Journal of Automated Reasoning (2023) 67:35
https://doi.org/10.1007/s10817-023-09683-1

Lower Bounds for QCDCL via Formula Gauge

Benjamin Böhm[1] · Olaf Beyersdorff[1]

Received: 26 June 2022 / Accepted: 29 August 2023 / Published online: 27 September 2023

Abstract
QCDCL is one of the main algorithmic paradigms for solving quantified Boolean formulas (QBF). We design a new technique to show lower bounds for the running time in QCDCL algorithms. For this we model QCDCL by concisely defined proof systems and identify a new width measure for formulas, which we call *gauge*. We show that for a large class of QBFs, large (e.g. linear) gauge implies exponential lower bounds for QCDCL proof size. We illustrate our technique by computing the gauge for a number of sample QBFs, thereby providing new exponential lower bounds for QCDCL. Our technique is the first bespoke lower bound technique for QCDCL.

Keywords QBF · QCDCL · Proof complexity · Resolution · Lower bounds

1 Introduction

The satisfiability problem for propositional formulas (SAT) is one of the central problems of computer science. Traditionally perceived as a hard problem due to its NP completeness, SAT is nowadays very efficiently tackled by SAT solvers, building on the paradigm of conflict-driven clause learning (CDCL) [29], which solve problems in even millions of variables on many industrial problems.

The success of SAT solving has been transferred to computationally even more challenging settings, with quantified Boolean formulas (QBF) receiving key attention during the last decade [12]. One of the main approaches to QBF solving lifts CDCL to the quantified level, resulting in QCDCL [36]. In addition to QCDCL there are a number of further competing approaches to QBF solving [22, 26, 30]. Due to its PSPACE completeness, QBFs allow to

An extended abstract of this paper appeared in the proceedings of SAT'21 [16]. Research was supported by grants from the Carl-Zeiss Foundation and DFG grant BE 4209/3-1. The first author is the main author of this work, both authors contributed to research and writing. This research is theoretical and there is no additional data. There are no competing interests arising from this research.

✉ Olaf Beyersdorff
olaf.beyersdorff@uni-jena.de

Benjamin Böhm
benjamin.boehm@uni-jena.de

[1] Friedrich Schiller University Jena, Jena, Germany

encode many problems more succinctly, thus allowing to tackle even further applications [33].

Understanding which formulas are hard for (Q)CDCL is one of the most fascinating questions, both from a theoretical and a practical point of view. The main approach to this problem is through interpreting runs of SAT and QBF solvers on unsatisfiable formulas as formal proofs of their unsatisfiability. Since learned clauses in CDCL are derivable in resolution, it was noted early on that each run of a CDCL solver on an unsatisfiable formula can be efficiently translated into a resolution refutation [3]. Somewhat surprisingly, the converse holds as well, and when allowing arbitrary non-deterministic decision schemes, CDCL and propositional resolution are equivalent [31]. However, practical CDCL using decision schemes such as VSIDS [35] is exponentially weaker than the full resolution system [34].

Nevertheless, practical CDCL schemes are simulated by resolution and thus proof size lower bounds for resolution translate into lower bounds for CDCL running time. To obtain such lower bounds we can utilise the vast proof complexity machinery of resolution lower bound techniques [24] to show a plethora of lower bounds for combinatorial, random, and further formulas. Indeed, resolution is arguably the best-understood proof system, intensively studied long before the advent of SAT solving.

The situation is somewhat more intricate regarding the relation between QCDCL and Q-resolution, the latter being the simplest and most-studied analogue of propositional resolution for QBF [23]. The first result regarding their relative strength is due to Janota [21], who proved that practical QCDCL does *not* simulate Q-resolution. This can be interpreted as the QBF analogue of Vinyals result for practical CDCL vs resolution [34] (though [21] actually predates [34]). In contrast, the celebrated result on the equivalence of non-deterministic CDCL and resolution [31] does *not* lift to QBF as very recently shown in [7]: (non-deterministic) QCDCL and Q-resolution are incomparable, i.e., there exist formulas exponentially hard for Q-resolution, but easy for QCDCL, and vice versa.

This leaves us with the conundrum of how to show lower bounds for QCDCL. Though we understand Q-resolution fairly well and have a number of dedicated techniques for lower bounds in that system [5, 6, 8, 10, 13–15], unlike in the SAT case, these do not automatically apply to QCDCL.

The existing information on QCDCL lower bounds can be summarized as follows. In addition to the above-mentioned lower bound of [21] for practical QCDCL, we showed in [7] that under certain conditions, lower bounds from Q-resolution can be lifted to QCDCL. Also, while QCDCL runs on false QBFs cannot be efficiently transformed into Q-resolution proofs, they can be translated into long-distance Q-resolution proofs, an exponentially stronger proof system designed to model clause learning in QCDCL [2, 18]. However, we only have very few examples of hard formulas for long-distance Q-resolution [1, 10, 14], which again are lifted from Q-resolution hardness.

In summary, it is fair to say that QCDCL is rather poorly understood from a theoretical point of view and in particular lower bound techniques that would allow to show exponential lower bounds for QCDCL are lacking.

Our Contributions. We devise the *first dedicated lower bound technique for QCDCL* (with arbitrary clause learning mechanisms including those used in practise). In contrast to previous lower bounds for QCDCL, our technique does not import Q-resolution hardness and thus applies to different formulas, regardless of whether they are hard for Q-resolution or not (note that although Q-resolution was shown to be incomparable to QCDCL [7], basically all QCDCL lower bounds were imported from long-distance Q-resolution and hence are lower bounds also for Q-resolution). We already mention at this point though, that our technique is not completely general, but is restricted to Σ_3^b-formulas that meet a certain XT-condition, considered already in [7].

Technically, our approach rests on interpreting QCDCL runs in a formal framework of proof systems, already used in [7]. Further, we define a property of long-distance Q-resolution proofs, which we call *quasi level-ordered*. This is inspired by the notion of level-ordered proofs, introduced in [22], where the order of resolution steps in proofs must follow the quantification order in the prefix. Quasi level-order proofs relax that condition (Definition 4).

Our lower bound technique then rests on two steps: (1) We show that for Σ_3^b-formulas with the XT-condition, QCDCL proofs can be efficiently translated into quasi level-ordered Q-resolution proofs. (2) We define a new measure called the *gauge* of a QBF and show that large (i.e. linear) gauge implies exponential size in quasi level-ordered Q-resolution. Together, (1) and (2) imply that formulas with the XT-property and large gauge are hard for QCDCL (our main Theorem 6).

We illustrate our technique on a couple of examples on which computing the gauge is fairly straightforward. Thus, though showing (1) and (2) above is rather technical, the lower bound technique itself is quite easily applicable.

It is also interesting to mention that our new notion of gauge is some kind of width measure on clauses. Showing proof size lower bounds via width lower bounds is a very well-explored theme in proof complexity, both propositionally [4] and in QBF [6, 9]. We show, however, that gauge and proof width are not related in general.

Organisation. The remainder of this article is organised as follows. We start in Sect. 2 by reviewing notions from QBF, including Q-resolution and long-distance Q-resolution. In Sect. 3 we sketch QCDCL and explain how to model it as a formal proof system QCDCL. In Sect. 4 we introduce a new notion of quasi level-ordered proofs and give an algorithm to translate QCDCL proofs into quasi-level ordered Q-resolution. Section 5 introduces our lower bound method for quasi-level ordered proofs via the gauge measure, which we apply in Sect. 6 to a number of old and new QBF families. We conclude in Sect. 7 with some open questions.

2 Preliminaries

Propositional and Quantified Formulas. Variables and negated variables are called *literals*, i.e., for a variable x we can form two literals: x and its negation $\bar{x}$. We denote the corresponding variable as $\text{var}(x) := \text{var}(\bar{x}) := x$.

A *clause* is a disjunction of literals, sometimes also viewed as a set of literals. The *empty clause* is the clause consisting of zero literals, denoted $(\bot)$. Terms are conjunctions of literals. Again, terms can be considered as sets of literals. A *CNF* (*conjunctive normal form*) is a conjunction of clauses. For $C = \ell_1 \vee \ldots \vee \ell_m$ we define $\text{var}(C) := \{\text{var}(\ell_1), \ldots, \text{var}(\ell_m)\}$. For a CNF $\phi = C_1 \wedge \ldots \wedge C_n$ we define $\text{var}(\phi) := \bigcup_{i=1}^n \text{var}(C_i)$. A clause C is called *tautological*, if there is a variable x with $x, \bar{x} \in C$.

An *assignment* σ of a set of variables X is a non-tautological set of literals, such that for all $x \in X$ there is $\ell \in \sigma$ with $\text{var}(\ell) = x$. The restriction of a clause C by an assignment σ is defined as $C|_\sigma := \top$ (true) if $C \cap \sigma \neq \emptyset$, and $\bigvee_{\ell \in C, \ell \notin \sigma} \ell$ otherwise. One can interpret σ as an operator that sets all literals from σ to the Boolean constant 1. We denote the set of assignments of X by $\langle X \rangle$.

A *QBF* (*quantified Boolean formula*) $\Phi = \mathcal{Q} \cdot \phi$ is a propositional formula ϕ (also called *matrix*) together with a *prefix* $\mathcal{Q}$. A prefix $Q_1 x_1 Q_2 x_2 \ldots Q_k x_k$ consists of variables $x_1, \ldots, x_k$

and quantifiers $Q_1, \dots, Q_k \in \{\exists, \forall\}$. We obtain an equivalent formula if we unite adjacent quantifiers of the same type. Therefore we can always assume that our prefix is in the form of $\mathcal{Q} = Q'_1 X_1 Q'_2 X_2 \dots Q'_s X_s$ with non-empty sets of variables $X_1, \dots, X_s$ and quantifiers $Q'_1, \dots, Q'_s \in \{\exists, \forall\}$ such that $Q'_i \neq Q'_{i+1}$ for $i \in [s-1]$. For a variable x in $\mathcal{Q}$ we denote the *quantifier level* with respect to $\mathcal{Q}$ by $\text{lv}(x) = \text{lv}_\Phi(x) = i$, if $x \in X_i$. Variables from Φ are called *existential*, if the corresponding quantifier is $\exists$, and *universal* if the quantifier is $\forall$.

A QBF with CNF matrix is called a *QCNF*. We require that all clauses from a matrix of a QCNF are non-tautological, otherwise we just delete these clauses. We further require that all variables in the matrix appear in the prefix. Since we will only discuss refutational proof systems, we only consider false QCNFs.

A QBF can be interpreted as a game between two players $\exists$ and $\forall$. These players have to assign the respective variables one by one along the quantifier order from left to right. The $\forall$-player wins the game if and only if the matrix of the QBF gets falsified by this assignment. It is well known that for every false QBF $\Phi = \mathcal{Q} \cdot \phi$ there exists a winning strategy for the $\forall$-player.

Q-Resolution and Long-Distance Q-Resolution. Let C_1 and C_2 be two clauses. Let also ℓ be an existential literal with $\text{var}(\ell) \notin \text{var}(C_1) \cup \text{var}(C_2)$. Then the *resolvent* of $C_1 \vee \ell$ and $C_2 \vee \bar{\ell}$ over ℓ is defined as

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

Let $C := u_1 \vee \dots \vee u_m \vee x_1 \vee \dots \vee x_n \vee v_1 \vee \dots \vee v_s$ be a clause from Φ, where $u_1, \dots, u_m, v_1, \dots, v_s$ are universal literals, $x_1, \dots, x_n$ are existential literals and $v_1, \dots, v_s$ are exactly those literals $v \in C$ such that v is universal and $\text{lv}(v) > \text{lv}(x_i)$ for all $i \in [n]$. Then we can perform a reduction step and obtain

$$\text{red}(C) := (u_1 \vee \dots \vee u_m \vee x_1 \vee \dots \vee x_n).$$

For a CNF $\phi = \{C_1, \dots, C_k\}$ we define $\text{red}(\phi) := \{\text{red}(C_1), \dots, \text{red}(C_k)\}$.

Q-resolution [23] is a refutational proof system for false QCNFs. A Q-resolution proof π of a clause C from a QCNF $\Phi = \mathcal{Q} \cdot \phi$ is a sequence of clauses $\pi = C_1, \dots, C_m$ with $C_m = C$. Each C_i has to be derived by one of the following three rules:

- *Axiom:* $C_i \in \phi$;
- *Resolution:* $C_i = C_j \overset{x}{\bowtie} C_k$ for some $j, k < i$ and $x \in \text{var}_\exists(\Phi)$, and C_i is non-tautological;
- *Reduction:* $C_i = \text{red}(C_j)$ for some $j < i$.

Note that none of our axioms are tautological by definition. A *refutation* of a QCNF Φ is a proof of the empty clause $(\bot)$.

To model clause learning in QCDCL, the proof system long-distance Q-resolution was introduced in [2, 36]. This extension of Q-resolution allows to derive universal tautologies under specific conditions such that the resulting system is still sound (this would not be the case for arbitrary tautologies). As in Q-resolution, there are three rules by which a clause C_i can be derived. The axiom and reduction rules are identical to Q-resolution, but the resolution rule is changed to

- *Resolution (long-distance):* $C_i = C_j \overset{x}{\bowtie} C_k$ for some $j, k < i$ and $x \in \text{var}_\exists(\Phi)$. The resolvent C_i is allowed to contain a tautology $u \vee \bar{u}$ if u is a universal variable. If $u \in \text{var}(C_j) \cap \text{var}(C_k)$, then we additionally require $\text{lv}(u) > \text{lv}(x)$.

Note that a long-distance Q-resolution proof without tautologies is just a Q-resolution proof.

If $\pi = C_1, \ldots, C_m$ is a proof, we define a *path* in π as a subsequence $C_{i_1}, \ldots, C_{i_s}$ of π, such that each C_{i_j} is a parental clause of $C_{i_{j+1}}$ for each $j \in [s-1]$, either by resolution or reduction.

3 QCDCL as a Formal Proof System

In this section we review quantified conflict-driven clause learning (QCDCL) and its formalisation as a proof system from [7]. This provides the formal framework for our subsequent proof complexity analysis.

QCDCL is the quantified version of the well-known CDCL algorithm (see [29, 35] for further details on CDCL, and [19, 28, 36] for QCDCL). Let $\Phi = \mathcal{Q} \cdot \phi$ be a false QCNF. Roughly speaking, QCDCL consists of two interleaved processes: *propagation* and *learning*.

In the *propagation process* we generate assignments with the goal to either find a satisfying assignment or to obtain a conflict. We start with clauses from ϕ that force us to assign literals such that we do not falsify these clauses (called unit clauses). The underlying idea of this process is *unit propagation*. One can think of a clause $x_1 \vee \ldots \vee x_n$ as an implication $(\bar{x}_1 \wedge \ldots \wedge \bar{x}_{n-1}) \rightarrow x_n$. That is, if we already assigned the literals $\bar{x}_1, \ldots, \bar{x}_{n-1}$, then we are forced to assign x_n in order to satisfy this clause. In QBF, we also insert reduction steps into this process, i.e., we are interested in clauses that become unit after reduction. For example, the clause $(\bar{x}_1 \wedge \ldots \wedge \bar{x}_{n-1}) \rightarrow (x_n \vee u)$ for an existential literal x_n and a universal literal u with $\text{lv}(x_n) < \text{lv}(u)$ can also be used as a ground clause for propagating x_n.

Performing unit propagation, the goal is to prevent a conflict for as long as possible. However, it is not guaranteed that we can even perform any unit propagations by just starting with the formula. Therefore we will make *decisions*, i.e., we assign literals without any solid reason. With the aid of these decisions (one can also think of assumptions) we can provoke further unit propagations. Since decision making is one of the non-deterministic components of the algorithm, we only make decisions if there are no more unit propagations available. In QCDCL these decisions follow the quantification order, i.e., we always decide a variable from the leftmost quantifier block.

After obtaining a conflict, i.e., falsifying a clause, we start the *clause learning process*. Here the underlying idea is to use Q-resolution resp. long-distance Q-resolution. We start with the clause that caused the conflict and resolve it with clauses that implied previous literals in the assignment in the reverse propagation order. At the end we get a clause such that is derived from existing clauses by long-distance Q-resolution. We add the learned clause to ϕ, backtrack to a state before we assigned all literals of this clause and restart the propagation process. The algorithm ends when we learn the empty clause $(\bot)$ and therefore obtain a refutation of Φ.

QCDCL has to handle both refutations of false formulas as well as prove the validity of true formulas. Therefore one would additionally need to implement *cube learning* (or *term learning*) for satisfying assignments. Since we are only interested in refutations (otherwise we could not compare with Q-resolution), we will omit this aspect of QCDCL.

To prove rigorous lower bounds on the running time of QCDCL we cast QCDCL as a formal proof system. We recall the relevant details from [7], where we fully formalised all components of QCDCL. Each QCDCL run consists of backtracking steps and restarts.

Between them we create *trails*, in which we store all information on decisions and unit propagations.

Definition 1 (trails, repeated from [7]) Let $\Phi = \mathcal{Q} \cdot \phi$ be a QCNF in n variables. A *trail* $\mathcal{T}$ for Φ is a sequence of literals (or $\bot$) of variables from Φ with some specific properties. We distinguish two types of literals in $\mathcal{T}$: *decision literals*, that can be both existential and universal, and propagated literals, that are either existential or $\bot$. We write a trail $\mathcal{T}$ as

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

where we denote decision literals by d_i and propagated literals by $p_{(i,j)}$. We are not allowed to make a new decision unless there are no more propagations possible. Also, decision literals have to be level-ordered, i.e., we have to choose a leftmost quantified variable (still unassigned) as the next decision.

There are some further requirements on $\mathcal{T}$, for which we refer to [7]. However, as they are not crucial for our lower bounds, we can safely ignore them for now.

For unit propagation we need the notion of *unit clauses* that allow us to assign a variable without making a decision. We call a clause C a *unit clause* if $\text{red}(C) = (x)$ for an existential literal x or $x = \bot$.

The next definition presents the main framework for the analysis of QCDCL as a proof system. After having defined trails in a general way, we want to specify the way a trail can be generated during a QCDCL run.

Definition 2 (QCDCL proof systems [7]) Let $\Phi = \mathcal{Q} \cdot \phi$ be a QCNF. We call a triple of sequences

$$\iota = ((\mathcal{T}_1, \ldots, \mathcal{T}_m), (C_1, \ldots, C_m), (\pi_1, \ldots, \pi_m))$$

a QCDCL *proof* from Φ of a clause C, if for all $i \in [m]$ the trail $\mathcal{T}_i$ uses the QCNF $\mathcal{Q} \cdot (\phi \cup \{C_1, \ldots, C_{i-1}\})$, where C_j is a clause learnable from $\mathcal{T}_j$ and $C_m = C$. Each π_i is the long-distance Q-resolution derivation of the clause C_i from $\mathcal{Q} \cdot (\phi \cup \{C_1, \ldots, C_{i-1}\})$ that we learned from the trail $\mathcal{T}_i$.

Between two trails $\mathcal{T}_i$ and $\mathcal{T}_{i+1}$ we backtrack to some point which we can choose freely. Backtracking to the start (before any variable was assigned) is called restarting. If $C = (\bot)$ we call ι a *refutation*.

By sticking together $\pi_1, \ldots, \pi_m$, we obtain a long-distance Q-resolution derivation π of C from Φ. We identify QCDCL proofs with this exact π.

We require that all trails are naturally created, which means that we are not allowed to skip unit propagations if they are possible, as we explained before. A more detailed description of this condition is given in [7].

We remark that though QCDCL proofs are basically long-distance Q-resolution derivations (i.e., QCDCL is simulated by long-distance Q-resolution), these system are not equal as QCDCL imposes a particular structure on long-distance Q-resolution proofs. Indeed, long-distance Q-resolution is exponentially stronger than QCDCL (cf. [7]).

4 Quasi Level-Ordered Proofs

For the remainder of this article we will entirely focus on Σ_3^b formulas and throughout fix the prefix $\exists X \forall U \exists T$, where X, U, and T are pairwise disjoint and non-empty sets of variables.

Our ultimate aim will be to develop a lower bound technique for such formulas for QCDCL. Conceptually, our technique is inspired by an approach for level-ordered proofs, which is why we recall that notion from [22].

Definition 3 ([22]) A long-distance Q-resolution proof π from a QCNF Φ of a clause C is called *level-ordered* if for each path P in π and two resolution steps in P over variables ℓ_1 and ℓ_2 the following holds: if the resolution over ℓ_1 is closer to the root C than the resolution over ℓ_2, then $\mathrm{lv}(\ell_1) \leq \mathrm{lv}(\ell_2)$.

For level-ordered proofs one can devise lower bounds as follows. A level-ordered long-distance Q-resolution refutation π of a Σ_3^b-formula $\Phi = \exists X \forall U \exists T \cdot \phi$ always starts with T-resolutions and ends with X-resolutions. We then count the clauses consisting only of X-literals at the transitions from a T-resolution to some X-resolution. For each $\tau \in \langle X \rangle$ we can find such a clause C_τ that is falsified by τ. Note that C_τ does not necessarily need to contain literals from all X-variables. Hence, the C_τ clauses do not need to be pairwise distinct. However, we will show that for a particular class of formulas, the C_τ clauses cannot be too small. Therefore each C_τ can only cover few assignments $\tau' \in \langle X \rangle$ and the number of these clauses is still exponential.

We will use this idea in a more general setting by introducing the notion of *quasi level-ordered* proofs where only the existence of these C_τ is required.

Definition 4 A long-distance Q-resolution refutation π of a Σ_3^b formula with prefix $\exists X \forall U \exists T$ is called *quasi level-ordered*, if for each assignment $\tau \in \langle X \rangle$ there exists an X-clause C_τ which is falsified by τ and the subproof $\pi_{C_\tau} \subseteq \pi$ of C_τ is level-ordered.

Clearly, level-ordered proofs are quasi level-ordered, but the converse does not hold in general.

In Sect. 5 we will devise a lower bound technique for quasi level-ordered proofs. To get the connection to QCDCL, we show that each QCDCL refutation of Σ_3^b formulas with a special property can be efficiently transformed into a quasi level-ordered Q-resolution refutation. The property needed is the *XT-property*, which we recall from [7].

Definition 5 [7] Let Φ be a QCNF of the form $\exists X \forall U \exists T \cdot \phi$. We call a clause C in the variables of Φ

- *X-clause*, if $\mathrm{var}(C) \cap U = \emptyset$ and $\mathrm{var}(C) \cap T = \emptyset$,
- *T-clause*, if $\mathrm{var}(C) \cap X = \emptyset$, $\mathrm{var}(C) \cap U = \emptyset$ and $\mathrm{var}(C) \cap T \neq \emptyset$,
- *XT-clause*, if $\mathrm{var}(C) \cap X \neq \emptyset$, $\mathrm{var}(C) \cap U = \emptyset$ and $\mathrm{var}(C) \cap T \neq \emptyset$,
- *XUT-clause*, if $\mathrm{var}(C) \cap X \neq \emptyset$, $\mathrm{var}(C) \cap U \neq \emptyset$ and $\mathrm{var}(C) \cap T \neq \emptyset$.

We say that Φ fulfils the *XT-property* if ϕ contains no XT-clauses as well as no unit T-clauses and there do not exist two T-clauses $C_1, C_2 \in \phi$ that are resolvable.

Intuitively, this says that there is no direct connection between the X- and T-variables, i.e., Φ does not contain clauses with X- and T-variables, but no U-variables. This XT-property allows us to prove several properties regarding QCDCL refutations.

Lemma 1 [7] *Let Φ be a QCNF that fulfils the XT-property. Then the following holds:*

1. *It is not possible to derive XT-clauses by* long-distance Q-resolution.
2. *It is not possible to resolve two XUT-clauses over an X-literal in a* QCDCL *proof.*
3. *Each* QCDCL *refutation of Φ is a* Q-resolution *refutation (not just a* long-distance Q-resolution *refutation).*

Algorithm 1

$M_X := \{m\}$;
$M_{XUT} := \emptyset$;
$L := \emptyset$;
$\pi' := \pi$;
$i := 1$;
while $M_X \neq \emptyset$ **do**
  while $M_X \neq \emptyset$ **do**
    choose $c \in M_X$ maximal;
    if subproof π_{C_c} of C_c is level-ordered **then**
      add c to L;
    else
      if last step in π'_{C_c} was a resolution over X, say $C_c = C_d \overset{x}{\bowtie} C_e$ **then**
        add d and e to M_X;
      else
        Under all transitions from X-resolutions to T-resolutions in π'_{C_c} of the form $C_d \overset{x}{\bowtie} C_e = C_f$ and $C_f \overset{t}{\bowtie} C_g = C_j$ let $\{d, e\}$ be maximal with respect to $\preccurlyeq$;
        W.l.o.g. let C_d be the XUT-clause and C_e be the X-clause (otherwise swap d and e);
        add (d, e, c) to M_{XUT};
        add e to M_X;
      end if
    end if
    delete c from M_X;
  end while
  $M^{(i)}_{XUT} := M_{XUT}$;
  $i := i + 1$;
  while $M_{XUT} \neq \emptyset$ **do**
    Choose $(d, e, c) \in M_{XUT}$;
    Let $C_d, C_{a_1}, C_{a_2}, \ldots, C_{a_k}, C_c$ be the path from C_d to C_c. Since C_c is an X-clause, all T-literals from C_d have to be resolved away. Let $C_{a_1} = C_d \overset{x}{\bowtie} C_e$, $C_{a_j} = C_{a_{j-1}} \overset{r_j}{\bowtie} C_{b_{j-1}}$ for T-variables r_j, some indices b_{j-1}, $j = 2, \ldots, k$ and $C_c = \text{red}(C_{a_k})$;
    Add the clauses $C_{a'_2} := C_d \overset{r_1}{\bowtie} C_{b_1}$, $C_{a'_j} := C_{a'_{j-1}} \overset{r_j}{\bowtie} C_{b_{j-1}}$ for $j = 3, \ldots, k$ and $C_{a'_{k+1}} := \text{red}(C_{a'_k})$. If somewhere the resolution does not work due to a lacking literal r_j or x, we define the corresponding $C_{a'_j}$ as the clause that lacks this literal. The $C_{a'_j}$ are inserted at the end of the proof.;
    add a'_{k+1} to M_X;
    delete (d, e, c) from M_{XUT};
  end while
end while

For the following results, we will always assume that all three properties from Lemma 1 are fulfilled.

Now we will work towards the transformation of QCDCL proofs into quasi level-ordered Q-resolution refutations. This transformation is described as an algorithm in the following theorem.

Intuitively, the algorithm takes as input a long-distance Q-resolution refutation π that was extracted from a QCDCL refutation of a QCNF that fulfils the XT-property (which is a Q-resolution proof by Lemma 1) and adds a polynomial number of clauses (and resolution steps), such that the obtained proof is quasi level-ordered (i.e., it contains C_τ for each $\tau \in \langle X \rangle$, c.f. Definition 4). The idea is that all the C_τ from the definition of quasi level-ordered proofs are already somehow contained in π, but they might be hidden in XUT-clauses. The algorithm detects these XUT-clauses and eliminates the T-literals by changing the order of resolutions over X-variables and T-variables. Note that the algorithm will only add parts to π and never delete anything (which is fine as quasi level-ordered proofs only require the existence of these C_τ, but they do not need to contribute to the refutation).

Initially, the algorithm checks if the proof of the empty clause is already level-ordered. If this is the case, then there is nothing to do as the technique for level-ordered proofs can be applied to find the C_τ as described above. Otherwise, the algorithm tries to find the last step in the subproof of the currently considered clause (which is the empty clause at the beginning, but might also be non-empty later) that violated the level order, which is just the last transition from an X-resolution to a T-resolution. When this transition is found, we have detected a path of clauses that starts with one X-resolution step, followed by an at most linear-sized sequence of T-resolution steps that ends with the currently considered clause (c.f. Figure 2). The algorithm then adds a path of at most the same size that now starts with all the T-resolutions and reductions (we do not need the X-resolution at the and as we are only interested in parent clauses, c.f. Figure 4). Constructing this path can be done easily because we know (by Lemma 1) that only one of the two clauses from the X-resolution step can contain T-literals (if were both did, we would need to create two new paths, and therefore potentially increase the proof size exponentially in further loops). The clause at the end of this added path as well as the X-clause from the previous X-resolution step will become an observed clause for the next loop (in Figure 4, these would be clauses $C_{a'_{k+1}}$ and C_e).

Theorem 2 *Let Φ be a Σ_3^b QCNF that fulfils the XT-property. Then, using Algorithm 1, each* QCDCL *refutation π (or more formally, the extracted* Q-resolution *refutation π) of Φ can be efficiently transformed into a quasi level-ordered* Q-resolution *refutation π' of Φ with $|\pi'| \in \mathcal{O}(|\pi|^4)$.*

Proof First, because of the XT-property each refutation extracted from a QCDCL run is also a Q-resolution refutation (cf. Lemma 1). That means we will only consider π as the Q-resolution proof that was extracted from the QCDCL run.

Let $\pi = C_1, \ldots, C_m = \bot$. Note that clauses could occur more than once in a proof since we cannot simply shorten a proof in QCDCL. Hence we will use indices to identify clauses in a proof. Each index not only determines the clause itself, but also its position in the proof. This is the reason why we will only use indices in the algorithm in order to store information about a particular clause.

Technically, we define an order that will help us determine if a resolution $C_d \bowtie C_e$ takes place before or after another resolution $C_{d'} \bowtie C_{e'}$ in a given proof.

For this we define a total order $\preccurlyeq$ on $\{\{d, e\} : d, e \in \mathbb{N}, \ d \neq e\}$ as follows:

$$A \preccurlyeq B \Leftrightarrow \max A < \max B \text{ or } (\max A = \max B \text{ and } \min A \leq \min B).$$

We use the notation $A \prec B$ for $A \preccurlyeq B$ and $A \neq B$.

We sketch how the transformation (Algorithm 1) works: Throughout the whole process we work with two sets M_X and M_{XUT}. The set M_X contains indices of X-clauses, where initially we start with $M_X = \{m\}$ (remember that $C_m = (\bot)$). For each $c \in M_X$ we check whether the clause C_c has a level-ordered subproof. If the subproof is not level-ordered, and if the last step before C_c (i.e., the last step in the subproof π_{C_c}) was an X-resolution, we just add the indices both parent clauses of C_c to M_X and delete c from it. Otherwise, if the subproof is not level-ordered, but the last step before C_c was no X-resolution, we search for the last transition that violates the level-order condition. This must be a transition from an X-resolution to a T-resolution. After this transition there will be only T-resolutions until

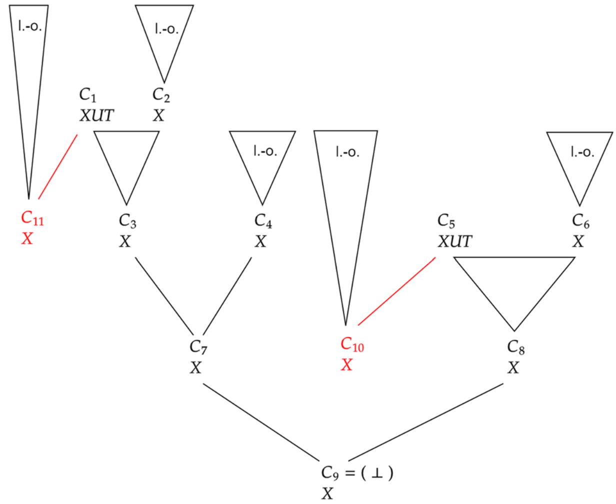

Fig. 1 Sketch of the functionality of the algorithm. Below each clause C_j we specify the type of clause (X- or XUT-clause). Newly added parts are coloured red. Triangles labeled with "l.-o." are level-ordered subproofs, otherwise they are not level-ordered and we can find a transition from an X-resolution to a T-resolution. The corresponding clause C_c is then one of the C_τ clauses for a particular τ

we reach C_c. One of the parent clauses of this X-resolution, which we call C_d and C_e, is an X-clause and the other one is an XUT-clause due to the XT-property (Lemma 1). The index of the X-clause (either d or e) is again stored in M_X, while we delete c from M_X. However, for the XUT-clauses, which are stored as triples (d, e, c) in M_{XUT} (where C_d is the XUT-clause), we have to add several clauses to the proof, including a new X-clause $C_{a'}$. This clause $C_{a'}$ is then added to M_X as well, and the loop repeats until there are no more clauses in M_X left. Note that these added clauses will be part of a dead end in the proof and therefore are not necessary for the refutation itself. However, we need these new clauses for a counting argument in our lower bound technique.

We will show that at the end we return a proof that is quasi level-ordered. More specifically, the X-clauses we detect during the run whose subproofs are level-ordered will be exactly the clauses C_τ from the definition of quasi level-ordered proofs. This holds because, starting from the empty clause, whenever we detect an X-resolution we can choose which parent clause we will consider next. Hence we can choose the polarity of the X-variable we resolve over in the current step. At the end, this last X-clause (whose subproof is level-ordered) only consists of variables with the right polarity as previously chosen. Figure 1 depicts how the algorithm transforms a proof. □

Claim 1 Each step is well-defined and the algorithm terminates.

Proof Let us consider the first inner while loop from line 7 to 22. For each $c \in M_X$ that we delete during the loop, we will add d and e (or sometimes only one of them) to M_X such that C_d and C_e both have smaller depth than C_c. Therefore this loop will repeat only finitely often.

Note that for each Q-resolution proof that is not level-ordered, we can find at least one transition from an X-resolution to a T-resolution. Because of the XT-property, we do not have any XT-clauses and also no X-resolutions over two XUT-clauses. The only two remaining

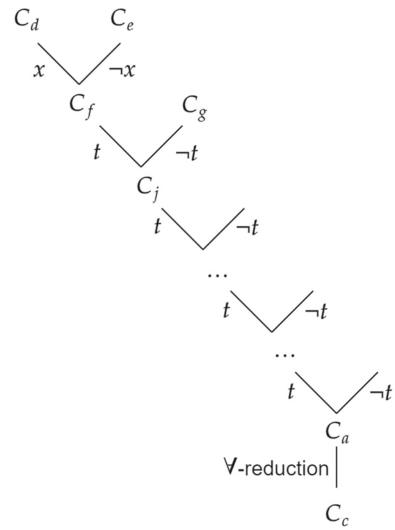

Fig. 2 Last transition from an X-resolution $C_d \overset{x}{\bowtie} C_e$ to a T-resolution $C_f \overset{t}{\bowtie} C_g$ in the subproof of C_c as it is detected in line 12 of Algorithm 1. The literals x (resp. t) are placeholders for some X- (resp. T-)literals. The T-literals do not need to be equal

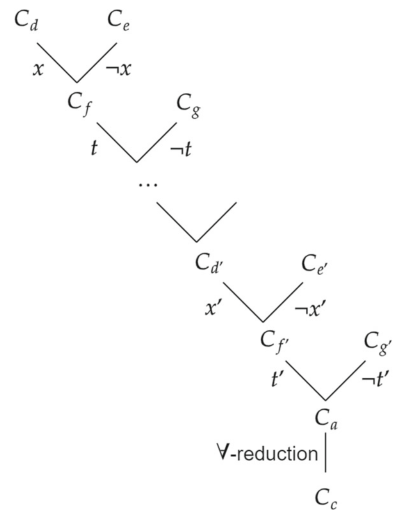

Fig. 3 Suppose that after the detected $\{d, e\}$ there is another set $\{d', e'\}$ which initializes a transition from an X-resolution to a T-resolution. However, this would contradict the maximality of $\{d, e\}$ since we would have $d' > \max\{d, e\}$ and therefore $\{d, e\} \prec \{d', e'\}$

possibilities for X-resolutions are between two X-clauses or between an XUT-clause and an X-clause. Let $C_d \overset{x}{\bowtie} C_e = C_f$ and $C_f \overset{t}{\bowtie} C_g = C_j$ be the transition we detected in the algorithm and as sketched in Figure 2. The case where both C_d and C_e are X-clauses is impossible since the next step is a T-resolution. So we can assume that we find an XUT- and an X-clause. For each $(d, e, c) \in M_{XUT}$ we have that C_d is the XUT-clause and C_e is the X-clause.

There cannot be another transition from an X-resolution to a T-resolution on a path downwards starting with the above transition since this would contradict the maximality of $\{d, e\}$, cf. Figure 3. Hence the found transition is indeed the (or "a") last one.

In the second inner while-loop from line 25 to 31 we will add only finitely many new clauses to the proof. Note that all added clauses are inserted after the original clauses. Since we have only added finitely many triples to M_{XUT} until this point, we will repeat this loop only finitely often, as well.

Let us now concentrate on the outer loop from line 6 to 32. We will show that this loop will repeat only $|\pi|^2$ times.

For each iteration i let

$$K_i := \max_{\preccurlyeq}\{\{d, e\} : (d, e, c) \in M^{(i)}_{XUT} \text{ for some index} c \in \mathbb{N}\}.$$

For each $(d_i, e_i, c_i) \in M^{(i)}_{XUT}$ let c'_i be the index a'_{k+1} of the clause we add to π' corresponding to (d_i, e_i, c_i) as described in the algorithm. If these c'_i are contained in a triple in the next $M^{(i+1)}_{XUT}$, say $(d_{i+1}, e_{i+1}, c'_i) \in M^{(i+1)}_{XUT}$, then we have $\{d_{i+1}, e_{i+1}\} \prec \{d_i, e_i\}$. We cannot have $\{d_i, e_i\} = \{d_{i+1}, e_{i+1}\}$ simply due to the fact that c'_i has no path to the resolution $C_{d_i} \overset{x}{\bowtie} C_{e_i}$ since we skipped the resolution with C_{e_i}. We cannot get $\{d_i, e_i\} \prec \{d_{i+1}, e_{i+1}\}$ either because otherwise we would have chosen $\{d_{i+1}, e_{i+1}\}$ instead of $\{d_i, e_i\}$ when we considered c_i in the iteration before.

We conclude that we have $K_{i+1} \prec K_i$ for each iteration i. These K_i are sets consisting of indices from original clauses since the corresponding clauses C_d and C_e got resolved over an X-variable in π and new clauses that were added during the run of the algorithm appear only in resolution steps over T-literals and reduction steps. Hence we can argue that we will repeat the outer while-loop at most $|\pi|^2$ times. □

Claim 2 At the end, we have $|\pi'| \in \mathcal{O}(|\pi|^4)$.

Proof We have to count the number of clauses we add to π' in each iteration. A visualization of this part of the algorithm can be seen in Figure 4. Let $\pi'_{(q)}$ be the current proof π' after the q^{th} time we added a path to π' in line 28. For each q we prove by induction that each possible path in $\pi'_{(q)}$ has at most length $|\pi|$. For $q = 0$ this is trivial since $\pi'_{(0)} = \pi$. Let the statement be true for $\pi'_{(q)}$ and consider the case $\pi'_{(q+1)}$. Let $C_{j_1}, \ldots, C_{j_\ell}$ be a path in $\pi'_{(q+1)}$. If all of these clauses were already contained in $\pi'_{(q)}$, then the result follows immediately. Therefore let us suppose the path contains some clauses we have newly added, say that C_{j_p} is the leftmost new clause compared to $\pi'_{(q)}$. But then all clauses $C_{j_p}, C_{j_{p+1}}, \ldots, C_{j_\ell}$ are new clauses as well, since each new clause in inserted at the end of the proof. By the method we constructed the clauses $C_{j_p}, C_{j_{p+1}}, \ldots, C_{j_\ell}$ in line 28, we conclude that these clauses are some of the $C_{a'_2}, \ldots, C_{a'_{k+1}}$, say $C_{a'_v}, \ldots, C_{a'_w}$. But then we can find another path $C_{j_1}, \ldots, C_{j_{p-1}}, C_{a_v}, \ldots, C_{a_w}$ (we have to set $C_{a_w} := C_c$ if $w = k+1$ and we have to insert C_{a_1} after $C_{j_{p-1}}$ if $C_{j_{p-1}} = C_d$), which has the same (or even a greater) length as the original path and is completely contained in $\pi'_{(q)}$. Hence, the original path has length at most $|\pi|$.

All in all, for each $(d, e, c) \in M_{XUT}$ we will add a path of length at most $|\pi|$. Each c can occur only once in the triples in M_{XUT}. After we added the path corresponding to (d, e, c), we can ignore potential future occurrences of (d, e, c) if this particular triple is detected more than once in the algorithm.

We want to show next that in each outer while-loop we will only add at most $|\pi|^2$ new clauses (resp. at most $|\pi|$ paths) to π'. The number of added paths in the i^{th} loop is determined by $|M^{(i)}_{XUT}|$. For $i = 1$ this is obvious, as all c in $(d, e, c) \in M^{(1)}_{XUT}$ are indices from original formulas and each c appears only once. Let us now assume that $M^{(i')}_{XUT} \leq |\pi|$ for all $i' < i$. Then we have to show that $M^{(i)}_{XUT} \leq |\pi|$ holds as well (for $i > 1$). For each $(d, e, c) \in M^{(i)}_{XUT}$ it holds that c is either some a'_{k+1} from the loop before, or an original index (because d and e from the upper loop can only be original indices, the only newly added index that can be

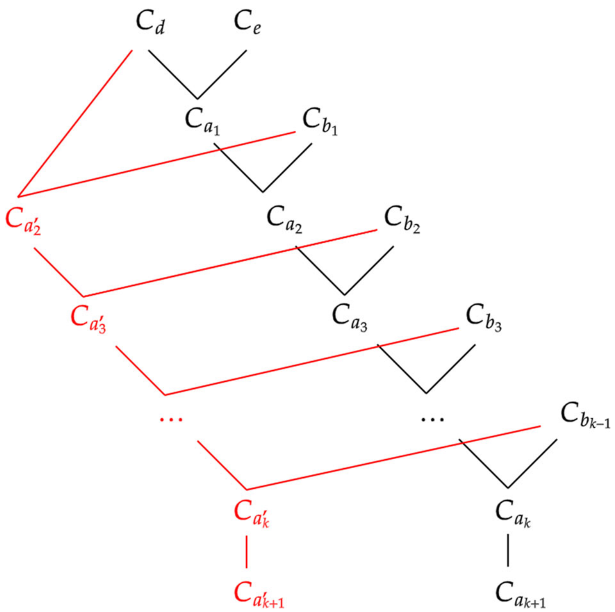

Fig. 4 Visualization of lines 27 and 28 in Algorithm 1. Newly added clauses and resolutions are coloured in red

contained in a triple is the last index a'_{k+1} of some added path from the loop before), from which we get $|M^{(i)}_{XUT}| \leq |\pi| + |M^{(i-1)}_{XUT}|$. However, if the c from some $(d, e, c) \in M^{(i)}_{XUT}$ was original, it must hold $(d, e, c) \notin M^{(i-1)}_{XUT}$. In fact, c cannot be included in any triple (d', e', c) from any $M^{(i')}_{XUT}$ for $i' < i$. Therefore, we can restrict the above inequality even more: $|M^{(i)}_{XUT}| \leq (|\pi| - |M^{(i-1)}_{XUT}|) + |M^{(i-1)}_{XUT}| = |\pi|$.

We conclude that in each outer while-loop we will add at most $|\pi|^2$ new clauses to π'. Since we will repeat the outer loop at most $|\pi|^2$ times, the new proof π' will at the end consist of at most $\mathcal{O}(|\pi|^4)$ clauses. □

Claim 3 π' is quasi level-ordered.

Proof Let us now briefly explain why the obtained proof is quasi level-ordered. We just have to argue how one can find the clauses C_τ for each $\tau \in \langle X \rangle$. For this, we simply backtrace all steps of the algorithm. We start by defining the empty clause as the currently considered clause and check whether its derivation is already level-ordered. If this is the case, then we can set $C_\tau = (\bot)$ for all τ. Otherwise, we look at the last X-resolution step in this derivation and find two clauses that got resolved over X. Depending on the choice of τ, we choose that clause which contains the X-literal that got negated by τ (as we would do in the level-ordered setting). At most one of these clauses is an XUT-clause (by Lemma 1), the other one is always an X-clause. If the clause we have chosen is the X-clause, then this X-clause becomes the new currently observed clause. If it is the XUT-clause, then we go into the path that was newly added during the algorithm and its last clause becomes the next observed clause (which, by construction, consists of the same X-literals as the original XUT-clause, but lacks all T- and U-literals). That also means that all observed clauses are X-clauses.

This search ends as soon as we get to a clause that has a level-ordered derivation, which will happen when the considered clause is an axiom (in the worst case). Because we always chose the right X-literal, this clause will be falsified by τ and can therefore serve as C_τ.

In more detail, we prove that the clauses we have added to L are exactly the clauses C_τ from the definition of quasi level-ordered proofs. Let us fix an assignment $\tau \in \langle X \rangle$. Starting from $C_m = (\bot)$, for each X-clause C_c we check if the subproof π'_{C_c} is level ordered. If it is not, we can find clauses $C_d, C_e \in \pi'_{C_c}$ as described in the algorithm that are resolved over an X-literal x. We pick the clause which contains x if $\tau(x) = 0$ and the other clause otherwise. If we pick an XUT-clause, say C_d, then we have to jump to the corresponding X-clause $C_{a'_{k+1}}$ which we have added when we chose $(d, e, c) \in M_{XUT}$ in the second inner while-loop. Note that $C_{a'_{k+1}}$ is a subclause of $C_c \vee x$ (resp. $C_c \vee \bar{x}$) since we only omitted the resolution with C_e over x. We continue by checking the subproof of C_d (resp. C_e or $C_{a'_{k+1}}$).

At the end, when the X-clause C_c has finally a level-ordered subproof π'_{C_c}, we will stop there and we set $C_\tau := C_c$ since we have $\tau(x) = 0$ for each $x \in C_c$. Therefore C_τ is falsified by τ. □

Algorithm 1 can be easily modified to also transform long-distance Q-resolution refutations by adding more case distinctions to line 16. However, this might lead to an exponential blow up.

We give an example of a formula with a refutation which we transform into a quasi level-ordered refutation.

Example 1 Let Ψ be the QCNF with prefix $\exists X \forall U \exists T$ with $X = \{x, y\}$, $U = \{u\}$, $T = \{s, t\}$ and the matrix

$$(u \vee \bar{s}) \wedge (x \vee u \vee s) \wedge (\bar{u} \vee \bar{s}) \wedge (y \vee u \vee s) \wedge (\bar{x} \vee u \vee \bar{s}) \wedge (x \vee \bar{y}) \\ \wedge (y \vee u \vee t) \wedge (\bar{s} \vee \bar{t}).$$

Further, let π be the Q-resolution refutation of Ψ as represented in Figure 5. We want to transform this proof π to a quasi level-ordered proof π' by carrying out the instructions as described in the algorithm. Note that for the sake of simplicity this proof is exceptionally not necessarily a QCDCL proof since finding a QCDCL proof that is representative enough to serve as an example is not a trivial thing to do. However, π fulfils at least the properties we need in order to get polynomially transformed, namely we never resolve two XUT-clauses over X. Also, π is most likely not the shortest possible refutation of Ψ, as one can see that the clause $C_6 = y \vee u \vee s$ is derived although $C_1 = y \vee u \vee s$ is an axiom clause.

First, we have $M_X = \{16\}$ and $M_{XUT} = \emptyset$. The proof of C_{16}, which is just π itself, is obviously not level ordered. The last transition from an X-resolution to a T-resolution is at $C_{11} \overset{y}{\bowtie} C_{12} = C_{13}$ to $C_{13} \overset{s}{\bowtie} C_{14} = C_{15}$. Since the last step in π was no X-resolution, we have to add the triple $(12, 11, 16)$ to M_{XUT} (note that the first number of the triple has to be the index of the XUT-clause). Further, we add 11 to M_X and delete 16 from it. The subproof $\pi_{C_{11}}$ of C_{11} is not level-ordered either. The last X- to T-transition in $\pi_{C_{11}}$ is $C_4 \overset{x}{\bowtie} C_5 = C_6$ to $C_6 \overset{s}{\bowtie} C_7 = C_8$. Because the last step in $\pi_{C_{11}}$ was a reduction and no X-resolution, we have to add $(5, 4, 11)$ to M_{XUT} and replace 11 with 4 in M_X. Now, the subproof π_{C_4} is level-ordered, so we can add 4 to L and delete it from M_X. Because M_X is now empty, we can continue by adding new clauses to π.

First, we add the clauses $C_{17} = C_{12} \overset{s}{\bowtie} C_{14}$ and $C_{18} = \text{red}(C_{17})$. This new path, which can be seen in Figure 6, corresponds to the triple $(12, 11, 16)$, that can now be deleted from M_{XUT}. After this we have to add 18 to M_X and continue with the next available triple from

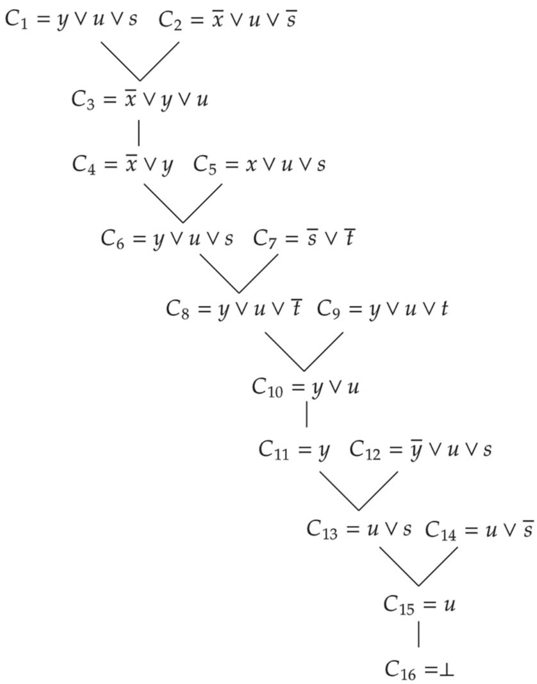

Fig. 5 Q-resolution refutation of Ψ

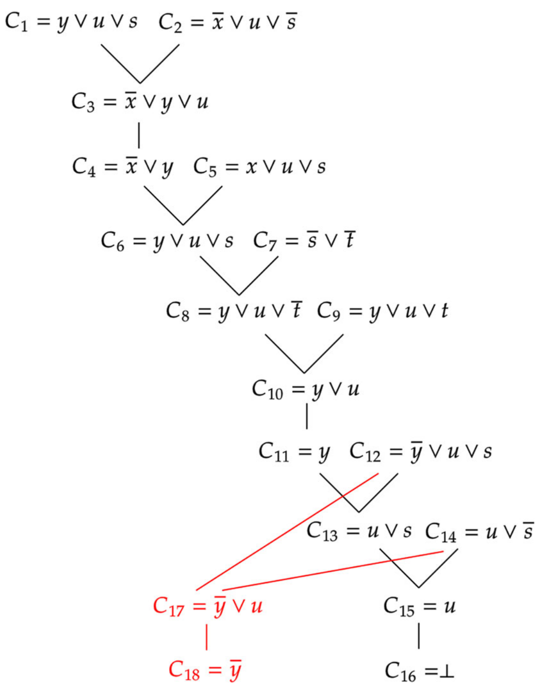

Fig. 6 Adding the new path of clauses corresponding to $(12, 11, 16) \in M_{XUT}$

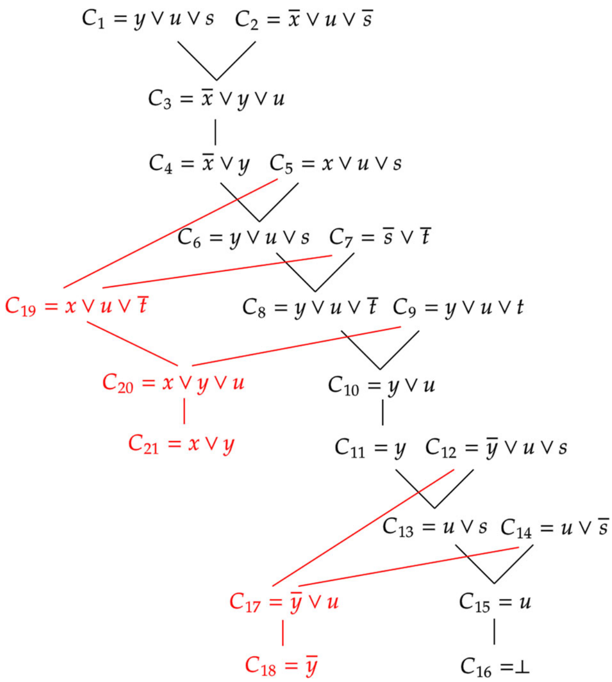

Fig. 7 Adding the new path of clauses corresponding to $(5, 4, 11) \in M_{XUT}$. This new proof π' is now quasi level-ordered

M_{XUT}, which is $(5, 4, 11)$. We add the clauses $C_{19} = C_5 \overset{s}{\bowtie} C_7$, $C_{20} = C_9 \overset{t}{\bowtie} C_{19}$ and $C_{21} = \text{red}(C_{20})$ to π, delete $(5, 4, 11)$ from M_{XUT} and add 21 to M_X. After that, M_{XUT} is empty and $M_X = \{18, 21\}$.

In the next iteration, we have to consider the subproofs $\pi_{C_{18}}$ and $\pi_{C_{21}}$, which are luckily both level-ordered. That means we can immediately delete both 18 and 21 from M_X and add them to L. Both M_X and M_{XUT} are now empty and hence our algorithm terminates. Our new proof π' which is represented in Figure 7 is now quasi level-ordered. The clauses whose indexes are contained in L are exactly the clauses C_τ we need for a quasi level-ordered proof. More precisely, $L = \{4, 18, 21\}$ with $C_{x\mapsto 1, y\mapsto 1} = C_{x\mapsto 0, y\mapsto 1} = C_{18} = (\bar{y})$, $C_{x\mapsto 1, y\mapsto 0} = C_4 = \bar{x} \vee y$ and $C_{x\mapsto 0, y\mapsto 0} = C_{21} = x \vee y$.

5 A Lower Bound Technique via Gauge

Now that we have proven that QCDCL is simulated by quasi level-ordered proofs, we continue by introducing a measure for Σ_3^b QCNFs that will provide an exponential lower bound for quasi level-ordered refutations of these formulas.

Definition 6 For a Σ_3^b QCNF Φ with prefix $\exists X \forall U \exists T$ let W_Φ be the set of all Q-resolution derivations π from Φ of some X-clause such that π only contains T-resolution and reduction steps. We define the *gauge* of Φ as

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

Intuitively, gauge(Φ) is the minimal number of X-literals that are necessarily piled up in a level-ordered Q-resolution derivation in which we want to get rid of all T-literals (hence we consider proofs of X-clauses).

Before showing how gauge lower bounds imply proof size lower bounds let us consider an example for which we recall the $\mathtt{CR}_n$ formulas from [22].

Definition 7 ([22]) The QCNF $\mathtt{CR}_n$ consists of the quantifier prefix

$$\exists x_{(1,1)}, \ldots, x_{(1,n)}, x_{(2,1)}, \ldots, x_{(2,n)}, \ldots, x_{(n,1)}, \ldots, x_{(n,n)} \forall u \exists s_1, \ldots, s_n, t_1, \ldots, t_n$$

and matrix clauses $(x_{(i,j)} \vee u \vee s_i)$, $(\bar{x}_{(i,j)} \vee \bar{u} \vee t_j)$ for $i, j \in [n]$ as well as $\bigvee_{i \in [n]} \bar{s}_i$ and $\bigvee_{i \in [n]} \bar{t}_i$.

The $\mathtt{CR}_n$ formulas describe a 'completion' game on an $(n \times n)$-matrix (cf. [22]): The universal player has to set u to false iff for all i there exists a j such that $x_{(i,j)}$ is set to false. Otherwise, there exists an i such that for each j, the literal $x_{(i,j)}$ is set to true, hence the universal player has to set u to true.

It is readily checked that the $\mathtt{CR}_n$ formulas fulfil the XT-property. We can now compute their gauge. Note that according to our convention, the T-variables comprise of all variables $s_1, \ldots, s_n, t_1, \ldots, t_n$.

Lemma 3 *We have gauge*$(\mathtt{CR}_n) = n$.

Proof Since there are no X-clauses as axioms, we necessarily need to resolve over T somehow. For this we need T-literals of negative polarity, hence each $\pi \in W_{\mathtt{CR}_n}$ contains $\bigvee_{i \in [n]} \bar{s}_i$ or $\bigvee_{i \in [n]} \bar{t}_i$. In each $\pi \in W_{\mathtt{CR}_n}$ every T-literal has to be resolved away. For this reason we need the corresponding clauses $x_{(i,j)} \vee u \vee s_i$ or $\bar{x}_{(i,j)} \vee \bar{u} \vee t_j$. Because we cannot resolve over X in $\pi \in W_{\mathtt{CR}_n}$, there are at least n X-literals that are piled up and therefore gauge$(\mathtt{CR}_n) = n$. □

Towards our lower bound technique we now estimate the size of derivations of X-clauses in terms of gauge.

Lemma 4 *Let Φ be a Σ_3^b QCNF. Let π be a level-ordered* Q-resolution *proof from Φ of a non-tautological X-clause D with $|D| = c$. Then $|\pi| \geq 2^{gauge(\Phi)-c}$.*

Proof Let $V := X \backslash \text{var}(D)$. For each assignment $\tau \in \langle V \rangle$ we will find a path P_τ in π by going backwards starting from D. For each resolution step over some $x \in V$ we choose the path whose literals are negated by τ, hence we choose the clause that contains x if $\tau(x) = 0$ and the other clause otherwise. If there are resolution steps over variables from var(D), then we will always choose the literal from D. If we reach a reduction step, we will just expand the path by this one parental clause. If we detect a resolution step over a T-literal, we stop there.

Let C_τ be the clause at which we stop. Clearly, the subproof π_{C_τ} of C_τ is one of the derivations in W_Φ, hence $|C_\tau| \geq$ gauge(Φ). Then C_τ has to be a non-tautological X-clause with at least gauge(Φ) different X-literals. Then C_τ contains at least gauge$(\Phi) - c$ different X-literals whose variables are in V. These literals are negated by the assignment τ.

Now let a be the number of these clauses C_τ by summing over all τ. Since for each C_τ there are at most $|X| -$ gauge(Φ) variables that are not contained as some literal in the clause, there are at most $2^{|X|-\text{gauge}(\Phi)}$ paths that can lead to each C_τ. Multiplying with the number of C_τ gives us at least the number of assignments $\tau \in \langle V \rangle$, hence

$$2^{|X|-\text{gauge}(\Phi)} \cdot a \geq 2^{|X|-c}$$

$$\Leftrightarrow a \geq 2^{|X|-c}/2^{|X|-\text{gauge}(\Phi)} = 2^{\text{gauge}(\Phi)-c}.$$

Since each C_τ is a clause from π, we get $|\pi| \geq a \geq 2^{\text{gauge}(\Phi)-c}$. □

Note that the bound from Lemma 4 is an exact lower bound (no asymptotics involved). We will now use Lemma 4 to get a lower bound for quasi level-ordered Q-resolution refutations. We will do this with a similar counting argument as in Lemma 4 by counting the number of clauses C_τ in quasi level-ordered proofs.

Proposition 5 *Each quasi level-ordered* Q-resolution *refutation of a* Σ_3^b *QCNF* Φ *has size* $2^{\Omega(gauge(\Phi))}$.

Proof Let π be the shortest quasi level-ordered refutation of Φ. By the definition of quasi level-ordered proofs we can find clauses C_τ for each $\tau \in \langle X \rangle$.

Let $h := \min_{\tau \in \langle X \rangle} |C_\tau|$. By Lemma 4 we get $|\pi| \geq 2^{\text{gauge}(\Phi)-h}$, hence $h \geq \text{gauge}(\Phi) - \log|\pi|$. Each clause C_τ can have at most $2^{|X|-h}$ assignments $\alpha \in \langle X \rangle$ such that $C_\alpha = C_\tau$. Let $a := |\{C_\tau : \tau \in \langle X \rangle\}|$, then $a \cdot 2^{|X|-h} \geq 2^{|X|}$ and thus

$$|\pi| \geq a \geq 2^h \geq 2^{\text{gauge}(\Phi)-\log|\pi|} = \frac{2^{\text{gauge}(\Phi)}}{|\pi|}.$$

We conclude that $|\pi|^2 \in 2^{\Omega(\text{gauge}(\Phi))}$. □

We combine Theorem 2 and Proposition 5 above and obtain a lower bound for QCDCL on formulas with the XT-property.

Theorem 6 *Each* QCDCL *refutation of a* Σ_3^b *QCNF* Φ *that fulfils the XT-property has size* $2^{\Omega(gauge(\Phi))}$.

6 Applications of the Lower Bound Technique

We now apply our new lower bound technique via gauge to show exponential lower bounds for QCDCL proof size (and thereby for QCDCL running time) for a number of QBF families. First, by combining Lemma 3 with Theorem 6 we obtain hardness for the $\texttt{CR}_n$ formulas from [22].

Corollary 7 *The formulas* $\texttt{CR}_n$ *require exponential-size proofs in* QCDCL.

With this result we gain an improved separation between Q-resolution and QCDCL. It was already shown in [7] that Q-resolution and QCDCL are incomparable. This involves constructing QBFs that are easy for QCDCL, but hard for Q-resolution, and vice versa. One direction is shown via the `QParity` formulas (Definition 9 below), which are hard for Q-resolution [14], but easy in QCDCL [7]. For the other direction, [7] used the Trapdoor [7] and Lonsing formulas [28], both of which are easy for Q-resolution, but hard for QCDCL. However, both QBF families incorporate the propositional pigeonhole principle (PHP) and the hardness of these formulas for QCDCL rests entirely on the hardness of PHP for propositional resolution [20]. This is somewhat unsatisfactory, as the hardness results do not refer to quantification and in particular do not hold in the presence of NP oracles (cf. [11, 25] for a detailed formal account on how to equip QBF proofs with NP oracles or equivalently QBF solving with SAT calls).

Our improved separation is shown in Corollary 7 above, as these formulas are hard in QCDCL, but easy in Q-resolution [22]. Unlike the separations from [7], this hardness result does not make any reference to propositional hardness but also holds under NP oracles in the framework of [11] (at least if each axiom of our formula contains at least one T-literal, otherwise the formula might be trivial to refute with just X-resolutions and without reductions). This is due to the fact that one could simply replace $|\pi|$ with the number of reduction steps in π in Lemma 4 and Proposition 5 (note that we can assume that there is a reduction step before each C_τ) and hardness in the presence of NP oracles basically corresponds to counting reduction steps (cf. [11]).

We also note that Janota [21] already proved hardness of the QBFs $\texttt{CR}_n$ for QCDCL with UIP learning. Corollary 7 improves on that result as well as our hardness result holds for arbitrary learning schemes in QCDCL.

As our second example we introduce the following formulas.

Definition 8 Let $\texttt{ENarrow}_n := \exists x_1, \dots, x_{n+1} \forall u_1, \dots, u_{n+1} \exists t_1, \dots, t_n \cdot \psi_n$ with the matrix ψ_n containing the clauses:

$$
\begin{aligned}
& x_1 \vee u_1 \vee t_1,\ \bar{x}_1 \vee \bar{u}_1 \vee t_1, \\
& x_i \vee u_i \vee \bar{t}_{i-1} \vee t_i,\ \bar{x}_i \vee \bar{u}_i \vee \bar{t}_{i-1} \vee t_i, \qquad for \quad i = 2, \dots, n \\
& x_{n+1} \vee u_{n+1} \vee \bar{t}_n,\ \bar{x}_{n+1} \vee \bar{u}_{n+1} \vee \bar{t}_n.
\end{aligned}
$$

It is easy to see that $\texttt{ENarrow}_n$ fulfils the XT-property. $\texttt{ENarrow}_n$ can be interpreted as a variant of $\texttt{Eqality}_n$ (cf. [5] or the corresponding definition later), in which the universal player wins by setting all u_i equal to x_i. The only difference is that instead of having a long T-clause, the T-literals are now connected chain-like by only using XUT-clauses. Next we will show an exponential lower bound for $\texttt{ENarrow}_n$ in QCDCL.

Lemma 8 *We have gauge*$(\texttt{ENarrow}_n) = n + 1$.

Proof Let $\pi \in W_{\texttt{ENarrow}_n}$. Define the sets of clauses

$$
\begin{aligned}
Z_1 &:= \{x_1 \vee u_1 \vee t_1,\ \bar{x}_1 \vee \bar{u}_1 \vee t_1\} \\
Z_i &:= \{x_i \vee u_i \vee \bar{t}_{i-1} \vee t_i,\ \bar{x}_i \vee \bar{u}_i \vee \bar{t}_{i-1} \vee t_i\} \qquad \text{for} \quad i = 2, \dots, n \\
Z_{n+1} &:= \{x_{n+1} \vee u_{n+1} \vee \bar{t}_n,\ \bar{x}_{n+1} \vee \bar{u}_{n+1} \vee \bar{t}_n\}.
\end{aligned}
$$

Let C be an axiom clause in π. Then C has to be contained in some set Z_i as above.

<u>Case 1:</u> $C \in Z_1$.

Then we have to get rid of $t_1 \in C$, hence we need a clause from Z_2. But then we have to get rid of t_2 and so on:

$Z_1 \rightsquigarrow Z_2 \rightsquigarrow \dots \rightsquigarrow Z_n \rightsquigarrow Z_{n+1}$.

We conclude that π has to contain at least one clause from each Z_j, $j \in [n+1]$. Therefore we have to pile up $n + 1$ X-literals.

<u>Case 2:</u> $C \in Z_i$ for some $i \in \{2, \dots, n\}$.

Then we have to get rid of $\bar{t}_{i-1}$ and $t_i \in C$, hence we need a clause from Z_{i-1} and Z_{i+1}. After this we have to resolve over $\bar{t}_{i-2}$ and t_{i+1} and so on, leading to a chain of resolutions

$Z_1 \leftsquigarrow \dots \leftsquigarrow Z_{i-1} \leftsquigarrow Z_i \rightsquigarrow Z_{i+1} \rightsquigarrow \dots \rightsquigarrow Z_{n+1}$.

Again, we conclude that π has to contain at least one clause from each Z_j, $j \in [n + 1]$. Therefore we have to pile up $n + 1$ X-literals.

<u>Case 3:</u> $C \in Z_{n+1}$.

This works similarly to Case 1, except that we start at Z_{n+1} and go backwards: $Z_1 \leftsquigarrow Z_2 \leftsquigarrow \dots \leftsquigarrow Z_n \leftsquigarrow Z_{n+1}$. ☐

Corollary 9 *The QBFs mathtt$ENarrow_n$ require exponential-size proofs in* QCDCL.

The gauge of a formula is obviously some width measure and it seems natural to wonder how it relates to the notion of the existential proof width[1] of long-distance Q-resolution refutations of a formula as studied in [6, 9, 17]. However, it turns out that these two measures are not directly related. On the one hand, it is easy to see that $\texttt{ENarrow}_n$ has long-distance Q-resolution refutations of constant existential clause width. Hence these formulas have small (constant) existential proof width, but linear gauge.

On the other hand, there are also formulas with constant gauge and linear proof width. For this we revisit the parity formula from [14].

Definition 9 [14] $\texttt{QParity}_n$ consists of the prefix $\exists x_1 \ldots x_n \forall u \exists t_2 \ldots t_n$ and the matrix

$$x_1 \vee x_2 \vee \bar{t}_2,\ x_1 \vee \bar{x}_2 \vee t_2,\ \bar{x}_1 \vee x_2 \vee t_2,\ \bar{x}_1 \vee \bar{x}_2 \vee \bar{t}_2,$$
$$x_i \vee t_{i-1} \vee \bar{t}_i,\ x_i \vee \bar{t}_{i-1} \vee t_i,\ \bar{x}_i \vee t_{i-1} \vee t_i,\ \bar{x}_i \vee \bar{t}_{i-1} \vee \bar{t}_i \quad \text{for} i \in \{3, \ldots, n\}$$
$$u \vee t_n,\ \bar{u} \vee \bar{t}_n.$$

The formulas $\texttt{QParity}_n$ are built on the parity function. In more detail, the universal player only wins by setting u equal to $x_1 \oplus \ldots \oplus x_n$. Each t_i encodes the partial sum $x_1 \oplus \ldots \oplus x_i$.

It was shown in [6, 9] that $\texttt{QParity}_n$ requires linear proof width. Here we modify this formula such that proof width remains unaffected, but gauge is small. Let $\texttt{mQParity}_n$ be the modified variant of this formula that consists of the prefix $\exists x_1, \ldots, x_n, y \forall u \exists t_2, \ldots, t_n$ and the matrix $(\bar{y}) \wedge \bigwedge_{C \in \texttt{QParity}_n} (y \vee C)$. Obviously, because of the unit clause $(\bar{y})$, we have gauge$(\texttt{mQParity}_n) = 1$, but still linear proof width (since we can simply consider the proof width of the proof restricted to the assignment $y \mapsto 0$, which is exactly a refutation of $\texttt{QParity}_n$).

We will see later that we can also use the $\texttt{QParity}_n$ formulas to show that large gauge alone is not sufficient to guarantee QCDCL hardness, but some further assumption such as the XT-condition is needed.

We continue with the equality formula from [5] as a further example of hard formulas for QCDCL. In [7] QCDCL hardness of $\texttt{Equality}_n$ was already proven by lifting Q-resolution hardness of these formulas to QCDCL. However, with our new lower bound technique it is possible to prove QCDCL hardness directly without importing Q-resolution lower bounds.

Definition 10 [5] The formula $\texttt{Equality}_n$ is defined as the QCNF

$$\exists x_1 \ldots x_n \forall u_1 \ldots u_n \exists t_1 \ldots t_n \cdot (\bar{t}_1 \vee \ldots \vee \bar{t}_n) \wedge \bigwedge_{i=1}^{n} ((\bar{x}_i \vee \bar{u}_i \vee t_i) \wedge (x_i \vee u_i \vee t_i)).$$

Proposition 10 *We have gauge*$(\texttt{Equality}_n) = n$. *Consequently the formulas are exponentially hard for* QCDCL.

Proof Let $\pi \in W_{\texttt{Equality}_n}$. Since none of the axioms are X-clauses, we have to resolve over T somehow. For this we need the clause $\bar{t}_1 \vee \ldots \vee \bar{t}_n$. But that means we have to resolve over each t_i at least once in π, and therefore we will pile up all n X-variables.

□

[1] The existential width of a clause is defined as the number of existential literals in this clause. The existential proof width is defined as the maximal existential width over all clauses in this proof.

Our next example illustrates that some further condition (such as the XT-property) is indeed required for our lower bound method to work. For this we will take another look at the parity formula $\texttt{QParity}_n$. These formulas are known to be hard for Q-resolution [14], but easy for QCDCL [7]. Nevertheless, we show that $\texttt{QParity}_n$ has large gauge. Hence this measure alone is not sufficient to imply QCDCL hardness.

Proposition 11 *We have gauge*$(\texttt{QParity}_n) = n$.

Proof We define the following sets of clauses:

$$
\begin{aligned}
Z_1 &:= \{x_1 \vee x_2 \vee \bar{t}_2,\ x_1 \vee \bar{x}_2 \vee t_2,\ \bar{x}_1 \vee x_2 \vee t_2,\ \bar{x}_1 \vee \bar{x}_2 \vee \bar{t}_2\} \\
Z_i &:= \{x_{i+1} \vee t_i \vee \bar{t}_{i+1},\ x_{i+1} \vee \bar{t}_i \vee t_{i+1},\ \bar{x}_{i+1} \vee t_i \vee t_{i+1},\ \bar{x}_{i+1} \vee \bar{t}_i \vee \bar{t}_{i+1}\} \\
Z_n &:= \{u \vee t_n,\ \bar{u} \vee \bar{t}_n\}
\end{aligned}
$$

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

We show that each $\pi \in W_{\texttt{QParity}_n}$ needs at least one clause from each Z_j as an axiom, hence $\pi \cap Z_j \neq \emptyset$ for every $j \in [n]$.

Assume that there is a proof $\pi \in W_{\texttt{QParity}_n}$ of an X-clause C and $j \in [n]$ with $\pi \cap Z_j = \emptyset$. Let S be the set of all symmetries σ on $\texttt{QParity}_n$ such that $\sigma(x_k) \in \{x_k, \bar{x}_k\}$ for each $k \in [n]$ and $\sigma(t_k)$ is chosen such that $\sigma(\texttt{QParity}_n) \subseteq \texttt{QParity}_n$ (one has to make sure that $\sigma(t_k) = \sigma(x_1) \oplus \ldots \oplus \sigma(x_k)$).

Then for each such $\sigma \in S$ we can derive $\sigma(C)$ via $\sigma(\pi)$ and still have $\sigma(\pi) \cap Z_j = \emptyset$. But then we could easily construct a refutation by just using $\{\sigma(C) : \sigma \in S\}$. Then $\texttt{QParity}_n$ without the clauses from Z_j would still be a false QCNF. However, this is not possible since we can construct a winning strategy A for $\texttt{QParity}_n \backslash Z_j$:

$$
\begin{aligned}
A(x_k) &:= 0 \text{ for all} k \in [n] \\
A(t_\ell) &:= 0 \text{ for all} \ell \in \{2, \ldots, j\} \\
A(t_{\ell'}) &:= 1 \oplus u \text{ for all} \ell' \in \{j+1, \ldots, n\}
\end{aligned}
$$

Therefore our assumption is false and we get $\pi \cap Z_j \neq \emptyset$. Using one clause from each Z_j results in piling up all variables $x_1, \ldots, x_n$ in some polarity, hence gauge$(\texttt{QParity}_n) = n$. □

The next example is a formula that follows the same approach as $\texttt{Equality}_n$ from [5], where the universal player had to fulfil the task of assigning the U-variables in the same way as the existential X-variables. However, we can replace this task with another, more complex one. In our case, the universal player has to detect palindromes in the word that was input by the existential player.

Example 2 Let $\texttt{QPalin}_n$ be the QCNF with prefix

$$\exists X \forall U \exists T$$

with

$$
\begin{aligned}
X &= \left\{x_j : j \in \{1, \ldots, n\}\right\} \\
U &= \left\{u_{k,i}, v_k : k \in \{0, \ldots n-1\}, i \in \left\{1, \ldots, \left\lfloor \frac{n}{2} \right\rfloor\right\}\right\} \\
T &= \left\{t_{k,i}, s_k : k \in \{0, \ldots n-1\}, i \in \left\{1, \ldots, \left\lfloor \frac{n}{2} \right\rfloor\right\}\right\}
\end{aligned}
$$

where the indices from the X-variables are interpreted as integers modulo n. Let the matrix of the formula consist of the following clauses:

$$\begin{aligned}
&x_{i+k} \vee x_{n-i+1+k} \vee \bar{u}_{k,i} \vee t_{k,i},\ \bar{x}_{i+k} \vee \bar{x}_{n-i+1+k} \vee \bar{u}_{k,i} \vee t_{k,i}\\
&x_{i+k} \vee \bar{x}_{n-i+1+k} \vee u_{k,i} \vee \bar{t}_{k,i},\ \bar{x}_{i+k} \vee x_{n-i+1+k} \vee u_{k,i} \vee \bar{t}_{k,i}\\
&\bar{v}_k \vee \bar{t}_{k,1} \vee \ldots \vee \bar{t}_{k,\lfloor \frac{n}{2} \rfloor} \vee s_k\\
&v_k \vee t_{k,i} \vee s_k,\ \bar{s}_0 \vee \ldots \vee \bar{s}_{n-1}
\end{aligned}$$

for $k \in \{0, \ldots n-1\}$, $i \in \left\{1, \ldots, \left\lfloor \frac{n}{2} \right\rfloor\right\}$.

Intuitively, the X-variables represent the word in which palindromes have to be detected. We not only check the word $x_1 \ldots x_n$ itself, but also all shifted variants $x_{1+k} \ldots x_{n+k}$. The $u_{k,i}$ encode whether or not, in the word $x_{1+k} \ldots x_{n+k}$, the i^{th} letter from the left and the i^{th} letter from the right are the same. If the universal player assigns the $u_{k,i}$ correctly, the existential player is forced to set $t_{k,i}$ equal to $u_{k,i}$ in their turn at the end. If and only if the word $x_{1+k} \ldots x_{n+k}$ was a palindrome, the universal player sets v_k to true. Hence, if $x_{1+k} \ldots x_{n+k}$ was a palindrome, v_k as well as $t_{k,i}$ for each i is set to true, hence the existential player has to set s_k to true. Otherwise, if it was not a palindrome, then there was an i such that x_{i+k} and $x_{n-i+1+k}$ were assigned differently, therefore $t_{k,i}$ and v_k are both false and s_k must be set to true, as well. However, setting all s_k to true falsifies the matrix.

In a nutshell: Let τ be a total assignment of X. There is a winning strategy for the universal player by setting $u_{k,i}$ to 1 if and only if $\tau(x_{i+k}) = \tau(x_{n-i+k})$ and v_k to 1 if and only if the word $\tau(x_{1+k}) \ldots \tau(x_{n+k})$ is a palindrome. Then the existential player has to set each s_k to 1, negating the last clause in the matrix.

Remark 1 $\texttt{QPalin}_n$ fulfils the XT-property.

Lemma 12 *We have gauge*$(\texttt{QPalin}_n) \in \Omega(\sqrt{n})$.

Proof Let $\pi \in W_{\texttt{QPalin}_n}$. Since we do not have any X-clauses as axioms, we need to resolve over T-variables at least once. We partition the matrix of $\texttt{QPalin}_n$ into the following sets:

$$\begin{aligned}
Z^+_{k,i} &:= \{x_{i+k} \vee x_{n-i+1+k} \vee \bar{u}_{k,i} \vee t_{k,i}, \bar{x}_{i+k} \vee \bar{x}_{n-i+1+k} \vee \bar{u}_{k,i} \vee t_{k,i}\}\\
Z^-_{k,i} &:= \{x_{i+k} \vee \bar{x}_{n-i+1+k} \vee u_{k,i} \vee \bar{t}_{k,i}, \bar{x}_{i+k} \vee x_{n-i+1+k} \vee u_{k,i} \vee \bar{t}_{k,i}\}\\
P_k &:= \{\bar{v}_k \vee \bar{t}_{k,1} \vee \ldots \vee \bar{t}_{k,\lfloor \frac{n}{2} \rfloor} \vee s_k\}\\
N_{k,i} &:= \{v_k \vee t_{k,i} \vee s_k\}\\
S &:= \{\bar{s}_0 \vee \ldots \vee \bar{s}_{n-1}\}
\end{aligned}$$

Let $C \in \pi$ be an axiom. We claim that $S \subseteq \pi$, for which we will distinguish four cases.

Case 1: $C \in Z^+_{k,i}$ for some $k \in \{0, \ldots, n-1\}$ and $i \in \left\{1, \ldots, \left\lfloor \frac{n}{2} \right\rfloor\right\}$.

Then we have to resolve away $t_{k,i}$, which can only be done with the clause in P_k since the clauses from $Z^-_{k,i}$ are blocked because of the $u_{k,i}$. But now we have introduced s_k which we can only resolve with the clause from S.

Case 2: $C \in Z^-_{k,i}$ for some $k \in \{0, \ldots, n-1\}$ and $i \in \left\{1, \ldots, \left\lfloor \frac{n}{2} \right\rfloor\right\}$.

To get rid of $\bar{t}_{k,i}$, we have to use the clause from $N_{k,i}$ since $Z^+_{k,i}$ is blocked as before. But then we have introduces s_k and we will need the clause from S.

Case 3: $C \in P_k$ or $C \in N_{k,i}$ for some $k \in \{0, \ldots, n-1\}$ and $i \in \left\{1, \ldots, \left\lfloor \frac{n}{2} \right\rfloor\right\}$.

Then we have $s_k \in C$ and we need to use the clause from S in order to resolve it away.

Case 4: $C \in S$.

This case is trivial.

We have shown that in each case we have $S \subseteq \pi$. That means we have to resolve over each s_k in π. For each $k \in \{0, \ldots, n-1\}$ we can choose if we want to use P_k or $N_{k,i}$ to get rid of the literal $\bar{s}_k$. If we choose P_k, then we have to resolve over each $t_{k,i}$ for $i \in \left\{1, \ldots, \left\lfloor \frac{n}{2} \right\rfloor\right\}$ by using the clauses from $Z^+_{k,i}$. However, if we choose $N_{k,i}$, it suffices to resolve over only one $t_{k,i}$ for some particular i. Hence, we only have to use one clause from $Z^-_{k,i}$ for only one i. In the worst case (that means in the case with the least resolutions over $t_{k,i}$), we will always pick $N_{k,i}$. More specific, for each $k \in \{0, \ldots, n-1\}$ there exists an $i_k \in \left\{1, \ldots, \left\lfloor \frac{n}{2} \right\rfloor\right\}$ such that π contains at least one clause from Z^+_{k,i_k} or Z^-_{k,i_k}. That means we will pile up at least the X-variables x_{i_k+k}, x_{n-i_k+1+k} for each $k \in \{0, \ldots, n-1\}$. If we can find a lower bound for the number of these X-variables, then this is also a lower bound for gauge($\texttt{QPalin}_n$).

First of all, it its obvious that for each k we have $x_{i_k+k} \neq x_{n-i_k+1+k}$ since $i_k + k \not\equiv n - i_k + 1 + k \pmod{n}$. Next, we define the following sets of variables:

$$X_k := \{x_{i_k+k}, x_{n-i_k+1+k}\}$$

for $k \in \{0, \ldots, n-1\}$. As we have argued above, we already know that the gauge of $\texttt{QPalin}_n$ is $\Omega(|\bigcup_k X_k|)$. Note that if a pair $\{x_j, x_\ell\}$ is equal to X_k, then their position in the word $x_{1+k} \ldots x_{n+k}$ is symmetric. If n is odd, then each pair $\{x_j, x_\ell\}$ can represent at most one X_k. For example, for $n = 5$ the pair $\{x_3, x_4\}$ can at most be X_3 since they are only symmetric in the word $x_4 x_5 x_1 x_2 x_3$. However, if n is even then each pair then each pair $\{x_j, x_\ell\}$ can represent up to two X_k. For example, for $n = 4$ the pair $\{x_1, x_4\}$ is symmetric in both $x_1 x_2 x_3 x_4$ and $x_3 x_4 x_1 x_2$.

We conclude that for odd n we have $|\{X_0, \ldots, X_{n-1}\}| = n$ and for even n we have $|\{X_0, \ldots, X_{n-1}\}| \geq \frac{n}{2}$. Now, with m different variables we could create at most $\mathcal{O}(m^2)$ different pairs X_k. Hence we need at least $\Omega(\sqrt{n})$ different variables to create $\mathcal{O}(n)$ different pairs X_k, and therefore $|\bigcup_k X_k| \in \Omega(\sqrt{n})$ and also gauge($\texttt{QPalin}_n$) $\in \Omega(\sqrt{n})$. ◻

Corollary 13 *The QCNF* $\texttt{QPalin}_n$ *needs* QCDCL *refutations of size* $2^{\Omega(\sqrt{n})}$.

7 Conclusion

We initiated the study of devising lower bound methods tailored to QCDCL. At the moment our techniques only apply to Σ_3^b-formulas. Though this is a quite relevant class of QBFs, also prominently represented in QBF benchmarks [27, 32], it would be very interesting to extend the method to QBFs of higher quantifier complexity.

In another direction, future research should explore further conditions (besides the XT-condition considered here) that allow to efficiently translate QCDCL into quasi level-ordered proofs and thus enable to show lower bounds via gauge.

Funding Open Access funding enabled and organized by Projekt DEAL.

References

1. Balabanov, V., Widl, M., Jiang, J.-H.R.: QBF resolution systems and their proof complexities. In: Proceedings of the Theory and Applications of Satisfiability Testing (SAT), pp. 154–169 (2014)
2. Balabanov, V., Jiang, J.-H.R.: Unified QBF certification and its applications. Form. Methods Syst. Des. **41**(1), 45–65 (2012)
3. Beame, P., Kautz, H.A., Sabharwal, A.: Towards understanding and harnessing the potential of clause learning. J. Artif. Intell. Res. (JAIR) **22**, 319–351 (2004)
4. Ben-Sasson, E., Wigderson, A.: Short proofs are narrow—resolution made simple. J. ACM **48**(2), 149–169 (2001)
5. Beyersdorff, O., Blinkhorn, J., Hinde, L.: Size, cost, and capacity: a semantic technique for hard random QBFs. Logical Methods in Computer Science **15**(1) (2019)
6. Beyersdorff, O., Blinkhorn, J., Mahajan, M.: Hardness characterisations and size-width lower bounds for QBF resolution. In: Proceedings of the ACM/IEEE Symposium on Logic in Computer Science (LICS), pp. 209–223 (2020)
7. Beyersdorff, O., Böhm, B.: Understanding the relative strength of QBF CDCL solvers and QBF resolution. In: Proceedings of the Innovations in Theoretical Computer Science (ITCS), pp. 12–11220 (2021)
8. Beyersdorff, O., Bonacina, I., Chew, L., Pich, J.: Frege systems for quantified Boolean logic. J. ACM **67**(2) (2020)
9. Beyersdorff, O., Chew, L., Mahajan, M., Shukla, A.: Are short proofs narrow? QBF resolution is not so simple. ACM Transactions on Computational Logic **19** (2018)
10. Beyersdorff, O., Chew, L., Mahajan, M., Shukla, A.: Feasible interpolation for QBF resolution calculi. Logical Methods in Computer Science **13** (2017)
11. Beyersdorff, O., Hinde, L., Pich, J.: Reasons for hardness in QBF proof systems. ACM Transactions on Computation Theory **12**(2) (2020)
12. Beyersdorff, O., Janota, M., Lonsing, F., Seidl, M.: Quantified Boolean formulas. In: Biere, A., Heule, M., van Maaren, H., Walsh, T. (eds.) Handbook of Satisfiability. Frontiers in Artificial Intelligence and Applications, pp. 1177–1221. IOS Press, Amsterdam (2021)
13. Beyersdorff, O.: Proof complexity of quantified Boolean logic—a survey. In: Benini, M., Beyersdorff, O., Rathjen, M., Schuster, P. (eds.) Mathematics for Computation (M4C), pp. 353–391. World Scientific, Singapore (2022)
14. Beyersdorff, O., Chew, L., Janota, M.: New resolution-based QBF calculi and their proof complexity. ACM Trans. Comput. Theory **11**(4), 26–12642 (2019)
15. Beyersdorff, O., Chew, L., Sreenivasaiah, K.: A game characterisation of tree-like Q-resolution size. J. Comput. Syst. Sci. **104**, 82–101 (2019)
16. Böhm, B., Beyersdorff, O.: Lower bounds for QCDCL via formula gauge. In: Li, C.-M., Manyà, F. (eds.) Theory and Applications of Satisfiability Testing—SAT 2021, pp. 47–63. Springer, Cham (2021)
17. Clymo, J., Beyersdorff, O.: Relating size and width in variants of Q-resolution. Inf. Process. Lett. **138**, 1–6 (2018)
18. Egly, U., Lonsing, F., Widl, M.: Long-distance resolution: proof generation and strategy extraction in search-based QBF solving. In: Proceedings of the Logic for Programming, Artificial Intelligence, and Reasoning (LPAR), pp. 291–308 (2013)
19. Giunchiglia, E., Narizzano, M., Tacchella, A.: Clause/term resolution and learning in the evaluation of quantified Boolean formulas. J. Artif. Intell. Res. **26**, 371–416 (2006)
20. Haken, A.: The intractability of resolution. Theor. Comput. Sci. **39**, 297–308 (1985)
21. Janota, M.: On Q-Resolution and CDCL QBF solving. In: Proceedings of the International Conference on Theory and Applications of Satisfiability Testing (SAT), pp. 402–418 (2016)
22. Janota, M., Marques-Silva, J.: Expansion-based QBF solving versus Q-resolution. Theor. Comput. Sci. **577**, 25–42 (2015)
23. Kleine Büning, H., Karpinski, M., Flögel, A.: Resolution for quantified Boolean formulas. Inf. Comput. **117**(1), 12–18 (1995)
24. Krajíček, J.: Proof Complexity. Encyclopedia of Mathematics and Its Applications, vol. 170. Cambridge University Press, Cambridge (2019)
25. Lonsing, F., Egly, U., Seidl, M.: Q-resolution with generalized axioms. In: Proceedings of the Theory and Applications of Satisfiability Testing (SAT), pp. 435–452 (2016)
26. Lonsing, F., Egly, U.: DepQBF 6.0: A search-based QBF solver beyond traditional QCDCL. In: Proceedings of the International Conference on Automated Deduction (CADE), pp. 371–384 (2017)
27. Lonsing, F., Egly, U.: Evaluating QBF solvers: quantifier alternations matter. In: Proceedings of the Principles and Practice of Constraint Programming (CP), pp. 276–294 (2018)

28. Lonsing, F.: Dependency schemes and search-based QBF solving: theory and practice. PhD thesis, Johannes Kepler University Linz (2012)
29. Marques Silva, J.P., Lynce, I., Malik, S.: Conflict-driven clause learning SAT solvers. In: Handbook of Satisfiability. IOS Press, Amsterdam (2009)
30. Peitl, T., Slivovsky, F., Szeider, S.: Dependency learning for QBF. J. Artif. Intell. Res. **65**, 180–208 (2019)
31. Pipatsrisawat, K., Darwiche, A.: On the power of clause-learning SAT solvers as resolution engines. Artif. Intell. **175**(2), 512–525 (2011)
32. Pulina, L., Seidl, M.: The 2016 and 2017 QBF solvers evaluations (QBFEVAL'16 and QBFEVAL'17). Artif. Intell. **274**, 224–248 (2019)
33. Shukla, A., Biere, A., Pulina, L., Seidl, M.: A survey on applications of quantified Boolean formulas. In: Proceedings of the IEEE International Conference on Tools with Artificial Intelligence (ICTAI), pp. 78–84 (2019)
34. Vinyals, M.: Hard examples for common variable decision heuristics. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI) (2020)
35. Zhang, L., Madigan, C.F., Moskewicz, M.W., Malik, S.: Efficient conflict driven learning in Boolean satisfiability solver. In: Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 279–285 (2001)
36. Zhang, L., Malik, S.: Conflict driven learning in a quantified Boolean satisfiability solver. In: Proceedings of the IEEE/ACM International Conference on Computer-aided Design (ICCAD), pp. 442–449 (2002)