nq	52	23	29	0	379K
CAQE	43	17	26	3	397K

(a) $\leq$ 2 quantifier alternations.

Solver	*S*	⊥	⊤	*U*	*Time*
DQ-linldq	81	50	31	2	120K
DQ	79	47	32	0	119K
DQ-ncl	79	47	32	0	120K
DQ-lin	78	47	31	0	123K
QSTS	72	44	28	3	132K
DQ-nq	56	37	19	0	159K
GhostQ	56	31	25	2	160K
DQ-n	55	36	19	0	159K
DQ-ncu	55	36	19	0	159K
AIGSolve	54	25	29	9	161K
QESTO	49	33	16	0	179K
CAQE	46	29	17	2	182K
RAReQS	43	33	10	0	180K

(b) $\geq$ 3 quantifier alternations.

[APPENDIX] Experiments

Table : Related to Table 1b: solver performance on 402 filtered and *preprocessed* instances partitioned into 270 instances with at most two (3a) and 132 with three or more quantifier alternations (3b).

Solver	*S*	$\bot$	$\top$	*U*	*Time*
RAReQS	157	79	78	8	227K
QESTO	138	66	72	0	255K
QSTS	136	62	74	2	255K
CAQE	118	49	69	2	298K
GhostQ	111	46	65	1	304K
DQ	107	43	64	1	311K
DQ-lin	106	42	64	0	311K
DQ-ncl	105	43	62	0	312K
DQ-n	105	41	64	0	313K
DQ-linldq	104	40	64	0	315K
AIGSolve	102	49	53	7	313K
DQ-nq	102	39	63	0	322K
DQ-ncu	102	40	62	0	323K

(a) ≤ 2 quantifier alternations.

Solver	*S*	$\bot$	$\top$	*U*	*Time*
DQ-ncl	83	51	32	0	96K
DQ	81	49	32	0	98K
DQ-linldq	81	51	30	2	102K
DQ-lin	78	48	30	0	105K
DQ-ncu	76	48	28	0	112K
QSTS	75	50	25	1	107K
DQ-n	75	46	29	0	112K
DQ-nq	72	45	27	0	113K
QESTO	69	45	24	0	120K
CAQE	64	42	22	0	136K
RAReQS	62	45	17	1	131K
AIGSolve	51	27	24	6	151K
GhostQ	46	26	20	0	162K

(b) ≥ 3 quantifier alternations.

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.

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

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

[BLS11] Armin Biere, Florian Lonsing, and Martina Seidl.
Blocked Clause Elimination for QBF.
In *CADE*, volume 6803 of *LNCS*, pages 101–115. Springer, 2011.

[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.

References II

[JKMSC16] Mikolás Janota, William Klieber, João Marques-Silva, and Edmund Clarke.
Solving QBF with Counterexample Guided Refinement.
Artif. Intell., 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.

[KBKF95] Hans Kleine Büning, Marek Karpinski, and Andreas Flögel.
Resolution for Quantified Boolean Formulas.
Inf. Comput., 117(1):12–18, 1995.

[LB10a] Florian Lonsing and Armin Biere.
DepQBF: A Dependency-Aware QBF Solver.
JSAT, 7(2-3):71–76, 2010.

[LB10b] Florian Lonsing and Armin Biere.
Integrating Dependency Schemes in Search-Based QBF Solvers.
In *SAT*, volume 6175 of *LNCS*, pages 158–171. Springer, 2010.

References III

[LBB+15] 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.

[LE14] Florian Lonsing and Uwe Egly.
Incremental QBF Solving.
In *CP*, volume 8656 of *LNCS*, pages 514–530. Springer, 2014.

[LE15] Florian Lonsing and Uwe Egly.
Incrementally Computing Minimal Unsatisfiable Cores of QBFs via a Clause Group Solver API.
In *SAT*, volume 9340 of *LNCS*, pages 191–198. Springer, 2015.

[LE17] Florian Lonsing and Uwe Egly.
Evaluating QBF Solvers: Quantifier Alternations Matter.
CoRR, abs/1701.06612, 2017.
technical report.

[LES16] Florian Lonsing, Uwe Egly, and Martina Seidl.
Q-Resolution with Generalized Axioms.
In *SAT*, volume 9710 of *LNCS*, pages 435–452. Springer, 2016.

References IV

[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.

[LEVG13] Florian Lonsing, Uwe Egly, and Allen Van Gelder.
Efficient Clause Learning for Quantified Boolean Formulas via QBF Pseudo Unit Propagation.
In *SAT*, volume 7962 of *LNCS*, pages 100–115. Springer, 2013.

[NPL+12] Aina Niemetz, Mathias Preiner, Florian Lonsing, Martina Seidl, and Armin Biere.
Resolution-Based Certificate Extraction for QBF - (Tool Presentation).
In *SAT*, volume 7317 of *LNCS*, pages 430–435. Springer, 2012.

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

[WRMB17] Ralf Wimmer, Sven Reimer, Paolo Marin, and Bernd Becker.
HQSpre - An Effective Preprocessor for QBF and DQBF.
In *Proc. of the 23rd Int. Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2017)*, volume 10205 of *LNCS*, pages 373–390. Springer, 2017.

References V

[ZM02] Lintao Zhang and Sharad Malik.
Conflict Driven Learning in a Quantified Boolean Satisfiability Solver.
In *ICCAD*, pages 442–449. ACM / IEEE Computer Society, 2002.