Technical Report AC-TR-19-004

May 2019

Proof Complexity of Fragments of Long-Distance Q-Resolution

Tomáš Peitl, Friedrich Slivovsky, and Stefan Szeider

This is the authors' copy of a paper that is to appear in the proceedings of SAT 2019, the 22nd International Conference on Theory and Applications of Satisfiability Testing, Lisbon, Portugal, July 9-12, 2019

www.ac.tuwien.ac.at/tr

Proof Complexity of Fragments of Long-Distance Q-Resolution[⋆]

Tomáš Peitl, Friedrich Slivovsky, and Stefan Szeider

Algorithms and Complexity Group, TU Wien, Vienna, Austria
{peitl,fs,sz}@ac.tuwien.ac.at

Abstract. Q-resolution is perhaps the most well-studied proof system for Quantified Boolean Formulas (QBFs). Its proof complexity is by now well understood, and several general proof size lower bound techniques have been developed. The situation is quite different for long-distance Q-resolution (LDQ-resolution). While lower bounds on LDQ-resolution proof size have been established for specific families of formulas, we lack semantically grounded lower bound techniques for LDQ-resolution.
In this work, we study restrictions of LDQ-resolution. We show that a specific lower bound technique based on bounded-depth strategy extraction does not work even for *reductionless Q-resolution* by presenting short proofs of the QParity formulas. Reductionless Q-resolution is a variant of LDQ-resolution that admits merging but no universal reduction. We also prove a lower bound on the proof size of the *completion principle* formulas in reductionless Q-resolution. This shows that two natural fragments of LDQ-resolution are incomparable: Q-resolution, which allows universal reductions but no merging, and reductionless Q-resolution, which allows merging but no universal reductions. Finally, we develop semantically grounded lower bound techniques for fragments of LDQ-resolution, specifically tree-like LDQ-resolution and regular reductionless Q-resolution.

1 Introduction

The effectiveness of modern satisfiability (SAT) solvers has established propositional logic as the language of choice for encoding hard combinatorial problems from areas such as formal verification [8, 33] and AI planning [27]. However, since the computational complexity of these problems usually exceeds the complexity of SAT, propositional encodings of such problems can be exponentially larger than their original descriptions. This imposes a limit on the problem instances that can be feasibly solved even with extremely efficient SAT solvers, and has prompted research on decision procedures for more succinct logical formalisms such as Quantified Boolean Formulas (QBFs).

QBFs augment propositional formulas with existential and universal quantification over truth values and can be exponentially more succinct. The downside of this conciseness is that the satisfiability problem of QBFs is PSPACE-

[⋆] This research was partially supported by FWF grants P27721 and W1255-N23.

complete [29], and in spite of substantial progress in solver technology, practically relevant instances remain hard to solve. Unlike in the case of SAT, where Conflict-Driven Clause Learning (CDCL) [15] has emerged as the single dominant solving paradigm, there is a variety of competing solver architectures for QBF, most of which are either based on a generalization of CDCL (QCDCL) [11, 35], quantifier expansion [7, 17], or clausal abstraction [18, 26, 30].

Research in proof complexity has provided valuable insights into the theoretical limits of different solving approaches and their relative strengths and weaknesses. *Q-resolution* [21] is perhaps the most well-studied QBF proof system, largely due to the fact that it is used by QCDCL solvers for proof generation (see Section 3). Early proof size lower bounds for Q-resolution relied on propositional hardness or ad-hoc arguments [21]. Semantically grounded lower bound techniques based on strategy extraction have been developed only recently [6, 4]. These techniques identify properties of winning strategies extracted from proofs and use them to derive proof size lower bounds. They not only help us prove lower bounds for new classes of formulas but afford a better understanding of what *kinds* of problems certain proof systems can solve efficiently.

Long-distance Q-resolution is a variant of Q-resolution which allows the derivation of syntactically tautological clauses in certain cases [1] and which is arguably the most natural proof system for use in a QCDCL solver [35, 13]. Although lower bounds for long-distance Q-resolution have been proved [3, 6], we lack semantically grounded lower bound techniques for this proof system. In this paper, we present results on the proof complexity of restricted versions of long-distance Q-resolution:

1. We prove an exponential lower bound on the *reductionless* Q-resolution [9] proof size of a class of QBFs with short Q-resolution refutations [20].
2. We observe that the QParity formulas [6] have short proofs in reductionless Q-resolution. It has already been shown that these formulas have short (linear) proofs in long-distance Q-resolution [12], and in fact these proofs are reductionless. In combination with the first result, this proves the incomparability of Q-resolution and reductionless Q-resolution. It also marks the breakdown of a semantically grounded lower bound technique for Q-resolution [6]—strategies corresponding to reductionless Q-resolution proofs cannot be efficiently represented by bounded-depth circuits.
3. Finally, we develop semantically grounded lower bound techniques for restricted subsystems of long-distance Q-resolution. Specifically, we show that the strategy functions computed by proofs in *regular* reductionless Q-resolution are read-once branching programs, and that *tree-like* long-distance Q-resolution proofs correspond to bounded depth circuits.

2 Preliminaries

We assume a countably infinite set V of propositional *variables* and consider *propositional formulas* constructed from V using the connectives $\neg$ (negation), $\wedge$ (conjunction), $\vee$ (disjunction), $\rightarrow$ (implication), and $\leftrightarrow$ (the biconditional).

The *size* $|\varphi|$ of a propositional formula φ is number of variable occurrences in φ plus the number of connectives. Given a propositional formula φ, we write $var(\varphi)$ to denote the set of variables occurring in φ. A *literal* is a variable v or a negated variable $\neg v$. A *clause* is a finite disjunction of literals. A clause is *tautological* if it contains both v and $\neg v$ for some variable v. A propositional formula is in *conjunctive normal form (CNF)* if it is a finite conjunction of non-tautological clauses. An *assignment* (or *variable assignment*) is a function that maps a subset $X \subseteq V$ of variables to the set $\{0, 1\}$ of truth values. Given a propositional formula φ and an assignment $\tau : X \to \{0, 1\}$ with $var(\varphi) \subseteq X$, we let $\varphi[\tau]$ denote the truth value obtained by evaluating φ under τ. The formula φ is *satisfied* by τ if $\varphi[\tau] = 1$, otherwise it is *falsified* by τ.

We consider Quantified Boolean Formulas in *Prenex Conjunctive Normal Form (PCNF)*. A PCNF formula $\mathcal{F} = \mathcal{Q}.\varphi$ consists of a *quantifier prefix* $\mathcal{Q}$ and a propositional formula φ in conjunctive normal form, called the *matrix* of $\mathcal{F}$. The quantifier prefix is a sequence $Q_1x_1 \dots Q_nx_n$ where $Q_i \in \{\exists, \forall\}$ and the x_i are pairwise distinct variables for $1 \leq i \leq n$. The quantifier prefix defines an ordering $<_\mathcal{F}$ on its variables as $x_i <_\mathcal{F} x_j$ for $1 \leq i < j \leq n$. We assume that $\{x_1, \dots, x_n\} = var(\varphi)$ and write $var(\mathcal{F}) = var(\varphi)$. The set of *existential* variables of $\mathcal{F}$ is $var_\exists(\mathcal{F}) = \{\, x_i \mid 1 \leq i \leq n, Q_i = \exists \,\}$, and the set of *universal* variables of $\mathcal{F}$ is $var_\forall(\mathcal{F}) = \{\, x_i \mid 1 \leq i \leq n, Q_i = \forall \,\}$.

A *strategy function* for a universal variable $u \in var_\forall(\mathcal{F})$ is a Boolean function $f_u : 2^{D^u_\mathcal{F}} \to \{0, 1\}$. Here, $D^u_\mathcal{F} = \{\, v \in var_\exists(\mathcal{F}) \mid v <_\mathcal{F} u \,\}$ is the set of existential variables to the left of u in the quantifier prefix and $2^{D^u_\mathcal{F}}$ denotes the set of assignments of $D^u_\mathcal{F}$. A *universal winning strategy* for a PCNF formula $\mathcal{F}$ is a family $\{\, f_u \mid u \in var_\forall(\mathcal{F}) \,\}$ of strategy functions with the following property. Let $\tau : var(\mathcal{F}) \to \{0, 1\}$ be an assignment satisfying $\tau(u) = f_u(\tau|_{D^u_\mathcal{F}})$ for each universal variable u, where $\tau|_{D^u_\mathcal{F}}$ denotes the restriction of τ to $D^u_\mathcal{F}$. Then the matrix of $\mathcal{F}$ is falsified by τ. A PCNF formula $\mathcal{F}$ is *false* if there exists a universal winning strategy for $\mathcal{F}$ and *true* otherwise.

3 Q-Resolution Proof Systems

In this section, we are going to introduce several clausal proof systems for PCNF formulas. The original *Q-resolution* proof system consists of propositional resolution and the *universal reduction* rule for dealing with universally quantified variables. This system—which is displayed in Figure 1—was shown to be sound and complete for false PCNF formulas [21]. Different variants of Q-resolution can be identified as the proof systems underlying search-based QBF solvers. For instance, the traces of certain DPLL-style algorithms can be mapped to Q-resolution proofs [14]. However, since ordinary Q-resolution explicitly forbids tautological resolvents, this requires that literals that would result in a tautology be resolved away in a recursive manner, a process that was shown to require exponential time in the worst case [32].

The exponential overhead can be avoided by a more intricate analysis of the implication graph [23], but arguably the more natural solution is to allow

$$\frac{}{C}\ \text{(input clause)} \qquad\qquad \frac{C_1 \vee e \qquad \neg e \vee C_2}{C_1 \vee C_2}\ \text{(resolution)}$$

An *input clause* $C \in \varphi$ can be used as an axiom. From two clauses $C_1 \vee e$ and $\neg e \vee C_1$, the *resolution* rule can derive the clause $C_1 \vee C_2$ provided that it is non-tautological. Here, e is an existential variable called the *pivot*.

$$\frac{C \vee \ell}{C}\ \text{(universal reduction)}$$

The *universal reduction* rule can derive the clause C from $C \vee \ell$ if $var(\ell)$ is universal and there is no existential variable $e \in var(C)$ with $\ell <_{\mathcal{F}} e$.

Fig. 1. Derivation rules of Q-resolution for a PCNF formula $\mathcal{F} = \mathcal{Q}.\varphi$.

tautological clauses during learning. This was the approach taken by an early version of CDCL for QBF [35], but the resulting proof system was only studied and proven sound under the name of *long-distance Q-resolution* much later [1]. Long-distance Q-resolution involves a generalized resolution rule that allows for the derivation of tautologies, or equivalently, of *merged literals*. A merged literal u^* is generated for a universal variable u upon resolving a clause $C_1 \vee e \vee u$ with a clause $C_2 \vee \neg e \vee \neg u$. Here, it is required that $e < u$ in the quantifier prefix. Since u^* is essentially a shorthand for $u \vee \neg u$, we let $\overline{u^*} = u^*$ and $var(u^*) = u$. The resolution rule of long-distance Q-resolution is shown in Figure 2. The long-distance Q-resolution (LDQ-resolution) proof system is comprised of the input clause rule, long-distance resolution, and universal reduction.

$$\frac{C_1 \vee e \qquad \neg e \vee C_2}{(C_1 \setminus \overline{C_2}) \vee (C_2 \setminus \overline{C_1}) \vee \{\, u^* \mid u \in var(C_1 \cap \overline{C_2}) \,\}}\ \text{(long-distance resolution)}$$

The *long-distance resolution* rule can derive clauses containing merged literals u^* for universal variables u. We require that $e <_{\mathcal{F}} u$ for each such variable $u \in var(C_1 \cap \overline{C_2})$, and that $C_1 \vee C_2$ does not contain an existential variable and its negation.

Fig. 2. The long-distance resolution rule for a PCNF formula $\mathcal{F}$.

The QBF solver GhostQ [22] uses a restricted version of long-distance Q-resolution without universal reduction. Indeed, in a search-based solver that assigns variables in the order of the quantifier prefix, universal reduction is not required to derive a learned clause. One only needs to identify purely universal clauses, which are treated as if they were empty. The corresponding traces can be construed as derivations in the proof system shown in Figure 3. We refer to this system—which was first studied under the name of *Q^w-resolution* [9]—as *reductionless Q-resolution*.

$$\frac{}{C} \text{ (input clause)}$$

Every input clause $C \in \varphi$ can be used as an axiom.

$$\frac{C_1 \vee e \qquad \neg e \vee C_2}{(C_1 \setminus \overline{C_2}) \vee (C_2 \setminus \overline{C_1}) \vee \{\, u^* \mid u \in \mathit{var}(C_1 \cap \overline{C_2}) \,\}} \text{ (long-distance resolution)}$$

It is required that $e <_{\mathcal{F}} u$ for each variable $u \in \mathit{var}(C_1 \cap \overline{C_2})$ and that $C_1 \vee C_2$ does not contain an existential variable and its negation.

Fig. 3. Derivation rules of reductionless Q-resolution for a PCNF formula $\mathcal{F} = \mathcal{Q}.\varphi$.

As usual, we consider *derivations* in these proof systems which are sequences $C_1, \ldots, C_k$ of clauses such that each clause C_i is an axiom or derived from clauses appearing earlier in the sequence using one of the proof rules. The *size* of a derivation is the number k of clauses in the sequence. A *refutation* is a derivation of the empty clause or, in the case of reductionless Q-resolution, a derivation of a purely universal clause.

In the following sections, we will sometimes write $C_1 \odot_x C_2$ to denote the resolvent of C_1 and C_2 on pivot x. If the pivot is understood we may simply write $C_1 \odot C_2$.

4 A Lower Bound for Reductionless Q-resolution

We generalize an exponential lower bound for *level-ordered* Q-resolution [19] for the *completion principle* formulas CR_n defined below. A Q-resolution derivation is level-ordered if the order of pivot variables encountered on any path in the derivation follows the order in the quantifier prefix. A level-ordered Q-resolution refutation can be turned into a reductionless Q-resolution refutation simply by omitting the reduction steps.

Definition 1 ([19]). *Let*

$$\mathrm{CR}_n = \underset{1\le i,j\le n}{\exists} x_{ij}\ \forall z\ \overset{n}{\underset{i=1}{\exists}} a_i\ \overset{n}{\underset{j=1}{\exists}} b_j \quad \left(\bigwedge_{1\le i,j\le n} A_{ij} \wedge B_{ij} \right) \wedge A \wedge B,$$

where

$$A_{ij} = x_{ij} \vee z \vee a_i, \qquad A = \overline{a_1} \vee \cdots \vee \overline{a_n},$$
$$B_{ij} = \overline{x_{ij}} \vee \overline{z} \vee b_j, \qquad B = \overline{b_1} \vee \cdots \vee \overline{b_n}.$$

We will prove the following result.

Theorem 1. *Any reductionless Q-resolution refutation of the formula* CR_n *has size at least* 2^n.

In the following we assume $n \geq 2$ (Theorem 1 obviously holds for $n = 1$), Π is a reductionless Q-resolution refutation of CR_n, C is a clause of Π, and $C_\perp$ the conclusion of Π (recall that a reductionless Q-resolution refutation ends in a purely universal clause). For the purposes of this subsection, we will consider a merged literal u^* as a shorthand for $u \vee \neg u$.

Claim 1. For all $1 \leq i \leq n$ and $1 \leq j \leq n$,

– if $a_i \in C$ or $x_{ij} \in C$ then $z \in C$,
– if $b_j \in C$ or $\overline{x_{ij}} \in C$ then $\overline{z} \in C$.

Proof. The statement holds for input clauses and universal literals are never removed from clauses. □

Claim 2. For all $1 \leq i \leq n$ and $1 \leq j \leq n$,

– if $\overline{a_i} \in C$ then $z \in C$ or $C = A$,
– if $\overline{b_j} \in C$ then $\overline{z} \in C$ or $C = B$.

Proof. The statement holds for input clauses. Let C be the resolvent of C_1 and C_2 and assume without loss of generality that $\overline{a_i} \in C_1$. By induction hypothesis either $z \in C_1$, in which case $z \in C$, or $C_1 = A$, in which case the resolution step is over some pivot a_j. That means $a_j \in C_2$, and by Claim 1 we have $z \in C_2$ and so $z \in C$. □

Claim 3.

– For all $1 \leq i \leq n$, if $\overline{z} \notin C \wedge \overline{a_i} \notin C \wedge C \neq B$, then there is j, such that $x_{ij} \in C$,
– For all $1 \leq j \leq n$, if $z \notin C \wedge \overline{b_j} \notin C \wedge C \neq A$, then there is i, such that $\overline{x_{ij}} \in C$.

Proof. By induction on the proof size. The statement clearly holds for input clauses. Let C be the resolvent of C_1 and C_2. Since $\overline{z} \notin C$, both $\overline{z} \notin C_1$ and $\overline{z} \notin C_2$. Since $\overline{a_i} \notin C$, either $\overline{a_i} \notin C_1$ or $\overline{a_i} \notin C_2$, assume the first. If $C_1 = B$, then the resolution step can only resolve away one of the literals, and so there is j such that $\overline{b_j} \in C$. By Claim 2 (since $C \neq B$) we have $\overline{z} \in C$, a contradiction. Therefore $C_1 \neq B$ and by induction hypothesis there is j such that $x_{ij} \in C_1$. Unless the resolution step is on x_{ij}, we have $x_{ij} \in C$, so it remains to prove that this is indeed not the case. If x_{ij} were the pivot, then $\overline{x_{ij}} \in C_2$, hence $\overline{z} \in C_2$, a contradiction. □

Claim 4. The conclusion $C_\perp$ contains z^*.

Proof. $C_\perp$ contains no existential literals and is distinct from A and B, so in order not to be in contradiction with Claim 3, the statement has to hold. □

Claim 5. If $z, \overline{z} \in C$, then for all $1 \leq i, j \leq n$ we have $a_i, \overline{a_i}, b_j, \overline{b_j} \notin C$.

Proof. Consider the last C that violates this implication. Clearly $C \neq C_\perp$. Therefore, there is C_0 and C_1 such that C_1 is the resolvent of C and C_0. Clearly $z, \overline{z} \in C_1$. Since C_1 no longer violates the condition, there is no literal right of z in C_1. Hence C_0 is neither A nor B. Therefore by Claim 1 and Claim 2, a literal on z is in C_0. Since C violates the implication, there is a literal right of z in C, but since there is none in C_1, the pivot variable must be right of z. That means the resolution step is illegal, a contradiction. □

Claim 6. If C is the resolvent of C_1 and C_2 and $z, \overline{z} \in C$, then neither C_1 nor C_2 contains any of the literals $a_i, \overline{a_i}, b_j, \overline{b_j}$.

Proof. If $z, \overline{z} \in C_1$ or $z, \overline{z} \in C_2$, then the statement follows from Claim 5. Otherwise, the resolution step merges z and $\overline{z}$. That means the pivot must be left of z, and so any of the literals $a_i, \overline{a_i}, b_j, \overline{b_j}$ would end up in C if contained in any of the premises, a contradiction with Claim 5. □

For the next claims, we need to introduce sets M and S as follows. We let

$$M = \{\, C \in \Pi \mid z^* \in C \,\}$$

be the set of clauses which contain a merged literal. We define S as the "boundary" of M, i.e., the set of clauses that do not contain a merged literal but have a direct descendant that does, formally

$$S = \{\, C \in \Pi \mid C \notin M \text{ and there are } C_0, C_1 \in \Pi \text{ s.t. } C_1 = C \odot C_0 \text{ and } C_1 \in M \,\}.$$

Claim 7. If $C \in S$, then $a_i, \overline{a_i}, b_j, \overline{b_j} \notin C$.

Proof. Follows from Claim 6 as $C \in S$ has a direct descendant with z^*. □

Claim 8. If $C \in S$ then $|C \setminus \{z, \overline{z}\}| \geq n$.

Proof. By Claim 7 and the fact that $C \notin M$ we get that all preconditions of one of the rows of Claim 3 are satisfied for C, which means that either for all i there is a j such that $x_{ij} \in C$, or for all j there is an i such that $\overline{x_{ij}} \in C$, in both cases a total of n distinct literals, none of which is a literal on z. □

Proof (of Theorem 1). Consider $\Pi' = S \cup M$. Clauses in M have direct ancestors only in M or S (any direct ancestor that is not in M is by definition in S). Since $C_\perp \in M$, Π' is a reductionless Q-resolution refutation of S. If we disregard literals on z, it is in fact a resolution refutation of the propositional formula S, which means that S is unsatisfiable. By Claim 8, every clause in S has at least n literals, and so it excludes at most 2^{n^2-n} of the assignments to the variables of S. Therefore, S must have at least 2^n clauses in order to exclude all assignments and be unsatisfiable. □

Corollary 1. *Reductionless Q-resolution does not p-simulate tree-like Q-resolution.*

Proof. Since CR_n have short proofs in tree-like Q-resolution [20], the separation follows from Theorem 1. □

Remark 1. Theorem 1 has another interesting consequence. Since QCDCL with dependency learning can solve CR_n in polynomial time [25], Theorem 1 implies that in order to harness the full power of QCDCL with dependency learning, one has to perform universal reduction during clause learning. This is in contrast with "ordinary" QCDCL where universal reduction is not required to derive a learned clause [22].

5 Short Proofs of QParity in Reductionless Q-Resolution

In this section, we prove that the QParity formulas, which require exponentially long proofs in Q-resolution [6], have short proofs in reductionless Q-resolution. It has already been shown that these formulas have short proofs in long-distance Q-resolution [12, Theorem 9]. We simply observe that these proofs—which we reproduce below for the sake of completeness—are in fact reductionless.

Definition 2 ([6]). *Let* $\mathrm{QParity}_n = \exists x_1, \dots, x_n\, \forall z\, \exists t_2, \dots, t_n.\, \phi_n$, *where*

$$\phi_n = T_2^1 \wedge T_2^2 \wedge T_2^3 \wedge T_2^4 \wedge \left(\bigwedge_{i=3}^{n} T_i^1 \wedge T_i^2 \wedge T_i^3 \wedge T_i^4 \right) \wedge Z^1 \wedge Z^2, \text{ and}$$

$$\begin{aligned}
T_2^1 &= x_1 \vee x_2 \vee \overline{t_2}, & T_i^1 &= t_{i-1} \vee x_i \vee \overline{t_i}, & Z^1 &= t_n \vee \overline{z},\\
T_2^2 &= x_1 \vee \overline{x_2} \vee t_2, & T_i^2 &= t_{i-1} \vee \overline{x_i} \vee t_i, & Z^2 &= \overline{t_n} \vee z.\\
T_2^3 &= \overline{x_1} \vee x_2 \vee t_2, & T_i^3 &= \overline{t_{i-1}} \vee x_i \vee t_i, & &\\
T_2^4 &= \overline{x_1} \vee \overline{x_2} \vee \overline{t_2}, & T_i^4 &= \overline{t_{i-1}} \vee \overline{x_i} \vee \overline{t_i}, & &
\end{aligned}$$

Theorem 2. *There is a reductionless Q-resolution refutation of* $\mathrm{QParity}_n$ *of size* $6n - 5$.

Proof. For $2 \leq i \leq n-1$ and $1 \leq j \leq 2$, we let $Z_i^1 = t_i \vee z^*$, $Z_i^2 = \overline{t_i} \vee z^*$, and $Z_n^j = Z^j$, and it is easy to verify that

$$Z_{i-1}^j = \left(T_i^{3j-2} \odot_{t_i} Z_i^1\right) \odot_{x_i} \left(T_i^{j+1} \odot_{t_i} Z_i^2\right).$$

Hence, we derive Z_2^1 and Z_2^2 in a total of $6(n-2)$ steps. Next, we have

$$(z^*) = \left(\left(T_2^1 \odot_{t_2} Z_2^1\right) \odot_{x_2} \left(T_2^2 \odot_{t_2} Z_2^2\right) \right) \odot_{x_1} \left(\left(T_2^4 \odot_{t_2} Z_2^1\right) \odot_{x_2} \left(T_2^3 \odot_{t_2} Z_2^2\right) \right),$$

and so the formula is refuted. The resolution steps on the x_i are sound, because $x_i < z$ for all $1 \leq i \leq n$. The total number of resolution steps is $6n - 5$. □

Corollary 2. *Q-resolution does not p-simulate reductionless Q-resolution.*

Remark 2. While strategies extracted from Q-resolution refutations (of PCNF formulas containing a single unviersal variable) correspond to bounded-depth circuits [6], Theorem 2 implies that reductionless Q-resolution proofs cannot be efficiently transformed into bounded-depth circuits.

6 Lower Bounds from Strategy Extraction

In this section, we will show how to extend the scope of lower bound techniques based on strategy extraction [6] to fragments of long-distance Q-resolution. We begin by observing that strategies extracted from reductionless Q-resolution proofs correspond to branching programs [5].

We briefly review the definition of a branching program and refer to the book by Wegener for more details [34]. A *branching program* or *binary decision diagram (BDD)* on a set X of variables is a directed acyclic graph with a unique source node and at most two sink nodes. Each node v is labelled with a variable $\lambda(v) \in X$, except for the sinks, which are labelled with 0 or 1. If there are two sink nodes, their labels must be distinct. Moreover, every node has exactly two outgoing edges labelled with 0 and 1, respectively. A path $v_1, \dots, v_n$ from the source of a branching program to its sink is *consistent* if the label of edge (v_i, v_{i+1}) agrees with the label of edge (v_j, v_{j+1}) whenever v_i and v_j are labelled with the same variable. A consistent path corresponds to an assignment in the obvious way. A branching program B on X computes a Boolean function $f(B)$ in the following way. Let $\tau : X \to \{0,1\}$ be an assignment. We follow the (consistent) path induced by τ to a sink node v, and set $f(B)(\tau) = \lambda(v)$. The *size* of a branching program is the number of nodes in it.

Let $\pi = C_1, \dots, C_k$ be a reductionless Q-resolution derivation from a PCNF formula $\mathcal{F}$. For each universal variable $u \in var_\forall(\mathcal{F})$, we construct a branching program $B^u_\mathcal{F}(\pi)$ in the following way [9, 5]. We first introduce two nodes v_0 and v_1 with $\lambda(v_0) = 0$ and $\lambda(v_1) = 1$. We now consider the clauses C_i in the order of their derivation and associate a node v_i with each one. Depending on how clause C_i was derived, we distinguish two cases:

1. If C_i is an input clause, we let
$$v_i = \begin{cases} v_0, & \text{if } u \in C_i; \\ v_1, & \text{otherwise.} \end{cases}$$
2. If C_i is the resolvent of clauses $C_j = C'_j \lor e$ and $C_l = \neg e \lor C'_l$, there are two possibilities depending on the order of variables e and u in the prefix:
   – If $e < u$, we introduce a fresh node v_i to $B^u_\mathcal{F}(\pi)$ and label it $\lambda(v) = e$. Moreover, we add a 0-labelled edge from v_i to v_j and a 1-labelled edge from v_i to v_l.
   – Otherwise, $u < e$ and we cannot have $u \in var(C_j \cap \overline{C_l})$ by the rules of reductionless Q-resolution (see Figure 3). If $u \in var(C_j)$, we let $v_i = v_j$. Otherwise, we let $v_i = v_l$.

Finally, we remove all nodes that cannot be reached from v_k. The following statement is immediate from the construction.

Lemma 1. *If $\pi = C_1, \dots, C_k$ is a reductionless Q-resolution derivation from a PCNF formula $\mathcal{F}$ and $u \in var_\forall(\mathcal{F})$ is a universal variable, then $B^u_\mathcal{F}(\pi)$ is a branching program on $D^u_\mathcal{F}$ of size at most k.*

Moreover, if the derivation π is a refutation, these branching programs compute a universal winning strategy. To show this, we first prove the following statement.

Lemma 2. *Let $\pi = C_1, \dots, C_k$ be a reductionless Q-resolution derivation from a PCNF formula $\mathcal{F}$. Let $\tau : \mathit{var}_\exists(\mathcal{F}) \to \{0, 1\}$ be an assignment that does not satisfy C_k, and let*

$$\sigma_{\mathcal{F}}^{\pi} = \{\, u \mapsto f(B_{\mathcal{F}}^{u}(\pi))(\tau|_{D_{\mathcal{F}}^{u}}) \mid u \in \mathit{var}_\forall(\mathcal{F}) \,\}$$

be the assignment computed by the branching programs $B_{\mathcal{F}}^{u}(\pi)$ in response. Then $C_i[\sigma_{\mathcal{F}}^{\pi} \cup \tau] = 0$ for some input clause $C_i \in \pi$.

Proof. We proceed by induction on the size k of the derivation. If $\pi = C_1$ then C_1 must be an input clause, and $B_{\mathcal{F}}^{u}(\pi)$ consists of a single node labelled 0 if $u \in C_1$ and labelled 1 if $\neg u \in C_1$. Accordingly, the function $f(B_{\mathcal{F}}^{u}(\pi))$ constantly returns an assignment that falsifies any universal literal on variable u. This proves the base case.

Suppose the statement of the lemma holds for derivations of size up to $k - 1$. If C_k is an input clause, the same reasoning as in the base case applies, so suppose C_k is derived by resolution from clauses $C_i = C_i' \lor e$ and $C_j = \neg e \lor C_j'$ with $1 \leq i, j < k$. Let $\pi_i = C_1, \dots, C_i$ and let $\pi_j = C_1, \dots, C_j$ be the derivations of the corresponding clauses. We claim that $\sigma_{\mathcal{F}}^{\pi}(u) = \sigma_{\mathcal{F}}^{\pi_i}(u)$ for each universal variable $u \in \mathit{var}(C_i)$ if $\tau(e) = 0$, and $\sigma_{\mathcal{F}}^{\pi}(u) = \sigma_{\mathcal{F}}^{\pi_j}(u)$ for each universal variable $u \in \mathit{var}(C_j)$ if $\tau(e) = 1$.

Choose a universal variable u and let $\tau' = \tau|_{D_{\mathcal{F}}^{u}}$ be the corresponding restriction of τ. We consider two cases. If $e < u$ it is not difficult to see that

$$f(B_{\mathcal{F}}^{u}(\pi))(\tau') = \begin{cases} f(B_{\mathcal{F}}^{u}(\pi_i))(\tau') \text{ if } \tau(e) = 0, \text{ and} \\ f(B_{\mathcal{F}}^{u}(\pi_j))(\tau') \text{ otherwise.} \end{cases}$$

Accordingly, we have $\sigma_{\mathcal{F}}^{\pi}(u) = \sigma_{\mathcal{F}}^{\pi_i}(u)$ if $\tau(e) = 0$ and otherwise $\sigma_{\mathcal{F}}^{\pi}(u) = \sigma_{\mathcal{F}}^{\pi_j}(u)$. On the other hand, if $u < e$ by construction of the branching program we have

$$f(B_{\mathcal{F}}^{u}(\pi)) = \begin{cases} f(B_{\mathcal{F}}^{u}(\pi_i)) \text{ if } u \in \mathit{var}(C_i), \text{ and} \\ f(B_{\mathcal{F}}^{u}(\pi_j)) \text{ otherwise.} \end{cases}$$

If $u \in \mathit{var}(C_i)$ then $\sigma_{\mathcal{F}}^{\pi}(u) = \sigma_{\mathcal{F}}^{\pi_i}(u)$. If $u \in \mathit{var}(C_j)$ as well then we must have $u \in C_i \cap C_j$ or $\neg u \in C_i \cap C_j$ since $u \notin \mathit{var}(C_i \cap \overline{C_j})$ by definition of reductionless Q-resolution. It follows that $f(B_{\mathcal{F}}^{u}(\pi_i))$ and $f(B_{\mathcal{F}}^{u}(\pi_j))$ compute the same constant function. Otherwise, if $u \notin \mathit{var}(C_i)$ then $\sigma_{\mathcal{F}}^{\pi}(u) = \sigma_{\mathcal{F}}^{\pi_j}(u)$. This proves the claim.

If $\tau(e) = 0$ then by induction hypothesis $C_l[\sigma_{\mathcal{F}}^{\pi_i} \cup \tau] = 0$ for an input clause $C_l \in \pi_i$. We can assume without loss of generality that π_i does not contain any universal variable besides those in the clause C_i. It follows from the claim that the assignment $\sigma_{\mathcal{F}}^{\pi} \cup \tau$ falsifies C_l as well. The case $\tau(e) = 1$ is symmetric. ☐

Lemma 3. *Let $\pi = C_1, \ldots, C_k$ be a reductionless Q-resolution refutation of a PCNF formula $\mathcal{F}$. The set $\{ f(B^u_{\mathcal{F}}(\pi)) \mid u \in var_\forall(\mathcal{F}) \}$ is a universal winning strategy.*

Proof. Because π is a refutation the clause C_k must not contain existential variables. Thus every assignment $\tau : var_\exists(\mathcal{F}) \to \{0, 1\}$ is an assignment that does not satisfy C_k, and by Lemma 2 the universal response $\sigma^\pi_{\mathcal{F}}$ (defined as in the statement of that lemma), in conjunction with the assignment τ, must falsify an input clause. □

These results allow us to translate lower bounds for branching programs to lower bounds on the size of reductionless Q-resolution refutations. Let $f : X \to \{0, 1\}$ be a Boolean function, let $\varphi(X)$ be a Boolean circuit encoding f, and let u be a variable not occurring in φ. Using Tseitin transformation [31], we can construct a CNF formula $\psi(X, u, Y)$ such that $\exists Y.\psi(X, u, Y)$ is logically equivalent to $\varphi(X) \neq u$. The PCNF formula $\mathcal{F} = \exists X \forall u \exists Y.\psi(X, u, Y)$ is a false PCNF formula with f as a unique universal winning strategy (cf. the lower bounds from strategy extraction for Q-resolution [6]). We call such a formula $\mathcal{F}$ a *PCNF encoding of f*.

Proposition 1. *Let $\mathcal{F}$ be a PCNF encoding of a Boolean function f such that any branching program computing f has size at least m. Then any reductionless Q-resolution refutation of $\mathcal{F}$ requires at least m clauses.*

Proof. Since f is the unique universal winning strategy for $\mathcal{F}$, the statement follows immediately from Lemma 1 and Lemma 3. □

To the best of our knowledge, the only lower bounds on the size of general branching programs for explicitly defined Boolean functions currently known are polynomial [24]. Accordingly, Proposition 1 does not yield strong lower bounds for reductionless Q-resolution. However, we can lift lower bounds for restricted classes of branching programs to lower bounds on the proof size in restricted versions of reductionless Q-resolution.

6.1 Regular Reductionless Q-Resolution

Every reductionless Q-resolution derivation $\pi = C_1, \ldots, C_k$ can be represented by a directed acyclic graph $G(\pi)$ on vertices $v_1, \ldots, v_k$ where v_i is labelled with C_i and there is an edge from v_j to v_i if $i < j$ and C_i is one of the clauses C_j was derived from (that is, edges are oriented from conclusions to premises). Each edge is labelled with the corresponding pivot variable.

A reductionless Q-resolution refutation $\pi = C_1, \ldots, C_k$ is *regular* if each variable occurs at most once as a label on any directed path starting from the vertex labelled with clause C_k. Each strategy function computed by such a proof corresponds to a so-called *read-once branching programs* or *free binary decision diagram (FBDD)*. A read-once branching program is a branching program where each variable is encountered at most once on any path from the source to a sink [34].

Lemma 4. *Let $\pi = C_1, \dots, C_k$ be a regular reductionless Q-resolution refutation of a PCNF formula $\mathcal{F}$. Then $B^u_{\mathcal{F}}(\pi)$ is a read-once branching program of size at most k for each universal variable $u \in var_\forall(\mathcal{F})$.*

Proof. Consider $B^u_{\mathcal{F}}(\pi)$ for any universal variable $u \in var_\forall(\mathcal{F})$. By construction, the sequence of variables encountered on any path starting from the source of $B^u_{\mathcal{F}}(\pi)$ is a subsequence of the pivot variables seen as edge labels on any path starting from the source of $G(\pi)$. In particular, every variable occurs at most once. Since $B^u_{\mathcal{F}}(\pi)$ is a branching program of size at most k by Lemma 1, it is in fact a read-once branching program of size at most k. □

The *FBDD size* of a Boolean function f is the size of the smallest read-once branching program representing f. We can transfer lower bounds on the FBDD size of Boolean functions into lower bounds on the regular reductionless Q-resolution proof size of certain PCNF formulas, as stated in the next result.

Proposition 2. *Let $\mathcal{F}$ be a PCNF encoding of a Boolean function f with FBDD size m. Any regular reductionless Q-resolution refutation of $\mathcal{F}$ has size at least m.*

Proof. The statement follows from Lemma 4 and Lemma 3. □

Unlike in the case of general branching programs, strong lower bounds on the FBDD size of many explicitly defined Boolean functions are known [34]. For instance, we can use the following result due to Bollig and Wegener [10].

Theorem 3 ([10]). *There is a Boolean function g in n variables that can be computed by a Boolean circuit of size $O(n^{3/2})$ but has FBDD size $\Omega(2^{\sqrt{n}})$.*

Corollary 3. *There is a Boolean function g in n variables with a PCNF encoding $\mathcal{F}$ of size polynomial in n such that any regular reductionless Q-resolution refutation of $\mathcal{F}$ has size $\Omega(2^{\sqrt{n}})$.*

6.2 Tree-like Long-Distance Q-Resolution

In this subsection, we are going to prove lower bounds on the size of *tree-like* long-distance Q-resolution. As in the case of reductionless Q-resolution, a long-distance Q-resolution derivation $\pi = C_1, \dots, C_k$ can be represented by a labelled DAG $G(\pi)$. A derivation π is called *tree-like* if the DAG $G(\pi)$ is a tree.

We want to show that every tree-like long-distance Q-resolution refutation of a PCNF encoding of a Boolean function f can be efficiently turned into a bounded-depth circuit computing f. First, we generalize the construction of the branching programs $B^u_{\mathcal{F}}$ described at the beginning of this section to long-distance Q-resolution derivations. Let $\pi = C_1, \dots, C_k$ be a long-distance Q-resolution derivation from a PCNF formula $\mathcal{F}$. For each universal variable $u \in var_\forall(\mathcal{F})$, we construct a labelled DAG $B^u_{\mathcal{F}}(\pi)$ in the same way as for a reductionless Q-resolution derivation, except for the following modification: if clause C_i is derived from a clause C_j by universal reduction and $u \in var(C_j) \setminus var(C_i)$, we set $v_i = v_0$, where $\lambda(v_0) = 0$. It is readily verified that we obtain a branching program of size at most k, as stated in the following lemma.

Lemma 5. *If $\pi = C_1, \ldots, C_k$ is a long-distance Q-resolution derivation from a PCNF formula $\mathcal{F}$ and $u \in var_\forall(\mathcal{F})$ is a universal variable, then $B^u_{\mathcal{F}}(\pi)$ is a branching program on $D^u_{\mathcal{F}}$ of size at most k.*

A universal winning strategy can be computed from a long-distance Q-resolution refutation as follows [2]. We maintain a kind of *decision list* [28] for each universal variable that is intended to encode a strategy function. Specifically, we consider sequences $L = (\varphi_1 \to \psi_1), \ldots, (\varphi_k \to \psi_k)$ where each of the φ_i and ψ_i are propositional formulas. Such a list, which we call a *generalized decision list*, represents a Boolean function f_L in the following way. Consider an assignment $\tau : \bigcup_{i=1}^{k} var(\varphi_i) \cup var(\psi_i) \to \{0,1\}$ to all the variables appearing in formulas on the list. If there is no index i with $1 \leq i \leq k$ such that τ satisfies φ_i, we define $f_L(\tau) = 1$. Otherwise, let i be the smallest index such that τ satisfies φ_i. Then $f_L(\tau) = \psi_i[\tau]$. The *size* of a decision list $L = (\varphi_1 \to \psi_1), \ldots, (\varphi_k \to \psi_k)$ is $|L| = \sum_{i=1}^{k}(|\varphi_i| + |\psi_i|)$.

Given a long-distance Q-resolution refutation $\pi = C_1, \ldots, C_k$ of a PCNF formula $\mathcal{F}$, we construct a family $\mathcal{L}_{\mathcal{F}}(\pi) = \{\, L_u \mid u \in var_\forall(\mathcal{F}) \,\}$ of generalized decision lists representing a universal winning strategy for $\mathcal{F}$. For $Q \in \{\exists, \forall\}$, let $C_i^Q = \{\, \ell \in C_i \mid var(\ell) \in var_Q(\mathcal{F}) \,\}$ denote the restriction of C_i to existential or universal literals. Moreover, for a Boolean function f, let $\phi(f)$ denote an encoding of f as a propositional formula. We consider applications of universal reduction in the same order as they appear in the proof. Let C_i be a clause derived by universal reduction from a clause C_j, and let $u \in var(C_j) \setminus var(C_i)$. Let $\pi_i = C_1, \ldots, C_i$ and $\pi_j = C_1, \ldots, C_j$ denote the subderivations ending in clauses C_i and C_j, respectively. We add an entry

$$\left(\overline{C_i^\exists} \wedge \bigwedge_{v \in var(C_i^\forall)} v \leftrightarrow \phi(f(B^v_{\mathcal{F}}(\pi_i))) \right) \to \phi(f(B^u_{\mathcal{F}}(\pi_j))) \qquad (1)$$

at the end of the decision list L_u. Observing that the functions $f(B^u_{\mathcal{F}}(\pi_j))$ correspond to the (negated) *phase functions* introduced for the purpose of efficiently extracting universal winning strategies from long-distance Q-resolution refutations [2], it can be verified that the strategy functions computed by the corresponding algorithm coincide with the functions computed by the decision lists defined according to (1).

Proposition 3 ([2]). *Let π be a long-distance Q-resolution refutation of a PCNF formula $\mathcal{F}$. The set $\{\, f_{L_u} \mid L_u \in \mathcal{L}_{\mathcal{F}}(\pi) \,\}$ is a universal winning strategy.*

We now argue that this winning strategy can be represented by a bounded-depth circuit for certain proofs in *tree-like* long-distance Q-resolution. Specifically, we will show that this is the case for every tree-like refutation of a PCNF encoding of a Boolean function. We first observe that the branching programs $B^u_{\mathcal{F}}$ for tree-like proofs are decision trees. A *decision tree* is a branching program that can be turned into a tree by deleting the sink nodes.

Lemma 6. *If $\pi = C_1, \ldots, C_k$ is a tree-like long-distance Q-resolution derivation from a PCNF formula $\mathcal{F}$, then $B^u_{\mathcal{F}}(\pi)$ is a decision tree for each universal variable $u \in var_\forall(\mathcal{F})$.*

Proof. It is not difficult to see that after deleting the sink nodes labelled with 0 and 1 from $B^u_{\mathcal{F}}(\pi)$, the corresponding DAG can be obtained from $G(\pi)$ by deleting vertices and edges as well as contracting induced paths. Since $G(\pi)$ is a tree, the result is also a tree. $\square$

Every decision tree can be efficiently translated into a CNF formula by taking the conjunction over the negations of its consistent paths [28]. Moreover, a generalized decision list $L = (\varphi_1 \rightarrow \psi_1), \ldots, (\varphi_k \rightarrow \psi_k)$ can be represented by a circuit

$$\phi(L) = \bigvee_{i=1}^{k} \left((\bigwedge_{j=1}^{i-1} \neg\varphi_j \wedge \varphi_i) \rightarrow \psi_i \right). \tag{2}$$

Lemma 7. *Let $L = (\varphi_1 \rightarrow \psi_1), \ldots, (\varphi_k \rightarrow \psi_k)$ be a generalized decision list such that d is the maximum depth of any formula φ_i and ψ_i, for $1 \leq i \leq k$. Then $\phi(L)$ is equivalent to f_L. Moreover, $\phi(L)$ has depth at most $d + 4$ and $|\phi(L)| = O(|L|^2)$.*

Let $\mathcal{F}$ be the PCNF encoding of a Boolean function f and consider a tree-like long-distance Q-resolution refutation π of $\mathcal{F}$. Because $\mathcal{F}$ contains only a single universal variable, each entry in a decision list of $\mathcal{L}_{\mathcal{F}}(\pi)$ given by (1) simplifies to $\overline{C} \rightarrow \phi(f(B^u_{\mathcal{F}}(\pi_j)))$, and the right hand side of this implication can be efficiently transformed into a CNF because $B^u_{\mathcal{F}}$ is a decision tree by Lemma 6. We thus obtain the following result.

Proposition 4. *There is a polynomial $p(\cdot)$ and a constant d such that, for any tree-like long-distance Q-resolution refutation π of the PCNF encoding of a function f, there exists a Boolean circuit of size at most $p(|\pi|)$ and depth at most d computing f.*

Proof. By Lemma 5 and Lemma 6, each labelled DAG $B^u_{\mathcal{F}}(\pi')$ is a decision tree of size at most $|\pi|$ for the universal variable u of $\mathcal{F}$ and each subproof π' of π. Each such decision tree can be efficiently encoded as a CNF formula and the decision list has no more than $|\pi|$ entries of size polynomial in $|\pi|$, so it follows from Lemma 7 that there is a polynomial $p(\cdot)$ such that $\phi(L_u)$ has size at most $p(|\pi|)$ and depth at most 6. Finally, $\{f_{L_u}\}$ is a universal winning strategy for $\mathcal{F}$ by Proposition 3, so f_{L_u} must coincide with f. $\square$

Since any bounded-depth circuit computing the n-bit parity function has size exponential in n [16], Proposition 4 allows us to obtain the following exponential lower bound on the size of refutations of QPARITY in tree-like long-distance Q-resolution.

Theorem 4. *Any refutation of* QPARITY_n *in tree-like long-distance Q-resolution requires size exponential in n.*

7 Conclusion

We studied the proof complexity of fragments of long-distance Q-resolution. We proved that reductionless Q-resolution cannot p-simulate even tree-like Q-resolution. Since reductionless Q-resolution can be used to derive learned clauses in QCDCL solvers [22], this is another indication that QCDCL[1] proofs correspond to a fairly weak fragment of (long-distance) Q-resolution [20]. The QParity formulas, on the other hand, have short refutations in reductionless Q-resolution. These formulas require Q-resolution refutations of exponential size [6], so Q-resolution and reductionless Q-resolution turn out to be incomparable.

The existence of short proofs of QParity also marks the breakdown of an elegant technique for obtaining lower bounds on the size of Q-resolution refutations through strategy extraction [6]. Evidently, strategies corresponding to reductionless Q-resolution proofs do not correspond to bounded-depth circuits.

We proved that arguments based on strategy extraction can nevertheless be used to obtain lower bounds for restricted versions of long-distance Q-resolution. Specifically, we showed that *regular* reductionless Q-resolution proofs correspond to *read-once branching programs*, and that *tree-like* long-distance Q-resolution proofs correspond to *bounded-depth* circuits, allowing us to transfer known lower bounds.

Obtaining a characterization of the strategies corresponding to (even reductionless) long-distance Q-resolution refutations that could be used in obtaining lower bounds remains as an intriguing open problem.

References

1. Valeriy Balabanov and Jie-Hong Roland Jiang. Unified QBF certification and its applications. *Formal Methods in System Design*, 41(1):45–65, 2012.
2. Valeriy Balabanov, Jie-Hong Roland Jiang, Mikoláš Janota, and Magdalena Widl. Efficient extraction of QBF (counter)models from long-distance resolution proofs. In Blai Bonet and Sven Koenig, editors, *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, pages 3694–3701. AAAI Press, 2015.
3. Valeriy Balabanov, Magdalena Widl, and Jie-Hong Roland Jiang. QBF resolution systems and their proof complexities. In Carsten Sinz and Uwe Egly, editors, *Theory and Applications of Satisfiability Testing - SAT 2014*, volume 8561 of *Lecture Notes in Computer Science*, pages 154–169. Springer Verlag, 2014.
4. Olaf Beyersdorff, Joshua Blinkhorn, and Luke Hinde. Size, cost, and capacity: A semantic technique for hard random QBFs. *Logical Methods in Computer Science*, 15(1), 2019.
5. Olaf Beyersdorff, Joshua Blinkhorn, and Meena Mahajan. Building strategies into QBF proofs. In *36th International Symposium on Theoretical Aspects of Computer Science, STACS 2019, March 13-16, 2019, Berlin, Germany*, pages 14:1–14:18, 2019.

[1] At least without techniques like dependency learning [25].

6. Olaf Beyersdorff, Leroy Chew, and Mikoláš Janota. Proof complexity of resolution-based QBF calculi. In Ernst W. Mayr and Nicolas Ollinger, editors, *32nd International Symposium on Theoretical Aspects of Computer Science, STACS 2015, March 4-7, 2015, Garching, Germany*, volume 30 of *LIPIcs*, pages 76–89. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015.
7. Armin Biere. Resolve and expand. In *Proceedings of SAT 2004 (Seventh International Conference on Theory and Applications of Satisfiability Testing, 10–13 May, 2004, Vancouver, BC, Canada)*, pages 59–70, 2004.
8. Armin Biere. Bounded model checking. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, pages 457–481. IOS Press, 2009.
9. Nikolaj Bjørner, Mikolás Janota, and William Klieber. On conflicts and strategies in QBF. In Ansgar Fehnker, Annabelle McIver, Geoff Sutcliffe, and Andrei Voronkov, editors, *20th International Conferences on Logic for Programming, Artificial Intelligence and Reasoning - Short Presentations, LPAR 2015, Suva, Fiji, November 24-28, 2015.*, volume 35 of *EPiC Series in Computing*, pages 28–41. EasyChair, 2015.
10. Beate Bollig and Ingo Wegener. A very simple function that requires exponential size read-once branching programs. *Inf. Process. Lett.*, 66(2):53–57, 1998.
11. Marco Cadoli, Marco Schaerf, Andrea Giovanardi, and Massimo Giovanardi. An algorithm to evaluate Quantified Boolean Formulae and its experimental evaluation. *Journal of Automated Reasoning*, 28(2), 2002.
12. Leroy Nicholas Chew. *QBF proof complexity*. PhD thesis, University of Leeds, UK, 2017.
13. Uwe Egly, Florian Lonsing, and Magdalena Widl. Long-distance resolution: Proof generation and strategy extraction in search-based QBF solving. In Kenneth L. McMillan, Aart Middeldorp, and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning - LPAR 2013*, volume 8312 of *Lecture Notes in Computer Science*, pages 291–308. Springer Verlag, 2013.
14. Enrico Giunchiglia, Massimo Narizzano, and Armando Tacchella. Clause/term resolution and learning in the evaluation of Quantified Boolean Formulas. *J. Artif. Intell. Res.*, 26:371–416, 2006.
15. Carla P. Gomes, Henry Kautz, Ashish Sabharwal, and Bart Selman. Satisfiability solvers. In *Handbook of Knowledge Representation*, volume 3 of *Foundations of Artificial Intelligence*, pages 89–134. Elsevier, 2008.
16. Johan Håstad. *Computational Limitations of Small-depth Circuits*. MIT Press, Cambridge, MA, USA, 1987.
17. Mikolás Janota, William Klieber, João Marques-Silva, and Edmund M. Clarke. Solving QBF with counterexample guided refinement. In Alessandro Cimatti and Roberto Sebastiani, editors, *Theory and Applications of Satisfiability Testing - SAT 2012*, volume 7317 of *Lecture Notes in Computer Science*, pages 114–128. Springer Verlag, 2012.
18. Mikolás Janota and Joao Marques-Silva. Solving QBF by clause selection. In Qiang Yang and Michael Wooldridge, editors, *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015*, pages 325–331. AAAI Press, 2015.
19. Mikoláš Janota and Joao Marques-Silva. Expansion-based QBF solving versus Q-resolution. *Theoretical Computer Science*, 577(0):25–42, April 2015.

20. Mikolá Janota. On Q-resolution and CDCL QBF solving. In Nadia Creignou and Daniel Le Berre, editors, *Theory and Applications of Satisfiability Testing - SAT 2016*, volume 9710 of *Lecture Notes in Computer Science*, pages 402–418. Springer Verlag, 2016.
21. H. Kleine Büning, M. Karpinski, and A. Flögel. Resolution for quantified Boolean formulas. *Information and Computation*, 117(1):12–18, 1995.
22. William Klieber, Samir Sapra, Sicun Gao, and Edmund M. Clarke. A non-prenex, non-clausal QBF solver with game-state learning. In Ofer Strichman and Stefan Szeider, editors, *Theory and Applications of Satisfiability Testing - SAT 2010*, volume 6175 of *Lecture Notes in Computer Science*, pages 128–142. Springer Verlag, 2010.
23. Florian Lonsing, Uwe Egly, and Allen Van Gelder. Efficient clause learning for quantified Boolean formulas via QBF pseudo unit propagation. In Matti Järvisalo and Allen Van Gelder, editors, *Theory and Applications of Satisfiability Testing - SAT 2013*, volume 7962 of *Lecture Notes in Computer Science*, pages 100–115. Springer Verlag, 2013.
24. 'E. I. Nechiporuk. A Boolean function. *Dokl. Akad. Nauk SSSR*, 169(4):765766, 1966.
25. Tomáš Peitl, Friedrich Slivovsky, and Stefan Szeider. Dependency learning for QBF. *Journal of Artificial Intelligence Research*, vol. 65, 2019.
26. Markus N. Rabe and Leander Tentrup. CAQE: A certifying QBF solver. In Roope Kaivola and Thomas Wahl, editors, *Formal Methods in Computer-Aided Design - FMCAD 2015*, pages 136–143. IEEE Computer Soc., 2015.
27. Jussi Rintanen. Planning and SAT. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, pages 483–504. IOS Press, 2009.
28. Ronald L. Rivest. Learning decision lists. *Machine Learning*, 2(3):229–246, 1987.
29. Larry J. Stockmeyer and Albert R. Meyer. Word problems requiring exponential time. In *Proc. Theory of Computing*, pages 1–9. ACM, 1973.
30. Leander Tentrup. Non-prenex QBF solving using abstraction. In Nadia Creignou and Daniel Le Berre, editors, *Theory and Applications of Satisfiability Testing - SAT 2016*, volume 9710 of *Lecture Notes in Computer Science*, pages 393–401. Springer Verlag, 2016.
31. G. S. Tseitin. On the complexity of derivation in propositional calculus. *Zap. Nauchn. Sem. Leningrad Otd. Mat. Inst. Akad. Nauk SSSR*, 8:23–41, 1968. Russian. English translation in J. Siekmann and G. Wrightson (eds.) *Automation of Reasoning. Classical Papers on Computer Science 1967–1970*, Springer Verlag, 466–483, 1983.
32. Allen Van Gelder. Contributions to the theory of practical quantified boolean formula solving. In Michela Milano, editor, *Principles and Practice of Constraint Programming*, pages 647–663, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
33. Yakir Vizel, Georg Weissenbacher, and Sharad Malik. Boolean satisfiability solvers and their applications in model checking. *Proceedings of the IEEE*, 103(11):2021–2035, 2015.
34. Ingo Wegener. *Branching Programs and Binary Decision Diagrams.* SIAM, 2000.
35. Lintao Zhang and Sharad Malik. Conflict driven learning in a quantified Boolean satisfiability solver. In Lawrence T. Pileggi and Andreas Kuehlmann, editors, *Proceedings of the 2002 IEEE/ACM International Conference on Computer-aided Design, ICCAD 2002, San Jose, California, USA, November 10-14, 2002*, pages 442–449. ACM / IEEE Computer Society, 2002.