Artificial Intelligence 336 (2024) 104194

Contents lists available at ScienceDirect

Artificial Intelligence

journal homepage: www.elsevier.com/locate/artint

QCDCL with cube learning or pure literal elimination – What is best? ☆

Benjamin Böhm [a,*], Tomáš Peitl [b,1,*], Olaf Beyersdorff [a,2,*]

[a] *Friedrich Schiller University Jena, Jena, Germany*
[b] *TU Wien, Vienna, Austria*

ARTICLE INFO

Keywords:
Lower bounds
Proof complexity
QBF proof systems
Quantified Boolean formulas
Resolution
SAT solving

ABSTRACT

Quantified conflict-driven clause learning (QCDCL) is one of the main approaches for solving quantified Boolean formulas (QBF). We formalise and investigate several versions of QCDCL that include cube learning and/or pure-literal elimination, and formally compare the resulting solving variants via proof complexity techniques. Our results show that almost all of the QCDCL variants are exponentially incomparable with respect to proof size (and hence solver running time), pointing towards different orthogonal ways how to practically implement QCDCL.

1. Introduction

SAT solving has revolutionised the way we perceive computationally hard problems. Determining the satisfiability of propositional formulas (SAT) has traditionally been viewed as intractable due to its NP completeness. In contrast, modern SAT solvers today routinely solve huge industrial instances of SAT from a wide variety of application domains [10]. This success of solving has not stopped at SAT, but in the last two decades was lifted to increasingly more challenging computational settings, with solving quantified Boolean formulas (QBF)—a PSPACE-complete problem—receiving key attention [8].

Conflict driven clause learning (CDCL) is the main paradigm of modern SAT solving [24]. Based on the classic DPLL algorithm from the 1960s, it combines a number of advanced features, including clause learning, efficient Boolean constraint propagation, decision heuristics, restart strategies, and many more. In QBF there exist several competing approaches to solving, with lifting CDCL to the quantified level in the form of QCDCL as one of the main paradigms [30], implemented e.g. in the state-of-the-art solvers DepQBF [22] and Qute [25].

For SAT/QBF solving, two questions of prime theoretical and practical importance are: (1) why are SAT/QBF solvers so effective and on which formulas do they fail? (2) Which solving ingredients are most important for their performance?

For (1), proof complexity offers the main theoretical approach to analyse the strength of solving [16,8,9]. In a breakthrough result, [26] and [1] established that CDCL on unsatisfiable formulas is equivalent to the resolution proof system, in the sense that from a CDCL run a resolution proof can be efficiently extracted [3], and conversely, each resolution proof can be efficiently simulated by CDCL [26]. Hence the well-developed proof-complexity machinery for proof size lower bounds in resolution [21] is directly applicable to show lower bounds for running time in CDCL.

☆ An extended abstract of this paper was published at IJCAI’22 [14].

* Corresponding authors.
E-mail address: benjamin.boehm@uni-jena.de (B. Böhm).

[1] Supported by FWF grant J-4361 (Austrian Science Fund).
[2] Supported by the Carl Zeiss Foundation and DFG grant BE 4209/3-1.

https://doi.org/10.1016/j.artint.2024.104194
Received 28 June 2023; Received in revised form 11 July 2024; Accepted 27 July 2024

Available online 8 August 2024

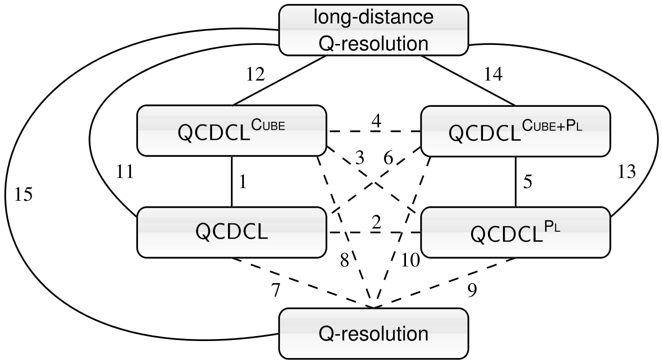

Fig. 1. Hasse diagram of the simulation order of QCDCL proof systems. Solid lines represent p-simulations and exponential separations (where the system depicted above is the stronger one). Dashed lines represent separations in both directions (i.e., incomparability). Details of the simulations and separations are depicted in Tables 1 and 2.

The latter simulation of resolution by CDCL assumes a strong 'non-deterministic' version of CDCL, whereas practical CDCL (using decision heuristics such as VSIDS) has been recently proved to be exponentially weaker than resolution [29]. In contrast, an analogous proof-theoretic characterisation is not known for QCDCL, and in particular QCDCL has recently been shown to be incomparable to Q-Resolution [4], the QBF analogue of propositional resolution [19].

Regarding question (2) above, there are some experimental studies [27,17,20], but no rigorous theoretical results are known on which (Q)CDCL ingredients are most crucial for performance. Of course, gaining such a theoretical understanding would also be very valuable in guiding future solving developments.

In this paper, we contribute towards question (2) in QBF.

Our contributions. Following the approach of [4], we model QCDCL as rigorously defined proof systems that are amenable to a proof-complexity analysis. This involves formalising individual QCDCL ingredients, such as clause and cube learning and different variants of Boolean constraint propagation. These components can then be 'switched' on or off, resulting in a number of different QCDCL solving approaches that we can formally investigate. Throughout we adopt the most common variable selection strategy of practical QCDCL that decides variables in the order of the prefix from left to right. More flexible decision strategies are used in dependency learning [25] or in [15,13].

Though we present the different QCDCL solving paradigms as formal proof systems in order to theoretically investigate them, they still retain a certain algorithmic flavour. We also note that in contrast to most conventional proof systems, the QCDCL systems are not rule-based, but are defined using QCDCL trails.

Our results can be summarised as follows.

(a) QCDCL with or without cube learning. In contrast to SAT solving, where there is somewhat of an asymmetry between satisfiable and unsatisfiable formulas, QCDCL implements a dual approach for false and true QBFs. In addition to learning clauses (as in CDCL) when running into a conflict under the current assignment, QCDCL also learns terms (or cubes) in the case a satisfying assignment is found (or a previously learned cube is satisfied). While cube learning is necessary to make QCDCL solving complete on true QBFs, it is less clear what the effect of cube learning is on false QBFs (and we only consider those throughout the paper as we cast all our variants in terms of refutational proof systems, in accordance with the proof complexity analysis of SAT [16]).

Here we establish the perhaps surprising result that even for false QBFs, cube learning can be advantageous, in the sense that QCDCL without cube learning (as a proof system for false QBFs) is exponentially weaker than QCDCL with cube learning (Theorems 5.3 and 6.11).

(b) QCDCL with or without pure-literal elimination. In its simplest form, Boolean constraint propagation, used to construct trails in (Q)CDCL, implements unit propagation. However, further methods can be additionally employed (and are considered in pre- and in-processing [11]). One of the classic mechanisms is pure-literal elimination, setting a pure literal (which occurs in only one polarity) to the obvious value. This is e.g. implemented in DepQBF and an efficient implementation is described by [23].

We show that QCDCL with or without pure-literal elimination results in incomparable proof systems (Theorem 5.13), i.e., there are QBFs that are easy in QCDCL with pure literal elimination, but hard in plain QCDCL, and vice versa (the latter is perhaps more surprising).

(c) Comparing QCDCL extensions. Given the preceding results, it is natural (and possibly most interesting for practice) to ask how the different QCDCL extensions compare with each other. We consider QCDCL with cube learning, QCDCL with pure-literal elimination but without cube learning, and QCDCL with both cube learning and pure-literal elimination. Except for the simulation of the second by the third system, we again obtain incomparability results between the systems with exponential separations (Theorem 6.5). We further show that all these systems are incomparable to Q-Resolution, again via exponential separations (Theorem 7.1). An overview of the systems and their relations is given in Fig. 1.

Technically, our results rest on formalising QCDCL systems as proof calculi and exhibiting specific QBFs for their separations. The latter includes both the explicit construction of short QCDCL runs and proving exponential proof size lower bounds for the relevant

Table 1
P-simulations and separations of proof systems from Fig. 1 (i.e. the solid lines).

No	Simulation Theorem	Separation Formula	easy for	hard for	Theorem
1	Proposition 5.1	$\mathtt{Eq}_n$	$\mathsf{QCDCL}^{\mathsf{CUBE}}$	QCDCL	Theorem 5.3
5	Proposition 5.1	$\mathtt{BulkyEq}_n$	$\mathsf{QCDCL}^{\mathsf{CUBE+PL}}$	$\mathsf{QCDCL}^{\mathsf{PL}}$	Theorem 6.11
11	by Def.	$\mathtt{CR}_n$	LD Q-Res	QCDCL	[12,18]
12	by Def.	$\mathtt{MirrorCR}_n$	LD Q-Res	$\mathsf{QCDCL}^{\mathsf{CUBE}}$	Proposition 7.1
13	by Def.	$\mathtt{MirrorCR}_n$	LD Q-Res	$\mathsf{QCDCL}^{\mathsf{PL}}$	Proposition 7.1
14	by Def.	$\mathtt{MirrorCR}_n$	LD Q-Res	$\mathsf{QCDCL}^{\mathsf{CUBE+PL}}$	Proposition 7.1
15	by Def.	$\mathtt{Eq}_n$	LD Q-Res	Q-Res	[6]

Table 2
Separations between incomparable proof systems from Fig. 1 (i.e. the dashed lines).

No	Formula	easy for	hard for	Theorem
2	$\mathtt{PLTrap}_n$	QCDCL	$\mathsf{QCDCL}^{\mathsf{PL}}$	Proposition 5.11, Proposition 5.12
	$\mathtt{Eq}_n$	$\mathsf{QCDCL}^{\mathsf{PL}}$	QCDCL	Proposition 5.4, [4]
3	$\mathtt{PLTrap}_n$	$\mathsf{QCDCL}^{\mathsf{CUBE}}$	$\mathsf{QCDCL}^{\mathsf{PL}}$	Proposition 5.11, Proposition 5.12
	$\mathtt{TwinEq}_n$	$\mathsf{QCDCL}^{\mathsf{PL}}$	$\mathsf{QCDCL}^{\mathsf{CUBE}}$	Proposition 6.3, Proposition 6.4
4	$\mathtt{PLTrap}_n$	$\mathsf{QCDCL}^{\mathsf{CUBE}}$	$\mathsf{QCDCL}^{\mathsf{CUBE+PL}}$	Proposition 6.7
	$\mathtt{TwinEq}_n$	$\mathsf{QCDCL}^{\mathsf{CUBE+PL}}$	$\mathsf{QCDCL}^{\mathsf{CUBE}}$	Proposition 6.3, Proposition 6.4
6	$\mathtt{PLTrap}_n$	QCDCL	$\mathsf{QCDCL}^{\mathsf{CUBE+PL}}$	Proposition 5.12, Proposition 6.7
	$\mathtt{Eq}_n$	$\mathsf{QCDCL}^{\mathsf{CUBE+PL}}$	QCDCL	Proposition 5.4, [4]
7	$\mathtt{QParity}_n$	QCDCL	Q-Res	[4,7]
	$\mathtt{CR}_n$	Q-Res	QCDCL	[12,18]
8	$\mathtt{QParity}_n$	$\mathsf{QCDCL}^{\mathsf{CUBE}}$	Q-Res	Theorem 7.1, [7]
	$\mathtt{MirrorCR}_n$	Q-Res	$\mathsf{QCDCL}^{\mathsf{CUBE}}$	Proposition 7.1
9	$\mathtt{QParity}_n$	$\mathsf{QCDCL}^{\mathsf{CUBE}}$	Q-Res	Theorem 7.1, [7]
	$\mathtt{MirrorCR}_n$	Q-Res	$\mathsf{QCDCL}^{\mathsf{CUBE}}$	Proposition 7.1
10	$\mathtt{QParity}_n$	$\mathsf{QCDCL}^{\mathsf{CUBE}}$	Q-Res	Theorem 7.1, [7]
	$\mathtt{MirrorCR}_n$	Q-Res	$\mathsf{QCDCL}^{\mathsf{CUBE}}$	Proposition 7.1

calculi. For the lower bounds, we identify a property of proofs (called primitivity here) that allows to use proof-theoretic machinery of [12] in the context of our QCDCL systems.

Our theoretical results on the strength of different QCDCL variants are empirically confirmed by experiments with state-of-the-art QCDCL solvers (cf. Section 8).

Organisation. We start in Section 2 by reviewing QBFs and Q-Resolution. In Section 3 we model variants of QCDCL as formal proof systems and develop a lower bound technique for such systems in Section 4. Sections 5 to 8 then contain our results on the relative strength of QCDCL variants. We conclude in Section 9 with an outlook on future research.

2. Preliminaries

Propositional and quantified formulas. Variables x and negated variables $\bar{x}$ are called *literals*. We denote the corresponding variable as $\text{var}(x) := \text{var}(\bar{x}) := x$. We will sometimes also use $\bot$ (*verum*) and $\top$ (*falsum*) as meta-literals.

A *clause* is a disjunction of literals, interpreted as a set of literals. A *unit clause* (ℓ) contains only one literal. The *empty clause* consists of zero literals, denoted $(\bot)$. A clause C is called *tautological* if $\{\ell, \bar{\ell}\} \subseteq C$ for some literal ℓ.

A *cube* is a conjunction of literals, viewed as a set of literals. We define a *unit cube* of a literal ℓ, denoted by $[\ell]$, and the *empty cube* $[\top]$ with 'empty literal' $\top$. A cube D is *contradictory* if $\{\ell, \bar{\ell}\} \subseteq D$ for some literal ℓ. If C is a clause or a cube, we define $\text{var}(C) := \{\text{var}(\ell) : \ell \in C\}$. The negation of a clause $C = \ell_1 \vee \ldots \vee \ell_m$ is the cube $\neg C := \bar{\ell}_1 \wedge \ldots \wedge \bar{\ell}_m$.

A *(total) assignment* σ of a set of variables V is a non-tautological set of literals such that for all $x \in V$ there is some $\ell \in \sigma$ with $\text{var}(\ell) = x$. A *partial assignment* σ of V is an assignment of a subset $W \subseteq V$. A clause C is *satisfied* by an assignment σ if $C \cap \sigma \neq \emptyset$. A cube D is *falsified* by σ if $\neg D \cap \sigma \neq \emptyset$. A clause C not satisfied by σ can be *restricted* by σ, defined as $C|_\sigma := \bigvee_{\ell \in C, \bar{\ell} \notin \sigma} \ell$. Similarly we can restrict a non-falsified cube D as $D|_\sigma := \bigwedge_{\ell \in D \setminus \sigma} \ell$.

A *CNF* (conjunctive normal form) is a conjunction of clauses and a *DNF* (disjunctive normal form) is a disjunction of cubes. We restrict a CNF (DNF) ϕ by an assignment σ as $\phi|_\sigma := \bigwedge_{C \in \phi \text{ non-satisfied}} C|_\sigma$ (resp. $\phi|_\sigma := \bigvee_{D \in \phi \text{ non-falsified}} D|_\sigma$). For a CNF (DNF) ϕ and an assignment σ, if $\phi|_\sigma = \emptyset$, then ϕ is *satisfied* (*falsified*) by σ.

A literal ℓ that appears in a clause of a CNF ϕ is called *pure* in ϕ if $\bar{\ell}$ does not occur in ϕ.

A *QBF* (quantified Boolean formula) $\Phi = \mathcal{Q} \cdot \phi$ consists of a propositional formula ϕ, called the *matrix*, and a *prefix* $\mathcal{Q}$. A *prefix* $\mathcal{Q} = \mathcal{Q}'_1 V_1 \ldots \mathcal{Q}'_s V_s$ consists of non-empty and pairwise disjoint sets of variables $V_1, \ldots, V_s$ and quantifiers $\mathcal{Q}'_1, \ldots, \mathcal{Q}'_s \in \{\exists, \forall\}$ with

$\mathcal{Q}'_i \neq \mathcal{Q}'_{i+1}$ for $i \in [s-1]$. For a variable x in $\mathcal{Q}$, the *quantifier level* is $\mathrm{lv}(x) := \mathrm{lv}_\Phi(x) := i$, if $x \in V_i$. For $\mathrm{lv}_\Phi(\ell_1) < \mathrm{lv}_\Phi(\ell_2)$ we write $\ell_1 <_\Phi \ell_2$.

For a QBF $\Phi = \mathcal{Q} \cdot \phi$ with ϕ a CNF (DNF), we call Φ a *QCNF* (*QDNF*). We write $\mathfrak{C}(\Phi) := \phi$ (resp. $\mathfrak{D}(\Phi) := \phi$). Φ is an *AQBF* (augmented QBF), if $\phi = \psi \vee \chi$ with CNF ψ and DNF χ. Again we write $\mathfrak{C}(\Phi) := \psi$ and $\mathfrak{D}(\Phi) := \chi$.

We restrict a QCNF (QDNF) $\Phi = \mathcal{Q} \cdot \phi$ by an assignment σ as $\Phi|_\sigma := \mathcal{Q}|_\sigma \cdot \phi|_\sigma$, where $\mathcal{Q}|_\sigma$ is obtained by deleting all variables from $\mathcal{Q}$ that appear in σ. Analogously, we restrict an AQBF $\Phi = \mathcal{Q} \cdot (\psi \vee \chi)$ as $\Phi|_\sigma := \mathcal{Q}|_\sigma \cdot (\psi|_\sigma \vee \chi|_\sigma)$.

(Long-distance) Q-resolution and Q-consensus. Let C_1 and C_2 be two clauses (cubes). Let ℓ be a literal with $\mathrm{var}(\ell) \notin \mathrm{var}(C_1) \cup \mathrm{var}(C_2)$. The *resolvent* of $C_1 \vee \ell$ and $C_2 \vee \bar{\ell}$ over ℓ is defined as

$$(C_1 \vee \ell) \overset{\ell}{\bowtie} (C_2 \vee \bar{\ell}) := C_1 \vee C_2$$

(resp. $(C_1 \wedge \ell) \overset{\ell}{\bowtie} (C_2 \wedge \bar{\ell}) := C_1 \wedge C_2$).

Let $C := \ell_1 \vee \ldots \vee \ell_m$ be a clause from a QCNF or AQBF Φ such that $\ell_i \leqslant_\Phi \ell_j$ for all $i < j$, $i, j \in [m]$. Let k be minimal such that $\ell_k, \ldots, \ell_m$ are universal. Then we can perform a *universal reduction* step and obtain

$$\mathrm{red}^\forall(C) := \ell_1 \vee \ldots \vee \ell_{k-1}.$$

Analogously, we perform *existential reduction* on cubes. Let $D := \ell_1 \wedge \ldots \wedge \ell_m$ be a cube of a QDNF or AQBF Φ with $\ell_i \leqslant_\Phi \ell_j$ for all $i < j$, $i, j \in [m]$. Let k be minimal such that $\ell_k, \ldots, \ell_m$ are existential. Then $\mathrm{red}^\exists(D) := \ell_1 \wedge \ldots \wedge \ell_{k-1}$.

As defined by Kleine Büning et al. [19], a Q-resolution (Q-consensus) proof π from a QCNF (QDNF) or AQBF Φ of a clause (cube) C is a sequence of clauses (cubes) $\pi = (C_i)_{i=1}^m$, such that $C_m = C$ and for each C_i one of the following holds:

- *Axiom:* $C_i \in \mathfrak{C}(\Phi)$ (resp. $C_i \in \mathfrak{D}(\Phi)$);
- *Resolution:* $C_i = C_j \overset{x}{\bowtie} C_k$ with x existential (univ.), $j, k < i$, and C_i non-tautological (non-contradictory);
- *Reduction:* $C_i = \mathrm{red}^\forall(C_j)$ (resp. $C_i = \mathrm{red}^\exists(C_j)$) for some $j < i$.

We call C the *root* of π. [2] introduced an extension of Q-resolution (Q-consensus) proofs to long-distance Q-resolution (long-distance Q-consensus) proofs by replacing the resolution rule by

- *Resolution (long-distance):* $C_i = C_j \overset{x}{\bowtie} C_k$ with x existential (universal) and $j, k < i$. The resolvent C_i is allowed to contain tautologies such as $u \vee \bar{u}$ (resp. contradictions such as $u \wedge \bar{u}$), if u is universal (existential). If there is a universal (existential) $u \in \mathrm{var}(C_j) \cap \mathrm{var}(C_k)$, then we require $x <_\Phi u$.

Note that in Q-resolution (resp. Q-consensus) proofs there are no tautologies or contradictions allowed at all.

A Q-resolution (Q-consensus) or long-distance Q-resolution (Q-consensus) proof from Φ of the empty clause $(\bot)$ (the empty cube $[\top]$) is called a *refutation* (*verification*) of Φ. In that case, Φ is called *false* (*true*).

A proof system S *p-simulates* a system S', if every S' proof can be transformed in polynomial time into an S proof of the same formula.

3. Formal calculi for QCDCL versions

In this section we model different versions of QCDCL as formal proof systems (we sketch this only here; for background on QCDCL cf. [8]). For this we need to formalise QCDCL ingredients. We start with trails. A *trail* $\mathcal{T}$ for a QCNF Φ is a finite sequence of literals from Φ, including the empty literals $\bot$ and $\top$. In general, a trail has the form

$$\mathcal{T} = (p_{(0,1)}, \ldots, p_{(0,g_0)}; \mathbf{d_1}, p_{(1,1)}, \ldots, p_{(1,g_1)}; \ldots; \mathbf{d_r}, p_{(r,1)}, \ldots, p_{(r,g_r)}), \tag{3.1}$$

where the d_i are *decision literals* and $p_{(i,j)}$ are *propagated literals*. Decision literals are written in **boldface**. We use a semicolon before each decision to mark the end of a decision level. We write $x <_\mathcal{T} y$ if $x, y \in \mathcal{T}$ and x is left of y in $\mathcal{T}$.

Trails can be interpreted as (partial) assignments. If $\mathcal{T}$ is a trail, then $\mathcal{T}[i, j]$, for $i \in \{0, \ldots, r\}$ and $j \in \{0, \ldots, g_i\}$, is defined as the *subtrail* that contains all literals from $\mathcal{T}$ left of (and excluding) $p_{(i,j)}$ (resp. d_i, if $j = 0$). We define $\mathcal{T}[0, 0]$ as the empty trail. A trail $\mathcal{T}$ has *run into conflict* if $\bot \in \mathcal{T}$ or $\top \in \mathcal{T}$.

For each propagated literal $p_{(i,j)}$ in a trail $\mathcal{T}$ the formula must contain a clause or a cube that caused this propagation by becoming a unit clause or cube. We denote such a clause/cube by $\mathrm{ante}_\mathcal{T}(p_{(i,j)})$. We can only propagate existential literals via clauses and universal literals via cubes. Further restrictions will be dictated by the respective QCDCL version and their specific rules. Note that antecedent clauses occur in CDCL as well.

Example 3.1. Throughout the section, we shall demonstrate the various notions on a run of QCDCL on the formula

$$\Phi := \exists x \forall u \exists t (x \vee u \vee t) \wedge (\bar{x} \vee \bar{u} \vee \bar{t}) \wedge (x \vee u \vee \bar{t}) \wedge (\bar{x} \vee \bar{u} \vee t).$$

A trail is a record of the state of a QCDCL algorithm. For example, QCDCL running on the above formula could start with the branching decision $\bar{x}$, followed by the decision $\bar{u}$, and propagation $\bar{t}$. The trail at that moment would be $\mathcal{T}_0 = (\bar{\mathbf{x}}; \bar{\mathbf{u}}, \bar{t})$.

Simply put, our QCDCL proof systems can be interpreted as sequences of trails. These trails cannot be created arbitrarily, but have to follow special rules, depending on the variant. We consider the following four QCDCL variants:

- QCDCL, which can be seen as the plain variant where we can only make decisions following the level order of the quantifier prefix, make propagations using clauses and use classic clause learning. We will never learn or use cubes and pure-literal elimination is turned off.
- $\mathsf{QCDCL}^{\mathsf{CUBE}}$ is an extension of QCDCL in which we can learn cubes and use them for propagations. Decisions are still level-ordered and pure-literal elimination is turned off.
- $\mathsf{QCDCL}^{\mathsf{PL}}$ is an extension of QCDCL, where we decide literals out of order if they are pure in the current configuration (pure-literal elimination). All other decisions (which we call regular decisions) are still level ordered. Cube learning is turned off.
- $\mathsf{QCDCL}^{\mathsf{CUBE+PL}}$ is an extension of $\mathsf{QCDCL}^{\mathsf{PL}}$, in which cube learning is now allowed (as in $\mathsf{QCDCL}^{\mathsf{CUBE}}$).

For a formal definition of these four variants, the following table show which subsequently described rules hold for each variant. They are classified into propagation, decision and conflict rules.

QCDCL	$\mathsf{QCDCL}^{\mathsf{CUBE}}$	$\mathsf{QCDCL}^{\mathsf{PL}}$	$\mathsf{QCDCL}^{\mathsf{CUBE+PL}}$
EP	AP	EP	AP
LOD	LOD	PLD	PLD
CC	AC	CC	AC

(Existential propagation rule) EP: Each $p_{(i,j)}$ is either an existential literal from Φ or the empty literal $\bot$. For each $p_{(i,j)}$ there exists a clause $\mathrm{ante}_{\mathcal{T}}(p_{(i,j)}) \in \mathfrak{C}(\Phi)$ such that $\mathrm{red}^{\forall}\left(\mathrm{ante}_{\mathcal{T}}(p_{(i,j)})|_{\mathcal{T}[i,j]}\right) = (p_{(i,j)})$.

(Arbitrary propagation rule) AP: Each $p_{(i,j)}$ is some literal from Φ or one of the empty literals $\bot$ or $\top$. If $p_{(i,j)}$ is existential or $\bot$, then the condition from EP applies. If $p_{(i,j)}$ is universal or $\top$, then there exists a cube $\mathrm{ante}_{\mathcal{T}}(p_{(i,j)}) \in \mathfrak{D}(\Phi)$ such that $\mathrm{red}^{\exists}\left(\mathrm{ante}_{\mathcal{T}}(p_{(i,j)})|_{\mathcal{T}[i,j]}\right) = \left[\bar{p}_{(i,j)}\right]$.

We call such a clause (cube) $\mathrm{ante}_{\mathcal{T}}(p_{(i,j)})$ an *antecedent clause* (*antecedent cube*). The next rules specify how decisions are made.

(Level-ordered decision rule) LOD: For each d_i we have that $\Phi|_{\mathcal{T}[i,0]}$ does not contain unit or empty clauses or cubes. Also, $\mathrm{lv}_{\Phi|_{\mathcal{T}[i,0]}}(d_i) = 1$, i.e., decisions are level-ordered.

(Pure literal decision rule) PLD: For each d_i we have that $\Phi|_{\mathcal{T}[i,0]}$ does not contain any unit or empty clauses or cubes. Also, if there are pure literals in $\mathfrak{C}(\Phi|_{\mathcal{T}[i,0]})$, then the following holds: If d_i is existential, then d_i has to be pure in $\mathfrak{C}(\Phi|_{\mathcal{T}[i,0]})$. Otherwise, if d_i is universal, then $\bar{d_i}$ has to be pure in $\mathfrak{C}(\Phi|_{\mathcal{T}[i,0]})$. In that case we will underline $\mathbf{d_i}$ in $\mathcal{T}$. However, if $\mathfrak{C}(\Phi|_{\mathcal{T}[i,0]})$ does not contain any pure literals, then $\mathrm{lv}_{\Phi|_{\mathcal{T}[i,0]}}(d_i) = 1$, i.e., decision literals which are not pure have to be level-ordered.

From now on, we will distinguish *regular* decisions (not underlined) and decisions via *pure literal elimination* (underlined).

Example 3.2. Recall the trail $\mathcal{T}_0$ from Example 3.1. Upon closer inspection we may notice that after the decision $\bar{x}$, the literal u is pure, and thus the decision $\bar{u}$ can be performed by pure literal elimination. We obtain the trail $\mathcal{T}_1 = (\bar{\mathbf{x}}; \underline{\bar{\mathbf{u}}}, \bar{t})$.

The last pair of rules will determine how we handle conflicts in trails.

(Clause conflict rule) CC: If $\bot \in \mathcal{T}$, then $\bot = p_{(r,g_r)}$ and there is no point $[i,j]$ except $[r,g_r]$ such that there exists some $C \in \mathfrak{C}(\Phi|_{\mathcal{T}[i,j]})$ with $\mathrm{red}^{\forall}(C) = (\bot)$, i.e., we cannot delay conflicts, but are forced to handle them as soon as possible.

(Arbitrary conflict rule) AC: If $\bot \in \mathcal{T}$, then $\top \notin \mathcal{T}$ and vice versa. If there is an $\ell \in \{\bot, \top\}$ with $\ell \in \mathcal{T}$, then $\ell = p_{(r,g_r)}$ and there is no point $[i,j]$ except $[r,g_r]$ such that there exists some $C \in \mathfrak{C}(\Phi|_{\mathcal{T}[i,j]})$ or $D \in \mathfrak{D}(\Phi|_{\mathcal{T}[i,j]})$ with $\mathrm{red}^{\forall}(C) = (\bot)$ or $\mathrm{red}^{\exists}(D) = [\top]$.

Note that decisions can only be made if there are no more propagations possible and pure literal decisions always have a higher priority than regular decisions. Also, conflicts have a higher priority than propagations of proper (existential or universal) literals. Hence, we will never skip conflicts, propagations or pure literal decisions. Trails that follow these principles are called *natural*.

Example 3.3. Continuing Example 3.2, we may notice that the clause $(x \vee u \vee t)$ is falsified under the trail $\mathcal{T}_1$, and thus the trail may be extended by propagating $\bot$ to $\mathcal{T}_2 = (\bar{\mathbf{x}}; \underline{\bar{\mathbf{u}}}, \bar{t}, \bot)$. The antecedent for the propagation of $\bar{t}$ is $\mathrm{ante}_{\mathcal{T}_2}(\bar{t}) = (x \vee u \vee \bar{t})$, and the antecedent for $\bot$ is $\mathrm{ante}_{\mathcal{T}_2}(\bot) = (x \vee u \vee t)$. Now, the trail $\mathcal{T}_2$ has run into a conflict. Notice that we have indeed performed all conflict detection, unit propagation, and pure-literal elimination as soon as possible, in line with the various rules.

After a trail has run into a conflict, or if all variables are assigned, we can start the learning process.

Definition 3.4 *(learnable constraints)*. Let $\mathcal{T}$ be a trail for Φ of the form (3.1) with $p_{(r,g_r)} \in \{\bot, \top\}$. Starting with $\text{ante}_{\mathcal{T}}(\bot)$ (resp. $\text{ante}_{\mathcal{T}}(\top)$) we reversely resolve over the antecedent clauses (cubes) that propagated the existential (universal) variables, until we stop at some arbitrarily chosen point. The clause (cube) we so derive is a *learnable constraint*. We denote the set of learnable constraints by $\mathfrak{L}(\mathcal{T})$.

We can also learn cubes from trails that did not run into conflict. If $\mathcal{T}$ is a total assignment of the variables from Φ, then we define the set of learnable constraints as the set of cubes $\mathfrak{L}(\mathcal{T}) := \{\text{red}^{\exists}(D) | \ D \subseteq \mathcal{T} \text{ and } D \text{ satisfies } \mathfrak{C}(\Phi)\}$.

Example 3.5. As we attempt to recover from the conflict $\mathcal{T}_2$ has run into in Example 3.3, we backtrack and apply long-distance Q-resolution. We first resolve the two antecedent clauses $(x \vee u \vee \bar{t})$ and $(x \vee u \vee t)$, obtaining $(x \vee u)$, and after universal reduction (x). There are no more antecedent clauses to resolve with, so the full set of learnable clauses is $\mathfrak{L}(\mathcal{T}_2) = \{(x)\}$. We will next learn the only possible clause (x), backtrack, and continue down a different trail. The sequence of trails thus constructed constitutes a QCDCL proof system.

Definition 3.6 *(QCDCL proof systems)*. Let S be one of the previously described variants QCDCL, QCDCL$^{\text{CUBE}}$, QCDCL$^{\text{PL}}$, QCDCL$^{\text{CUBE+PL}}$. An S proof ι from a QCNF $\Phi = \mathcal{Q} \cdot \phi$ of a clause or cube C is a sequence of triples

$$\iota := [(\mathcal{T}_i, C_i, \pi_i)]_{i=1}^{m},$$

where $C_m = C$, each $\mathcal{T}_i$ is a trail of Φ_i, each $C_i \in \mathfrak{L}(\mathcal{T}_i)$ is one of the constraints we can learn from each trail and π_i is the long-distance Q-resolution or long-distance Q-consensus proofs from Φ_i of C_i we obtain by performing the steps in Definition 3.4. If necessary, we set $\pi_i := \emptyset$. We will denote the set of trails in ι as $\mathfrak{T}(\iota)$.

The QCNF or AQBF Φ_i is defined as follows: If S is one of QCDCL or QCDCL$^{\text{PL}}$, then we set $\Phi_1 := \Phi$ and

$$\Phi_{j+1} := \mathcal{Q} \cdot \left(\mathfrak{C}(\Phi_j) \wedge C_j\right).$$

However, if $S \in \{\text{QCDCL}^{\text{CUBE}}, \text{QCDCL}^{\text{CUBE+PL}}\}$, then the Φ_i are AQBFs defined as $\Phi_1 := \mathcal{Q} \cdot \left(\mathfrak{C}(\Phi) \vee \emptyset\right)$ and

$$\Phi_{j+1} := \begin{cases} \mathcal{Q} \cdot \left((\mathfrak{C}(\Phi_j) \wedge C_j) \vee \mathfrak{D}(\Phi_j)\right) & \text{if } C_j \text{ is a clause,} \\ \mathcal{Q} \cdot \left(\mathfrak{C}(\Phi_j) \vee (\mathfrak{D}(\Phi_j) \vee C_j)\right) & \text{if } C_j \text{ is a cube,} \end{cases}$$

for $j = 1, \dots, m-1$.

Furthermore, we require that $\mathcal{T}_1$ is a natural S trail and for each $2 \leqslant i \leqslant m$ there is a point $[a_i, b_i]$ such that $\mathcal{T}_i[a_i, b_i] = \mathcal{T}_{i-1}[a_i, b_i]$ and $\mathcal{T}_i \backslash \mathcal{T}_i[a_i, b_i]$ has to be a natural S trail for $\Phi_i|_{\mathcal{T}_i[a_i,b_i]}$. This process is called *backtracking*. We will also say that after $\mathcal{T}_{i-1}$ we backtrack back to the point $[a_i, b_i]$. If $\mathcal{T}_{i-1}[a_i, b_i] = \emptyset$, then this is also called a *restart*.

Note that we only require $\mathcal{T}_i \backslash \mathcal{T}_i[a_i, b_i]$ to be natural. However, since the first part always belongs to a previous trail, and the first trail in the proof is always natural, we can nevertheless use the notion of antecedent clauses for the whole trail $\mathcal{T}_i$. In particular, for all $\mathcal{T}_i$ either EP or AP holds, which we need for the learning process.

Unfortunately we cannot claim the same for LOD and PLD, because for a decision d_i in a trail $\mathcal{T}_k \in \mathfrak{T}(\iota)$ it might happen that $\Phi_k|_{\mathcal{T}_k[i,0]}$ contains unit or empty clauses or literals after clause learning and backtracking. However, we can still assume that the decisions are level-ordered, since the condition $\text{lv}_{\Phi_k|_{\mathcal{T}_k[i,0]}}(d_i) = 1$ is not affected by new clauses. Also, it could happen that a literal d_i that was originally decided by pure literal elimination in some trail $\mathcal{T}_k$ might not pure in $\mathfrak{C}(\Phi_{k+1}|_{\mathcal{T}_{k+1}[i,0]})$ anymore because of a new clause C_k. Nevertheless, this will not cause too many difficulties since we can always find the original trail (here: $\mathcal{T}_k$) in which d_i was in fact decided as a pure literal. Thus, when we say that a literal was decided by pure literal elimination in a trail $\mathcal{T}$, we will always refer to this original trail.

If $C = C_m = (\bot)$, then ι is called an S *refutation* of Φ. If $C = C_m = [\top]$, then ι is called an S *verification* of Φ. The proof ends once we have learned $(\bot)$ or $[\top]$.

If C is a clause, we can stick together the long-distance Q-resolution derivations from $\{\pi_1, \dots, \pi_m\}$ and obtain a long-distance Q-resolution proof from Φ of C, which we call $\mathfrak{R}(\iota)$. Similarly, if C is a cube, we can stick together the long-distance Q-consensus derivations and obtain a long-distance Q-consensus proof $\mathfrak{R}(\iota)$ from Φ of C.

The *size* of ι is defined as $|\iota| := \sum_{i=1}^{m} |\mathcal{T}_i|$. Obviously, we have $|\mathfrak{R}(\iota)| \in \mathcal{O}(|\iota|)$.

We say that S *p-simulates* another system S', if every S' proof ι' can be transformed in polynomial time into an S proof ι of the same formula.

Example 3.7. Finally, let us see the whole QCDCL run in one place. We begin with the decision $\bar{x}$, pure-literal elimination $\bar{u}$, propagation $\bar{t}$ and then $\bot$ to obtain the trail $\mathcal{T}_2$. We learn (x) and backtrack just before the last non-pure-literal decision to the empty trail $\mathcal{T}_3 = (;)$. The newly learned unit clause propagates x, u is then decided by pure-literal elimination, t propagated by the antecedent $(\bar{x} \vee \bar{u} \vee t)$, and $\bot$ by the antecedent $(\bar{x} \vee \bar{u} \vee \bar{t})$, arriving at the conflict trail $\mathcal{T}_4 = (x; \underline{u}, t, \bot)$. As we resolve backwards, we identify as learnable first the reduced unit clause $(\bar{x})$, and after resolving with $\text{ante}_{\mathcal{T}_4}(x) = (x)$ also the empty clause. By learning the empty clause, we complete the proof. In this particular case, because we used pure-literal elimination but no cube learning, we obtain a QCDCL$^{\text{PL}}$ proof.

Theorem 3.8. QCDCL*,* QCDCL$^{\textit{CUBE}}$*,* QCDCL$^{\textit{PL}}$*, and* QCDCL$^{\textit{CUBE+PL}}$ *are sound and complete proof systems.*

Proof. We start with the soundness. All Φ_i have the same truth value. In fact, either the newly added clauses (cubes) are derived from already known clauses (cubes) by long-distance Q-resolution (long-distance Q-consensus), which is a sound proof system, or we have added a cube $D \in \mathfrak{L}(\mathcal{T}_j)$ that can be extended to an assignment σ which satisfies $\mathfrak{C}(\Phi_j)$ and $\text{red}^{\exists}(\sigma) = D$. If adding such a D to $\mathfrak{D}(\Phi_j)$ would have changed the truth value from false for Φ_j to true for Φ_{j+1}, then there would be a strategy for the universal player that falsifies $\mathfrak{C}(\Phi_j) \vee \mathfrak{D}(\Phi_j)$ and the existential player would have a strategy that satisfies $\mathfrak{C}(\Phi_j) \vee \mathfrak{D}(\Phi_j) \vee D$. If both players play their strategy on Φ_{j+1}, then this would not satisfy $\mathfrak{C}(\Phi_j)$, but would satisfy D (and w.l.o.g. also σ). But then $\mathfrak{C}(\Phi_j)$ would be satisfied, contradiction.

For the completeness, we refer to [4] for a detailed argumentation, in which the completeness of QCDCL is proven, but provide a brief sketch here. We first show that we are always able to learn so-called *asserting clauses*, which are learnable clauses that become unit after backtracking. Because of this property, we can argue that these asserting clauses must be new (otherwise they would not trigger a new unit propagation). Since a QBF has finitely many variables, we can only learn finitely many new clauses until we reach the empty clause or cube at some point. Because each QCDCL refutation can be interpreted as $\text{QCDCL}^{\text{CUBE}}$ refutation, we immediately obtain completeness for $\text{QCDCL}^{\text{CUBE}}$.

For the two systems with pure literal elimination, we will argue similarly as in [4] and claim that we are also always able to learn asserting clauses. First, it is always possible to let a trail run into a conflict by deciding the universal literals according to a winning strategy for the universal player. We can assume that in this winning strategy universal pure literals are immediately set to false, since this will never be disadvantageous for the universal player. At some point, we will falsify the matrix and obtain a conflict, from which we can start clause learning.

In [4] we described how one can find asserting clauses in a conflicting trail for a particular QCDCL variant (which we have not defined here) in which we are allowed to decide universal literals earlier than it would be allowed with the LOD rule. This construction can be transferred to QCDCL^{PL} because universal pure literals are decided earlier, as well. We can ignore pure literal elimination for existential literals because they will always occur at a dead end (we cannot use them for further propagations). That means even if a trail contains existential literals that are decided out-of-order as pure literals, they will not interfere with finding asserting clauses as they will simply be ignored by clause learning.

We conclude that from each trail we will be able to learn asserting clauses that are always new. Since we only have a finite number of literals, there are also only a finite number of clauses to learn. At some point, we will learn the empty clause ($\bot$) and our QCDCL^{PL} proof ends. Due to the fact that QCDCL^{PL} proofs can be interpreted as $\text{QCDCL}^{\text{CUBE+PL}}$ proofs, we conclude that both systems are complete. □

We highlight that these systems formally model QCDCL solving as used in practice (cf. [8]).

4. Proving lower bounds for QCDCL systems

Throughout the paper we will concentrate on Σ_3^b QCNFs which we always assume to have the form $\Phi = \exists X \forall U \exists T \cdot \phi$ for non-empty blocks of variables X, U, and T.

A literal ℓ is an X-literal, if $\text{var}(\ell) \in X$. Analogously, we get U- and T-literals and variables. A clause $C \in \mathfrak{C}(\Phi)$ is an *X-clause*, if all its literals are X-literals. The empty clause ($\bot$) is also an X-clause. Analogously, we define *T-clauses*. A clause $C \in \mathfrak{C}(\Phi)$ is an *XT-clause*, if it contains at least one X-literal, at least one T-literal, but no U-literals; analogously we define *UT-clauses*. A clause $C \in \mathfrak{C}(\Phi)$ is an *XUT-clause* if it contains at least one X-, U- and T- literal.

Definition 4.1. We say that Φ fulfils the *XT-property*, if $\mathfrak{C}(\Phi)$ contains no XT-clauses, no T-clauses that are unit (or empty) and no two T-clauses from $\mathfrak{C}(\Phi)$ are resolvable.

As shown by [12], clause learning does not affect the XT-property, i.e., a formula Φ with the XT-property will still fulfil it during the whole QCDCL run even after having added new clauses to $\mathfrak{C}(\Phi)$.

Next we recall the definition of formula gauge from [12], which represents a measure that can be used for lower bounds (for a brief depiction of the gauge lower bound method, see Fig. 2).

Definition 4.2 *([12]).* For a QCNF Φ as above we define W_Φ to be the set of all Q-resolution derivations π from Φ of some X-clause such that π only contains resolutions over T-variables and reduction steps. We set

$$\text{gauge}(\Phi) := \min\{|C| : C \text{ is the root of some } \pi \in W_\Phi\}.$$

We now define fully reduced and primitive proofs. Our lower bound technique will then work for fully reduced primitive refutations of formulas that fulfil the XT-property.

Definition 4.3. A long-distance Q-resolution refutation π of a QCNF Φ is called *fully reduced*, if the following holds: For each clause $C \in \pi$ that contains universal literals that are reducible, the reduction step has to be performed immediately and C cannot be used otherwise in the proof.

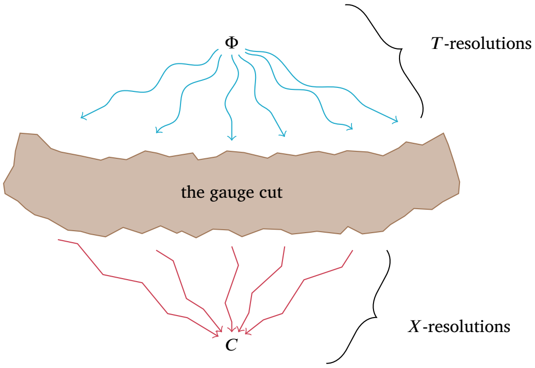

Fig. 2. The gauge lower bound method in brief. Theorem 4.5 postulates that primitive refutations of XT-formulas can be decomposed into a collection of subderivations like that shown, of some X-clause C, and in which all T-variables are resolved away before any X-variables. We can thus find a cut in each subderivation, and by Definition 4.2, the clauses in the cut have width at least gauge(Φ). By a combination of the width bound and a simple model counting argument, we obtain an exponential bound on the size of the original refutation.

For example, consider the clause $x \vee u \vee \bar{w}$ under the prefix $\exists x \forall u, w$. This clause is not fully reduced and can therefore not be used for resolution steps in a fully reduced proof. In order to use this clause in a resolution step, we need to reduce it to the clause $\mathrm{red}^{\forall}(x \vee u \vee \bar{w}) = (x)$.

Each proof $\mathfrak{R}(\iota)$ that was extracted from a QCDCL proof ι is automatically fully reduced, as we perform reduction steps as soon as possible during clause learning. On the other hand, primitivity does not hold for proofs $\mathfrak{R}(\iota)$ in general. In fact, the main work in proving our hardness results will be to show that specific extracted proofs are primitive.

Definition 4.4. A long-distance Q-resolution proof π from a Σ_3^b formula with the XT-property is *primitive*, if there are no two XUT-clauses in π that are resolved over an X-variable.

For example, consider the clauses $x \vee u \vee t$ and $\bar{x} \vee u \vee t$ under the prefix $\exists x \forall u \exists t$. Although these two clauses are resolvable in long-distance Q-resolution (even in Q-resolution), such a resolution step is not allowed in primitive long-distance Q-resolution.

Since it is not possible to derive tautological clauses in fully reduced primitive proofs, we may also refer to them as (fully reduced) primitive Q-resolution proofs.

It follows from [12], that these two conditions suffice to show lower bounds via gauge.

Theorem 4.5 *([12]). Each fully reduced primitive Q-resolution refutation of a Σ_3^b QCNF Φ that fulfils the XT-property has size $2^{\Omega(\mathrm{gauge}(\Phi))}$.*

Proof Sketch. We refer to the lower bound technique for so-called *quasi level-ordered* Q-resolution refutations (it is not necessary to define this notion here) devised in [12]. Its main result [12, Theorem 12] is an analogous statement of Theorem 4.5 for QCDCL refutations. However, for this result to hold, it is actually sufficient to start with a fully reduced primitive Q-resolution refutation, which is a weaker requirement. Hence, the gauge lower bound for fully reduced primitive Q-resolution refutations (Theorem 4.5) directly follows from [12, Theorem 12] with the identical proof, although the notion 'fully reduced primitive' was not used there. □

The next two results represent the main methodology for most of our hardness results throughout the paper.

Lemma 4.6. *Let $\mathcal{T}$ be a trail in a QCDCL, QCDCL^{CUBE}, QCDCL^{PL} or QCDCL$^{CUBE+PL}$ proof from a QCNF Φ with the XT-property. Then for each T-literal $t_1 \in \mathcal{T}$, which was not decided by pure literal elimination, there is a U-literal $u \in \mathcal{T}$ with $u <_{\mathcal{T}} t_1$.*

Proof. If t_1 was decided regularly, then the situation is clear because we can only decide T-literals if and only if all U-variables were assigned before. Therefore we can assume that there is no T-literal $t' \in \mathcal{T}$ with $t' \leqslant_{\mathcal{T}} t_1$ such that t' was a regular decision. □

Proposition 4.7. *Let ι be a QCDCL, QCDCL^{CUBE}, QCDCL^{PL} or QCDCL$^{CUBE+PL}$ refutation of a QCNF Φ that fulfils the XT-property. If $\mathfrak{R}(\iota)$ is not primitive, then there exists a trail $\mathcal{T} \in \mathfrak{T}(\iota)$ such that there is a U-literal $u \in \mathcal{T}$ and an X-literal $x \in \mathcal{T}$ with $u <_{\mathcal{T}} x$. Additionally, u cannot be a regular decision literal.*

Proof. If $\mathfrak{R}(\iota)$ is not primitive, then there are two XUT-clauses $C, D \in \mathfrak{R}(\iota)$ that are resolved over an X-variable x, say $x \in C$ and $\bar{x} \in D$. One of these clauses has to be an antecedent clause of some trail $\mathcal{T} \in \mathfrak{T}(\iota)$, w.l.o.g. let C be the antecedent clause $\mathrm{ante}_{\mathcal{T}}(x)$. Let $\bar{t} \in C$ be one of the T-literals from C. In particular, we have $t \in \mathcal{T}$ and $t <_{\mathcal{T}} x$. Because t was not a pure literal decision (we have $\bar{t} \in C$) and because of Lemma 4.6, there is a U-literal $u \in \mathcal{T}$ with $u <_{\mathcal{T}} t$. We conclude that also $u <_{\mathcal{T}} x$ holds.

Since we can only decide U-literals regularly if all X-variables are assigned in some polarity in $\mathcal{T}$, it is impossible for u to be a regular decision literal. □

Basically, this result tells us that for a non-primitive proof $\Re(\iota)$ of some S proof ι, where S is one of our four QCDCL variants, ι needs to consist of a trail that assigns a U-literal out-of-order (i.e., before we have assigned all X-literals).

Since neither cube learning nor pure literal elimination is allowed in QCDCL, we can immediately conclude:

Corollary 4.8. *Let ι be a* QCDCL *refutation of a QCNF Φ that fulfils the XT-property. Then $\Re(\iota)$ is primitive.*

We remark that some of the QBFs we introduce in the paper are not minimally false, i.e., we have added extra clauses to formulas that were false already. Although this is unusual in proof complexity, practical (false) instances are not guaranteed to be minimally false. Therefore it is natural to also consider these QBFs when investigating QCDCL systems. These algorithmic proof systems have to utilise all clauses, even if they are redundant for Q-resolution refutations.

5. Plain QCDCL vs. extensions with cubes/PL

We start by examining the influence of cube learning on our QCDCL variant. For false formulas we can always prevent learning cubes by just deciding the universal variables according to a winning strategy for the universal player, which will cause a conflict on the current trail. Thus cube learning will never be disadvantageous in principle.

Proposition 5.1. $\mathsf{QCDCL}^{\mathit{CUBE}}$ *p-simulates* QCDCL *and* $\mathsf{QCDCL}^{\mathit{CUBE+PL}}$ *p-simulates* $\mathsf{QCDCL}^{\mathit{PL}}$.

Proof. A QCDCL ($\mathsf{QCDCL}^{\mathsf{PL}}$) proof translates into a $\mathsf{QCDCL}^{\mathsf{CUBE}}$ ($\mathsf{QCDCL}^{\mathsf{CUBE+PL}}$) proof where all trails run into conflict and no cubes are learnt. □

We recall the equality formulas $\mathtt{Eq}_n$ of [5]. These are QCNFs with prefix

$$\exists x_1 \dots x_n \forall u_1 \dots u_n \exists t_1 \dots t_n$$

and matrix

$$(\bar{t}_1 \vee \dots \vee \bar{t}_n) \wedge \bigwedge_{i=1}^{n} ((\bar{x}_i \vee \bar{u}_i \vee t_i) \wedge (x_i \vee u_i \vee t_i)).$$

The formulas are known to be hard for Q-resolution [5] and also for QCDCL [4]. In contrast, we show that they are easy in QCDCL with cube learning.

Proposition 5.2. *There exist polynomial-size* $\mathsf{QCDCL}^{\mathit{CUBE}}$ *refutations of* $\mathtt{Eq}_n$.

Proof. First we learn the cubes $x_i \wedge \bar{u}_i$ and $\bar{x}_i \wedge u_i$ for $i = 1, \dots, n-1$. In order to learn $x_1 \wedge \bar{u}_1$, we can use the trail

$$\mathcal{T}_1 := (\mathbf{x_1}; \dots; \mathbf{x_n}; \bar{\mathbf{u}}_\mathbf{1}; \dots; \bar{\mathbf{u}}_\mathbf{n}; \bar{\mathbf{t}}_\mathbf{1}; \mathbf{t_2}; \dots; \mathbf{t_n}).$$

Then the partial assignment $x_1 \wedge \bar{u}_1 \wedge \bar{t}_1 \wedge t_2 \wedge \dots \wedge t_n$ satisfies the matrix of $\mathtt{Eq}_n$. Reducing this cube existentially results in $x_1 \wedge \bar{u}_1$, hence $x_1 \wedge \bar{u}_1 \in \mathfrak{L}(\mathcal{T}_1)$.

Learning $\bar{x}_1 \wedge u_1$ works analogously. Note that the previously learned cube does not interfere with the learning of this cube.

Having already learned the $2i$ cubes from 1 to i, let us now explain how to learn the two cubes for $i+1$. We create the following trail:

$$\mathcal{T}_{i+1} := (\mathbf{x_1}, u_1, t_1; \dots; \mathbf{x_i}, u_i, t_i; \mathbf{x_{i+1}}; \dots; \mathbf{x_n}; \bar{\mathbf{u}}_\mathbf{i+1}; \dots; \bar{\mathbf{u}}_\mathbf{n}; \bar{\mathbf{t}}_\mathbf{i+1}; \mathbf{t_{i+2}}; \dots; \mathbf{t_n})$$

with

$$\mathrm{ante}_{\mathcal{T}_{i+1}}(u_j) = x_j \wedge \bar{u}_j,$$
$$\mathrm{ante}_{\mathcal{T}_{i+1}}(t_j) = \bar{x}_j \vee \bar{u}_j \vee t_j$$

for $j = 1, \dots, i$.

Again, the partial assignment $x_{i+1} \wedge \bar{u}_{i+1} \wedge t_1 \wedge \dots \wedge t_i \wedge \bar{t}_{i+1} \wedge t_{i+2} \wedge \dots \wedge t_n$ satisfies the matrix of $\mathtt{Eq}_n$. This can be reduced to the cube $x_{i+1} \wedge \bar{u}_{i+1}$, which we will learn. As before, learning $\bar{x}_{i+1} \wedge u_{i+1}$ works analogously.

After we have learned all of these $2n-2$ cubes, we will go on with clause learning in which we will successively learn the clauses

$$L_i := \bar{x}_i \vee \bar{u}_i \vee \bigvee_{j=i+1}^{n} (u_j \vee \bar{u}_j) \vee \bigvee_{k=1}^{i-1} \bar{t}_k$$

$$R_i := x_i \vee u_i \vee \bigvee_{j=i+1}^{n} (u_j \vee \bar{u}_j) \vee \bigvee_{k=1}^{i-1} \bar{t}_k$$

for $i = 2, \dots, n-1$.

We start with the following trails:

$$\mathcal{U}_{n-1} := (\mathbf{x_1}, u_1, t_1; \dots ; \mathbf{x_{n-1}}, u_{n-1}, t_{n-1}, \bar{t}_n, x_n, \bot)$$

with

$$\mathrm{ante}_{\mathcal{U}_{n-1}}(u_j) = x_j \wedge \bar{u}_j$$
$$\mathrm{ante}_{\mathcal{U}_{n-1}}(t_j) = \bar{x}_j \vee \bar{u}_j \vee t_j$$
$$\mathrm{ante}_{\mathcal{U}_{n-1}}(\bar{t}_n) = \bar{t}_1 \vee \dots \vee \bar{t}_n$$
$$\mathrm{ante}_{\mathcal{U}_{n-1}}(x_n) = x_n \vee u_n \vee t_n$$
$$\mathrm{ante}_{\mathcal{U}_{n-1}}(\bot) = \bar{x}_n \vee \bar{u}_n \vee t_n$$

for $j = 1, \dots, n-1$. We resolve over $x_n, \bar{t}_n$ and t_{n-1} and get L_{n-1}. Analogously, we can learn R_{n-1}.

Suppose we have already learned $L_{n-1}, R_{n-1}, \dots, L_i, R_i$ for some $i \in \{3, \dots, n-1\}$. Let us now construct trails from which we can learn L_{i-1} and R_{i-1}:

$$\mathcal{U}_{i-1} := (\mathbf{x_1}, u_1, t_1; \dots ; \mathbf{x_{i-1}}, u_{i-1}, t_{i-1}, x_i, \bot)$$

with

$$\mathrm{ante}_{\mathcal{U}_{i-1}}(u_j) = x_j \wedge \bar{u}_j,$$
$$\mathrm{ante}_{\mathcal{U}_{i-1}}(t_j) = \bar{x}_j \vee \bar{u}_j \vee t_j$$
$$\mathrm{ante}_{\mathcal{U}_{i-1}}(x_i) = R_i$$
$$\mathrm{ante}_{\mathcal{U}_{i-1}}(\bot) = L_i$$

for $j = 1, \dots, i-1$. We resolve over x_i and t_{i-1} and get L_{i-1}. Again, analogously we can derive R_{i-1}.

After we have finished learning L_2 and R_2, we can create the last two trails as follows:

$$\mathcal{U}_1 := (\mathbf{x_1}, u_1, t_1, x_2, \bot)$$

with

$$\mathrm{ante}_{\mathcal{U}_1}(u_1) = x_1 \wedge \bar{u}_1$$
$$\mathrm{ante}_{\mathcal{U}_1}(t_1) = \bar{x}_1 \vee \bar{u}_1 \vee t_1$$
$$\mathrm{ante}_{\mathcal{U}_1}(x_2) = R_2$$
$$\mathrm{ante}_{\mathcal{U}_1}(\bot) = L_2.$$

We resolve over x_2 and t_1 and obtain the unit clause $(\bar{x}_1)$. Then the last trail will not contain any decision:

$$\mathcal{U}'_1 := (\bar{x}_1, \bar{u}_1, t_1, x_2, \bot)$$

with

$$\mathrm{ante}_{\mathcal{U}'_1}(\bar{x}_1) = (\bar{x}_1)$$
$$\mathrm{ante}_{\mathcal{U}'_1}(u_1) = x_1 \wedge \bar{u}_1$$
$$\mathrm{ante}_{\mathcal{U}'_1}(t_1) = \bar{x}_1 \vee \bar{u}_1 \vee t_1$$
$$\mathrm{ante}_{\mathcal{U}'_1}(x_2) = R_2$$
$$\mathrm{ante}_{\mathcal{U}'_1}(\bot) = L_2.$$

Resolving over all existential variables leads to the empty clause. □

As the formulas $\mathtt{Eq}_n$ require exponential-sized QCDCL refutations [4] we obtain:

Theorem 5.3. QCDCL^{CUBE} *is exponentially stronger than* QCDCL.

Next we will look at the influence of pure literal elimination. Now, the effect of pure literal elimination is similar to cube learning: they enable out-of-order decisions that can shorten the refutations. This again manifests in $\mathtt{Eq}_n$.

Proposition 5.4. $\mathtt{Eq}_n$ *has poly-size* QCDCL^{PL} *(and* QCDCL$^{CUBE+PL}$*) refutations.*

Proof. The refutation is similar to the one in Proposition 5.2, except that instead of learning cubes, we will use pure literal elimination to decide the universal literals out of order. We will again learn the clauses L_i and R_i for $i = 2, \dots, n-1$.

We start with the following trails:

$$\mathcal{U}_{n-1} := (\mathbf{x_1}; \underline{\mathbf{u_1}}, t_1; \dots; \mathbf{x_{n-1}}; \underline{\mathbf{u_{n-1}}}, t_{n-1}, \bar{t}_n, x_n, \bot)$$

with

$$\begin{aligned}
\mathrm{ante}_{\mathcal{U}_{n-1}}(t_j) &= \bar{x}_j \vee \bar{u}_j \vee t_j \\
\mathrm{ante}_{\mathcal{U}_{n-1}}(\bar{t}_n) &= \bar{t}_1 \vee \dots \vee \bar{t}_n \\
\mathrm{ante}_{\mathcal{U}_{n-1}}(x_n) &= x_n \vee u_n \vee t_n \\
\mathrm{ante}_{\mathcal{U}_{n-1}}(\bot) &= \bar{x}_n \vee \bar{u}_n \vee t_n
\end{aligned}$$

for $j = 1, \dots, n-1$. We resolve over $x_n, \bar{t}_n$ and t_{n-1} and get L_{n-1}. In an analogous way we can learn R_{n-1}.

Suppose we have already learned $L_{n-1}, R_{n-1}, \dots, L_i, R_i$ for some $i \in \{3, \dots, n-1\}$. Let us now construct trails from which we can learn L_{i-1} and R_{i-1}:

$$\mathcal{U}_{i-1} := (\mathbf{x_1}; \underline{\mathbf{u_1}}, t_1; \dots; \mathbf{x_{i-1}}; \underline{\mathbf{u_{i-1}}}, t_{i-1}, x_i, \bot)$$

with

$$\begin{aligned}
\mathrm{ante}_{\mathcal{U}_{i-1}}(t_j) &= \bar{x}_j \vee \bar{u}_j \vee t_j \\
\mathrm{ante}_{\mathcal{U}_{i-1}}(x_i) &= R_i \\
\mathrm{ante}_{\mathcal{U}_{i-1}}(\bot) &= L_i
\end{aligned}$$

for $j = 1, \dots, i-1$. We resolve over x_i and t_{i-1} and get L_{i-1}. Again, analogously we can derive R_{i-1}. Note that, in our case, the learned clauses will not interfere with pure literal elimination. Once we have learned L_i and R_i, we will not need to make the literals from $u_i, \dots, u_n$ pure any more. Also, say we learn L_i before R_i, once we decide $\bar{x}_i$ in order to learn R_i, we will also make L_i true. Therefore pure literal elimination behaves (almost) symmetrically.

After we have finished learning L_2 and R_2, we can create the last two trails as follows:

$$\mathcal{U}_1 := (\mathbf{x_1}; \underline{\mathbf{u_1}}, t_1, x_2, \bot)$$

with

$$\begin{aligned}
\mathrm{ante}_{\mathcal{U}_1}(t_1) &= \bar{x}_1 \vee \bar{u}_1 \vee t_1 \\
\mathrm{ante}_{\mathcal{U}_1}(x_2) &= R_2 = x_2 \vee u_2 \vee \bigvee_{j=3}^{n} (u_j \vee \bar{u}_j) \vee \bar{t}_1 \\
\mathrm{ante}_{\mathcal{U}_1}(\bot) &= L_2 = \bar{x}_2 \vee u_2 \vee \bigvee_{j=3}^{n} (u_j \vee \bar{u}_j) \vee \bar{t}_1.
\end{aligned}$$

We resolve over x_2 and t_1 and obtain the unit clause $(\bar{x}_1)$. Then the last trail will not contain any decision:

$$\mathcal{U}'_1 := (\bar{x}_1, \underline{\bar{\mathbf{u}}_\mathbf{1}}, t_1, x_2, \bot)$$

with

$$\begin{aligned}
\mathrm{ante}_{\mathcal{U}'_1}(\bar{x}_1) &= (\bar{x}_1) \\
\mathrm{ante}_{\mathcal{U}'_1}(t_1) &= \bar{x}_1 \vee \bar{u}_1 \vee t_1 \\
\mathrm{ante}_{\mathcal{U}'_1}(x_2) &= R_2 \\
\mathrm{ante}_{\mathcal{U}'_1}(\bot) &= L_2.
\end{aligned}$$

Resolving over all existential variables leads to the empty clause. □

Although pure literal elimination helps to refute $\mathtt{Eq}_n$, it turns out that pure literal elimination can also be disadvantageous. It might be a fallacy to think that pure existential literals should be satisfied in the same way as unit clauses in unit propagation. We will construct formulas in which pure literal elimination thwarts finding a convenient conflict and therefore short refutations.

We construct these formulas in stages, starting with $\texttt{MirrorCR}_n$. In turn, these QBFs are based on the Completion Principle $\texttt{CR}_n$ of [18], known to be hard for QCDCL [18,12]. The "Mirror"-modification adds new symmetries to the formula, causing pure literals to appear too late to make a difference.

Definition 5.5. The QCNF $\texttt{MirrorCR}_n$ consists of the prefix

$$\exists x_{(1,1)},\dots,x_{(n,n)}\forall u\exists a_1,\dots,a_n,b_1,\dots,b_n$$

and the matrix

$$\begin{array}{ll} x_{(i,j)}\vee u\vee a_i & \bar a_1\vee\dots\vee\bar a_n \\ \bar x_{(i,j)}\vee \bar u\vee b_j & \bar b_1\vee\dots\vee\bar b_n \\ x_{(i,j)}\vee \bar u\vee \bar a_i & a_1\vee\dots\vee a_n \\ \bar x_{(i,j)}\vee u\vee \bar b_j & b_1\vee\dots\vee b_n \quad \text{for } i,j\in[n]. \end{array}$$

It is easy to see that $\texttt{MirrorCR}_n$ fulfil the XT-property. Additionally, we can show:

Proposition 5.6. *The CNF* $\mathfrak{C}(\texttt{MirrorCR}_n)$ *is unsatisfiable and gauge*$(\texttt{MirrorCR}_n)\geqslant n-1$.

Proof. We first show the unsatisfiability of the matrix. Assume otherwise. Let σ be a satisfying assignment for $\mathfrak{C}(\texttt{MirrorCR}_n)$. We can assume that σ is a total assignment. W.l.o.g. let $u\in\sigma$. We distinguish two cases:

Case 1: For all $i\in\{1,\dots,n\}$ there exists a $j\in\{1,\dots,n\}$ such that $\bar x_{(i,j)}\in\sigma$. Then we need $\bar a_i\in\sigma$ for all $i=1,\dots,n$, which falsifies the clause $a_1\vee\dots\vee a_n$.

Case 2: There is an $i\in\{1,\dots,n\}$ such that for all $j\in\{1,\dots,n\}$ we have $x_{(i,j)}\in\sigma$. Then we need $b_j\in\sigma$ for all $j=1,\dots,n$, which falsifies the clause $\bar b_1\vee\dots\vee\bar b_n$.

In each case we can conclude that it is not possible to construct a satisfying assignment for $\mathfrak{C}(\texttt{MirrorCR}_n)$.

We now prove gauge$(\texttt{MirrorCR}_n)\geqslant n-1$.

Since $\texttt{MirrorCR}_n$ contains no X-clauses as axioms, we have to resolve over some a_i or b_j somehow. Obviously, it is not possible to resolve $x_{(i,j)}\vee u\vee a_i$ and $x_{(i,j)}\vee\bar u\vee\bar a_i$ or $\bar x_{(i,j)}\vee\bar u\vee b_j$ and $\bar x_{(i,j)}\vee u\vee\bar b_j$. That means we have to use the other axioms. Because of the symmetry, we can assume that we use the clause $\bar a_1\vee\dots\vee\bar a_n$ somehow. Then we have to get rid of all $\bar a_i$. This can be done via the clauses $x_{(i,j)}\vee u\vee a_i$, or we use the clause $a_1\vee\dots\vee a_n$. However, to use the latter clause we have to get rid of at least $n-1$ different a_i in another way first, which is only possible with the aid of the clauses $x_{(i,j)}\vee\bar u\vee\bar a_i$. We conclude that we will pile up at least $n-1$ different X-literals. □

Applying Theorem 4.5 we infer:

Corollary 5.7. $\texttt{MirrorCR}_n$ *needs exponential-size fully reduced primitive Q-resolution refutations.*

All we have to do now is to show that all $\mathsf{QCDCL}^{\mathsf{PL}}$ refutations of $\texttt{MirrorCR}_n$ are primitive. Then the gauge lower bound applies. We will show that for a non-primitive refutation of $\texttt{MirrorCR}_n$ we would need to decide literals by pure literal elimination, and before each pure literal elimination we have to perform another one, which is a contradiction.

Proposition 5.8. *From each* $\mathsf{QCDCL}^{\mathsf{PL}}$ *refutation of* $\texttt{MirrorCR}_n$ *we can extract a fully reduced primitive Q-resolution refutation of the same size.*

Proof. Let ι be a $\mathsf{QCDCL}^{\mathsf{PL}}$ refutation of $\texttt{MirrorCR}_n$. We will show that $\mathfrak{R}(\iota)$ is primitive.

Assume not. Then by Proposition 4.7 there exists a trail $\mathcal{T}\in\mathfrak{T}(\iota)$ such that there is an X-literal $x\in\mathcal{T}$ and a U-literal $v\in\mathcal{T}$ with $v<_{\mathcal{T}}x$ and v is not a regular decision literal. Let us say that $\text{var}(x)=x_{(k,m)}$ for some $k,m\in\{1,\dots,n\}$.

That means we have decided v (which is either u or $\bar u$) out of order via pure literal elimination. We show that this is not possible before we have assigned all X-literals.

Claim 1: There is a T-literal t_1 such that $t_1<_{\mathcal{T}}v<_{\mathcal{T}}x$.

W.l.o.g. let $v=\bar u$. We need to satisfy the clauses $\bar x_{(i,j)}\vee\bar u\vee b_j$ and $x_{(i,j)}\vee\bar u\vee\bar a_i$ for each $i,j\in\{1,\dots,n\}$ without assigning u. Since we want to propagate x later, we cannot assign the X-variable $x_{(k,m)}$ in order to satisfy these clauses. That means we need to set b_m to true and a_k to false. If we set $t_1:=b_m$, then we get $t_1<_{\mathcal{T}}v<_{\mathcal{T}}x$.

Claim 2: For each T-literal t_j with $t_j<_{\mathcal{T}}v<_{\mathcal{T}}x$ there is another T-literal t_{j+1} such that $t_{j+1}<_{\mathcal{T}}t_j<_{\mathcal{T}}v<_{\mathcal{T}}x$.

Because of $t_j<_{\mathcal{T}}v$, the T-literal t_j cannot be a regular decision. Either t_j was decided as a pure literal, or it was propagated. If it was a pure literal, then we needed to satisfy one of the clauses $\bar a_1\vee\dots\vee\bar a_n$, $\bar b_1\vee\dots\vee\bar b_n$, $a_1\vee\dots\vee a_n$ or $b_1\vee\dots\vee b_n$. This is only possible if we assigned another T-literal t_{j+1} before, hence $t_{j+1}<_{\mathcal{T}}t_j<_{\mathcal{T}}v<_{\mathcal{T}}x$. However, if t_j was propagated, then there is the antecedent clause $F:=\text{ante}_{\mathcal{T}}(t_j)$. Due to the XT-property, F cannot be unit. Then there is another literal $t_j\neq\ell\in F$. Because the formula only contains one U-variable, ℓ can only be an X- or a T-literal. Again, by the XT-property, F cannot be an XT-clause and therefore ℓ has to be a T-literal, which needs to be falsified by the current trail. Therefore, if we set $t_{j+1} := \bar{\ell}$, we get $t_{j+1} <_{\mathcal{T}} t_j <_{\mathcal{T}} v <_{\mathcal{T}} x$.

We proved that $\mathfrak{R}(\iota)$ has to be primitive, otherwise the trail $\mathcal{T}$ would contain infinitely many T-literals t_j. □

Corollary 5.9. *The QBFs* $\texttt{MirrorCR}_n$ *require exponential-size* QCDCL^{PL} *refutations.*

Next we embed this formula into a new QCNF $\texttt{PLTrap}_n$.

Definition 5.10. The QCNF $\texttt{PLTrap}_n$ has the prefix

$$\exists y, x_{(1,1)}, \dots, x_{(n,n)} \forall u \exists a_1, \dots, a_n, b_1, \dots, b_n, a, b.$$

Its matrix contains all clauses from $\texttt{MirrorCR}_\texttt{n}$ and additionally $(y \vee a)$, $(\bar{a} \vee b)$, $(\bar{a} \vee \bar{b})$, $(a \vee b)$, and $(a \vee \bar{b})$.

Proposition 5.11. $\texttt{PLTrap}_n$ *needs exponential-size* QCDCL^{PL} *and* QCDCL$^{CUBE+PL}$ *refutations.*

Proof. Because $\mathfrak{C}(\texttt{PLTrap}_n)$ contains $\mathfrak{C}(\texttt{MirrorCR}_n)$, which is unsatisfiable, the matrix of $\texttt{PLTrap}_n$ is unsatisfiable, as well. Therefore cube learning will never be applied and it suffices to consider QCDCL$^{\text{PL}}$ refutations.

Let ι be a QCDCL$^{\text{PL}}$ refutation of $\texttt{PLTrap}_n$. We will show that each trail of $\mathfrak{T}(\iota)$ can only contain literals from $\texttt{MirrorCR}_n$ or y. Then ι can be interpreted as a QCDCL$^{\text{PL}}$ refutation of $\texttt{MirrorCR}_n$ where the only difference is the assignment of y, which does not affect clause learning in any form. Then the result follows by Corollary 5.9.

In each QCDCL$^{\text{PL}}$ trail, we will set y to true due to pure literal elimination. That means the clause $y \vee a$ will never become the unit clause (a).

After this, we have to assign the variables from $\texttt{MirrorCR}_n$. We will show that for each trail $\mathcal{T} \in \mathfrak{T}(\iota)$ we have $\{a, \bar{a}, b, \bar{b}\} \cap \mathcal{T} = \emptyset$.

First of all, it is obvious that pure literal elimination of a or b is impossible at any time due to the four clauses $\bar{a} \vee b$, $a \vee b$, $\bar{a} \vee \bar{b}$ and $a \vee \bar{b}$. In fact, if, for example, we would like to make b pure, then we have to satisfy the clauses $\bar{a} \vee \bar{b}$ and $a \vee \bar{b}$, which cannot be done without setting b to false.

Next, let us assume that there is some literal $\ell \in \{a, \bar{a}, b, \bar{b}\}$ that was propagated in some trail $\mathcal{T} \in \mathfrak{T}(\iota)$. In particular, let $\mathcal{T}$ be the first trail in which we propagated a literal $\ell \in \{a, \bar{a}, b, \bar{b}\}$. Since $y \vee a$ can never be used as an antecedent clause for a, we have $\text{ante}_{\mathcal{T}}(\ell) \in \{\bar{a} \vee b, a \vee b, \bar{a} \vee \bar{b}, a \vee \bar{b}\}$. But then we would need another $\ell \neq \ell' \in \{a, \bar{a}, b, \bar{b}\}$ with $\ell' \in \mathcal{T}$ and $\ell' <_{\mathcal{T}} \ell$. If we suppose that ℓ was the first propagation of a literal from $\{a, \bar{a}, b, \bar{b}\}$, then we conclude that ℓ' has to be a regular decision.

We will now argue that we get a contradiction if there is a trail $\mathcal{T} \in \mathfrak{T}(\iota)$ in which we have decided a literal $\ell' \in \{a, \bar{a}, b, \bar{b}\}$. Because of the level-ordered decision rule LOD, there exists $v \in \{u, \bar{u}\}$ with $v \in \mathcal{T}$ and $v <_{\mathcal{T}} \ell'$. We can only decide v if we have assigned all existential literals left of v. In particular, for each $i, j = 1, \dots, n$ there is a literal $\ell_{(i,j)} \in \{x_{(i,j)}, \bar{x}_{(i,j)}\}$ with $\ell_{(i,j)} \in \mathcal{T}$ and $\ell_{(i,j)} <_{\mathcal{T}} v$. We now distinguish two cases.

<u>Case 1:</u> For all $i \in \{1, \dots, n\}$ there exists a $j \in \{1, \dots, n\}$ with $\ell_{(i,j)} = \bar{x}_{(i,j)}$.

Then if $v = u$, we will gain unit clauses $(\bar{a}_i)$ for $i = 1, \dots, n$ from the clauses $x_{(i,j)} \vee \bar{u} \vee \bar{a}_i$, which can be used for unit propagations that lead to a conflict in the clause $a_1 \vee \dots \vee a_n$. Otherwise, if $v = \bar{u}$, then we will get unit clauses (a_i) from the clauses $x_{(i,j)} \vee u \vee a_i$ and a conflict in $\bar{a}_1 \vee \dots \vee \bar{a}_n$.

<u>Case 2:</u> There exists an $i \in \{1, \dots, n\}$ such that for all $j \in \{1, \dots, n\}$ it holds $\ell_{(i,j)} = x_{(i,j)}$.

This case is analogous to Case 1 with unit clauses (b_j) (resp. $(\bar{b}_j)$) and a conflict in $\bar{b}_1 \vee \dots \vee \bar{b}_n$ (resp. $b_1 \vee \dots \vee b_n$).

In each case we will get a conflict in some clause. That means the trail $\mathcal{T}$ would run into a conflict before we would have the chance to decide ℓ'. That shows that ℓ' cannot be decided at any point. We conclude that no trail from ι can contain a literal from $\{a, \bar{a}, b, \bar{b}\}$. □

Not having to follow the PLD rule, we can construct short proofs of $\texttt{PLTrap}_n$ by focusing on the new clauses over a, b.

Proposition 5.12. $\texttt{PLTrap}_n$ *has polynomial-size* QCDCL *refutations.*

Proof. The shortest refutation only consists of two trails. We start with

$$\mathcal{T} := (\bar{\mathbf{y}}, a, b, \bot)$$

with

$$\text{ante}_{\mathcal{T}}(a) = y \vee a$$

$$\text{ante}_{\mathcal{T}}(b) = \bar{a} \vee b$$

$$\text{ante}_{\mathcal{T}}(\bot) = \bar{a} \vee \bar{b}.$$

We resolve over b and learn the unit clause $(\bar{a})$.

The final trail looks as follows:

$$\mathcal{U} := (\bar{a}, b, \bot)$$

with

$$\text{ante}_{\mathcal{U}}(\bar{a}) = (\bar{a})$$
$$\text{ante}_{\mathcal{U}}(b) = a \vee b$$
$$\text{ante}_{\mathcal{U}}(\bot) = a \vee \bar{b},$$

from which we can learn the empty clause by resolving over everything. □

We conclude that pure literal elimination is advantageous for $\mathtt{Eq}_n$, but not for $\mathtt{PLTrap}_n$. Therefore we obtain:

Theorem 5.13. *QCDCL^{PL} and QCDCL are incomparable as well as QCDCL$^{CUBE+PL}$ and QCDCL.*

6. Cube learning vs. pure literal elimination

As shown in Section 5, cube learning improves QCDCL, while adding pure literal elimination leads to incomparable systems. Thus it is natural to directly compare cube learning and pure literal elimination. Because of the results above, we cannot use $\mathtt{Eq}_n$ for a potential separation. However, we can modify the QBFs such that they remain easy for QCDCL^{PL}, while eliminating the benefits from cube learning.

Definition 6.1. The QCNF $\mathtt{TwinEq}_n$ has the prefix

$$\exists x_1, \dots, x_n \forall u_1, \dots, u_n, w_1, \dots, w_n \exists t_1, \dots, t_n.$$

Its matrix contains the clauses from $\mathtt{Eq}_n$ together with $x_i \vee w_i \vee t_i$ and $\bar{x}_i \vee \bar{w}_i \vee t_i$ for $i \in [n]$.

The main idea of this twin construction is to ensure that all potential cubes consist of at least two universal variables. We can do the same construction for other QCNFs.

Obviously, $\mathtt{TwinEq}_n$ fulfils the XT-property. We compute gauge($\mathtt{TwinEq}_n$) $= n$ and hence infer by Theorem 4.5:

Proposition 6.2. *Fully reduced primitive Q-resolution refutations of $\mathtt{TwinEq}_n$ have exponential size.*

Proof. We need to show gauge($\mathtt{TwinEq}_n$) $= n$, then the result follows by Theorem 4.5.

Since we have to resolve over T somehow, we have to use the clause $\bar{t}_1 \vee \dots \vee \bar{t}_n$. Hence, we have to resolve over each t_i at least once, and therefore we will pile up x_i or $\bar{x}_i$ in each resolution step due to the XUT-axioms. □

We show that QCDCL^{CUBE} refutations of $\mathtt{TwinEq}_n$ are primitive by proving that it is impossible to propagate U-literals before having assigned all X-literals.

Proposition 6.3. *Each QCDCL^{CUBE} refutation of $\mathtt{TwinEq}_n$ has at least exponential size.*

Proof. We will prove that from each QCDCL^{CUBE} refutation of $\mathtt{TwinEq}_n$ we can extract a fully reduced primitive Q-resolution refutation of the same size. Let ι be a QCDCL^{CUBE} refutation of $\mathtt{TwinEq}_n$. We will show that $\mathfrak{R}(\iota)$ is primitive.

Assume not. Then by Proposition 4.7 there exists a trail $\mathcal{T} \in \mathfrak{T}(\iota)$ such that there is an X-literal $x \in \mathcal{T}$ and a U-literal $u \in \mathcal{T}$ with $u <_{\mathcal{T}} x$. Also, u cannot be a regular decision in $\mathcal{T}$.

Hence, we have propagated u before x. Universal propagation can only be performed via cubes. Let us now consider how the initial cubes from $\mathtt{TwinEq}_n$ look like.

Assume that the cube A is a (not necessarily total) assignment that satisfies the matrix of $\mathtt{TwinEq}_n$. We have to satisfy the clause $\bar{t}_1 \vee \dots \vee \bar{t}_n$, hence there is a $j \in \{1, \dots, n\}$ with $\bar{t}_j \in A$. Then we also have to satisfy the four clauses

$$x_j \vee u_j \vee t_j$$
$$\bar{x}_j \vee \bar{u}_j \vee t_j$$
$$x_j \vee w_j \vee t_j$$
$$\bar{x}_j \vee \bar{w}_j \vee t_j.$$

That means x_j has to appear in some polarity in A, say $x_j \in A$. But then we need to set both u_j and w_j to false, thus $\bar{u}_j, \bar{w}_j \in A$.

We conclude that each (reduced) cube has to contain one of the subcubes

$$x_j \wedge \bar{u}_j \wedge \bar{w}_j$$
$$\bar{x}_j \wedge u_j \wedge w_j$$

for some $j \in \{1, \dots, n\}$. This also causes that none of these cubes are resolvable.

We observe that all cubes that can be used for universal unit propagation contain at least two universal literals. Since we needed one of these cubes as antecedent cube of some universal literal in our trail $\mathcal{T}$, we would have needed to decide or propagate another universal literal before. Having only finitely many universal literals, we would have needed to decide one universal literal before propagating x, which is a contradiction to our decision rule LOD.

This shows that $\mathfrak{R}(\iota)$ is indeed primitive. □

Having shown that $\mathtt{TwinEq}_n$ is hard for QCDCL$^{\mathsf{CUBE}}$, it remains to prove that it is easy for QCDCL$^{\mathsf{PL}}$.

Proposition 6.4. $\mathtt{TwinEq}_n$ *has polynomial-size* QCDCL^{PL} *refutations.*

Proof. The proof is similar to the one in Proposition 5.4, except one change: Each time some universal literal is getting pure, say u_i, then also w_i becomes pure as well. That means each time we decide some u_i (resp. $\bar{u}_i$) in the trail by pure literal elimination, we also have to do the same to w_i (resp. $\bar{w}_i$) in the next decision level. However, this does not affect anything concerning unit propagation or clause learning.

To give an example: The trail $\mathcal{U}_{n-1}$ from Proposition 5.4 will now look like

$$\mathcal{U}_{n-1} := (\mathbf{x_1}; \underline{\mathbf{u_1}}, t_1; \underline{\mathbf{w_1}}; \dots; \mathbf{x_{n-2}}; \underline{\mathbf{u_{n-2}}}, t_{n-2}; \underline{\mathbf{w_{n-2}}};$$
$$\mathbf{x_{n-1}}; \underline{\mathbf{u_{n-1}}}, t_{n-1}, \bar{t}_n, x_n, \bot). \quad \square$$

For the other separation we use $\mathtt{PLTrap}_n$, which is hard for QCDCL$^{\mathsf{PL}}$ by Proposition 5.11, but still easy for QCDCL$^{\mathsf{CUBE}}$ by Proposition 5.1 and 5.12. Therefore we conclude:

Theorem 6.5. QCDCL^{CUBE} *is incomparable to* QCDCL^{PL}.

We have seen earlier that the QCDCL system including pure literal elimination is incomparable to the system without. Now we will prove that this incomparability still holds with cube learning turned on. By Proposition 5.1, we obtain that QCDCL$^{\mathsf{CUBE+PL}}$ p-simulates QCDCL$^{\mathsf{PL}}$. Therefore we get from Proposition 6.4:

Corollary 6.6. $\mathtt{TwinEq}_n$ *has poly-size* QCDCL$^{CUBE+PL}$ *refutations.*

Since $\mathtt{TwinEq}_n$ is hard for QCDCL$^{\mathsf{CUBE}}$, this gives us the first separation between QCDCL$^{\mathsf{CUBE+PL}}$ and QCDCL$^{\mathsf{CUBE}}$. The other direction can be shown with $\mathtt{PLTrap}_n$.

Proposition 6.7. $\mathtt{PLTrap}_n$ *has poly-size* QCDCL^{CUBE} *refutations.*

Proof. The short proofs in QCDCL$^{\mathsf{CUBE}}$ follow from Propositions 5.1 and 5.12. □

Hence we get:

Theorem 6.8. QCDCL$^{CUBE+PL}$ *and* QCDCL^{CUBE} *are incomparable.*

We now consider the relation between QCDCL$^{\mathsf{CUBE+PL}}$ and QCDCL$^{\mathsf{PL}}$. We introduce another modification of $\mathtt{Eq}_n$, which we call $\mathtt{BulkyEq}_n$, where we add two clauses.

Definition 6.9. The QCNF $\mathtt{BulkyEq}_n$ is obtained from $\mathtt{Eq}_n$ by adding the clauses $u_1 \vee \dots \vee u_n \vee t_1 \vee \dots \vee t_n$ and $\bar{u}_1 \vee \dots \vee \bar{u}_n \vee t_1 \vee \dots \vee t_n$ to the matrix.

As $\mathtt{Eq}_n$, this formula fulfils the XT-property and has a high gauge value ($\geqslant n-1$). By Theorem 4.5 we infer that $\mathtt{BulkyEq}_n$ needs exponential-size fully reduced primitive Q-resolution refutations. Similarly to $\mathtt{MirrorCR}_n$, we can then show that pure literal elimination does not shorten proofs because of the two additional 'bulky' clauses that prevent pure literals to occur early in trails. Therefore $\mathtt{BulkyEq}_n$ is hard for QCDCL$^{\mathsf{PL}}$. On the other hand, we can explicitly construct short proofs in QCDCL$^{\mathsf{CUBE+PL}}$. Therefore we get:

Proposition 6.10. *The formula* $\mathtt{BulkyEq}_n$ *has poly-size* $\mathsf{QCDCL}^{CUBE+PL}$ *refutations, but needs exponential-size* QCDCL^{PL} *refutations.*

Proof. ***Part 1:*** $\mathtt{BulkyEq}_n$ needs exponential-size $\mathsf{QCDCL}^{\mathsf{PL}}$ refutations.

We first prove gauge($\mathtt{BulkyEq}_n$) $\geqslant n-1$. To derive an X-clause, we have to use $\bar{t}_1 \vee \ldots \vee \bar{t}_n$ somehow. That means we have to resolve over each t_i. We can resolve with $u_1 \vee \ldots \vee u_n \vee t_1 \vee \ldots \vee t_n$ or $\bar{u}_1 \vee \ldots \vee \bar{u}_n \vee t_1 \vee \ldots \vee t_n$ only after we have resolved away at least $n-1$ different T-variables otherwise. That means we have pile up at least $n-1$ different X-literals by using the clauses $x_i \vee u_i \vee t_i$ or $\bar{x}_i \vee \bar{u}_i \vee t_i$. Hence gauge($\mathtt{BulkyEq}_n$) $\geqslant n-1$.

We will now prove that from each $\mathsf{QCDCL}^{\mathsf{PL}}$ refutation of $\mathtt{BulkyEq}_n$ we can extract a fully reduced primitive Q-resolution refutation of the same size. Let ι be a $\mathsf{QCDCL}^{\mathsf{PL}}$ refutation of $\mathtt{BulkyEq}_n$. We will show that $\mathfrak{R}(\iota)$ is primitive.

Assume not. Then by Proposition 4.7 there exists a trail $\mathcal{T} \in \mathfrak{T}(\iota)$ such that there is an X-literal $x \in \mathcal{T}$ and a U-literal $u \in \mathcal{T}$ with $u <_{\mathcal{T}} x$ and u is not a regular decision literal.

Since cube learning is disabled, this universal literal u had to be decided by pure literal elimination. We will show that pure literal elimination of the universal literal u before deciding or propagating all X-variables is not possible. Define $M := \{u_i, \bar{u}_i, t_i, \bar{t}_i \,:\, i = 1, \ldots, n\}$.

Claim 1: There exists some $\ell_1 \in M$ such that $\ell_1 <_{\mathcal{T}} u <_{\mathcal{T}} x$.

In order to make u pure, we have to satisfy one of the clauses $u_1 \vee \ldots \vee u_n \vee t_1 \vee \ldots \vee t_n$ or $\bar{u}_1 \vee \ldots \vee \bar{u}_n \vee t_1 \vee \ldots \vee t_n$. In particular, we need some $\ell_1 \in M$ with $\ell_1 <_{\mathcal{T}} u <_{\mathcal{T}} x$.

Claim 2: For each $\ell_j \in M$ with $\ell_j <_{\mathcal{T}} u <_{\mathcal{T}} x$ there exists some $\ell_{j+1} \in M$ such that $\ell_{j+1} <_{\mathcal{T}} \ell_j <_{\mathcal{T}} u <_{\mathcal{T}} x$

If ℓ_j was decided via pure literal elimination, we can use a similar argument as in Claim 1 (now we have satisfied one of the three clauses $u_1 \vee \ldots \vee u_n \vee t_1 \vee \ldots \vee t_n$, $\bar{u}_1 \vee \ldots \vee \bar{u}_n \vee t_1 \vee \ldots \vee t_n$ or $\bar{t}_1 \vee \ldots \vee \bar{t}_n$) and conclude that we need some $\ell_{j+1} \in M$ with $\ell_{j+1} <_{\mathcal{T}} \ell_j <_{\mathcal{T}} u <_{\mathcal{T}} x$. However, if ℓ_j was not decided as a pure literal, then it has to be a T-literal that was propagated. Note that we cannot have decided ℓ_j regularly because of $\ell_j <_{\mathcal{T}} x$ and $\ell_j <_{\mathcal{T}} u$. That means there is an antecedent clause $F := \mathrm{ante}_{\mathcal{T}}(\ell_j)$. Due to the XT-property, F cannot be a unit clause. That means there is another literal $\ell_j \neq \ell \in F$. If ℓ is a U- or a T-literal, then we can set $\ell_{j+1} := \bar{\ell}$. If ℓ is an X-literal, then there is at least one U-literal $v \in F$, again because of the XT-property. Then we can set $\ell_{j+1} := \bar{v}$.

We have proven that if $\mathfrak{R}(\iota)$ is not primitive, then $\mathcal{T}$ has to contain an endless number of literals ℓ_j, which is obviously not possible since the formula only consists of finitely many variables. That means $\mathfrak{R}(\iota)$ has to be primitive.

Part 2: $\mathtt{BulkyEq}_n$ has polynomial-size $\mathsf{QCDCL}^{\mathsf{CUBE+PL}}$ refutations.

We start with the learning of exactly two cubes: $x_1 \wedge \bar{u}_1$ and $\bar{x}_1 \wedge u_1$. We do this via the following two trails:

$$\mathcal{T} := (\mathbf{x_1}; \ldots; \mathbf{x_n}; \mathbf{\bar{u}_1}; \ldots; \mathbf{\bar{u}_n}; \mathbf{\bar{t}_1}; \mathbf{t_2}; \ldots; \mathbf{t_n})$$

$$\mathcal{T}' := (\mathbf{\bar{x}_1}; \ldots; \mathbf{\bar{x}_n}; \mathbf{u_1}; \ldots; \mathbf{u_n}; \mathbf{\bar{t}_1}; \mathbf{t_2}; \ldots; \mathbf{t_n})$$

Unfortunately we cannot continue learning the other cubes as in Proposition 5.2 since this will be blocked by pure literal elimination. However, we can use this effect to our advantage by simulating the missing cubes in this way.

Let us now start the learning of the clauses L_i and R_i for $i = 2, \ldots, n-1$ from the proof of Proposition 5.2.

We begin by constructing the following trail:

$$\mathcal{U}_{n-1} := (\mathbf{x_1}, u_1, t_1; \mathbf{x_2}; \underline{\mathbf{u_2}}, t_2, \ldots, \mathbf{x_{n-2}}; \underline{\mathbf{u_{n-2}}}, t_{n-2}; \mathbf{x_{n-1}}, \underline{\mathbf{u_{n-1}}}, t_{n-1}, \bar{t}_n, x_n, \bot)$$

with the same antecedent constraint as in Proposition 5.2 (except of the pure literals $u_2, \ldots, u_{n-2}$) and the same learned clause L_{n-1}. Analogously we can learn R_{n-1}.

We go on with the trails $\mathcal{U}_{n-2}, \ldots, \mathcal{U}_2$ in the same way as in Proposition 5.2 where we learn $L_{n-2}, \ldots, L_2$, except that the literals $u_2, \ldots, u_{i-1}$ in $\mathcal{U}_{i-1}$ are now pure literals and not propagated via cubes. However, this does not affect the clause learning process in any aspect. The same is obviously true for the analogous trails in which we learn $R_{n-2}, \ldots, R_2$.

We finish the proof with the last two trails $\mathcal{U}_1$ and $\mathcal{U}'_1$ exactly as in Proposition 5.2. ☐

As for the systems without pure literal elimination, we get:

Theorem 6.11. $\mathsf{QCDCL}^{CUBE+PL}$ *is exponentially stronger than* QCDCL^{PL}.

7. The QCDCL systems vs. Q-resolution

[4] showed incomparability of Q-resolution and QCDCL. We now lift this to the other QCDCL variants introduced here. For one separation, we can use the QCNF $\mathtt{QParity}_n$ from [7], which have short QCDCL refutations. These formulas have prefix $\exists x_1 \ldots x_n \forall u \exists t_1 \ldots t_n$ and clauses

$$x_1 \vee \bar{t}_1,\ \bar{x}_1 \vee t_1,\ u \vee t_n,\ \bar{u} \vee \bar{t}_n,$$

$$x_i \vee t_{i-1} \vee \bar{t}_i,\ x_i \vee \bar{t}_{i-1} \vee t_i,$$

$\bar{x}_i \vee t_{i-1} \vee t_i,\ \bar{x}_i \vee \bar{t}_{i-1} \vee \bar{t}_i \quad$ for $i \in \{2, \dots, n\}$.

Theorem 7.1. *QCDCL, QCDCL^{CUBE}, QCDCL^{PL} and QCDCL$^{CUBE+PL}$ are all incomparable to Q-resolution. In detail, the QBFs $\mathtt{QParity}_n$ have polynomial-size refutations in QCDCL, QCDCL^{CUBE}, QCDCL^{PL}, and QCDCL$^{CUBE+PL}$, but need exponential-size Q-resolution refutations. On the other hand, $\mathtt{MirrorCR}_n$ have polynomial-size Q-resolution refutations, but need exponential-size QCDCL, QCDCL^{CUBE}, QCDCL^{PL}, and QCDCL$^{CUBE+PL}$ refutations.*

Proof. Claim 1: $\mathtt{QParity}_n$ has polynomial-size QCDCL and QCDCL$^{\text{CUBE}}$ refutations.

It was proven in [4] that $\mathtt{QParity}_n$ has short QCDCL refutations. And because of Proposition 5.1, the formula is easy for QCDCL$^{\text{CUBE}}$, as well.

Claim 2: $\mathtt{QParity}_n$ has polynomial-size QCDCL$^{\text{PL}}$ and QCDCL$^{\text{CUBE+PL}}$ refutations.

We will show that we will never find pure literals while creating QCDCL$^{\text{PL}}$ trails. In fact, the only way in making a literal ℓ pure is to create a unit clause (ℓ), which would immediately lead to the propagation of ℓ or a conflict.

For example, suppose the literal t_i is pure at some point in the trail. Then the clauses $x_i \vee t_{i-1} \vee \bar{t}_i$ and $\bar{x}_i \vee \bar{t}_{i-1} \vee \bar{t}_i$ must have been satisfied by the current assignment of the trail. Since we have not assigned t_i yet, we have to set either x_i to true and t_{i-1} to false, or x_i to false and t_{i-1} to true. In both cases we would obtain the unit clause (t_i) by apply this assignment to either $x_i \vee \bar{t}_{i-1} \vee t_i$ or $\bar{x}_i \vee t_{i-1} \vee t_i$.

The same holds for the universal variable u. For u or $\bar{u}$ to be pure, we need to set t_n to false or true. But then we would obtain the unit clause (u) or $(\bar{u})$, which would immediately lead to a conflict. We conclude that the polynomial-size QCDCL refutation of $\mathtt{QParity}_n$ is a QCDCL$^{\text{PL}}$ refutation as well. And because QCDCL$^{\text{CUBE+PL}}$ p-simulates QCDCL$^{\text{PL}}$, $\mathtt{QParity}_n$ is also easy for QCDCL$^{\text{CUBE+PL}}$.

Claim 3: $\mathtt{QParity}_n$ needs exponential-size Q-resolution refutations.

This was already proven in [7, Proposition 4.2].

Claim 4: $\mathtt{MirrorCR}_n$ needs exponential-size QCDCL, QCDCL$^{\text{CUBE}}$, QCDCL$^{\text{PL}}$ and QCDCL$^{\text{CUBE+PL}}$ refutations.

Because of Proposition 5.6, each trail $\mathcal{T}$ in a QCDCL$^{\text{CUBE}}$ or QCDCL$^{\text{CUBE+PL}}$ refutation runs into a conflict. Therefore we will always learn clauses and no cubes. Then each QCDCL$^{\text{CUBE}}$ refutation can be interpreted as a QCDCL refutation and each QCDCL$^{\text{CUBE+PL}}$ refutation can be interpreted as a QCDCL$^{\text{PL}}$ refutation. The rest follows by Corollary 4.8, 5.7 and 5.9.

Claim 5: $\mathtt{MirrorCR}_n$ has polynomial-size Q-resolution refutations.

This follows directly from the fact that $\mathtt{MirrorCR}_n$ extends the original QCNF $\mathtt{CR}_n$, which has polynomial-size Q-resolution refutations [18]. We will just ignore the clauses that are not contained in $\mathtt{CR}_n$. ☐

8. Experiments

We study proof systems in the hope that proof-complexity results will translate to running-time complexity for solvers. In this section we do our reality check to see whether this hypothesis holds up experimentally.

We picked the solver DepQBF [22], as it is the only one that supports pure-literal elimination (PLE) and also has the ability to turn cube learning (SDCL) off.[3] We additionally ran Qute [25] when we wanted to confirm DepQBF's surprising behaviour. Though, out of the systems discussed above and defined in Section 3 Qute supports only QCDCL$^{\text{CUBE}}$, and so is not well suited for most of our experiments.

We ran DepQBF on each of the formulas used for separations in this paper, as well as on the PCNF track of the QBF Evaluation 2020.[4] We set the time limit on each formula to 10 minutes (no memory limit). The computation was performed on a machine with two 16-core Intel® Xeon® E5-2683 v4@2.10GHz CPUs and 512 GB RAM running Ubuntu 20.04.3 LTS on Linux 5.4.0-48, and organised with GNU Parallel [28].

We discovered that *heuristics* have a significant impact on performance on theoretically easy formulas (theoretically hard formulas must be hard for solvers as well, and we confirm this in every case). We thus decided to run DepQBF with each available heuristic, in order to paint a full picture. In total, we evaluated 24 configurations of DepQBF—with and without PLE and with and without SDCL, and for each of these four, with each of all 6 available heuristics.[5]

---

[3] In order to obtain vanilla QCDCL in DepQBF, we set `--traditional-qcdcl --long-dist-res --dep-man=simple --no-dynamic-nenofex --no-trivial-truth --no-trivial-falsity`.

[4] http://www.qbflib.org/qbfeval20.php.

[5] We used the parameter `--dec-heur=` to set the decision heuristic in DepQBF and `--phase-heuristic` for the closest corresponding setting for Qute. For each of the solvers, this sets the heuristic that selects the polarity (aka *phase*) of a decision variable, once the decision variable itself has been selected.

The DepQBF heuristics are

- `simple`: always assign to false
- `random`: always flip a coin
- `falsify`: assign to the polarity that satisfies **fewer** not-yet satisfied original clauses (or take a previously cached assignment if one exists)
- `satisfy`: assign to the polarity that satisfies **more** not-yet satisfied original clauses (or take a previously cached assignment if one exists)
- `qtype`: assign to the polarity that satisfies **more if existential and fewer if universal** not-yet satisfied original clauses (or take a previously cached assignment if one exists)
- `sdcl`: assign to the polarity that occurs in **more already satisfied** original clauses (or take a previously cached assignment if one exists)

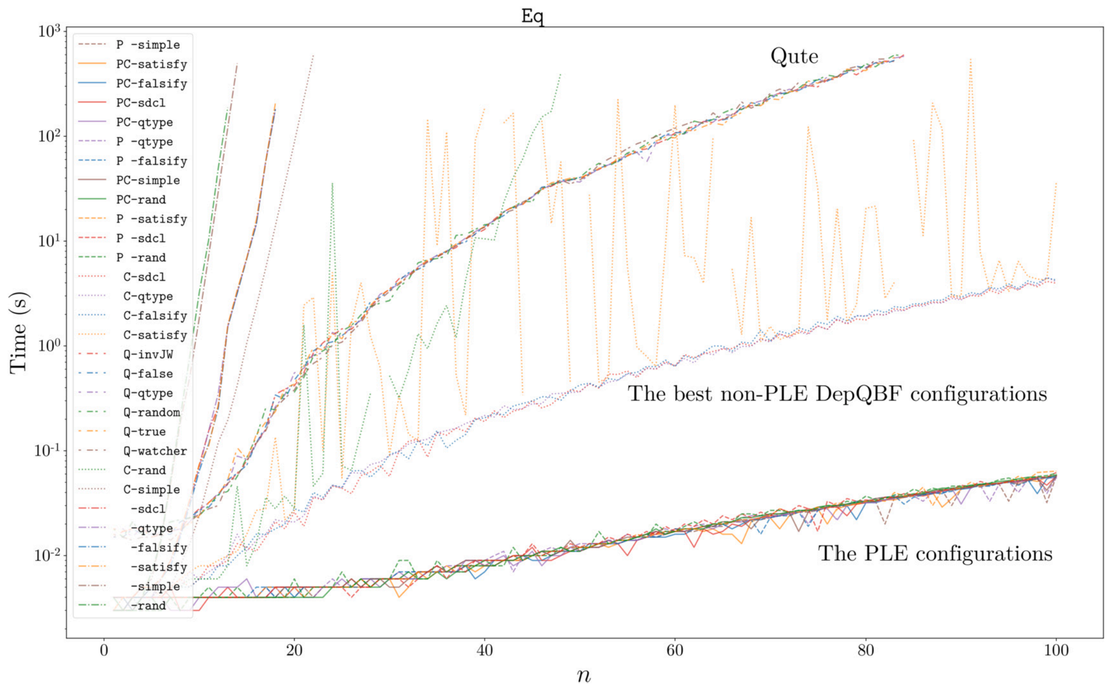

Fig. 3. Labels indicate whether PLE ("P*") and SDCL ("*C") are on, configurations of one kind have the same line style. Lines for Qute start with "Q", the remaining lines are for DepQBF. The rest of the label is the heuristic; configurations with the same heuristic share colour. Gaps in lines indicate time-outs at 10 minutes. The legend is sorted in descending order of performance. (For interpretation of the colours in the figure(s), the reader is referred to the web version of this article.)

Fig. 3 shows the results on Equality. While the formulas are easy regardless of the heuristic when using PLE, without PLE DepQBF's performance fluctuates depending on the heuristic, even though the formulas are 'easy' as long as SDCL is on. Qute's performance is more stable, but still much worse than DepQBF with PLE. The formulas get hard without both PLE and SDCL, in line with [4].

Fig. 4 shows DepQBF's behaviour on `PLTrap` and `TwinEq`. We see that here solver performance matches proof complexity almost perfectly. The only slight discrepancy is that `PLTrap` remains hard without PLE with the heuristics `satisfy` and `qtype`; but the reason for this is simply that these two heuristics fall into the same trap by suggesting to assign y to true at the beginning; indeed, if an existential literal is pure, and its variable chosen for decision, it is readily seen that by definition `satisfy` and `qtype` are equivalent to pure literal elimination.

`MirrorCR` is hard for every configuration (Fig. 5), as it should be. On the other hand, `BulkyEq` exhibits a similarly erratic behaviour as Equality (Fig. 6). We know that `BulkyEq` is easy for $\mathsf{QCDCL}^{\mathsf{CUBE+PL}}$, but hard for $\mathsf{QCDCL}^{\mathsf{PL}}$ (Proposition 6.10), yet somehow it seems the only configurations able to solve `BulkyEq` fast are ones without PLE (and with SDCL). It remains to be seen how PLE hurts solver performance here; we see no apparent 'poisoned pure literal' like in `PLTrap`.

Finally, Fig. 7 shows the performance of DepQBF in the default vanilla QCDCL configuration with and without pure-literal elimination on the PCNF track of QBF Evaluation 2020. With PLE, DepQBF solved 84 formulas, without only 80. 95 formulas were solved by at least one configuration. This serves as an illustration that benefits from pure-literal elimination can be observed outside of crafted proof-complexity formulas. A state-of-the-art solver configuration on industrial formulas would typically include a preprocessor and other techniques that go beyond vanilla QCDCL; here we aim to test just QCDCL with and without PLE.

While both PLE and SDCL make `Eq` easy, PLE seems easier to exploit. It seems this is because PLE can hardly be applied wrongly, while to benefit from cube learning, one must learn the right cubes. This suggests that in spite of its conceptual simplicity PLE can be quite useful, and perhaps should appear on Qute's feature roadmap.

9. Conclusion

While this paper only studies false formulas (in accordance with proof complexity conventions), we expect similar phenomena of incomparability on true formulas, which we leave for future work to explore. Interestingly, while cube learning is primarily needed for true QBFs, we have shown that it can also improve the running time on false instances.

Qute's heuristics are similar: `true` and `false` assign always true and always false respectively, `random` flips a coin once and caches the value (so only the first selection is random; propagation overwrites the cached value in both solvers). `qtype` is like in DepQBF, but without checking whether clauses are currently satisfied, and `invjw` is like `qtype`, but instead of counting clauses, it takes $\sum_{l\in C} 2^{|C|}$ over original clauses C where the literal l occurs.

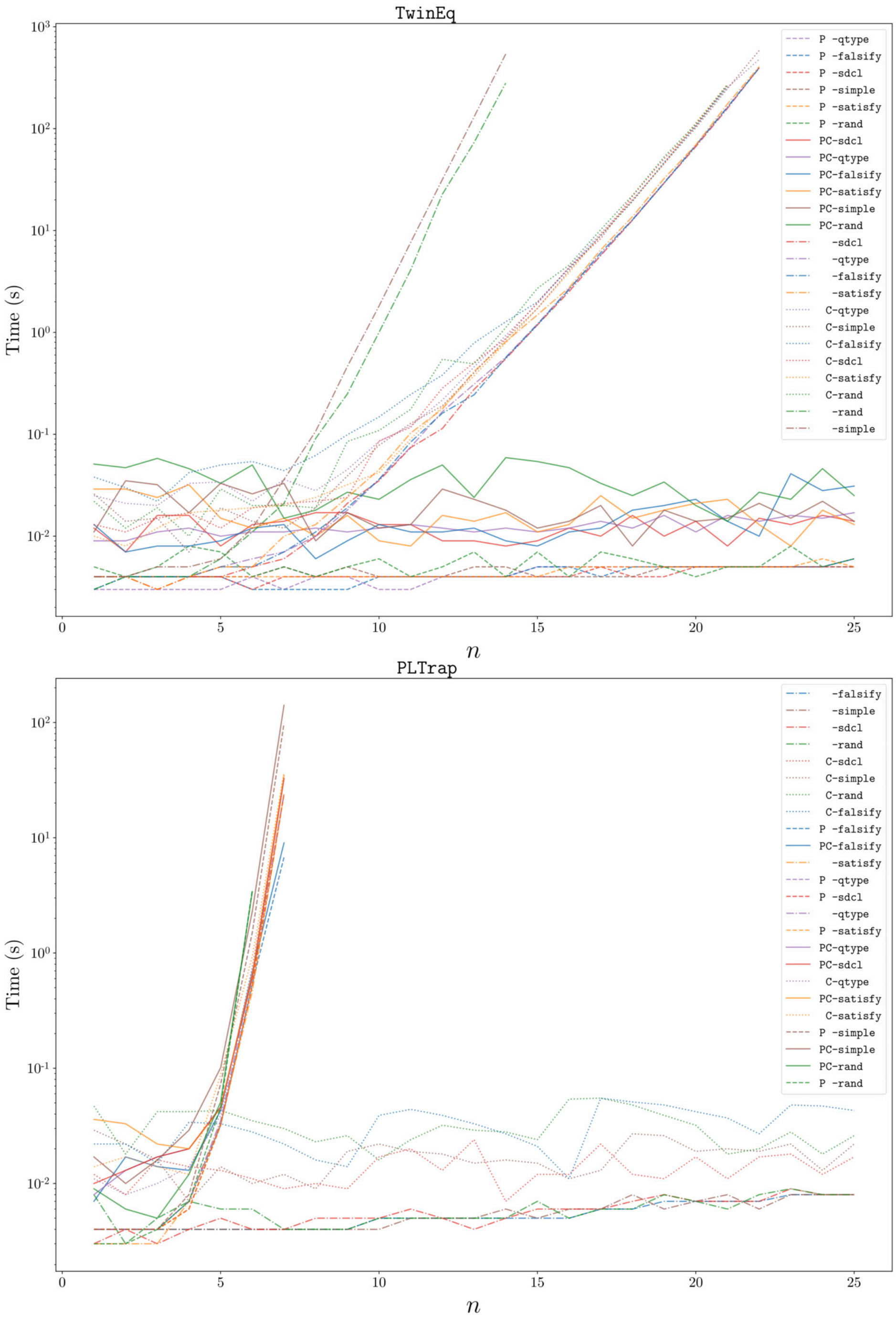

Fig. 4. TwinEq (above) and PLTrap (below) formulas documenting Theorem 6.8. Labels indicate whether PLE ("P*") and SDCL ("*C") are on, configurations of one kind have the same line style. The rest of the label is the heuristic; configurations with the same heuristic share colour. Gaps in lines indicate time-outs at 10 minutes. The legend is sorted in descending order of performance.

Technically, we believe that our new notion of primitive proofs has further potential for showing QCDCL lower bounds, also for QBFs of higher quantifier complexity. While previous results tried to lift lower bounds from Q-Resolution [4], primitivity also applies to QBFs easy for Q-Resolution, thus supplying new reasons for QCDCL hardness.

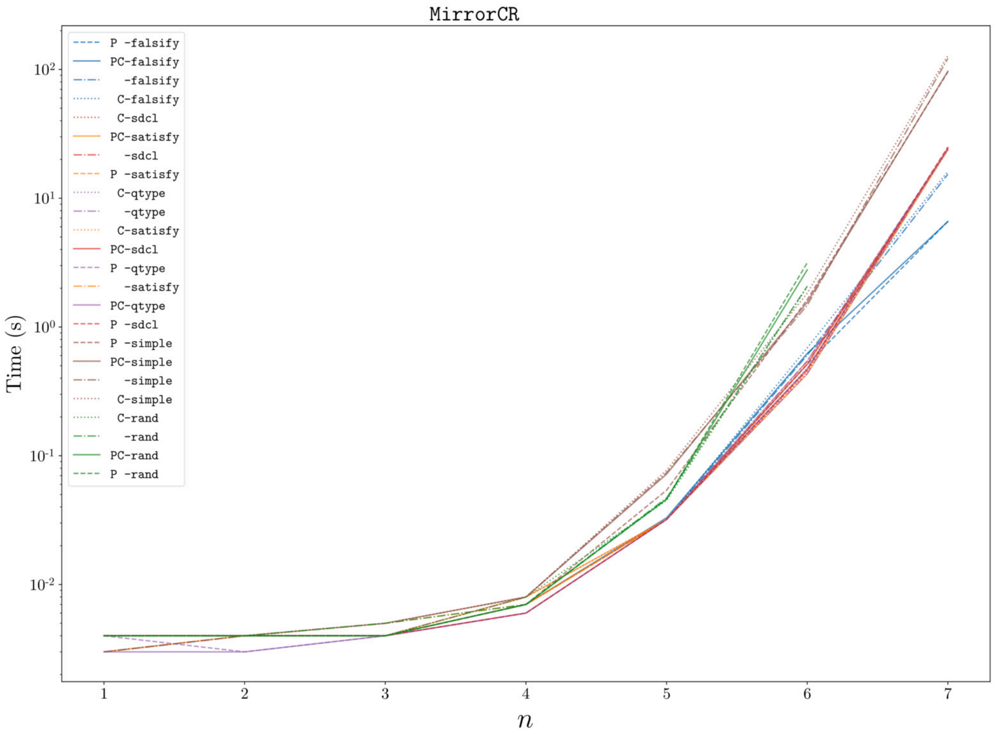

Fig. 5. MirrorCR, the same kind of plot as before. We tested the solver on up to $n = 10$, but all configurations timed out on $n \geqslant 8$.

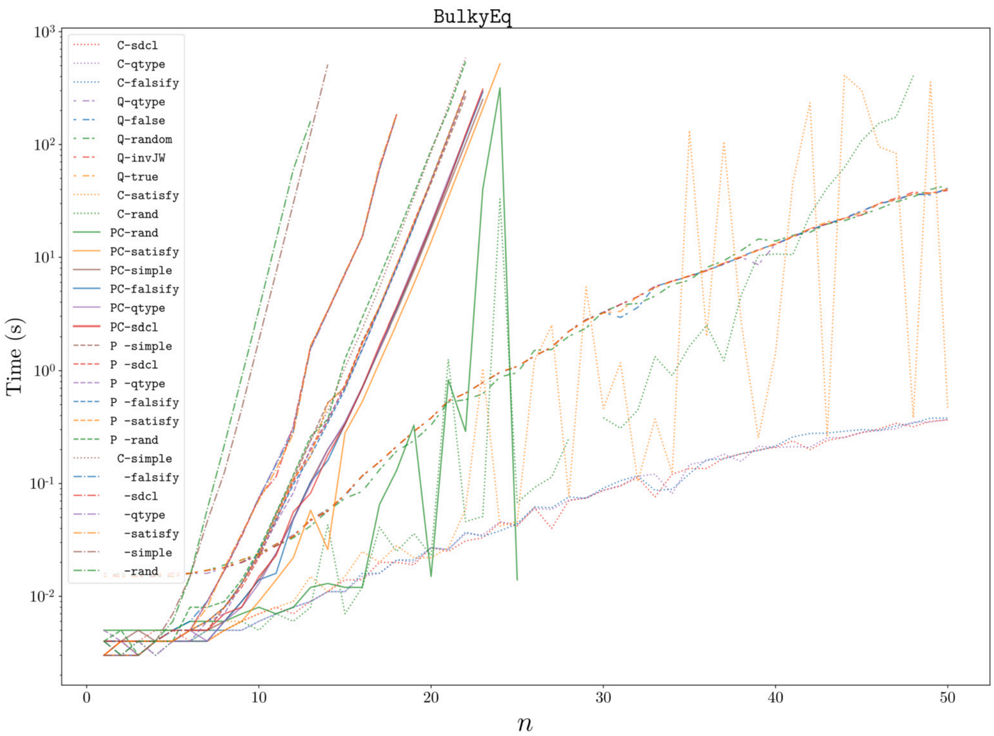

Fig. 6. BulkyEq. Lines for Qute start with "Q", the remaining lines are for DepQBF, otherwise the same kind of plot as before.

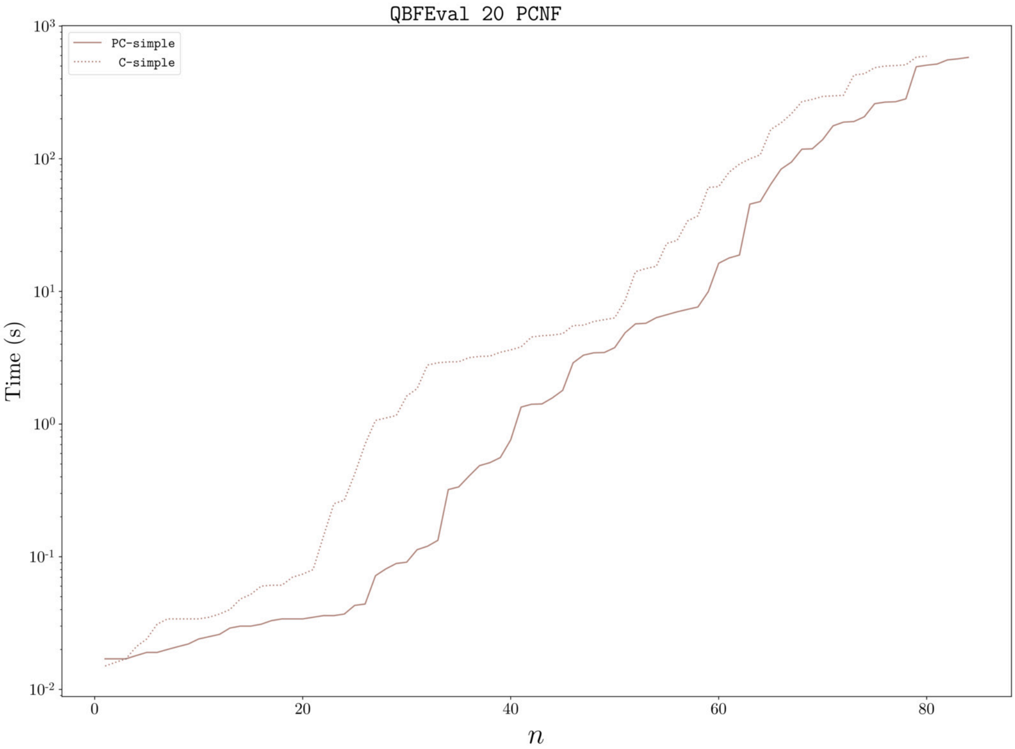

Fig. 7. DepQBF on the QBF Evaluation 2020 PCNF Track. Cactus plot: (x, y) means the configuration solved x instances in y seconds. Right and low is better. Labels like before.

CRediT authorship contribution statement

Benjamin Böhm: Conceptualization, Methodology, Writing – original draft, Writing – review & editing. **Tomáš Peitl:** Conceptualization, Funding acquisition, Methodology, Software, Writing – original draft, Writing – review & editing. **Olaf Beyersdorff:** Conceptualization, Funding acquisition, Methodology, Supervision, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

[1] Albert Atserias, Johannes Klaus Fichte, Marc Thurley, Clause-learning algorithms with many restarts and bounded-width resolution, J. Artif. Intell. Res. 40 (2011) 353–373.
[2] Valeriy Balabanov, Jie-Hong R. Jiang, Unified QBF certification and its applications, Form. Methods Syst. Des. 41 (1) (2012) 45–65.
[3] Paul Beame, Henry A. Kautz, Ashish Sabharwal, Towards understanding and harnessing the potential of clause learning, J. Artif. Intell. Res. 22 (2004) 319–351.
[4] Olaf Beyersdorff, Benjamin Böhm, Understanding the relative strength of QBF CDCL solvers and QBF resolution, Log. Methods Comput. Sci. 19 (2) (2023).
[5] Olaf Beyersdorff, Joshua Blinkhorn, Luke Hinde, Size, cost, and capacity: a semantic technique for hard random QBFs, Log. Methods Comput. Sci. 15 (1) (2019).
[6] Olaf Beyersdorff, Joshua Blinkhorn, Meena Mahajan, Building strategies into QBF proofs, in: STACS, in: LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum Fuer Informatik, 2019, pp. 14:1–14:18.
[7] Olaf Beyersdorff, Leroy Chew, Mikolás Janota, New resolution-based QBF calculi and their proof complexity, ACM Trans. Comput. Theory 11 (4) (2019) 26.
[8] Olaf Beyersdorff, Mikolás Janota, Florian Lonsing, Martina Seidl, Quantified Boolean formulas, in: Armin Biere, Marijn Heule, Hans van Maaren, Toby Walsh (Eds.), Handbook of Satisfiability, Frontiers in Artificial Intelligence and Applications, IOS Press, 2021, pp. 1177–1221.
[9] Olaf Beyersdorff, Proof complexity of quantified Boolean logic – a survey, in: Marco Benini, Olaf Beyersdorff, Michael Rathjen, Peter Schuster (Eds.), Mathematics for Computation (M4C), World Scientific, 2022, pp. 353–391.
[10] Armin Biere, Marijn Heule, Hans van Maaren, Toby Walsh (Eds.), Handbook of Satisfiability, Frontiers in Artificial Intelligence and Applications, IOS Press, 2021.
[11] Armin Biere, Matti Järvisalo, Benjamin Kiesl, Preprocessing in sat solving, in: Armin Biere, Marijn Heule, Hans van Maaren, Toby Walsh (Eds.), Handbook of Satisfiability, Frontiers in Artificial Intelligence and Applications, IOS Press, 2021, pp. 291–436.
[12] Benjamin Böhm, Olaf Beyersdorff, Lower bounds for QCDCL via formula gauge, J. Autom. Reason. 67 (4) (2023) 35.

[13] Benjamin Böhm, Olaf Beyersdorff, QCDCL vs QBF resolution: further insights, in: Meena Mahajan, Friedrich Slivovsky (Eds.), 26th International Conference on Theory and Applications of Satisfiability Testing (SAT), in: LIPIcs, vol. 271, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023, pp. 4:1–4:17.
[14] Benjamin Böhm, Tomás Peitl, Olaf Beyersdorff, QCDCL with cube learning or pure literal elimination - what is best?, in: Luc De Raedt (Ed.), Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI), 2022, pp. 1781–1787, ijcai.org.
[15] Benjamin Böhm, Tomás Peitl, Olaf Beyersdorff, Should decisions in QCDCL follow prefix order?, J. Autom. Reason. 68 (1) (2024) 5.
[16] Sam Buss, Jakob Nordström, Proof complexity and SAT solving, in: Armin Biere, Marijn Heule, Hans van Maaren, Toby Walsh (Eds.), Handbook of Satisfiability, Frontiers in Artificial Intelligence and Applications, IOS Press, 2021, pp. 233–350.
[17] Jan Elffers, Jesús Giráldez-Cru, Stephan Gocht, Jakob Nordström, Laurent Simon, Seeking practical CDCL insights from theoretical SAT benchmarks, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI), 2018, pp. 1300–1308, ijcai.org.
[18] Mikolás Janota, On Q-resolution and CDCL QBF solving, in: Proc. International Conference on Theory and Applications of Satisfiability Testing (SAT), 2016, pp. 402–418.
[19] Hans Kleine Büning, Marek Karpinski, Andreas Flögel, Resolution for quantified Boolean formulas, Inf. Comput. 117 (1) (1995) 12–18.
[20] Janne I. Kokkala, Jakob Nordström, Using resolution proofs to analyse CDCL solvers, in: Principles and Practice of Constraint Programming - 26th International Conference (CP), Springer, 2020, pp. 427–444.
[21] Jan Krajíček, Proof Complexity, Encyclopedia of Mathematics and Its Applications, vol. 170, Cambridge University Press, 2019.
[22] Florian Lonsing, Uwe Egly, DepQBF 6.0: a search-based QBF solver beyond traditional QCDCL, in: Proc. International Conference on Automated Deduction (CADE), 2017, pp. 371–384.
[23] Florian Lonsing, Dependency Schemes and Search-Based QBF Solving: Theory and Practice, PhD thesis, Johannes Kepler University Linz, 2012.
[24] João P. Marques Silva, Inês Lynce, Sharad Malik, Conflict-driven clause learning SAT solvers, in: Armin Biere, Marijn Heule, Hans van Maaren, Toby Walsh (Eds.), Handbook of Satisfiability, Frontiers in Artificial Intelligence and Applications, IOS Press, 2021.
[25] Tomás Peitl, Friedrich Slivovsky, Stefan Szeider, Dependency learning for QBF, J. Artif. Intell. Res. 65 (2019) 180–208.
[26] Knot Pipatsrisawat, Adnan Darwiche, On the power of clause-learning SAT solvers as resolution engines, Artif. Intell. 175 (2) (2011) 512–525.
[27] Karem A. Sakallah, João Marques-Silva, Anatomy and empirical evaluation of modern SAT solvers, Bull. Eur. Assoc. Theor. Comput. Sci. 103 (2011) 96–121.
[28] Ole Tange, GNU Parallel 20211222 ('Støjberg'), GNU Parallel is a general parallelizer to run multiple serial command line programs in parallel without changing them, December 2021.
[29] Marc Vinyals, Hard examples for common variable decision heuristics, in: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), 2020.
[30] Lintao Zhang, Sharad Malik, Conflict driven learning in a quantified Boolean satisfiability solver, in: Proc. IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2002, pp. 442–449.