Electronic Colloquium on Computational Complexity, Report No. 59 (2015)

Feasible Interpolation for QBF Resolution Calculi

Olaf Beyersdorff[1], Leroy Chew[1], Meena Mahajan[2], and Anil Shukla[2]

[1] School of Computing, University of Leeds, United Kingdom.
{o.beyersdorff,mm12lnc}@leeds.ac.uk
[2] The Institute of Mathematical Sciences, Chennai, India.
{meena,anilsh}@imsc.res.in

Abstract. In sharp contrast to classical proof complexity we are currently short of lower bound techniques for QBF proof systems. In this paper we establish the feasible interpolation technique for all resolution-based QBF systems, whether modelling CDCL or expansion-based solving. This both provides the first general lower bound method for QBF proof systems as well as largely extends the scope of classical feasible interpolation. We apply our technique to obtain new exponential lower bounds to all resolution-based QBF systems for a new class of QBF formulas based on the clique problem. Finally, we show how feasible interpolation relates to the recently established lower bound method based on strategy extraction [8].

1 Introduction

The main aim in proof complexity is to understand the complexity of theorem proving. Arguably, what is even more important is to establish techniques for lower bounds, and the recent history of computational complexity speaks volumes on how difficult it is to develop general lower bound techniques. Understanding the size of proofs is important for at least two reasons. The first is its tight relation to the separation of complexity classes: NP vs. coNP for propositional proofs, and NP vs. PSPACE in the case of proof systems for quantified boolean formulas (QBF). New superpolynomial lower bounds for specific proof systems rule out specific classes of non-deterministic poly-time algorithms for problems in co-NP or PSPACE, thereby providing an orthogonal approach to the predominantly machine-oriented view of computational complexity.

The second reason to study lower bounds for proofs is the analysis of SAT and QBF solvers: powerful algorithms that efficiently solve the classically hard problems of SAT and QBF for large classes of practically relevant formulas. Modern SAT solvers routinely solve industrial instances in even millions of variables for various applications. Even though QBF solving is at a much earlier state, due to its greater expressivity, QBF even applies to further fields such as formal verification or planning [6, 14, 27]. Each successful run of a solver on an unsatisfiable instance can be interpreted as a proof of unsatisfiability; and modern SAT solvers based on conflict-driven clause learning (CDCL) are known to implicitly generate resolution proofs. Thereby, understanding the complexity of resolution proofs directly translates into sharp bounds for the performance of CDCL-based SAT solvers.

The picture is more complex for QBF solving, as there exist two main, yet conceptually very different paradigms: CDCL-based and expansion-based solving. A variety of QBF resolution systems have been designed to capture the power of QBF solvers based on these paradigms. The core system of these is Q-Resolution (Q-Res), introduced in [19]. This has been augmented to capture ideas from CDCL solving, leading to long-distance resolution (LD-Q-Res) [3], universal resolution (QU-Res) [28], or its combinations like LQU$^+$-Res [4]. Powerful proof systems for expansion-based solving were recently developed in the form of

ISSN 1433-8092

∀Exp+Res [18], and the stronger IR-calc and IRM-calc [7]. Latest findings show that CDCL and expansion are indeed orthogonal paradigms as the underlying proof systems from the two categories are incomparable with respect to simulations [8].

Understanding which general techniques can be used to show lower bounds for proof systems is of paramount importance in proof complexity. For propositional proof systems we have a number of very effective techniques, most notably the size-width technique [5], deriving size from width bounds, game characterisations (e.g. [10, 26]), the approach via proof-complexity generators (cf. [21]), and feasible interpolation. Feasible interpolation, first introduced by Krajíček [20], is a particularly successful paradigm that transfers circuit lower bounds to size of proof lower bounds. The technique has been shown to be effective for resolution [20], cutting planes [25] and even strong Frege systems for modal and intuitionistic logics [16]. However, feasible interpolation fails for strong propositional systems as Frege systems under plausible cryptographic and number-theoretic assumptions [11, 12, 22].

The situation is drastically different for QBF proof systems, where we currently possess a very limited bag of techniques. At present we only have the very recent strategy extraction technique [8], which works only for Q-Res, a game characterisation of the very weak tree-like Q-Res [9], and ad-hoc lower bound arguments for various systems [8, 19]. In addition, the recent paper [4] develops methods to lift some previous lower bounds from Q-Res to stronger systems.

Our contributions

1. A general lower bound technique. We show that the feasible interpolation technique applies to all resolution-type QBF proof systems, whether expansion or CDCL based. This provides the first truly general lower bound technique for QBF proof systems, and—at the same time—hugely extends the scope of the feasible interpolation method.

In a nutshell, feasible interpolation works for true implications $A(\boldsymbol{p}, \boldsymbol{q}) \to B(\boldsymbol{p}, \boldsymbol{r})$ (or, equivalently, false conjunctions $A(\boldsymbol{p}, \boldsymbol{q}) \wedge \neg B(\boldsymbol{p}, \boldsymbol{r})$), which by Craig's interpolation theorem [13] possess interpolants $C(\boldsymbol{p})$ in the common variables $\boldsymbol{p}$. Such interpolants, even though they exist, may not be of polynomial size [23]. However, it may be the case that we can always efficiently extract such interpolants from a proof of the implication in a particular proof system P, and in this case, the system P is said to admit feasible interpolation. If we know that a particular class of formulas does not admit small interpolants (either unconditional or under suitable assumptions), then there cannot exist small proofs of the formulas in the system P. Here we show that this feasible interpolation theorem holds for arbitrarily quantified formulas $A(\boldsymbol{p}, \boldsymbol{q})$ and $B(\boldsymbol{p}, \boldsymbol{r})$ above, when the common variables $\boldsymbol{p}$ are existentially quantified before all other variables.

2. New lower bounds for QBF systems. As our second contribution we exhibit new hard formulas for QBF resolution systems. It is fair to say that we are currently quite short of hard examples: research so far has mainly concentrated on formulas of Kleine Büning, Karpinski, and Flögel [19] and their modifications [4, 8], a principle from [17], and a class of parity formulas recently introduced in [8]. This again is in sharp contrast with classical proof complexity where a wealth of different combinatorial principles as well as random formulas are known to be hard for resolution.

Our new hard formulas are QBF contradictions formalising the easy and appealing fact that a graph cannot both have and not have a k-clique. The trick is that in our formulation, each interpolant for these formulas has to solve the k-clique problem. Using our interpolation

theorem together with the exponential lower bound for the monotone circuit complexity of clique [1], we obtain exponential lower bounds for the clique-co-clique formulas in all CDCL and expansion-based QBF resolution systems.

We remark that conceptually our clique-co-clique formulas are different from and indeed simpler than the clique-colour formulas used for the interpolation technique in classical proof systems. This is due to the greater expressibility of QBF. Indeed it is not clear how the clique-co-clique principle could even be formulated succinctly in propositional logic.

3. Comparison to strategy extraction. On a conceptual level, we uncover a tight relationship between feasible interpolation and strategy extraction. Strategy extraction is a very desirable property of QBF proof systems and is known to hold for the main resolution-based systems. From a refutation of a false QBF, a winning strategy for the universal player can be efficiently extracted.

Like feasible interpolation, the lower bound technique based on strategy extraction from [8] also transfers circuit lower bounds to proof size bounds. However, instead of monotone circuit bounds as in the case of feasible interpolation, the strategy extraction technique imports AC^0 circuit lower bounds. Here we show that each feasible interpolation problem can be transformed into a strategy extraction problem, where the interpolant corresponds to the winning strategy of the universal player on the first universal variable. This clarifies that indeed feasible interpolation can be viewed as a special case of strategy extraction.

Organisation of the paper

The remaining part of the paper is organised as follows. In Section 2 we review the definitions and relations of relevant QBF proof systems. In Section 3 we start by recalling the overall idea for feasible interpolation and show interpolation theorems for the strongest CDCL-based system $\mathsf{LQU}^+\text{-}\mathsf{Res}$ as well as the strongest expansion-based proof system $\mathsf{IRM\text{-}calc}$. This implies feasible interpolation for all QBF resolution-based systems. Further we show that all these systems even admit monotone feasible interpolation. In Section 4 we obtain the new lower bounds for the clique-co-clique formulas. Section 5 reformulates interpolation as a strategy extraction problem.

2 Preliminaries

A literal is a boolean variable or its negation. We say a literal x is complementary to the literal $\neg x$ and vice versa. A *clause* is a disjunction of literals and a *term* is a conjunction of literals. The empty clause is denoted by $\Box$, and is semantically equivalent to false. A formula in *conjunctive normal form* (CNF) is a conjunction of clauses. For a literal $l = x$ or $l = \neg x$, we write $\mathrm{var}(l)$ for x and extend this notation to $\mathrm{var}(C)$ for a clause C. Let α be any partial assignment. For a clause C, we write $C|_\alpha$ for the clause obtained after applying the partial assignment α to C.

Quantified Boolean Formulas (QBFs) extend propositional logic with boolean quantifiers with the standard semantics that $\forall x.F$ is satisfied by the same truth assignments as $F|_{x=0} \wedge F|_{x=1}$ and $\exists x.F$ as $F|_{x=0} \vee F|_{x=1}$. We assume that QBFs are in *closed prenex form* with a CNF matrix, i.e, we consider the form $\mathcal{Q}_1 X_1 \dots \mathcal{Q}_k X_k.\phi$, where $\mathcal{Q}_i \in \{\exists, \forall\}$, $\mathcal{Q}_i \neq \mathcal{Q}_{i+1}$, and X_i are pairwise disjoint sets of variables. The formula ϕ is in CNF and is defined only on variables $X_1 \cup \ldots \cup X_k$. The propositional part ϕ is called the *matrix* and the rest the *prefix*. If $x \in X_i$, we say that x is at *level* i and write $\mathrm{lv}(x) = i$; we write $\mathrm{lv}(l)$ for $\mathrm{lv}(\mathrm{var}(l))$. The

index $\mathrm{ind}(x)$ provides the more detailed information on the actual position of x in the prefix, i.e. all variables are indexed by $1, \ldots, n$ from left to right.

Often it is useful to think of a QBF $\mathcal{Q}_1 X_1 \ldots \mathcal{Q}_k X_k . \phi$ as a *game* between the *universal* and the *existential player*. In the i-th step of the game, the player $\mathcal{Q}_i$ assigns values to all the variables X_i. The existential player wins the game iff the matrix ϕ evaluates to 1 under the assignment constructed in the game. The universal player wins iff the matrix ϕ evaluates to 0. Given a universal variable u with index i, a *strategy for* u is a function from all variables of index $< i$ to $\{0, 1\}$. A QBF is false iff there exists a *winning strategy* for the universal player, i.e. if the universal player has a strategy for all universal variables that wins any possible game ([15], [2, Sec. 4.2.2], [24, Chap. 19]).

Resolution-based calculi for QBF. We now give a brief overview of the main existing resolution-based calculi for QBF. We start by describing the proof systems modelling *CDCL-based QBF solving*; their rules are summarized in Figure 1. The most basic and important system is *Q-resolution (Q-Res)* by Kleine Büning et al. [19]. It is a resolution-like calculus that operates on QBFs in prenex form with CNF matrix. In addition to the axioms, Q-Res comprises the resolution rule S∃R and universal reduction ∀-Red (cf. Fig. 1).

$$\frac{}{C}\ (\text{Ax}) \qquad \frac{D \cup \{u\}}{D}\ (\forall\text{-Red}) \qquad \frac{D \cup \{u^*\}}{D}\ (\forall\text{-Red}^*)$$

C is a clause in the matrix. Literal u is universal and $\mathrm{lv}(u) \geq \mathrm{lv}(l)$ for all $l \in D$.

$$\frac{C_1 \cup U_1 \cup \{x\} \qquad C_2 \cup U_2 \cup \{\neg x\}}{C_1 \cup C_2 \cup U}\ (\text{Res})$$

We consider four instantiations of the Res-rule:
S∃R: x is existential.
If $z \in C_1$, then $\neg z \notin C_2$. $U_1 = U_2 = U = \emptyset$.
S∀R: x is universal. Other conditions same as S∃R.
L∃R: x is existential.
If $l_1 \in C_1, l_2 \in C_2$, $\mathrm{var}(l_1) = \mathrm{var}(l_2) = z$ then $l_1 = l_2 \neq z^*$. U_1, U_2 contain only universal literals with $\mathrm{var}(U_1) = \mathrm{var}(U_2)$. $\mathrm{ind}(x) < \mathrm{ind}(u)$ for each $u \in \mathrm{var}(U_1)$.
If $w_1 \in U_1, w_2 \in U_2$, $\mathrm{var}(w_1) = \mathrm{var}(w_2) = u$ then $w_1 = \neg w_2$, $w_1 = u^*$ or $w_2 = u^*$. $U = \{u^* \mid u \in \mathrm{var}(U_1)\}$.
L∀R: x is universal. Other conditions same as L∃R.

Fig. 1. The rules of CDCL-based proof systems

Long-distance resolution (LD-Q-Res) appears originally in the work of Zhang and Malik [29] and was formalized into a calculus by Balabanov and Jiang [3]. It merges complementary literals of a universal variable u into the special literal u^*. LD-Q-Res uses the rules L∃R, ∀-Red and ∀-Red* (cf. Fig. 1).

QU-resolution (QU-Res) [28] removes the restriction from Q-Res that the resolved variable must be an existential variable and allows resolution of universal variables. The rules of QU-Res are S∃R, S∀R and ∀-Red. *LQU⁺-Res* [4] extends LD-Q-Res by allowing short and long distance resolution pivots to be universal; however, the pivot is never a merged literal z^*. LQU⁺-Res uses the rules L∃R, L∀R, ∀-Red and ∀-Red*.

The second type of calculi models *expansion-based QBF solving*. These calculi are based on *instantiation* of universal variables: ∀Exp+Res [18], IR-calc, and IRM-calc [7]. All these calculi operate on clauses that comprise only existential variables from the original QBF, which are additionally *annotated* by a substitution to some universal variables, e.g. $\neg x^{0/u_1 1/u_2}$. For any annotated literal l^σ, the substitution σ must not make assignments to variables at a higher quantification level than l, i.e. if $u \in \mathsf{dom}(\sigma)$, then u is universal and $\mathrm{lv}(u) < \mathrm{lv}(l)$. To

preserve this invariant, we use the *auxiliary notation* $l^{[\sigma]}$, which for an existential literal l and an assignment σ to the universal variables filters out all assignments that are not permitted, i.e. $l^{[\sigma]} = l^{\{c/u \in \sigma \mid \mathrm{lv}(u) < \mathrm{lv}(l)\}}$.

$$\frac{}{\left\{x^{[\tau]} \mid x \in C, x \text{ is exist.}\right\}} \text{ (Ax)}$$

C is a non-tautological clause from the matrix.
$\tau = \{0/u \mid u \text{ is universal in } C\}$, where the notation $0/u$ for literals u is shorthand for $0/x$ if $u = x$ and $1/x$ if $u = \neg x$.

$$\frac{x^{\tau} \vee C_1 \qquad \neg x^{\tau} \vee C_2}{C_1 \cup C_2} \text{ (Res)}$$

$$\frac{C}{\mathsf{inst}(\tau, C)} \text{ (Instantiation)}$$

τ is an assignment to universal variables with $\mathsf{rng}(\tau) \subseteq \{0, 1\}$.

Fig. 2. The rules of IR-calc [7]

Axiom and instantiation rules as in IR-calc in Figure 2.

$$\frac{x^{\tau \cup \xi} \vee C_1 \qquad \neg x^{\tau \cup \sigma} \vee C_2}{\mathsf{inst}(\sigma, C_1) \cup \mathsf{inst}(\xi, C_2)} \text{ (Res)}$$

$\mathsf{dom}(\tau)$, $\mathsf{dom}(\xi)$ and $\mathsf{dom}(\sigma)$ are mutually disjoint.
$\mathsf{rng}(\tau) = \{0, 1\}$

$$\frac{C \vee b^{\mu} \vee b^{\sigma}}{C \vee b^{\xi}} \text{ (Merging)}$$

$\mathsf{dom}(\mu) = \mathsf{dom}(\sigma)$.
$\xi = \{c/u \mid c/u \in \mu, c/u \in \sigma\} \cup$
$\{*/u \mid c/u \in \mu, d/u \in \sigma, c \neq d\}$

Fig. 3. The rules of IRM-calc [7]

As annotations can be partial assignments, we use auxiliary operations of *completion* and *instantiation*. For assignments τ and μ, we write $\tau \veebar \mu$ for the assignment σ defined as follows: $\sigma(x) = \tau(x)$ if $x \in \mathsf{dom}(\tau)$, otherwise $\sigma(x) = \mu(x)$ if $x \in \mathsf{dom}(\mu)$. The operation $\tau \veebar \mu$ is called *completion* because μ provides values for variables not defined in τ. The operation is associative and therefore we can omit parentheses. For an assignment τ and an annotated clause C, the function $\mathsf{inst}(\tau, C)$ returns the annotated clause $\left\{l^{[\sigma \veebar \tau]} \mid l^{\sigma} \in C\right\}$. The system IR-calc is defined in Fig. 2.

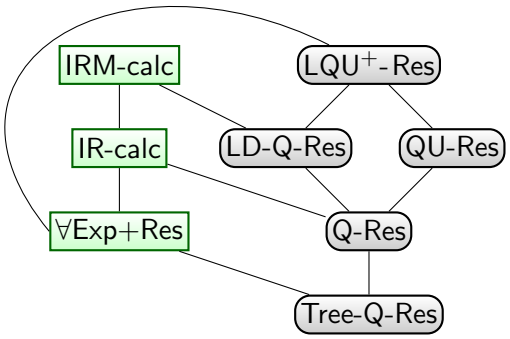

Fig. 4. The simulation order of QBF resolution systems. Systems on the left correspond to expansion-based solving, whereas the systems on the right are CDCL based.

The calculus IRM-calc further extends IR-calc by enabling annotations containing $*$. The rules of the calculus IRM-calc are presented in Fig. 3. The symbol $*$ may be introduced by the merge rule, e.g. by collapsing $x^{0/u} \vee x^{1/u}$ into $x^{*/u}$.

The simulation order of QBF resolution systems is shown in Figure 4. All proof systems have been exponentially separated (cf. [8]).

Definition 1. *For clauses C, D we write $C \preceq D$ if for any literal $l \in C$ we have $l \in D$ or $l^* \in D$ and for any $l^* \in C$ we have $l^* \in D$.*

For annotations τ and σ we say that $\tau \preceq \sigma$ if $\mathsf{dom}(\tau) = \mathsf{dom}(\sigma)$ and for any $c/u \in \tau$ we have $c/u \in \sigma$ or $*/u \in \sigma$ and for any $*/u \in \tau$ we have $*/u \in \sigma$. If C, D are annotated clauses, we write $C \preceq D$ if there is an injective function $f : C \hookrightarrow D$ such that for all $l^{\tau} \in C$ we have $f(l^{\tau}) = l^{\sigma}$ with $\tau \preceq \sigma$.

3 Feasible Interpolation and Feasible Monotone Interpolation

In this section we show that feasible interpolation and feasible monotone interpolation hold for $\mathsf{LQU}^+\text{-}\mathsf{Res}$ and $\mathsf{IRM\text{-}calc}$. We adapt the technique first used by Pudlák [25] to re-prove and generalise the result of Krajíček [20].

3.1 The setting

Consider a false QBF sentence $\mathcal{F}$ of the form

$$\exists \boldsymbol{p} \mathcal{Q} \boldsymbol{q} \mathcal{Q} \boldsymbol{r} \big[A(\boldsymbol{p}, \boldsymbol{q}) \wedge B(\boldsymbol{p}, \boldsymbol{r}) \big],$$

where, $\boldsymbol{p}$, $\boldsymbol{q}$, and $\boldsymbol{r}$ are mutually disjoint sets of propositional variables, $A(\boldsymbol{p}, \boldsymbol{q})$ is a CNF formula on variables $\boldsymbol{p}$ and $\boldsymbol{q}$, and $B(\boldsymbol{p}, \boldsymbol{r})$ is a CNF formula on variables $\boldsymbol{p}$ and $\boldsymbol{r}$. Thus $\boldsymbol{p}$ are the common variables between them. The $\boldsymbol{q}$ and $\boldsymbol{r}$ variables can be quantified arbitrarily, with any number of quantification levels. The sentence is equivalent to the following, not in prenex form

$$\exists \boldsymbol{p} \big[\mathcal{Q} \boldsymbol{q}. A(\boldsymbol{p}, \boldsymbol{q}) \wedge \mathcal{Q} \boldsymbol{r}. B(\boldsymbol{p}, \boldsymbol{r}) \big].$$

Definition 2. *Let $\mathcal{F}$ be a false QBF of the form $\exists \boldsymbol{p} \mathcal{Q} \boldsymbol{q} \mathcal{Q} \boldsymbol{r}. [A(\boldsymbol{p}, \boldsymbol{q}) \wedge B(\boldsymbol{p}, \boldsymbol{r})]$. An* interpolation circuit *for $\mathcal{F}$ is a boolean circuit G such that on every $0,1$ assignment $\boldsymbol{a}$ for $\boldsymbol{p}$ we have*

$$G(\boldsymbol{a}) = 0 \implies \mathcal{Q} \boldsymbol{q}. A(\boldsymbol{a}, \boldsymbol{q}) \text{ is false, and}$$
$$G(\boldsymbol{a}) = 1 \implies \mathcal{Q} \boldsymbol{r}. B(\boldsymbol{a}, \boldsymbol{r}) \text{ is false.}$$

We say that a QBF proof system S has feasible interpolation *if for any S-proof π of a QBF $\mathcal{F}$ of the form above, we can extract from π an interpolation circuit for $\mathcal{F}$ of size polynomial in the size of π.*

We say that S has monotone feasible interpolation *if the following holds: in the same setting as above, if $\boldsymbol{p}$ appears only positively in $A(\boldsymbol{p}, \boldsymbol{q})$, then we can extract from π a monotone interpolation circuit for $\mathcal{F}$.*

As our main results, we show that both $\mathsf{LQU}^+\text{-}\mathsf{Res}$ and $\mathsf{IRM\text{-}calc}$ have monotone feasible interpolation.

Before proving the interpolation theorems, we first outline the general idea:

Proof idea. Fix a proof system $S \in \{\mathsf{LQU}^+\text{-}\mathsf{Res}, \mathsf{IRM\text{-}calc}\}$ and an S-proof π of $\mathcal{F}$. Consider the following definition of a $\boldsymbol{q}$-clause and an $\boldsymbol{r}$-clause.

Definition 3. *We call a clause C in π a $\boldsymbol{q}$-clause (resp. $\boldsymbol{r}$-clause), if C contains only variables $\boldsymbol{p}, \boldsymbol{q}$ (resp. $\boldsymbol{p}, \boldsymbol{r}$). We also call C a $\boldsymbol{q}$-clause (resp. $\boldsymbol{r}$-clause), if C contains only $\boldsymbol{p}$ variables, but all its descendant clauses in the proof π are $\boldsymbol{q}$ (resp. $\boldsymbol{r}$)-clauses. In the case of $\mathsf{IRM\text{-}calc}$ the annotations are not considered and can be from either set.*

From π we construct a circuit C_π with the $\boldsymbol{p}$-variables as inputs: For each node u with clause C_u in the proof π, associate a gate g_u (or a constant-size circuit) in the circuit C_π. We then construct, for any assignment $\boldsymbol{a}$ to the $\boldsymbol{p}$ variables, another proof-like structure $\pi'(\boldsymbol{a})$. For each node u with clause C_u in the proof π, associate a clause $C'_{u,\boldsymbol{a}}$ in the structure $\pi'(\boldsymbol{a})$. Finally,

we obtain $\pi''(\boldsymbol{a})$ from the structure $\pi'(\boldsymbol{a})$ by instantiating $\boldsymbol{p}$ variables to the assignment $\boldsymbol{a}$ and doing some pruning, and show that $\pi''(\boldsymbol{a})$ is a valid proof in S. We then find that if $C_\pi(\boldsymbol{a}) = 0$, then $\pi''(\boldsymbol{a})$ uses only $\boldsymbol{q}$-clauses and thus is a refutation of $\mathcal{Q}\boldsymbol{q}.A(\boldsymbol{a}, \boldsymbol{q})$, and if $C_\pi(\boldsymbol{a}) = 1$, then $\pi''(\boldsymbol{a})$ uses only $\boldsymbol{r}$-clauses and thus is a refutation of $\mathcal{Q}\boldsymbol{r}.B(\boldsymbol{a}, \boldsymbol{r})$. Thus C_π is the desired interpolant circuit.

More precisely, we show (by induction on the height of u in π) that:

1. $C'_{u,\boldsymbol{a}} \preceq C_u$.
2. $g_u(\boldsymbol{a}) = 0 \implies C''_{u,\boldsymbol{a}}$ is a $\boldsymbol{q}$-clause and can be obtained from the clauses of $A(\boldsymbol{a}, \boldsymbol{q})$ alone using the rules of S.
3. $g_u(\boldsymbol{a}) = 1 \implies C''_{u,\boldsymbol{a}}$ is an $\boldsymbol{r}$-clause and can be obtained from the clauses of $B(\boldsymbol{a}, \boldsymbol{r})$ alone using the rules of S.

From the above, we have the following conclusion. Let r be the root of π. Then on any assignment $\boldsymbol{a}$ to the $\boldsymbol{p}$ variables we have:

(1) $C'_{r,\boldsymbol{a}} \preceq C_r = \square$, so $C'_{r,\boldsymbol{a}} = \square$. Therefore, $C''_{r,\boldsymbol{a}} = C'_{r,\boldsymbol{a}}|_{\boldsymbol{a}} = \square$.
(2) $g_r(\boldsymbol{a}) = 0 \implies \square$ is a $\boldsymbol{q}$-clause and can be obtained from the clauses of $A(\boldsymbol{a}, \boldsymbol{q})$ alone using the rules of system S. Hence by soundness of S, $\mathcal{Q}\boldsymbol{q}.A(\boldsymbol{a}, \boldsymbol{q})$ is false.
(3) $g_r(\boldsymbol{a}) = 1 \implies \square$ is an $\boldsymbol{r}$-clause and can be obtained from the clauses of $B(\boldsymbol{a}, \boldsymbol{r})$ alone using the rules of system S. Hence by soundness of S, $\mathcal{Q}\boldsymbol{q}.B(\boldsymbol{a}, \boldsymbol{r})$ is false.

Thus g_r, the output gate of the circuit, computes an interpolant.

3.2 Interpolants from LQU$^+$-Res proofs

We now implement the idea described above for LQU$^+$-Res.

Theorem 4. *LQU$^+$-Res has feasible interpolation.*

Proof. As mentioned in the proof idea, for an LQU$^+$-Res proof π of $\mathcal{F}$ we first describe the circuit C_π with input $\boldsymbol{p}$.

Construction of the circuit C_π: The DAG underlying the circuit is exactly the same as the DAG underlying the proof π. For each node u with clause C_u in π we associate a gate g_u as follows:

u **is a leaf node:** If $C_u \in A(\boldsymbol{p}, \boldsymbol{q})$ then g_u is a constant 0 gate. If $C_u \in B(\boldsymbol{p}, \boldsymbol{r})$ then g_u is a constant 1 gate.

u **is an internal node:** We distinguish four cases.

1. u was derived by a universal reduction step. In this case put a no-operation gate for g_u.
2. u corresponds to a resolution step with an existential variable $x \in \boldsymbol{p}$ as pivot. Nodes v and w are its two children, i.e.

$$\dfrac{\overbrace{C_1 \vee x}^{\text{node } v} \qquad \overbrace{C_2 \vee \neg x}^{\text{node } w}}{\underbrace{C}_{\text{node } u}}$$

In this case, put a selector gate $\mathsf{sel}(x, g_v, g_w)$ for g_u. Here, $\mathsf{sel}(x, a, b) = a$, when $x = 0$ and $\mathsf{sel}(x, a, b) = b$, when $x = 1$. That is, $\mathsf{sel}(x, a, b) = (\neg x \wedge a) \vee (x \wedge b)$. Note that all the variables in $\boldsymbol{p}$ are existential variables without annotations.

3. u corresponds to a resolution step with an existential or universal variable $x \in \boldsymbol{q}$ as pivot. Put an OR gate for g_u.
4. u corresponds to a resolution step with an existential or universal variable $x \in \boldsymbol{r}$ as pivot. Put an AND gate for g_u.

This completes the description of the circuit C_π.

Construction of π' and π'': Following our proof idea, we now construct a proof-like structure $\pi'(\boldsymbol{a})$, which depends on the assignment $\boldsymbol{a}$ to the $\boldsymbol{p}$ variables, the proof π of $\mathcal{F}$, and the circuit C_π. For each node u in π with clause C_u, we associate a clause $C'_{u,\boldsymbol{a}}$ in $\pi'(\boldsymbol{a})$.

From the structure $\pi'(\boldsymbol{a})$, we get another structure $\pi''(\boldsymbol{a})$ by instantiating $\boldsymbol{p}$ variables by the assignment $\boldsymbol{a}$ in each clause of $\pi'(\boldsymbol{a})$, cutting away any edge out of a node where the clause evaluates to 1, and deleting nodes which now have no path to the root node. That is, for each survived node u in $\pi''(\boldsymbol{a})$, the associated clause $C''_{u,\boldsymbol{a}}$ is equal to $C'_{u,\boldsymbol{a}}|_{\boldsymbol{a}}$.

We show (by induction on the height of u in π) that:

1. $C'_{u,\boldsymbol{a}} \preceq C_u$.
2. $g_u(\boldsymbol{a}) = 0 \implies C''_{u,\boldsymbol{a}}$ is a $\boldsymbol{q}$-clause and can be obtained from the clauses of $A(\boldsymbol{a}, \boldsymbol{q})$ alone using the rules of system $\mathsf{LQU}^+\text{-}\mathsf{Res}$.
3. $g_u(\boldsymbol{a}) = 1 \implies C''_{u,\boldsymbol{a}}$ is a $\boldsymbol{r}$-clause and can be obtained from the clauses of $B(\boldsymbol{a}, \boldsymbol{r})$ alone using the rules of system $\mathsf{LQU}^+\text{-}\mathsf{Res}$.

As described in the proof outline, this suffices to conclude that C_π computes an interpolant. We now present the construction details.

At leaf level: Let node u be a leaf in π. Then $C'_{u,\boldsymbol{a}} = C_u$; that is, we copy the clause as it is. Trivially, we have $C'_{u,\boldsymbol{a}} \preceq C_u$. By construction of C_π, the conditions concerning $g_u(\boldsymbol{a})$ and $C''_{u,\boldsymbol{a}}$ are satisfied.

At an internal node we distinguish four cases based on the rule that was applied.

At an internal node with universal reduction: Let node u be an internal node in π corresponding to a universal reduction step on some universal variable x or x^*. Let node v be its only child. Here we consider only the case where the universal literal is x. The case of x^* is identical. We have

$$\frac{C_v = \overbrace{D_v \vee x}^{\text{node } v}}{C_u = \underbrace{D_v}_{\text{node } u}}, \qquad x \text{ is a universal variable, } \forall l \in D_v, \mathrm{lv}(l) < \mathrm{lv}(x).$$

In this case, define $C'_{u,\boldsymbol{a}} = C'_{v,\boldsymbol{a}} \setminus \{x, \neg x, x^*\}$. By induction, $C'_{v,\boldsymbol{a}} \preceq C_v = D_v \vee x$. Therefore, $C'_{u,\boldsymbol{a}} = C'_{v,\boldsymbol{a}} \setminus \{x, \neg x, x^*\} \preceq D_v = C_u$.

If $g_u(\boldsymbol{a}) = 0$, then we know that $g_v(\boldsymbol{a}) = 0$ as $g_u(\boldsymbol{a}) = g_v(\boldsymbol{a})$. By the induction hypothesis, we know that $C''_{v,\boldsymbol{a}} = C'_{v,\boldsymbol{a}}|_{\boldsymbol{a}}$ is a $\boldsymbol{q}$-clause and can be derived using $A(\boldsymbol{a}, \boldsymbol{q})$ alone via $\mathsf{LQU}^+\text{-}\mathsf{Res}$. Recall that $C'_{u,\boldsymbol{a}} = C'_{v,\boldsymbol{a}} \setminus \{x, \neg x, x^*\}$ in this case. Since $\boldsymbol{a}$ is an assignment to the $\boldsymbol{p}$ variables and $x \notin \boldsymbol{p}$, $C'_{u,\boldsymbol{a}}|_{\boldsymbol{a}} = C''_{u,\boldsymbol{a}}$ is a $\boldsymbol{q}$-clause and can be derived using $A(\boldsymbol{a}, \boldsymbol{q})$ alone via $\mathsf{LQU}^+\text{-}\mathsf{Res}$. (Either $C''_{u,\boldsymbol{a}}$ already equals $C''_{v,\boldsymbol{a}}$, or x needs to be dropped. In the latter case, the condition on $\mathrm{lv}(x)$ is satisfied at $C''_{u,\boldsymbol{a}}$ because it is satisfied at C_v in π and $C''_{v,\boldsymbol{a}} \preceq C_v$. So we can drop x from $C''_{v,\boldsymbol{a}}$ to get $C''_{u,\boldsymbol{a}}$.)

The situation is dual for the case when $g_u(\boldsymbol{a}) = 1$; we get $\boldsymbol{r}$-clauses.

At an internal node with $\boldsymbol{p}$-resolution: Let node u in the proof π correspond to a resolution step with pivot $x \in \boldsymbol{p}$. Note that x is existential, as $\boldsymbol{p}$ variables occur only existentially in $\mathcal{F}$. We have

$$\frac{C_v = \overbrace{C_1 \vee U_1 \vee x}^{\text{node } v} \qquad \overbrace{C_2 \vee U_2 \vee \neg x}^{\text{node } w} = C_w}{C_u = \underbrace{C_1 \vee C_2 \vee U}_{\text{node } u}}.$$

In the assignment $\boldsymbol{a}$, if $x = 0$, then define $C'_{u,\boldsymbol{a}} = C'_{v,\boldsymbol{a}} \setminus \{x\}$ and if $x = 1$ then define $C'_{u,\boldsymbol{a}} = C'_{w,\boldsymbol{a}} \setminus \{\neg x\}$. By induction, we have $C'_{v,\boldsymbol{a}} \preceq C_v$ and $C'_{w,\boldsymbol{a}} \preceq C_w$. So, if $x = 0$, we have $C'_{u,\boldsymbol{a}} = C'_{v,\boldsymbol{a}} \setminus \{x\} \preceq C_1 \vee U_1 \preceq C_u$. If $x = 1$, we have $C'_{u,\boldsymbol{a}} \preceq C'_{w,\boldsymbol{a}} \setminus \{\neg x\} \preceq C_2 \vee U_2 \preceq C_u$.

In this case g_u is a selector gate. If $x = 0$ in the assignment $\boldsymbol{a}$, then $g_u(\boldsymbol{a}) = g_v(\boldsymbol{a})$ and $C''_{u,\boldsymbol{a}} = C''_{v,(a)}$. Since the conditions concerning $g_v(\boldsymbol{a})$ and $C''_{v,\boldsymbol{a}}$ are satisfied by induction, the conditions concerning $g_u(\boldsymbol{a})$ and $C''_{u,\boldsymbol{a}}$ are satisfied as well. Similarly, if $x = 1$, then $g_u(\boldsymbol{a}) = g_w(\boldsymbol{a})$ and $C''_{u,\boldsymbol{a}} = C''_{w,\boldsymbol{a}}$, and the statements that are inductively true at w hold at u as well.

At an internal node with $\boldsymbol{q}$-resolution: Let node u in the proof π correspond to a resolution step with pivot $x \in \boldsymbol{q}$. Note that x may be existential or universal. We have

$$\frac{C_v = \overbrace{C_1 \vee U_1 \vee x}^{\text{node } v} \qquad \overbrace{C_2 \vee U_2 \vee \neg x}^{\text{node } w} = C_w}{C_u = \underbrace{C_1 \vee C_2 \vee U}_{\text{node } u}}, \quad x \in \boldsymbol{q}.$$

In this case, we use the value of gate g_u in circuit C_π on input $\boldsymbol{a}$.

If $g_v(\boldsymbol{a}) = 1$ then define $C'_{u,\boldsymbol{a}} = C'_{v,\boldsymbol{a}}$. By induction, we know that $C''_{u,\boldsymbol{a}} = C''_{v,\boldsymbol{a}}$ is an $\boldsymbol{r}$-clause. Since x is a $\boldsymbol{q}$-variable and is not instantiated by $\boldsymbol{a}$, it must be the case that $x \notin C'_{v,\boldsymbol{a}}$. Thus $C'_{u,\boldsymbol{a}} = C'_{v,\boldsymbol{a}} \preceq C_v \setminus \{x\} \preceq C_u$.

Else if $g_w(\boldsymbol{a}) = 1$, define $C'_{u,\boldsymbol{a}} = C'_{w,\boldsymbol{a}}$. By a similar analysis as above, $C'_{u,\boldsymbol{a}} = C'_{w,\boldsymbol{a}} \preceq C_w \setminus \{\neg x\} \preceq C_u$.

If $g_v(\boldsymbol{a}) = g_w(\boldsymbol{a}) = 0$, and if $x \notin C'_{v,\boldsymbol{a}}$, define $C'_{u,\boldsymbol{a}} = C'_{v,\boldsymbol{a}}$. Otherwise, if $\neg x \notin C'_{w,\boldsymbol{a}}$, define $C'_{u,\boldsymbol{a}} = C'_{w,\boldsymbol{a}}$. It follows from induction that $C'_{u,\boldsymbol{a}} \preceq C_u$.

Else, define $C'_{u,\boldsymbol{a}}$ to be the resolvent of $C'_{v,\boldsymbol{a}}$ and $C'_{w,\boldsymbol{a}}$ on x. By induction, we know that $C'_{v,\boldsymbol{a}} \setminus \{x\} \preceq C_1 \vee U_1$ and $C'_{w,\boldsymbol{a}} \setminus \{\neg x\} \preceq C_2 \vee U_2$. Hence $C'_{u,\boldsymbol{a}} \preceq C_1 \vee C_2 \vee U = C_u$.

We need to verify the conditions on $g_u(\boldsymbol{a})$ and $C''_{u,\boldsymbol{a}}$. The case when $g_u(\boldsymbol{a}) = 1$ is immediate, since $C''_{u,\boldsymbol{a}}$ copies a clause known by induction to be an $\boldsymbol{r}$-clause. So now consider the case when $g_u(\boldsymbol{a}) = 0$. By induction, we know that both $C''_{v,\boldsymbol{a}} = C'_{v,\boldsymbol{a}}|_{\boldsymbol{a}}$ and $C''_{w,\boldsymbol{a}} = C'_{w,\boldsymbol{a}}|_{\boldsymbol{a}}$ are $\boldsymbol{q}$-clauses and can be derived using $A(\boldsymbol{a}, \boldsymbol{q})$ alone via $\mathsf{LQU}^+\text{-}\mathsf{Res}$.

We have three cases. If $C'_{u,\boldsymbol{a}} = C'_{v,\boldsymbol{a}}$ or $C'_{u,\boldsymbol{a}} = C'_{w,\boldsymbol{a}}$, then by induction we are done. Otherwise, $C'_{u,\boldsymbol{a}}$ is obtained from $C'_{v,\boldsymbol{a}}$ and $C'_{w,\boldsymbol{a}}$ via a resolution step on pivot x. Since $\boldsymbol{a}$ is an assignment to the $\boldsymbol{p}$ variables and $x \notin \boldsymbol{p}$, $C''_{u,\boldsymbol{a}}$ can be derived from $C''_{v,\boldsymbol{a}}$ and $C''_{w,\boldsymbol{a}}$ via the same (LD)-resolution step.

Note: A simple observation is that $C'_{u,\boldsymbol{a}}$ is always a subset of C_u with only one exception, which is that some special symbol u^* in C_u may be converted into u in $C'_{u,\boldsymbol{a}}$. This leads us to define the relation $\preceq$. Also, the resolution step in $\pi''(\boldsymbol{a})$ is applicable in $\mathsf{LQU}^+\text{-}\mathsf{Res}$ because (1) every mergable universal variable in $C''_{v,\boldsymbol{a}}$ and $C''_{w,\boldsymbol{a}}$ was also mergable earlier in C_v and

C_w in π. (2) Every common non-mergable existential variable in $C''_{v,\boldsymbol{a}}$ and $C''_{w,\boldsymbol{a}}$ was also a non-mergable existential variable in C_v and C_w. (3) Every non-mergable universal variable in $C''_{v,\boldsymbol{a}}$ and $C''_{w,\boldsymbol{a}}$ was also a non-mergable universal pair in C_v and C_w. (4) The operations do not disturb the levels of variables, therefore if variable x satisfies the level condition in π it satisfies it in $\pi''(\boldsymbol{a})$ as well.

At an internal node with $\boldsymbol{r}$-resolution: Let node u in π correspond to a resolution step with pivot $x \in \boldsymbol{r}$. This is dual to the case above. □

3.3 Interpolants from IRM-calc proofs

We now establish the interpolation theorem for the expansion-based calculi, following the same overall idea described in Section 3.1.

Theorem 5. *IRM-calc has feasible interpolation.*

Proof. This proof closely follows that of Theorem 4, but with several changes in the proof details. We describe the changes here.

Construction of the circuit C_π: The circuit construction is very similar to that for LQU$^+$-Res. Leaves and resolution nodes are treated as before. Instantiation and merging nodes are treated as the universal reduction nodes were; that is, the corresponding gates are no-operation gates.

Construction of π' and π'': As before we construct a proof-like structure $\pi'(\boldsymbol{a})$, which depends on the assignment $\boldsymbol{a}$ to the $\boldsymbol{p}$ variables, the proof π of $\mathcal{F}$, and the circuit C_π. For each node u in π, with clause C_u, we associate a clause $C'_{u,\boldsymbol{a}}$ in $\pi'(\boldsymbol{a})$, and let $C''_{u,\boldsymbol{a}}$ be the instantiation of $C'_{u,\boldsymbol{a}}$ by the assignment $\boldsymbol{a}$. We show (by induction on the height of u in π) that:

1. $C'_{u,\boldsymbol{a}} \preceq C_u$.
2. $g_u(\boldsymbol{a}) = 0 \implies C''_{u,\boldsymbol{a}}$ is a $\boldsymbol{q}$-clause and can be obtained from the clauses of $A(\boldsymbol{a}, \boldsymbol{q})$ alone using the rules of system IRM-calc.
3. $g_u(\boldsymbol{a}) = 1 \implies C''_{u,\boldsymbol{a}}$ is a $\boldsymbol{r}$-clause and can be obtained from the clauses of $B(\boldsymbol{a}, \boldsymbol{r})$ alone using the rules of system IRM-calc.

Once again, as described in the proof outline, this suffices to conclude that the circuit C_π computes an interpolant.

Recall that for annotated clauses, the meaning of $\preceq$ is slightly different and is given in Definition 2.

At a leaf level: Let node u be a leaf in π. Then $C'_{u,\boldsymbol{a}} = C_u$; that is, copy the clause as it is. Trivially, $C'_{u,\boldsymbol{a}} \preceq C_u$. The gates give the correct values by definition.

At an internal node with instantiation: Let node u be an internal node in π corresponding to an instantiation step by τ. And let node v be its only child. We know $C_u = \mathsf{inst}(\tau, C_v)$.

Suppose $l^{\sigma'} \in \mathsf{inst}(\tau, C'_{v,\boldsymbol{a}})$. Then for some ξ', $l^{\xi'} \in C'_{v,\boldsymbol{a}}$, and $l^{\sigma'} = l^{[\xi' \veebar \tau]}$; hence σ' is a subset of ξ' completed with τ. By induction we know that $C'_{v,\boldsymbol{a}} \preceq C_v$. We have an injective function $f : C'_{v,\boldsymbol{a}} \hookrightarrow C_v$ that demonstrates this. Let $f(l^{\xi'}) = l^{\xi}$. Hence $l^{\xi} \in C_v$ for some $\xi' \preceq \xi$. So $l^{\sigma} = l^{[\xi \veebar \tau]} \in C_u$. Since the annotations introduced by instantiation match, $\sigma' \preceq \sigma$. We use this to define a function $g : \mathsf{inst}(\tau, C'_{v,\boldsymbol{a}}) \to C_u$ where $g(l^{\sigma'}) = l^{\sigma}$. Now we find any l^{τ_1}, l^{τ_2} where $g(l^{\tau_1}) = g(l^{\tau_2}) = l^{\tau}$ and perform a merging step on l^{τ_1} and l^{τ_2}; note that the

resulting literal $l^{\tau'}$ will still satisfy $\tau' \preceq \tau$. Eventually we get a clause which we define as $\mathsf{instmerge}(\tau, C'_{v,\boldsymbol{a}}, C_u) = C'_{u,\boldsymbol{a}}$ where this function is injective. We will use this notation to refer to this process of instantiation then deliberate merging to get $\preceq C_u$.

Therefore $C'_{u,\boldsymbol{a}} \preceq C_u$.

If the node u is not pruned out in $\pi''(\boldsymbol{a})$, then $C''_{u,\boldsymbol{a}}$ contains no satisfied $\boldsymbol{p}$ literals; hence neither does $C'_{v,\boldsymbol{a}}$. Therefore $C''_{u,\boldsymbol{a}}$ is derived from $C''_{v,\boldsymbol{a}}$; this is a valid step in the proof system.

Because we only use instantiation and merging or a dummy step, $C''_{u,\boldsymbol{a}}$ is a $\boldsymbol{q}$-clause if and only if $C''_{v,\boldsymbol{a}}$ is a $\boldsymbol{q}$-clause. Therefore the no-operation gate g_u gives a valid result by induction.

At an internal node with merging: Let node u be an internal node in π corresponding to a merging step. Let node v be its only child. We have

$$\frac{C_v = D_v \vee b^{\mu} \vee b^{\sigma}}{C_u = D_v \vee b^{\xi}}$$

where $\mathsf{dom}(\mu) = \mathsf{dom}(\sigma)$ and ξ is obtained by merging the annotations μ, σ. That is, $\xi = \mathsf{AMerge}(\mu, \sigma) \triangleq \{c/u \mid c/u \in \mu, c/u \in \sigma\} \cup \{*/u \mid c/u \in \mu, d/u \in \sigma, c \neq d\}$. Note that $\mu, \sigma \preceq \mathsf{AMerge}(\mu, \sigma)$.

Note that from the induction hypothesis, $C'_{v(\boldsymbol{a})} \preceq C_v$, so there is an injective function $f : C'_{v(\boldsymbol{a})} \hookrightarrow C_v$. Suppose $C'_{v(\boldsymbol{a})}$ contains two distinct literals $b^{\mu'}$ and $b^{\sigma'}$ where $f(b^{\mu'}) = b^{\mu}$ and $f(b^{\sigma'}) = b^{\sigma}$. So $C'_{v(\boldsymbol{a})} = D'_v \vee b^{\mu'} \vee b^{\sigma'}$. Then let $C'_{v,\boldsymbol{a}} = D'_v \vee b^{\xi'}$, where $\xi' = \mathsf{AMerge}(\mu', \sigma')$. Otherwise let $C'_{u,\boldsymbol{a}} = C'_{v,\boldsymbol{a}}$.

We first observe whenever we do actual merging, if $c/u \in \xi'$ then one of the following holds:

1. $c/u \in \sigma'$. Then $c/u \in \sigma$ or $*/u \in \sigma$, and so $c/u \in \xi$ or $*/u \in \xi$.
2. $c/u \in \mu'$. Then $c/u \in \mu$ or $*/u \in \mu$, and so $c/u \in \xi$ or $*/u \in \xi$.
3. $e/u \in \mu'$, $d/u \in \sigma'$, $e \neq d$, in which case $*/u \in \xi$.

Since all other annotated literals are unaffected, $C'_{u,\boldsymbol{a}} \preceq C_u$. We never merge $\boldsymbol{p}$ literals as they have no annotations, so if $C''_{u,\boldsymbol{a}}$ is not pruned away, then $C''_{u,\boldsymbol{a}}$ is derived from $C''_{v,\boldsymbol{a}}$ via merging.

In case we do not merge, there might be some $b^{\sigma'} \in C'_{v,\boldsymbol{a}}$ with $\sigma' \preceq \sigma$, which is not removed by merging. However $\sigma' \preceq \sigma \preceq \xi$, so $C'_{u,\boldsymbol{a}} = C'_{v,\boldsymbol{a}} \preceq C_u$. As $C''_{u,\boldsymbol{a}} = C''_{v,\boldsymbol{a}}$, this is a valid inference step (in fact, a dummy step).

Because we only use merging or a dummy step, $C''_{u,\boldsymbol{a}}$ is a $\boldsymbol{q}$-clause if and only if $C''_{v,\boldsymbol{a}}$ is a $\boldsymbol{q}$-clause, therefore the no-operation gate g_u gives a valid result by induction.

At an internal node with $\boldsymbol{p}$-resolution: We do not have any annotations on $\boldsymbol{p}$-literals. So in this case we construct C'_u and C''_u exactly as we would for an LQU^+-Res proof.

At an internal node with $\boldsymbol{q}$-resolution: When we have a resolution step between nodes v and w on a $\boldsymbol{q}$ pivot to get node u, we have

$$\frac{C_v = x^{\tau \cup \xi} \vee D_v \qquad C_w = \neg x^{\tau \cup \sigma} \vee D_w}{C_u = \mathsf{inst}(\sigma, D_v) \cup \mathsf{inst}(\xi, D_w)}$$

where $\mathsf{dom}(\tau)$, $\mathsf{dom}(\xi)$ and $\mathsf{dom}(\sigma)$ are mutually disjoint, and $\mathsf{rng}(\tau) = \{0, 1\}$.

In order to do dummy instantiations we will need to define a $\{0, 1\}$ version of ξ and σ. So we define $\xi' = \{c/u \mid c/u \in \xi, c \in \{0, 1\}\} \cup \{0/u \mid */u \in \xi\}$, $\sigma' = \{c/u \mid c/u \in \sigma, c \in \{0, 1\}\} \cup \{0/u \mid */u \in \sigma\}$. This gives us the desirable property that $\xi' \preceq \xi$, $\sigma' \preceq \sigma$.

Now resuming the construction of C', we use information from the circuit to construct this. If $g_v(\boldsymbol{a}) = 1$, then we define $C'_{u,\boldsymbol{a}} = \mathsf{instmerge}(\sigma', C'_{v,\boldsymbol{a}}, C_u)$. Otherwise, if $g_w(\boldsymbol{a}) = 1$, then we define $C'_{u,\boldsymbol{a}} = \mathsf{instmerge}(\xi', C'_{w,\boldsymbol{a}}, C_u)$. In these cases, we know by the inductive claim that $C'_{u,\boldsymbol{a}}$ does not contain any $\boldsymbol{q}$ literals. Therefore $C'_{u,\boldsymbol{a}}$ is the correct instantiation (as $\xi' \preceq \xi$, $\sigma' \preceq \sigma$) of some subset of D_v or D_w. Hence $C'_{u,\boldsymbol{a}} \preceq C_u$. Furthermore since g_u is an OR gate evaluating to 1 and since $C''_{u,\boldsymbol{a}}$, an $\boldsymbol{r}$-clause, can be obtained by an instantiation step, our inductive claim is true.

Now suppose $g_v(\boldsymbol{a}) = 0$ and $g_w(\boldsymbol{a}) = 0$. If there is no $x^\mu \in C'_{v,\boldsymbol{a}}$ such that $\mu \preceq \tau \cup \xi$, then define $C'_{u,\boldsymbol{a}} = \mathsf{instmerge}(\sigma', C'_{v,\boldsymbol{a},C_u})$. Else, if there is no $\neg x^\mu \in C'_{w,\boldsymbol{a}}$ such that $\mu \preceq \tau \cup \sigma$, then define $C'_{u,\boldsymbol{a}} = \mathsf{instmerge}(\xi', C'_{w,\boldsymbol{a},C_u})$. In these cases we know that $C'_{u,\boldsymbol{a}}$ is the correct instantiation (as $\xi' \preceq \xi$, $\sigma' \preceq \sigma$) of some subset of D_v or D_w; hence $C'_{u,\boldsymbol{a}} \preceq C_u$. Furthermore, since g_u is an OR gate evaluating to 0, and since $C''_{u,\boldsymbol{a}}$, a $\boldsymbol{q}$-clause, can be obtained by an instantiation step, our inductive claim is true.

The final case is when $g_v(\boldsymbol{a}) = g_w(\boldsymbol{a}) = 0$ and $x^{\tau\cup\xi_1} \in C'_{v,\boldsymbol{a}}$ for some $\xi_1 \preceq \xi$ and $\neg x^{\tau\cup\sigma_1} \in C'_{w,\boldsymbol{a}}$ for some $\sigma_1 \preceq \sigma$. Here, because $\mathsf{dom}(\tau)$, $\mathsf{dom}(\xi)$ and $\mathsf{dom}(\sigma)$ are mutually disjoint, $\mathsf{dom}(\tau)$, $\mathsf{dom}(\xi_1)$ and $\mathsf{dom}(\sigma_1)$ are also mutually disjoint. Thus we can do the resolution step

$$\frac{C'_{v,\boldsymbol{a}} = x^{\tau\cup\xi_1} \vee D'_v \qquad C'_{w,\boldsymbol{a}} = \neg x^{\tau\cup\sigma_1} \vee D'_w}{\mathsf{inst}(\sigma_1, D'_v) \cup \mathsf{inst}(\xi_1, D'_w)}.$$

Since $\mathsf{instmerge}(\sigma_1, D'_v, C_u) \preceq \mathsf{inst}(\sigma, D_v)$ and $\mathsf{instmerge}(\xi_1, D'_w, C_u) \preceq \mathsf{inst}(\xi, D_w)$, we can follow up $\mathsf{inst}(\sigma_1, D'_v) \cup \mathsf{inst}(\xi_1, D'_w)$ with sufficient merging steps to get a clause $C' \preceq C_u$; we define this clause to be the clause $C'_{u,\boldsymbol{a}}$. By the inductive claim, both $C''_{v,\boldsymbol{a}}$ and $C''_{w,\boldsymbol{a}}$ are $\boldsymbol{q}$-clauses; hence $C''_{u,\boldsymbol{a}}$ is also a $\boldsymbol{q}$-clause and is obtained via a valid resolution step.

At an internal node with $\boldsymbol{r}$-resolution: When we have a resolution step between nodes u and v on an $\boldsymbol{r}$-literal, this is the dual of the previous case. □

3.4 Monotone Interpolation

To transfer known circuit lower bounds into size of proof bounds, we need a monotone version of the previous interpolation theorems, which we prove next.

Theorem 6. *LQU$^+$-Res and IRM-calc have monotone feasible interpolation.*

Proof. In previous subsections, we have shown that the circuit $C_\pi(\boldsymbol{p})$ is a correct interpolant for the QBF sentence $\mathcal{F}$. That is, if $C_\pi(\boldsymbol{p}) = 0$ then $\mathcal{Q}\boldsymbol{q}.A(\boldsymbol{a}, \boldsymbol{q})$ is false, and if $C_\pi(\boldsymbol{p}) = 1$ then $\mathcal{Q}\boldsymbol{r}.B(\boldsymbol{a}, \boldsymbol{r})$ is false.

However, if $\boldsymbol{p}$ occurs only positively in $A(\boldsymbol{p}, \boldsymbol{q})$ then we construct a monotone circuit $C^{mon}_\pi(\boldsymbol{p})$ such that, on every $0, 1$ assignment $\boldsymbol{a}$ to $\boldsymbol{p}$ we have

$$\begin{aligned} C^{mon}_\pi(\boldsymbol{a}) = 0 &\implies \mathcal{Q}\boldsymbol{q}.A(\boldsymbol{a}, \boldsymbol{q}) \text{ is false, and} \\ C^{mon}_\pi(\boldsymbol{a}) = 1 &\implies \mathcal{Q}\boldsymbol{r}.B(\boldsymbol{a}, \boldsymbol{r}) \text{ is false.} \end{aligned}$$

We obtain $C^{mon}_\pi(\boldsymbol{p})$ from $C_\pi(\boldsymbol{p})$ by replacing all selector gates $g_u = \mathsf{sel}(x, g_v, g_w)$ by the following monotone ternary connective: $g_u = (x \vee g_v) \wedge g_w$ where nodes v and w are the children of u in π. We also change the proof-like structure $\pi'(\boldsymbol{a})$; the construction is the same as before except that at $\boldsymbol{p}$-resolution nodes, the rule for fixing $C'_{u,\boldsymbol{a}}$ is also changed to reflect the monotone function used instead.

More precisely, the functions $\mathsf{sel}(x, g_v, g_w)$ and $g_u = (x \vee g_v) \wedge g_w$ differ only when $x = 0$, $g_v(\boldsymbol{a}) = 1$, and $g_w(\boldsymbol{a}) = 0$. We set $C'_{u,\boldsymbol{a}}$ to $C'_{w,\boldsymbol{a}} \setminus \{\neg x\}$ if $x = 1$ or if $x = 0$, $g_v(\boldsymbol{a}) = 1$ and $g_w(\boldsymbol{a}) = 0$, and to $C'_{v,\boldsymbol{a}} \setminus \{x\}$ otherwise.

We need to show that at the differing setting, the inductive statements relating the modified $C'_{u,\boldsymbol{a}}$, $g_u(\boldsymbol{a})$ and $C''_{u,\boldsymbol{a}}$ continue to hold. The relation $C''_{u,\boldsymbol{a}} \preceq C_u$ holds by induction. Now consider the gate values.

We know by induction that $g_v(\boldsymbol{a}) = 1$ means that $C''_{v,\boldsymbol{a}}$ is an $\boldsymbol{r}$-clause and can be derived from $B(\boldsymbol{a}, \boldsymbol{r})$ alone. When $x = 0$, $C'_{u,\boldsymbol{a}} = C'_{v,\boldsymbol{a}}$ and the selector gate will output the value of $g_v(\boldsymbol{a})$ which is a 1. Hence $C''_{u,\boldsymbol{a}}$ is an $\boldsymbol{r}$-clause. However observe that at this setting, $g_w(\boldsymbol{a}) = 0$, which means by induction that $C''_{w,\boldsymbol{a}}$ is a $\boldsymbol{q}$-clause and can be derived using $A(\boldsymbol{a}, \boldsymbol{q})$ clauses alone via the appropriate proof system. Thus by our assumption about $\boldsymbol{p}$ variables appearing only positively in A, the clause $C'_{w,\boldsymbol{a}}$ does not contain $\neg x$. Thus we can safely assign $C'_{u,\boldsymbol{a}} = C'_{w,\boldsymbol{a}}$. This completes the proof. □

4 New Exponential Lower Bounds for IRM-calc and LQU$^+$-Res

We now apply our interpolation theorems to obtain new exponential lower bounds for a new class of QBFs. The lower bound will be directly transferred from the following monotone circuit lower bound for the problem Clique(n, k), asking whether a given graph with n nodes has a clique of size k.

Theorem 7 (Alon, Boppana 87 [1]). *All monotone circuits that compute* Clique$(n, n/2)$ *are of exponential size.*

We now build the QBF. Fix an integer n (indicating the number of vertices of the graph) and let $\boldsymbol{p}$ be the set of variables $\{p_{uv} \mid 1 \leq u < v \leq n\}$. An assignment to $\boldsymbol{p}$ picks a set of edges, and thus an n-vertex graph. Let $\boldsymbol{q}$ be the set of variables $\{q_{iu} \mid i \in [\frac{n}{2}], u \in [n]\}$. We use the following clauses.

$$\begin{array}{rll} C_i &= q_{i1} \vee \cdots \vee q_{in} & \text{for } i \in [\frac{n}{2}] \\ D_{i,j,u} &= \neg q_{iu} \vee \neg q_{ju} & \text{for } i, j \in [\frac{n}{2}], i < j \text{ and } u \in [n] \\ E_{i,u,v} &= \neg q_{iu} \vee \neg q_{iv} & \text{for } i \in [\frac{n}{2}] \text{ and } u, v \in [n], u < v \\ F_{i,j,u,v} &= \neg q_{iu} \vee \neg q_{jv} \vee p_{uv} & \text{for } i, j \in [\frac{n}{2}], i < j \text{ and } u \neq v \in [n]. \end{array}$$

We can now express Clique$(n, n/2)$ as a polynomial-size QBF $\exists \boldsymbol{q}.A_n(\boldsymbol{p}, \boldsymbol{q})$, where

$$A_n(\boldsymbol{p}, \boldsymbol{q}) = \bigwedge_{i \in [\frac{n}{2}]} C_i \wedge \bigwedge_{i<j, u \in [n]} D_{i,j,u} \wedge \bigwedge_{i \in [\frac{n}{2}], u<v} E_{i,u,v} \wedge \bigwedge_{i<j, u \neq v} F_{i,j,u,v}.$$

Here the edge variables $\boldsymbol{p}$ appear monotone in $A_n(\boldsymbol{p}, \boldsymbol{q})$.

Likewise co-Clique$(n, n/2)$ can be written as a polynomial-size QBF $\forall \boldsymbol{r_1} \exists \boldsymbol{r_2}.B_n(\boldsymbol{p}, \boldsymbol{r_1}, \boldsymbol{r_2})$. To construct this we use a polynomial-size circuit that checks whether the nodes specified by $\boldsymbol{r_1}$ fail to form a clique in the graph given by $\boldsymbol{p}$. We then use existential variables $\boldsymbol{r_2}$ for the gates of the circuit and can then form a CNF $B_n(\boldsymbol{p}, \boldsymbol{r_1}, \boldsymbol{r_2})$ that represents the circuit computation.

Now we can form a sequence of false QBFs, stating that the graph encoded in $\boldsymbol{p}$ both has a clique of size $n/2$ (as witnessed by $\boldsymbol{q}$) and likewise does not have such a clique as expressed in the B part:

$$\Phi_n(\boldsymbol{p}, \boldsymbol{q}, \boldsymbol{r}) = \exists \boldsymbol{p} \exists \boldsymbol{q} \forall \boldsymbol{r_1} \exists \boldsymbol{r_2}.A_n(\boldsymbol{p}, \boldsymbol{q}) \wedge B_n(\boldsymbol{p}, \boldsymbol{r_1}, \boldsymbol{r_2}).$$

This formula has the unique interpolant $\textsc{Clique}(n, n/2)(\boldsymbol{p})$. But since all monotone circuits for this are of exponential size by Theorem 7 and monotone circuits of size polynomial in IRM-calc and LQU$^+$-Res proofs can be extracted by Theorem 6, all such proofs must be of exponential size, yielding:

Theorem 8. *The QBFs $\Phi_n(\boldsymbol{p}, \boldsymbol{q}, \boldsymbol{r})$ require exponential-size proofs in IRM-calc and LQU$^+$-Res.*

5 Feasible Interpolation vs. Strategy Extraction

Recall the two player game semantics of a QBF; every false QBF has a winning strategy for the universal player, where the strategy for each variable depends only on the variables played before. We now explain the relation between strategy extraction — one of the main paradigms for QBF systems — and feasible interpolation. In Section 3 we studied QBFs of the form $\mathcal{F} = \exists \boldsymbol{p} \mathcal{Q} \boldsymbol{q} \mathcal{Q} \boldsymbol{r}. [A(\boldsymbol{p}, \boldsymbol{q}) \wedge B(\boldsymbol{p}, \boldsymbol{r})]$. If we add a common universal variable b we can change it to an equivalent QBF

$$\mathcal{F}^b = \exists \boldsymbol{p} \forall b \, \mathcal{Q} \boldsymbol{q} \, \mathcal{Q} \boldsymbol{r}. [(A(\boldsymbol{p}, \boldsymbol{q}) \vee b) \wedge (B(\boldsymbol{p}, \boldsymbol{r}) \vee \neg b)] .$$

If $\mathcal{F}$ is false, then also $\mathcal{F}^b$ is false and thus the universal player has a winning strategy, including a strategy for $b = \sigma(\boldsymbol{p})$ for the common universal variable b.

Remark 9. Every winning strategy $\sigma(\boldsymbol{p})$ for b is an interpolant for $\mathcal{F}$, i.e., for every $0, 1$ assignment $\boldsymbol{a}$ of $\boldsymbol{p}$ we have

$$\begin{aligned} \sigma(\boldsymbol{a}) = 0 &\implies \mathcal{Q}\boldsymbol{q}.A(\boldsymbol{a}, \boldsymbol{q}) \text{ is false, and} \\ \sigma(\boldsymbol{a}) = 1 &\implies \mathcal{Q}\boldsymbol{r}.B(\boldsymbol{a}, \boldsymbol{r}) \text{ is false.} \end{aligned}$$

Proof. Suppose not. Then there are two possibilities.

- There is some $\boldsymbol{a}$ where $\sigma(\boldsymbol{a}) = 0$ and $\mathcal{Q}\boldsymbol{q}.A(\boldsymbol{a}, \boldsymbol{q})$ is true. Then setting $b = 0$ would satisfy $\mathcal{Q}\boldsymbol{r}.B(\boldsymbol{p}, \boldsymbol{r}) \vee \neg b$. But $\mathcal{Q}\boldsymbol{q}.A(\boldsymbol{a}, \boldsymbol{q}) \vee b$ is also satisfied. Hence this cannot be part of the winning strategy for the universal player.
- There is some $\boldsymbol{a}$ where $\sigma(\boldsymbol{a}) = 1$ and $\mathcal{Q}\boldsymbol{r}.B(\boldsymbol{a}, \boldsymbol{r})$ is true. This is the dual of the above. □

Theorem 10. *1. From each LQU$^+$-Res refutation π of $\mathcal{F}^b$ we can extract in polynomial time a boolean circuit for $\sigma(\boldsymbol{p})$, i.e., the part of the winning strategy for variable b.*

2. If in the same setting as above for $\mathcal{F}^b$, the variables $\boldsymbol{p}$ appear only positively in $A(\boldsymbol{p}, \boldsymbol{q})$, then we can extract a monotone boolean circuit for $\sigma(\boldsymbol{p})$ from a LQU$^+$-Res refutation π of $\mathcal{F}^b$ in polynomial time (in the size of π).

Proof. As we can compute the (monotone) interpolant when b is absent, we use the same proof with a few modifications for the new formula.

We first change the definition of $\boldsymbol{q}$ and $\boldsymbol{r}$-clauses to allow for b and $\neg b$ literals.

Definition 11. *We call any clause in the proof a $\boldsymbol{q}$-clause (resp. $\boldsymbol{r}$-clause) if it contains only variables $\boldsymbol{p}, \boldsymbol{q}$ or literal b (resp. $\boldsymbol{p}, \boldsymbol{r}$ or literal $\neg b$). We retain the inheritance property for clauses only containing $\boldsymbol{p}$ variables.*

Construction of the circuit C_π: When constructing the circuit, we now also need to consider a resolution step on the common universal variable b:

$$\frac{C_v = \overbrace{C_1 \vee U_1 \vee b}^{\text{node } v} \qquad \overbrace{C_2 \vee U_2 \vee \neg b}^{\text{node } w} = C_w}{C_u = \underbrace{C_1 \vee C_2 \vee U}_{\text{node } u}}.$$

Here we can arbitrarily pick one of v or w. For example here we pick v and let $g_u = g_v$ disregarding the input from g_w.

Construction of π' and π'': We slightly modify the invariants to include the new definitions. Additionally we make a small change to the first invariant.

1. $C'_{u,\boldsymbol{a}} \backslash \{b, \neg b\} \preceq C_u$.
2. $g_u(\boldsymbol{a}) = 0 \implies C''_{u,\boldsymbol{a}}$ is a $\boldsymbol{q}$-clause and can be obtained from the clauses of $A(\boldsymbol{a}, \boldsymbol{q})$ alone using the rules of $\mathsf{LQU}^+\text{-}\mathsf{Res}$.
3. $g_u(\boldsymbol{a}) = 1 \implies C''_{u,\boldsymbol{a}}$ is a $\boldsymbol{r}$-clause and can be obtained from the clauses of $B(\boldsymbol{a}, \boldsymbol{r})$ alone using the rules of $\mathsf{LQU}^+\text{-}\mathsf{Res}$.

Notice also that $b^* \notin C''_{u,\boldsymbol{a}}$ as b^* can only arise from a long distance resolution step on a $\boldsymbol{p}$ variable but these are instantiated and so never occur as pivots in this proof assuming the induction hypothesis.

We observe that the base cases work for the construction of π' and π''. The only new part of the inductive step is when we have

$$\frac{C_v = \overbrace{C_1 \vee U_1 \vee b}^{\text{node } v} \qquad \overbrace{C_2 \vee U_2 \vee \neg b}^{\text{node } w} = C_w}{C_u = \underbrace{C_1 \vee C_2 \vee U}_{\text{node } u}}.$$

To find $C'_{u,\boldsymbol{a}}$ we look at our choice of wiring in the circuit construction. If g_u is wired to g_v ($g_u = g_v$) then we take $C'_{u,\boldsymbol{a}}$ to equal $C'_{v,\boldsymbol{a}}$. Since $C'_{v,\boldsymbol{a}} \backslash \{b, \neg b\} \preceq C_v \backslash \{b, \neg b\} \preceq C_u$ we get $C'_{u,\boldsymbol{a}} \backslash \{b, \neg b\} \preceq C_u$. Since our choice of the clause is determined by our choice of wiring, then we retain our invariants in that way.

Notice that because of this case we never resolve a $\boldsymbol{q}$-clause with a $\boldsymbol{r}$ clause in π' so $b, \neg b$ will always be retained in their respective type of clauses.

From the above, we have the following conclusion. Let r be the root of π. Then on any assignment $\boldsymbol{a}$ to the $\boldsymbol{p}$ variables we have:

(1) $C'_{r,\boldsymbol{a}} \backslash \{b, \neg b\} \preceq C_r = \square$. Therefore, $C''_{r,\boldsymbol{a}} \setminus \{b, \neg b\} = \square$. But $C''_{r,\boldsymbol{a}}$ can contain at most one of these literals, which can be universally reduced to complete a refutation.

(2) $g_r(\boldsymbol{a}) = 0 \implies C''_{r,\boldsymbol{a}}$ is a $\boldsymbol{q}$-clause and can be obtained from the clauses of $A(\boldsymbol{a}, \boldsymbol{q})$ alone using the rules of system $\mathsf{LQU}^+\text{-}\mathsf{Res}$. Hence by soundness of $\mathsf{LQU}^+\text{-}\mathsf{Res}$, $\mathcal{Q}\boldsymbol{q}.A(\boldsymbol{a}, \boldsymbol{q})$ is false.

(3) $g_r(\boldsymbol{a}) = 1 \implies C''_{r,\boldsymbol{a}}$ is an $\boldsymbol{r}$-clause and can be obtained from the clauses of $B(\boldsymbol{a}, \boldsymbol{r})$ alone using the rules of system $\mathsf{LQU}^+\text{-}\mathsf{Res}$. Hence by soundness of $\mathsf{LQU}^+\text{-}\mathsf{Res}$, $\mathcal{Q}\boldsymbol{q}.B(\boldsymbol{a}, \boldsymbol{r})$ is false.

Thus g_r, the output gate of the circuit, computes $\sigma(\boldsymbol{p})$. $\square$

An analogous result to Theorem 10 also holds for IRM-calc.

Theorem 12. *1. From each IRM-calc refutation π of $\mathcal{F}^b$ we can extract in polynomial time a boolean circuit for $\sigma(\boldsymbol{p})$, i.e., the part of the winning strategy for variable b.*
2. If in the same setting as above for $\mathcal{F}^b$, the variables $\boldsymbol{p}$ appear only positively in $A(\boldsymbol{p}, \boldsymbol{q})$, then we can extract a monotone boolean circuit for $\sigma(\boldsymbol{p})$ from a IRM-calc refutation π of $\mathcal{F}^b$ in polynomial time (in the size of π).

Proof. We can use exactly the same constructions as in Theorem 5. The b literals do not affect the argument. □

As a corollary, the versions $\Phi_n^b(\boldsymbol{p}, \boldsymbol{q}, \boldsymbol{r})$ of the formulas from Sect. 4 also require exponential-size proofs in IRM-calc and LQU$^+$-Res.

Acknowledgements

This work was supported by the EU Marie Curie IRSES grant CORCON, grant no. 48138 from the John Templeton Foundation, EPSRC grant EP/L024233/1, and a Doctoral Training Grant from EPSRC (2nd author).

We thank Pavel Pudlák and Mikoláš Janota for helpful discussions on the relation between feasible interpolation and strategy extraction during the recent Dagstuhl Seminar 'Optimal Agorithms and Proofs' (14421).

References

1. Noga Alon and Ravi B. Boppana. The monotone circuit complexity of boolean functions. *Combinatorica*, 7(1):1–22, 1987.
2. Sanjeev Arora and Boaz Barak. *Computational Complexity – A Modern Approach*. Cambridge University Press, 2009.
3. Valeriy Balabanov and Jie-Hong R. Jiang. Unified QBF certification and its applications. *Formal Methods in System Design*, 41(1):45–65, 2012.
4. Valeriy Balabanov, Magdalena Widl, and Jie-Hong R. Jiang. QBF resolution systems and their proof complexities. In *SAT*, pages 154–169, 2014.
5. Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow - resolution made simple. *Journal of the ACM*, 48(2):149–169, 2001.
6. Marco Benedetti and Hratch Mangassarian. QBF-based formal verification: Experience and perspectives. *JSAT*, 5(1-4):133–191, 2008.
7. Olaf Beyersdorff, Leroy Chew, and Mikoláš Janota. On unification of QBF resolution-based calculi. In *MFCS, II*, pages 81–93, 2014.
8. Olaf Beyersdorff, Leroy Chew, and Mikoláš Janota. Proof complexity of resolution-based QBF calculi. In *STACS*. LIPIcs series, 2015.
9. Olaf Beyersdorff, Leroy Chew, and Karteek Sreenivasaiah. A game characterisation of tree-like Q-resolution size. In *LATA*. Springer, 2015.
10. Olaf Beyersdorff and Oliver Kullmann. Unified characterisations of resolution hardness measures. In *SAT*, pages 170–187, 2014.
11. Maria Luisa Bonet, Carlos Domingo, Ricard Gavaldà, Alexis Maciel, and Toniann Pitassi. Non-automatizability of bounded-depth Frege proofs. *Computational Complexity*, 13(1–2):47–68, 2004.
12. Maria Luisa Bonet, Toniann Pitassi, and Ran Raz. On interpolation and automatization for Frege systems. *SIAM Journal on Computing*, 29(6):1939–1967, 2000.
13. William Craig. Three uses of the Herbrand-Gentzen theorem in relating model theory and proof theory. *The Journal of Symbolic Logic*, 22(3):269–285, 1957.

14. Uwe Egly, Martin Kronegger, Florian Lonsing, and Andreas Pfandler. Conformant planning as a case study of incremental QBF solving. *CoRR*, abs/1405.7253, 2014.
15. Alexandra Goultiaeva, Allen Van Gelder, and Fahiem Bacchus. A uniform approach for generating proofs and strategies for both true and false QBF formulas. In *IJCAI*, pages 546–553, 2011.
16. Pavel Hrubeš. On lengths of proofs in non-classical logics. *Annals of Pure and Applied Logic*, 157(2–3):194–205, 2009.
17. Mikoláš Janota and Joao Marques-Silva. ∀Exp+Res does not p-simulate Q-resolution. International Workshop on Quantified Boolean Formulas, 2013.
18. Mikoláš Janota and Joao Marques-Silva. On propositional QBF expansions and Q-resolution. In *SAT*, pages 67–82, 2013.
19. Hans Kleine Büning, Marek Karpinski, and Andreas Flögel. Resolution for quantified Boolean formulas. *Inf. Comput.*, 117(1):12–18, 1995.
20. Jan Krajíček. Interpolation theorems, lower bounds for proof systems and independence results for bounded arithmetic. *J. Symb. Log.*, 62(2):457–486, 1997.
21. Jan Krajíček. *Forcing with random variables and proof complexity*, volume 382 of *Lecture Note Series*. London Mathematical Society, 2011.
22. Jan Krajíček and Pavel Pudlák. Some consequences of cryptographical conjectures for S_2^1 and EF. *Information and Computation*, 140(1):82–94, 1998.
23. Daniele Mundici. Tautologies with a unique Craig interpolant, uniform vs. nonuniform complexity. *Annals of Pure and Applied Logic*, 27:265–273, 1984.
24. Christos H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.
25. Pavel Pudlák. Lower bounds for resolution and cutting planes proofs and monotone computations. *The Journal of Symbolic Logic*, 62(3):981–998, 1997.
26. Pavel Pudlák. Proofs as games. *American Math. Monthly*, pages 541–550, 2000.
27. Jussi Rintanen. Asymptotically optimal encodings of conformant planning in QBF. In *AAAI*, pages 1045–1050. AAAI Press, 2007.
28. Allen Van Gelder. Contributions to the theory of practical quantified Boolean formula solving. In *CP*, pages 647–663, 2012.
29. Lintao Zhang and Sharad Malik. Conflict driven learning in a quantified Boolean satisfiability solver. In *ICCAD*, pages 442–449, 2002.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de