New Resolution-based QBF Calculi and Their Proof Complexity

OLAF BEYERSDORFF, Institute of Computer Science, Friedrich Schiller University Jena, Germany
LEROY CHEW, School of Computing, University of Leeds, UK
MIKOLÁŠ JANOTA, IST/INESC-ID, Universidade de Lisboa, Portugal

Modern QBF solvers typically use two different paradigms, conflict-driven clause learning (CDCL) solving or expansion solving. Proof systems for quantified Boolean formulas (QBFs) provide a theoretical underpinning for the performance of these solvers, with Q-Resolution and its extensions relating to CDCL solving and ∀Exp+Res relating to expansion solving. This paper defines two novel calculi, which are resolution-based and enable unification of some of the principal existing resolution-based QBF calculi, namely Q-resolution, long-distance Q-resolution and the expansion-based calculus ∀Exp+Res.

However, the proof complexity of the QBF resolution proof systems is currently not well understood. In this paper we completely determine the relative power of the main QBF resolution systems, settling in particular the relationship between the two different types of resolution-based QBF calculi: proof systems for CDCL-based solvers (Q-resolution, universal and long-distance Q-resolution) and proof systems for expansion-based solvers (∀Exp+Res and its generalizations IR-calc and IRM-calc defined here).

The most challenging part of this comparison is to exhibit hard formulas that underly the exponential separations of the aforementioned proof systems. To this end we exhibit a new and elegant proof technique for showing lower bounds in QBF proof systems based on strategy extraction. This technique provides a direct transfer of circuit lower bounds to lengths of proofs lower bounds. We use our method to show the hardness of a natural class of parity formulas for Q-resolution and universal Q-resolution. Variants of the formulas are hard for even stronger systems as long-distance Q-resolution and extensions.

With a completely different and novel counting argument we show the hardness of the prominent formulas of Kleine Büning et al. [51] for the strong expansion-based calculus IR-calc.

CCS Concepts: • **Theory of computation** → **Proof complexity**; *Automated reasoning*; • **Mathematics of computing** → *Solvers*;

Additional Key Words and Phrases: proof complexity, QBF, lower bound techniques, separations

ACM Reference Format:
Olaf Beyersdorff, Leroy Chew, and Mikoláš Janota. 2022. New Resolution-based QBF Calculi and Their Proof Complexity. *ACM Trans. Comput. Theory* 1, 1 (March 2022), 42 pages. https://doi.org/10.1145/3352155

1 INTRODUCTION

Proof complexity studies the complexity of proof measures in various formal systems, providing both sharp lower and upper bounds for the size of proofs of important combinatorial statements. One motivation for this research comes from its close connection to fundamental questions in computational complexity, and this connection has been present since the very beginnings of the field [32]. Another motivation is the tremendous success of SAT solvers, which today solve huge industrial instances of the NP-hard SAT problem with even millions of variables. Proof complexity provides the main theoretical tool for an understanding of the power and limitations of these algorithms. As most modern SAT solvers are based on resolution, this proof system has received

Authors' addresses: Olaf Beyersdorff, Institute of Computer Science, Friedrich Schiller University Jena, Germany, olaf.beyersdorff@uni-jena.de; Leroy Chew, School of Computing, University of Leeds, Woodhouse Lane, Leeds, West Yorkshire, LS2, UK, scslnc@leeds.ac.uk; Mikoláš Janota, IST/INESC-ID, Universidade de Lisboa, Portugal.

key attention; and many ingenious techniques have been devised to understand the complexity of resolution proofs (cf. [27, 68] for surveys).

During the last decade there has been great interest and research activity to extend the success of SAT solvers to the more powerful case of *quantified Boolean formulas (QBF)*. Due to its PSPACE completeness (even for restricted versions [2]), QBF is far more succinct than SAT and thus applies to further fields such as formal verification or planning [8, 34, 62]. As for SAT solvers, runs of QBF solvers produce witnesses of unsatisfiability (proofs), and there has been a lot of interest in the correspondence between the formal systems and solvers. In fact, QBF solvers can also provide witnesses in terms of proofs for true QBFs. However, in this article we only consider refutational systems that refute closed false QBFs.

In particular, Kleine Büning et al. [51] defined a resolution-like calculus called *Q-resolution* (Q-Res). There are several extensions of Q-Res; notably *long-distance Q-resolution* (LQ-Res) [3], which is more powerful than the standard Q-Res [35]. While Q-Res can only resolve on existential variables, the proof system *QU-Res*, introduced by Van Gelder [71], also allows to resolve on universal variables. Combining universal and long-distance Q-resolution, Balabanov et al. [5] considered the system LQU^{+}-Res. Q-Res and its extensions are important as they model QBF solving based on *conflict driven clause learning (CDCL)* [37].

Apart from CDCL, another main approach to QBF-solving is through *expansion of quantifiers* [7, 22, 47]. Recently, a proof system ∀Exp+Res was introduced [48] with the motivation to trace expansion-based QBF solvers. ∀Exp+Res also uses resolution, but is conceptually very different from Q-Res.

Conceptually, the variants of Q-resolution correspond to solving QBF by *search and clause-learning*. In contrast, expansion-based systems correspond to solving QBF by *rewriting quantifiers into Boolean connectives* — such rewrites may be iterative and may require introduction of fresh variables. Hence, the division between expansion-based systems and Q-resolution-based systems is naturally reflected in solving techniques.

1.1 Our contributions

In this paper we aim towards a significantly better understanding of QBF resolution proof systems and their proof complexity. Our main contributions are (A) two new resolution-type proof systems that naturally combine features of CDCL and expansion QBF solving and (B) the analysis of the proof complexity of these systems. In the following we explain our main results.

A. Unifying QBF resolution calculi. Our first contribution is the introduction of two new calculi that combine core ideas in CDCL and expansion solving.

A1. New QBF proof systems. We introduce two novel calculi IR-calc and IRM-calc, which are shown to be sound and complete for QBF. These calculi are able to simulate the basic existing QBF resolution-based calculi from both CDCL and expansion solving and thus combine core features of these two solving approaches in one system.[1] At the same time the new systems remain amenable to machine manipulation.

IR-calc takes influence from the expansion in ∀Exp+Res. While ∀Exp+Res uses only full expansion of all universal variables — resulting in a propositional CNF to which classical resolution is applied — our new system IR-calc works with partial expansions and thus allows to mix resolution and expansion steps. Working with partial expansions, which can be completed later in different ways, yields additional power over ∀Exp+Res and allows IR-calc to both simulate ∀Exp+Res and Q-Res.

[1]We remark that there are further features in QBF solving not captured by our system (cf. Section 1.2).

Our second system IRM-calc extends IR-calc by allowing merge steps for universal variables and more flexible annotations (using a new $*$ symbol which may stand for expansions by either 0 or 1). This takes inspiration from LQ-Res, and in fact IRM-calc can simulate LQ-Res, hence unifying CDCL- and expansion-based ideas.

A2. Strategy extraction. The semantics of a closed prenix QBF can be understood as a two-player game between $\exists$ and $\forall$ who alternatively pick values for the variables in the order of the quantifier prefix. In this game $\exists$ has a winning strategy if and only if the QBF is true, and $\forall$ has a winning strategy if and only if the formula is false. A QBF proof system has strategy extraction if given a refutation of a false QBF φ it is possible to efficiently compute a winning strategy for the universal player for φ. Here we show that both IR-calc and IRM-calc admit strategy extraction in polynomial time.

Indeed, unified certification of QBF solvers or certification of solvers combining expansion and DPLL is of immense practical importance [3, 35, 42] and presents one of the main motivations for our new systems.

B. Proof complexity of QBF resolution calculi. In general, it is fair to say that in comparison to propositional proofs, the complexity and relations between QBF proof systems are not well understood. In particular, it is crucial to understand which lower bound techniques are available for QBF proof systems.

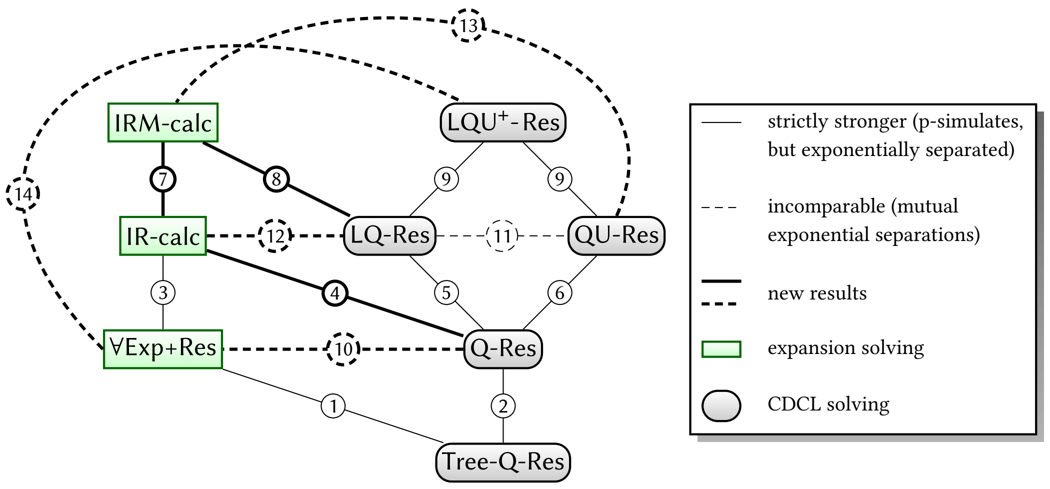
B1. A new lower bound method based on strategy extraction. We exhibit a new method to obtain lower bounds to the proof size in QBF proof systems, which directly allows to transfer circuit lower bounds to size of proof lower bounds. This method is based on the afore-mentioned property of *strategy extraction*, which is known to hold for many resolution-based QBF proof systems.

The basic idea of our method is both conceptually simple and elegant: If we know that a family φ_n of false QBFs requires large winning strategies, then proofs of φ_n must be large in all proof systems with feasible strategy extraction. Now we need suitable formulas φ_n. Starting with a language L — for which we know (or conjecture) circuit lower bounds — we construct a family of false QBFs φ_n such that every winning strategy of the universal player for φ_n will have to compute L for inputs of length n. Consequently, a circuit lower bound for L directly translates into a lower bound for the winning strategy and therefore the proof size.

Carefully implemented, our method yields *unconditional lower bounds*. For Q-Res (and QU-Res) it is known that strategy extraction is computationally easy [3]; it is in fact possible in AC^0 as we verify here. Using the hardness of parity for AC^0 we can therefore construct formulas $\text{QP}\textsc{arity}_n$ that require exponential-size proofs in Q-Res (and QU-Res).

Conceptually, our lower bound method via strategy extraction is similar to the feasible interpolation technique [54], which is one of the most successful techniques in classical proof complexity. In feasible interpolation, circuit lower bounds are also translated into proof size lower bounds. However, feasible interpolation only works for formulas of a special syntactic form, while our technique directly applies to arbitrary languages. It is a long-standing belief in the proof complexity community that there exists a direct connection between progress for showing lower bounds in circuit complexity and for proof systems (cf. [31]). For QBF proof systems our technique makes such a connection very explicit.

B2. Lower bounds for QBF proof systems. Our new lower bound method directly gives a new lower bound for Q-Res for the parity formulas. In addition, we transfer this lower bound to the stronger systems LQ-Res and QU-Res by arguing that neither long-distance nor universal resolution gives any advantage on a suitable modification of the parity formulas.



	Simulation/Separation			Incomparable
1	[48]	[48]	10	[48], Cor. 28
2	by def.	[25]	11	[5]
3	Thm. 18	[48], Thm. 16	12	Cor. 40, Cor. 56
4	Thm. 16	Cor. 28	13	Cor. 28, Cor. 60
5	by def.	[35]	14	Cor. 40, Cor. 56
6	by def.	[71]		
7	by def.	Thm. 19, Cor. 56		
8	Thm. 19	Cor. 40		
9	by def.	[5]		

Fig. 1. The simulation order of QBF resolution systems

For the strong system IR-calc we show that the strategy extraction method is not directly applicable (at least for unconditional bounds in the way we use it here). However, we use a completely different lower bound argument to obtain an exponential lower bound for the well-known formulas KBKF(t) of Kleine Büning, Karpinski and Flögel [51] in IR-calc. In the same work [51], where Q-Res was introduced, these formulas were suggested as hard formulas for Q-Res. In fact, a number of further separations of QBF proof systems builds on this [5, 35]. Here we show in a technically involved counting argument that the formulas are even hard for IR-calc. As IR-calc simulates Q-Res we obtain as a by-product a formal proof of the hardness of KBKF(t) in Q-Res.

B3. Separations between QBF proof systems. Our lower bounds imply a number of new separations and incomparability results. The three main new results are: *(i) IR-calc does not simulate LQ-Res; (ii) IRM-calc does not simulate QU-Res; (iii) LQU$^+$-Res does not simulate ∀Exp+Res.* All are in fact exponential separations. Items (i) and (ii) are obtained from lower bounds for KBKF(t) and a modification thereof suggested in [5], while (iii) follows from the lower bound on a variant of the parity formulas.[2] In contrast to separations by KBKF(t), all separations derived from (iii) even hold for formulas of bounded quantifier complexity.

By transitivity and together with previous simulation results these imply many further separations. Figure 1 depicts the simulation order of QBF resolution systems together with the separations. Combined with previous simulations and separations (cf. the table accompanying Figure 1) this yields a complete understanding of the simulation order of QBF resolution systems, i.e., any two systems depicted in Figure 1 either simulate each other or are exponentially separated (lines not depicted in Figure 1 follow by transitivity).

1.2 Relations to further work

Follow-up work. We point out that since the appearance of the conference versions [11, 12] — containing parts of the material of the present article — there has been interesting follow-up work on the results presented here, thus demonstrating the impact of this work. Recently our lower bound method based on strategy extraction has been generalised to much stronger systems than QBF resolution, namely QBF Frege systems defined in [10], thereby exploiting the full range of current circuit lower bounds for lower bounds in strong QBF proof systems [10, 21].

Another line of very recent research assessed the applicability of classical proof complexity techniques to the QBF resolution systems investigated and defined here. Specifically, feasible interpolation [54, 60] was shown to hold for all QBF resolution systems depicted in Figure 1, and indeed, this technique can be interpreted as a special case of our new strategy extraction technique [14]. In contrast, the seminal size-width technique for classical resolution [6] only works in very limited settings for tree-like Q-Res and tree-like IR-calc, but drastically fails for most other QBF resolution systems [15, 30]. Game-theoretic techniques from classical resolution [18–20, 61], however, can be generalised to QBF resolution [17, 28].

Finally, [16] shows that IR-calc can be lifted to a sound and complete proof system for the NEXPTIME-complete logic of dependency QBFs (DQBF), while all other lifted versions of QBF resolution systems from Figure 1 except ∀Exp+Res are either incomplete or unsound. Further, Egly [33] shows that IR-calc can be interpreted as a meaningful fragment of FO resolution.[3] These results confirm that IR-calc is a natural choice for a QBF resolution calculus.

Other approaches in QBF solving. In addition to the core features from CDCL and expansion solving modelled by the proof systems investigated here, practical QBF solving incorporates a number of further ideas, which are beyond the scope of the present paper. Most notably this concerns the use of *dependency schemes* [55, 56, 65, 66, 70], which enable de-linearizing the prefix and thus strengthen the performance of solvers. Dependency schemes are not modelled by the systems in Figure 1. However, some of the proof systems considered here can be parameterized by dependency schemes (cf. [9, 59, 69]).

Another approach not covered in this paper is *extension variables* [49, 53]. Some initial proof complexity investigation on the power of extension variables in QBF is contained in [13]. We also note that [33] contains an extension of IR-calc, where the propositional extension rule is permitted. If universal variables cannot only be replaced by truth values but also by existential variables of smaller level, then the propositional extension rule allows for a simulation of powerful quantifier rules known from sequent calculi for QBFs.

[2] Items (i) and (iii) are already stated and sketched in the conference version [12] of this paper, whereas the separation in item (ii) is posed in [12] as an open problem and is shown here for the first time.

[3] This also holds for other QBF resolution variants like Q-Res or QU-Res. However, FO resolution (together with suitable translations) is stronger than IR-calc and IRM-calc [33].

Finally, *preprocessing* is a powerful tool in QBF [22, 24, 26, 38, 67]. A variety of applied techniques are traceable by Q-resolution [46], but some of the techniques are expansion-based. We do not determine the relationship of preprocessing to the calculi studied here. A proof system for tracing preprocessing was recently developed in the form of QRAT [44].

True vs false QBFs. We concentrate in this paper on refuting false QBFs, but QBF solvers also need to cope with true formulas. While our results show separations for calculi (and associated solving approaches) on false QBFs, we do not investigate similar separations on true QBFs. We comment briefly on this.

There are two main approaches to show that a formula is true in QBF solving. *Term-resolution* is dual to Q-resolution—it operates on terms rather than clauses—with an additional rule that enables generating terms (in fact implicants) from the CNF matrix on the fly [39]. Another approach is to reason on the formula's negation, which is especially advantageous when the input is non-CNF [40, 41, 53, 72, 75]. We conjecture that the ideas presented here can be adapted to either of these approaches; some preliminary results exist in this direction [13, 45], however that is beyond the scope of this article. Some approaches certify true formulas by a winning strategy for the existential player [4, 42, 49]. Nevertheless, this is not a calculus in the traditional sense as verifying that a strategy is indeed a winning one is coNP-complete [52].

1.3 Organisation

The remainder of this article is organised as follows. Section 2 explains background on QBFs, including their semantics and the previously known QBF resolution proof systems.

In Section 3 we define the new proof systems IR-calc and IRM-calc, which generalise the expansion calculus ∀Exp+Res. In order to show soundness we prove that they have strategy extraction in polynomial time. We then show that IR-calc p-simulates both Q-Res and ∀Exp+Res and that IRM-calc p-simulates LQ-Res, implying that our new calculi IR-calc and IRM-calc are complete.

Sections 4 and 5 contain our lower bound arguments and techniques, based on strategy extraction (Section 4) and for the formulas from Kleine Büning et al. [51] (Section 5). These lower bounds also imply the separations shown in Figure 1.

2 PRELIMINARIES

Notions from complexity. We use standard notions from computational complexity [1]. The circuit class AC^0 contains all languages computable by a uniform family of Boolean circuits with $\neg$, $\wedge$, and $\vee$ gates, where each circuit's depth is bounded by a constant (cf. [73]).

Notions from logic. A *literal* is a Boolean variable or its negation. If l is a literal, then $\neg l$ denotes the *complementary literal*, i.e. $\neg\neg x = x$. A *clause* is a disjunction of literals (which we typically represent as a set) and a *term* is a conjunction of literals. The empty clause is denoted by $\bot$, which 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.

Given a propositional formula Φ and a partial assignment α, the *restriction* of Φ by α, denoted $\Phi|_\alpha$, results from substituting the values specified by α into Φ (and simplifying the formula accordingly).

Quantified Boolean Formulas (QBFs) [50] extend propositional logic with quantifiers with the standard semantics that $\forall x.\,\Psi$ is satisfied by the same truth assignments as $\Psi|_{x=0} \wedge \Psi|_{x=1}$ and $\exists x.\,\Psi$ as $\Psi|_{x=0} \vee \Psi|_{x=1}$. Unless specified otherwise, we assume that QBFs are in *closed prenex* form with a CNF *matrix*, i.e., we consider the form $\mathcal{Q}_1 X_1 \ldots \mathcal{Q}_k X_k.\,\phi$, where X_i are pairwise disjoint (ordered) sets of variables; $\mathcal{Q}_i \in \{\exists, \forall\}$ and $\mathcal{Q}_i \neq \mathcal{Q}_{i+1}$. The formula ϕ is in CNF and is defined only

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

C is a non-tautological clause in the original 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 an existential variable. If z is a literal with $\mathrm{var}(z) \neq x$ and $z \in C_1$, then $\neg z \notin C_2$. $U_1 = U_2 = U = \emptyset$.
S∀R: x is a universal variable. Otherwise same conditions as S∃R.
L∃R: x is an existential variable.
If $l_1 \in C_1, l_2 \in C_2$, and $\mathrm{var}(l_1) = \mathrm{var}(l_2) = z$ then $l_1 = l_2 \neq z^*$. U_1, U_2 contain only universal literals with $\mathrm{var}(U_1) = \mathrm{var}(U_2)$ and $\mathrm{ind}(x) < \mathrm{ind}(u)$ 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 $w_1 = \neg w_2$ or $w_1 = u^*$ or $w_2 = u^*$. $U = \{u^* \mid u \in \mathrm{var}(U_1)\}$.
L∀R: x is a universal variable (but not a merged literal z^*). Otherwise same conditions as L∃R.

proof system	rules
Q-Res	S∃R, ∀-Red
LQ-Res	L∃R, ∀-Red, ∀-Red*
QU-Res	S∃R, S∀R, ∀-Red
LQU$^+$-Res	L∃R, L∀R, ∀-Red, ∀-Red*

Fig. 2. The rules of CDCL-based proof systems

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 $\mathrm{lv}(\mathrm{var}(l))$. In contrast to the level, 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. Only occasionally we distinguish between a level and index, e.g. in the definition of LQ-Res below.

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, and both players have complete information on all previous moves. 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 level i, a *strategy for u* is a function, which maps assignments of 0/1 values to the variables of lower index than u to $\{0, 1\}$ (the intended response for u). 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 [42][1, Sec. 4.2.2][58, Chap. 19].

A *proof system* [32] for a language L over alphabet Γ (usually binary) is a polynomial-time computable partial function $f : \Gamma^\star \longrightarrow \Gamma^\star$ with $rng(f) = L$. If $f(x) = y$ then x is called an f-proof for y. If L consists of all propositional tautologies, then f is called a *propositional proof system*, and proof systems for the language TQBF of true QBFs are called *QBF proof systems*. Equivalently, we can consider refutation proof systems where we start with the negation of the formula that we want to prove and derive a contradiction.

A proof system S for L *simulates* a proof system P for L if there exists a polynomial p such that for all P-proofs π of x there is an S-proof π' of x with $|\pi'| \leq p(|\pi|)$. If such a proof π' can even be computed from π in polynomial time we say that S *p-simulates* P.

Resolution is one of the best studied propositional proof systems (cf. [68]). It is a refutational proof system manipulating unsatisfiable CNFs as sets of clauses. The only inference rule is

$$\frac{C \cup \{x\} \qquad D \cup \{\neg x\}}{C \cup D}$$

where C, D are clauses and x a variable. A *Resolution refutation* derives the empty clause $\bot$.

Resolution-based calculi for QBF. We now give a brief overview of the main existing resolution-based calculi for QBF, which model *CDCL-based QBF solving*. Their rules are summarized in Figure 2. Like Resolution, all of these proof systems are refutation systems operating with clauses. As before, a refutation in any of these systems has the empty clause as its last clause.

The most basic and important system is *Q-resolution (Q-Res)* by Kleine Büning et al. [51]. 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. 2). The ∀-Red rule allows to drop a universal literal u from the clause, provided u has highest level in the clause (i.e., no existential variable in the clause depends on u). Note that the rule S∃R is just the classical resolution rule on existential variables. However, it imposes the restriction not to derive tautological clauses. In propositional resolution this does not present a problem, but in Q-Res we need this restriction to guarantee soundness. Consider the example $\forall u \exists x.\, (x \vee u) \wedge (\neg x \vee \neg u)$. This is a true QBF. However, if tautological resolvents were allowed, we could derive $u \vee \neg u$ and then ∀-reduce to the empty clause.

Long-distance resolution (LQ-Res) appears originally in the work of Zhang and Malik [74] and was formalized into a calculus by Balabanov and Jiang [3]. The idea of long-distance resolution is to allow certain resolution steps that produce tautological clauses and are thus prohibited in Q-Res. However, this is only allowed under certain side-conditions, explained below. Instead of deriving a tautological clause containing $u \vee \neg u$, long-distance steps merge complementary universal literals u and $\neg u$ of a universal variable u into the special literal u^*. Syntactically, u^* is just a new variable, but intuitively, one may think of u^* as some kind of representation for $u \vee \neg u$.[4]

The mentioned side-conditions in the long-distance rule L∃R are as follows (cf. Figure 2). The pivot is the existential variable x. In addition, there are universal literals U_1 contained in the first clause and U_2 contained in the second clause, which all appear right of the pivot x in the quantifier prefix. Moreover, U_1 and U_2 contain exactly the same universal variables, but in opposite polarities (or already in its starred version u^*). In the long-distance resolution step, all variables in U_1 and U_2 will be merged, i.e., for each u appearing in U_1 and U_2 we include u^* in the resolvent.

LQ-Res uses the 'long-distance' resolution rule L∃R over existential pivots (generalising the 'short-distance' resolution rule S∃R over existentials) together with the universal reduction rules ∀-Red and ∀-Red*.

As a toy example consider the formula $\exists x \forall u.\, (x \vee u) \wedge (\neg x \vee \neg u)$, where we can perform a long-distance step with pivot x, leading to u^*, which can subsequently be ∀-reduced to the empty clause. However, the unsound step used in our earlier example $\forall u \exists x.\, (x \vee u) \wedge (\neg x \vee \neg u)$ is still prohibited in LQ-Res as the merged literals must have higher index than the pivot.

QU-resolution (QU-Res) [71] is a different extension of Q-Res, which removes the restriction from Q-Res that the resolved variable must be an existential variable and allows resolution of universal variables. However, it still forbids tautological resolvents. The rules of QU-Res are S∃R, S∀R, and ∀-Red.

[4]Note that complementary universal literals appear naturally in learned clauses in QBF solvers [74].

Finally, *LQU*$^+$*-Res* [5] combines LQ-Res and QU-Res by allowing short and long distance resolution over both existential or universal pivots. However, the pivot is never a merged literal z^*. LQU$^+$-Res uses the rules L∃R, L∀R, ∀-Red, and ∀-Red*.

For any of the resolution calculi above, the size of a derivation is the number of clauses in the proof.

3 EXPANSION CALCULI: DEFINITIONS, STRATEGY EXTRACTION, AND SIMULATIONS

In this section we define the new expansion calculi IR-calc and IRM-calc (Section 3.1), show their soundness by strategy extraction — a very desirable property in practice (Section 3.3) — and compare them to previously defined QBF resolution systems (Section 3.4).

3.1 Introducing the expansion calculi IR-calc and IRM-calc

In contrast to the QBF resolution systems from the previous section the expansion calculi we will consider here use only existential variables. Before giving the technical details, let us try to explain the idea of expansion of universal variables.

Consider the QBF $\exists x \forall u \exists y.\, \phi(x, u, y)$. Expanding the universal variable u, this is semantically equivalent to $\exists x \exists y^0 \exists y^1.\, \phi(x, 0, y^0) \wedge \phi(x, 1, y^1)$, where we need to create two fresh copies of the existential variables right of u (in this case we create two copies y^0 and y^1, but keep x). In fact, instead of just renaming the variable y to y^0 and y^1 it will be more convenient to record in the annotation, which expansion caused the renaming. In our example, we would name y^0 as $y^{0/u}$ and y^1 as $y^{1/u}$ to remember that we created the copies of y by expanding u.

In this way we expand all the universal variables in the QBF, which results in a purely existentially quantified formula. Of course, in general this will produce an exponential increase in the formula size. However, in some cases it may suffice to instantiate a universal variable in just one polarity and still preserve the falsity of the QBF. Consider the example $\forall u \exists x.\, (u \vee x) \wedge (u \vee \neg x)$, where we can just instantiate u by 0 to obtain the false QBF $\exists x^{0/u}.\, x^{0/u} \vee \neg x^{0/u}$. We refer to [48] for more background information on expansion.

This approach is in line with the workings of the highly competitive QBF solver RAReQS [47], which gradually expands all universal variables to obtain a purely existential formula that is then passed to a SAT solver. Proof theoretically, RAReQS corresponds to the calculus ∀Exp+Res [48], introduced below. On a technical note, the solver also simultaneously expands the formula's negation, which enables proving true formulas.

We now start to set up the formal framework allowing us to define our new calculi. The framework hinges on the concept of clauses that only contain existential variables with annotations.

Definition 1.

(1) *An* extended assignment *is a partial mapping from the universal variables to* $\{0, 1, *\}$. *We remark that this notation has nothing to do with the proof complexity convention to map unassigned variables to* $*$. *Rather, a variable u is mapped to* $*$ *if* u^* *appears in some clause, which will be made precise later.*
In contrast, an assignment has range $\{0, 1\}$, *as usual. To highlight the difference, we will sometimes refer to assignments as 0/1 assignments.*

(2) *An* annotated clause *is a clause where each existential literal is annotated by an extended assignment to universal variables.*

(3) *For an extended assignment* σ *to universal variables we write* $l^{[\sigma]}$ *to denote an annotated literal where* $[\sigma] = \{c/u \in \sigma \mid \mathrm{lv}(u) < \mathrm{lv}(l)\}$. *Thus* $l^{[\sigma]}$ *annotates l only with the variables in* σ *left*

of l in the quantifier prefix (which are the variables assigned before l in the game semantics of QBF).

(4) *Two (extended) assignments τ and μ are called* contradictory *if there exists a variable $x \in \mathsf{dom}(\tau) \cap \mathsf{dom}(\mu)$ with $\tau(x) \neq \mu(x)$.*

Further we define operations that let us modify annotations of a clause by *instantiation.*

Definition 2. *For (extended) assignments τ and μ, we write $\tau \circ \mu$ 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 $\tau \circ \mu$ is referred to as* completion *because μ provides values for variables that are not defined in τ.*

The completion operation is associative and therefore we can omit parentheses. In contrast, it is *not* commutative. The following properties hold: (i) For non-contradictory μ and τ, we have $\mu \circ \tau = \tau \circ \mu = \mu \cup \tau$. (ii) $\tau \circ \tau = \tau$.

We also need an auxiliary operation of instantiation, which completes the annotations of literals in a clause by a partial (extended) assignment.

Definition 3. *For an extended assignment τ and an annotated clause C, the function inst(τ, C) returns the annotated clause $\left\{l^{[\sigma\circ\tau]} \mid l^{\sigma} \in C\right\}$.*

We are now ready to describe the expansion QBF systems.

The first of these is the *calculus* $\forall$*Exp+Res from* [48]. In Figure 3 we present an adapted version of this calculus so that it is congruent with the new concepts presented here (semantically it is the same as in [48]). The axiom rule for ∀Exp+Res simply downloads a clause by picking a *total* 0/1 assignment to the universal variables, applies it to the clause, and records the used assignment in the annotations.

Note that, in ∀Exp+Res, τ is always a complete assignment to all universal variables. However, when appearing as annotation to an existential variable x, it is truncated to the universal variables left of x. This is reflected in the notation $x^{[\tau]}$ (cf. Definition 1). For example, if we have a formula $\forall u \exists x \forall v \exists y.\, \phi$ and τ assigns u and v to 0, then a clause $u \vee x \vee v \vee y$ from ϕ would be instantiated to $x^{0/u} \vee y^{0/u,0/v}$.

The resolution rule of ∀Exp+Res is just the propositional resolution rule. Note, however, that the pivot annotations need to match exactly in both parent clauses. This makes sense, because in our framework, different annotations formally lead to distinct variables.

In ∀Exp+Res, proofs can be easily split into two phases. The first consists only of axiom downloads, removing all universal variables and annotating the existential variables. This is followed by the second phase, consisting only of classical resolution steps on the new annotated variables, cf. Section 3.2 for an example.

We now describe *our first new system IR-calc.* Like ∀Exp+Res it completely eliminates universal variables and operates with annotated clauses. The main difference is that, unlike in ∀Exp+Res, where existentials are always annotated by all universals preceding them in the prefix, the system IR-calc can deal with partial assignments. The calculus introduces clauses from the matrix and allows to **I**nstantiate and **R**esolve clauses; hence the name IR-calc. It comprises the rules in Figure 4.

The axiom download rule is similar to ∀Exp+Res, but only annotates the existential literals with those preceding universals which actually occur in the clause. For example, if $\forall u \exists x \forall v \exists y.\, \phi$ contains the clause $x \vee v \vee y$ we download it in IR-calc as $x \vee y^{0/v}$, whereas in ∀Exp+Res we also need to choose a value for u and could either download it as $x^{0/u} \vee y^{0/u,0/v}$ or $x^{1/u} \vee y^{1/u,0/v}$.

The resolution rule is identical to ∀Exp+Res. Again, annotations in the pivot need to match precisely, and for this reason we also adopt an instantiation rule, which can increase annotations in the clause. This can enable further resolution steps. Again, in instantiation steps we need to

$$\frac{}{\{l^{[\tau]} \mid l \in C, l \text{ is an existential literal}\}}\ \text{(Axiom)}$$

C is a non-tautological clause from the original matrix and τ is a 0/1 assignment to *all* universal variables in the quantifier prefix that falsifies all universal literals in C.

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

C_1 and C_2 are clauses of annotated literals and x^τ is an annotated variable.

Fig. 3. The rules of ∀Exp+Res (adapted from [48])

$$\frac{}{\{l^{[\tau]} \mid l \in C, l \text{ is an existential literal}\}}\ \text{(Axiom)}$$

C is a non-tautological clause from the original matrix. τ is a *partial* 0/1 assignment to universal variables, which has exactly all universal variables from C as its domain and falsifies all universal literals in C, i.e., $\tau = \{0/u \mid u \in C, u \text{ is universal}\} \cup \{1/u \mid \neg u \in C, u \text{ is universal}\}$.

$$\frac{\{x^\tau\} \cup C_1 \qquad \{\neg x^\tau\} \cup C_2}{C_1 \cup C_2}\ \text{(Resolution)} \qquad \frac{C}{\mathsf{inst}(\tau, C)}\ \text{(Instantiation)}$$

C_1, C_2, and C are clauses of annotated literals.
τ is a partial 0/1 assignment to universal variables with $\mathsf{rng}(\tau) \subseteq \{0, 1\}$.

Fig. 4. The rules of IR-calc.

truncate the annotations to universal variables left of the annotated existential variable, as indicated by the notation $l^{[\sigma\circ\tau]}$ in Definition 3.

Unlike in the ∀Exp+Res system, IR-calc proofs are no longer separated into an annotation and a resolution phase, but can mix instantiation and resolution steps in the proof by "delayed instantiations" as and when they are needed. An example is contained in Section 3.2.

Note that in ∀Exp+Res, propositional variables are introduced so that their annotations assign *all* relevant variables. In this way, each literal corresponds to a value of a Skolem function in a specific point. In contrast, in IR-calc, variables are annotated "lazily", i.e., it enables us to reason about multiple points of Skolem functions at the same time. This is analogous to *specialization* of free variables by constants in first-order logic (FO). Similarly, resolution in IR-calc is analogous to resolution in Robinson's FO resolution [64]. The papers [16, 33] further explore the connection between IR-calc and FO resolution.[5] From the FO perspective, IR-calc appears to be a very natural proof system. In [16] this is confirmed by the fact that ∀Exp+Res and IR-calc are the only proof system among the resolution systems considered here that lift to the more succinct setting of dependency QBFs (DQBF).

Our second new system IRM-calc is an extension of IR-calc where we allow as annotations extended assignments with range $\{0, 1, *\}$. The purpose of $*$ is similar to the role of $*$ in LQ-Res (cf. the remarks in Section 2).

[5]We remark that the instantiation rule in Figure 4 can be omitted when the resolution rule is extended by unification which computes most general unifiers between the (FO image of) complementary annotated literals. Unification and most general unifiers guarantee that the refutation is as general as possible, i.e., instantiations are "minimal".

Axiom and instantiation rules as in IR-calc in Figure 4.

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

C_1 and C_2 are clauses of annotated literals. τ is a partial assignment to universal variables (i.e., $\mathsf{rng}(\tau) = \{0,1\}$) and ξ, σ are extended assignments (i.e., they assign universal variables to $\{0,1,*\}$). $\mathsf{dom}(\xi) \cap \mathsf{dom}(\sigma) = \emptyset$.

$$\frac{C \lor l^{\mu} \lor l^{\sigma}}{C \lor l^{\xi}} \text{ (Merging)}$$

C is a clause of annotated literals. l is a literal and μ and σ are extended assignments with $\mathsf{dom}(\mu) = \mathsf{dom}(\sigma)$. $\xi = \{c/u \mid c/u \in \mu, c/u \in \sigma\} \cup \{*/u \mid c/u \in \mu, d/u \in \sigma, c \neq d\}$

Fig. 5. The rules of IRM-calc.

Axioms and instantiation rules of IRM-calc are handled as in IR-calc, which do not create $*$. The annotation $*$ may be introduced by a new rule called *merging*. It merges two literals on the same existential variable to one copy where all conflicting annotations produce $*$. For example, the merge of $x^{0/u,0/v,0/w}$ and $x^{0/u,1/v,*/w}$ produces $x^{0/u,*/v,*/w}$.

The resolution rule is defined slightly differently, annotations of the pivot need not necessarily match before application of the rule, but are matched “on the fly" while resolving (the resolution rule is designed such that this matching is possible). More precisely, the resolution rule in Figure 5 uses pivot x, annotated as $x^{\tau\cup\xi}$ in the first parent clause and as $\neg x^{\tau\cup\sigma}$ in the second parent, where ξ and σ are partial extended assignments with disjoint domain. Thus, by instantiating the first parent with σ and the second parent with ξ we create a pivot $x^{\tau\cup\xi\cup\sigma}$ with matching annotations and obtain a sound resolution step. The resolution rule can now deal with $*$, but when $\xi = \sigma = \emptyset$ we have exactly the resolution rule from Figure 4. Thus IRM-calc encompasses IR-calc.

IRM-calc is defined in Figure 5, an example is contained in Section 3.2.

Intuitively, a unit clause $x^{0/u}$ asserts that any interpretation of the Skolem function for x must yield 1 whenever $u = 0$. Thus the clause $x^{1/u} \lor x^{0/u}$ asserts that for any value of the remaining relevant universal variables, the Skolem function must yield 1 on $u = 1$ or on $u = 0$. Note that this does *not* mean that the function has to be constantly 1. Merging enables compressing such clauses. For instance, in IR-calc, to refute the clauses $\{\neg x^{1/w},\ x^{0/u} \lor x^{1/u}\}$ one would apply the instantiation rule to obtain $\{\neg x^{0/u,1/w}, \neg x^{1/u,1/w}, x^{0/u,1/w} \lor x^{1/u,1/w}\}$ and then proceed as in classical propositional resolution. In contrast, IRM-calc enables merging the binary clause into the unit clause $x^{*/u}$, which in turn gives a contradiction with the original clause $\neg x^{1/w}$.

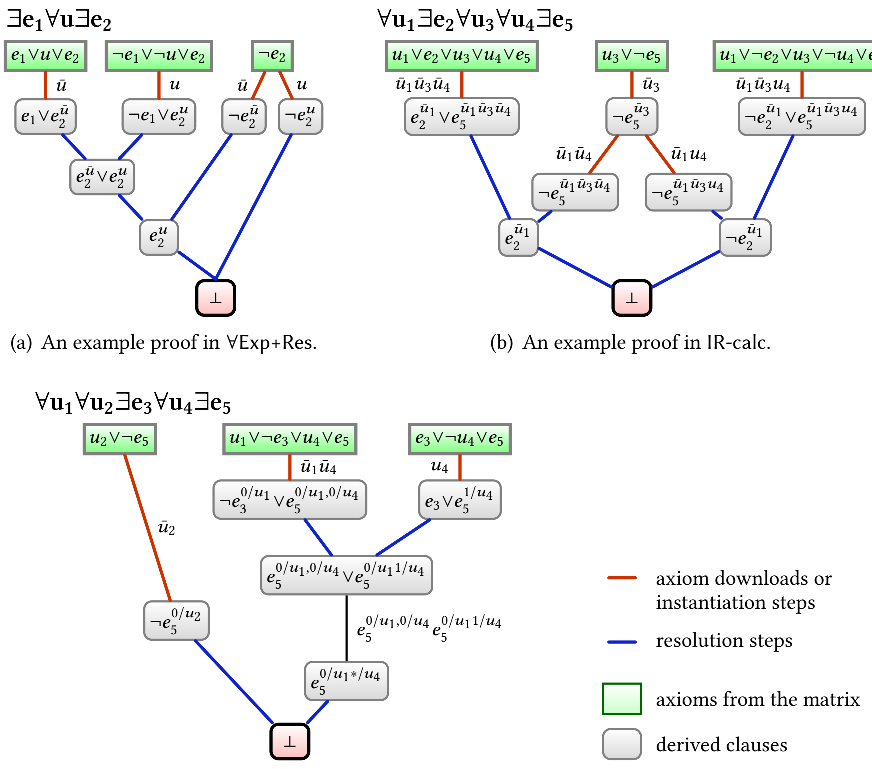
3.2 Proof examples

We illustrate the expansion calculi from Section 3.1 by a few examples. We write l^{u} as a shorthand for $l^{1/u}$ and $l^{\bar{u}}$ for $l^{0/u}$.

Example 4. *Figure 6(a) exemplifies a proof in ∀Exp+Res. In this case there is only one universal variable u. The variable needs to be instantiated whenever a new clause is introduced into the proof. In the case of the clause $\neg e_2$ there are two options for instantiation. In the case of the two other clauses only one of the instantiations is useful. Note that only e_2 is annotated by u because* $\mathrm{lv}(e_1) < \mathrm{lv}(u)$.

Example 5. *Figure 6(b) exemplifies a proof in IR-calc; most notably, $\neg e_5$ needs to be annotated only by $\bar{u}_3$ when it enters the proof.*

Example 6. *Figure 6(c) shows an IRM-calc refutation, containing $*$ in an annotation of e_5.*


$\exists e_1 \forall u \exists e_2$
$e_1 \vee u \vee e_2$
$\neg e_1 \vee \neg u \vee e_2$
$\neg e_2$
$\bar{u}$
u
$e_1 \vee e_2^{\bar{u}}$
$\neg e_1 \vee e_2^{u}$
$\neg e_2^{\bar{u}}$
$\neg e_2^{u}$
$e_2^{\bar{u}} \vee e_2^{u}$
e_2^{u}
$\perp$
$\forall u_1 \exists e_2 \forall u_3 \forall u_4 \exists e_5$
$u_1 \vee e_2 \vee u_3 \vee u_4 \vee e_5$
$u_3 \vee \neg e_5$
$u_1 \vee \neg e_2 \vee u_3 \vee \neg u_4 \vee e_5$
$\bar{u}_1 \bar{u}_3 \bar{u}_4$
$\bar{u}_3$
$\bar{u}_1 \bar{u}_3 u_4$
$e_2^{\bar{u}_1} \vee e_5^{\bar{u}_1 \bar{u}_3 \bar{u}_4}$
$\neg e_5^{\bar{u}_3}$
$\neg e_2^{\bar{u}_1} \vee e_5^{\bar{u}_1 \bar{u}_3 u_4}$
$\bar{u}_1 \bar{u}_4$
$\bar{u}_1 u_4$
$\neg e_5^{\bar{u}_1 \bar{u}_3 \bar{u}_4}$
$\neg e_5^{\bar{u}_1 \bar{u}_3 u_4}$
$e_2^{\bar{u}_1}$
$\neg e_2^{\bar{u}_1}$
$\perp$
$\forall u_1 \forall u_2 \exists e_3 \forall u_4 \exists e_5$
$u_2 \vee \neg e_5$
$u_1 \vee \neg e_3 \vee u_4 \vee e_5$
$e_3 \vee \neg u_4 \vee e_5$
$\bar{u}_1 \bar{u}_4$
u_4
$\neg e_3^{0/u_1} \vee e_5^{0/u_1, 0/u_4}$
$e_3 \vee e_5^{1/u_4}$
$\bar{u}_2$
$e_5^{0/u_1, 0/u_4} \vee e_5^{0/u_1 1/u_4}$
$\neg e_5^{0/u_2}$
$e_5^{0/u_1, 0/u_4}\ e_5^{0/u_1 1/u_4}$
$e_5^{0/u_1 * /u_4}$
$\perp$
axiom downloads or instantiation steps
resolution steps
axioms from the matrix
derived clauses

(a) An example proof in ∀Exp+Res.

(b) An example proof in IR-calc.

(c) An example proof in IRM-calc.

Fig. 6. Proof examples. Instantiation

Example 7. *Consider the (true) QBF $\exists x \forall u w \exists b. (x \vee u \vee b) \wedge (\neg x \vee \neg u \vee b) \wedge (u \vee w \vee \neg b)$. In both calculi, applications of the Axiom rule yield $(x \vee b^{0/u})$, $(\neg x \vee b^{1/u})$, and $(\neg b^{0/w, 0/u})$. IR-calc enables deriving $(b^{0/u} \vee b^{1/u})$ from the first two clauses. IRM-calc further enables deriving $b^{*/u}$ by merging. Intuitively, $(b^{0/u} \vee b^{1/u})$ means that the existential player must play so that for any assignment to w it holds that $b = 1$ if $u = 0$ or if $u = 1$. So for instance, the existential player might choose to play $b = 1$ if $w = 0, u = 1$ and if $w = 1, u = 0$ (and otherwise $b = 0$). The clause $b^{*/u}$ can be seen as a shorthand for the clause $(b^{0/u} \vee b^{1/u})$. Note that it would be* unsound *to derive the clause (b) (with no annotation). This would mean that b must be 1 regardless of the moves of the universal player. However, b needs to be 0 when $u = w = 0$ due to the axiom $(\neg b^{0/w, 0/u})$.*

Example 8. *Consider again the QBF $\exists x \forall u w \exists b. (x \vee u \vee b) \wedge (\neg x \vee \neg u \vee b) \wedge (u \vee w \vee \neg b)$ from Example 7. If the third clause of the formula is changed to $\neg b$, the formula becomes false, which is shown by instantiating $\neg b$ to $\neg b^{0/u}$ and to $\neg b^{1/u}$, using those to obtain x and $\neg x$ by resolution and deriving the empty clause.*

Example 9. *Consider the QBF $\exists x \forall u \exists bc. (x \vee u \vee b) \wedge (\neg x \vee \neg u \vee c) \wedge (\neg b \vee c) \wedge (\neg c)$. The following derivation is possible in IR-calc. Resolving $x \vee b^{0/u}$ and $\neg x \vee c^{1/u}$ yields $b^{0/u} \vee c^{1/u}$. Instantiating $\neg b \vee c$ by $0/u$ gives $\neg b^{0/u} \vee c^{0/u}$, resolving this with the previous resolvent yields $c^{0/u} \vee c^{1/u}$. Refutation can be obtained by instantiation of $\neg c$ once by $0/u$ and once by $1/u$ and subseqent two resolution steps. In IRM-calc it is possible to obtain $c^{*/u}$ by merging and resolve that directly with $\neg c$, which yields $\perp$.*

3.3 Soundness and extraction of winning strategies

The purpose of this section is twofold: to show how to obtain a *winning strategy* for the universal player from an IR-calc proof, and to show that IR-calc is *sound*. First we show how to obtain a winning strategy for the universal player from a proof. From this, the soundness of the calculus follows because a QBF is false if and only if such a strategy exists. We will then explain how to extend this technique to IRM-calc.

The approach we follow is similar to the one used for Q-Res [42] or LQ-Res [35]. We first explain this method informally and then give full details on how this is implemented for IR-calc and IRM-calc.

Informally, for the approach to work for a QBF proof system P we need two properties:

(1) The proof system P shall be *closed under restrictions*, i.e., if π is a P-proof of $QX.\,\Phi$ (where Φ may contain further quantifiers) and α is an assignment to the variables X, then we can construct from π and α a P-proof π_α of $\Phi|_\alpha$ in polynomial time.
(2) Restricting a proof π by an assignment to the leftmost block of existential variables to a proof π_α as in item 1, we can *read off from π_α a response* for the universal player for the next round.

We therefore consider QBFs in the form $\Gamma = \exists E \forall U.\,\Phi$, where E and U are sets of variables and Φ is a QBF (potentially with further quantification beginning with $\exists$). Suppose π is an IR-calc refutation of Γ, and let ϵ be a total assignment to the variable block E. The assignment ϵ represents a move of the existential player. We restrict π to a refutation π_ϵ of the restricted formula $\forall U.\,\Phi|_\epsilon$. To obtain a response of the universal player, we construct from π_ϵ an assignment μ to the variables U such that reducing π_ϵ by μ gives a refutation of $\Phi|_{\epsilon\cup\mu}$.

Now let $\pi_{\epsilon,\mu}$ be the proof resulting from restricting π_ϵ by μ. The game continues with $\Phi|_{\epsilon\cup\mu}$ and $\pi_{\epsilon,\mu}$. In each of these steps, two quantifier levels are removed from the given QBF and a refutation for each of the intermediate formulas is produced. This guarantees a winning strategy for the universal player because in the end the existential player will be faced with an unsatisfiable formula without universal variables.

We now implement this approach for IR-calc. We start with item 1, showing closure of IR-calc under restrictions. This indeed holds for all partial assignments to existential variables (not just those to the first existential block).

LEMMA 10. *Let π be an IR-calc refutation of Γ and let ϵ be a partial assignment of existential variables from Γ. We can then construct from π a refutation π_ϵ of $\Gamma|_\epsilon$ in polynomial time.*

PROOF. Let ϵ be a partial assignment to the existential variables of Γ and let $\pi = (C_1, \ldots, C_m = \bot)$. We describe the construction of the restricted proof π_ϵ and simultaneously argue that it is a valid refutation. For this we define inductively clauses C'_i such that

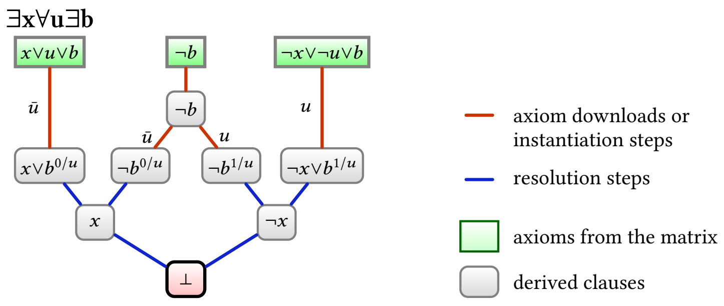
(1) if $C'_i \neq \top$ then $C'_i \subseteq C_i$ and C'_i has a valid IR-calc derivation in π_ϵ.
(2) if $C'_i = \top$ then ϵ satisfies C_i, i.e., $\epsilon(x) = 1$ for some $x^\tau \in C_i$.

We note that from these two conditions it follows that π_ϵ contains a refutation, because $C_m = \bot$ and hence $C'_m = \bot$, as ϵ does not satisfy C_m.

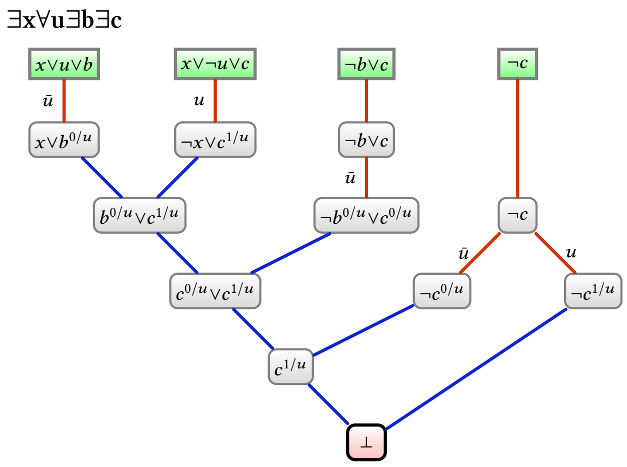
We now inductively construct C'_i and show the two items above.

Base case. Let C_i be derived by the axiom rule. If ϵ satisfies a literal l with $l^\tau \in C_i$ for some annotation τ we set $C'_i = \top$. Otherwise we define $C'_i = C_i \setminus \{l^\tau \mid l \text{ is a literal falsified by } \epsilon \text{ and } \tau \text{ is an annotation}\}$.

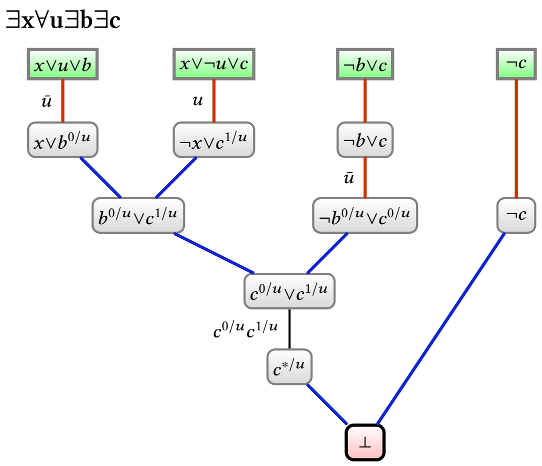
Instantiation. If C_i is derived by instantiating C_j by τ and $C'_j \neq \top$, then we set $C'_i = \mathsf{inst}(\tau, C'_j)$. Obviously, $C'_i \subseteq C_i$. If $C'_i = \top$ then also $C'_j = \top$ and ϵ satisfies C_j by inductive hypothesis. Hence ϵ also satisfies C_i in the sense of condition 2.



(a) IR-calc proof DAG for Example 8



(b) IR-calc proof DAG for Example 9



(c) IRM-calc proof DAG for Example 9

Fig. 7. Proof DAGs for Examples 8 and 9

Resolution. Assume now that C_i was derived by resolution from C_j and C_k with pivot $x^\tau \in C_j$ and $\neg x^\tau \in C_k$. We distinguish four cases.

In the first case we have $x^\tau \in C'_j$ and $\neg x^\tau \in C'_k$. We perform the resolution step and set $C'_i = (C'_j \setminus \{x^\tau\}) \cup (C'_k \setminus \{\neg x^\tau\})$. We then have $C'_i \subseteq (C_j \setminus \{x^\tau\}) \cup (C_k \setminus \{\neg x^\tau\}) = C_i$, fulfilling condition 1.

In the second case we have $x^\tau \notin C'_j$ and $C'_j \neq \top$, and set $C'_i = C'_j$. Then $C'_i \subseteq C_j \setminus \{x^\tau\} \subseteq C_i$, fulfilling condition 1. (In the case of $\neg x^\tau \notin C'_k$ and $C'_k \neq \top$ we analogously set $C'_i = C'_k$.)

The third case is $x^\tau \in C'_j$ and $C'_k = \top$. Then we set $C'_i = \top$. We have to show condition 2. By inductive hypothesis, ϵ satisfies C_k, hence ϵ satisfies some literal $l \in C_k$. Necessarily, $l \neq \neg x^\tau$, because otherwise ϵ falsifies x^τ and thus $x^\tau \notin C'_i$, contradicting our assumption. Thus $l \in C_k \setminus \{\neg x^\tau\} \subseteq C_i$, which satisfies condition 2. (The case $C'_j = \top$ and $\neg x^\tau \in C'_k$ is analogous.)

In the last case, $C'_j = C'_k = \top$. We set $C'_i = \top$. By inductive hypothesis, ϵ satisfies both C_j and C_k, hence by soundness of the resolution rule also C_i. □

Lemma 11. *Let π be an IRM-calc refutation of Γ and let ϵ be a partial assignment of existential variables from Γ. We can then construct from π a refutation π_ϵ of $\Gamma|_\epsilon$ in polynomial time.*

Proof. The proof is similar in structure, but we have to be careful of the following:

- We also need to consider the merge rule.
- IRM-calc has an instantiation part to the resolution rule, so dummy instantiation steps may have to occur in the restricted proof in place of resolution steps.
- We cannot instantiate $*$ annotations outside of the resolution rule, so we have to instantiate instead by a constant (e.g. 0). This does not cause any invalid inferences as $*$ is more restrictive than constants.
- Annotations may differ between the original and restricted proof on $*$, so the clauses in the restricted proof may not be subsets of the clauses in the original proof. In order to make sure the clauses in the restricted proof are smaller than the original, we inductively define an injection from the restricted clause to the original.
- To define such an injection, it will be convenient to treat contraction of two identical literals to one literal as a separate rule and not perform contraction automatically in the proof system. This does not affect the complexity of the calculus. Contraction can also be thought of as a special case of the merging rule.

Firstly, we have to define the restricted proof π_ϵ and then argue for its validity as an IRM-calc proof. The restriction is done as follows, each clause C in π is given a clause C' in π_ϵ. However, in the cases where C' is derived using the axiom rule but C is satisfied by ϵ, C' must be defined as a tautology $\top$ which will be eventually removed from the proof. Likewise if D is derived using a unary (binary) rule from C_1 (C_1 and C_2), if C'_1 (both C'_1 and C'_2) are $\top$ then D must be as well. For non-tautological clauses D' in π_ϵ we can map the literals back to the literals of the original clause D injectively. This injection we label as f_D, and it is important for us showing that our construction does in fact yield a valid IRM-calc proof.

Details of the restriction and the construction of the injections f_D are shown in Figure 8. We omit the tautological cases we have already discussed since we do not give an injection there. By induction on the derivation depth to derive D we show that this restriction yields a valid IRM-calc refutation π_ϵ of $\Phi|_\epsilon$.

The induction hypothesis states that any derived clause C' in the restricted proof corresponding to clause C in the original proof has a valid derivation $\pi_{C'}$. Further, if C' is non-tautologous there is an injection $f_C : C' \to C$, where $f_C(l^{\sigma'}) = l^\sigma$ and σ satisfies the following: $\mathrm{dom}(\sigma') = \mathrm{dom}(\sigma)$ and

Rule	Original proof step leading to clause D	Proof step restricted by ϵ & case condition	Construction of f_D
A	$\dfrac{}{\bigvee_{l\in C, l\in\exists} l^{[\tau]}}$ C is a clause in ϕ. τ falsifies all $\forall$ lit in C.	$\dfrac{}{\bigvee_{l\in C, l\in\exists, \mathrm{var}(l)\notin\mathrm{dom}(\epsilon)} l^{[\tau]}}$ when ϵ does not satisfy C.	$f_D(l^\tau) = l^\tau$
I	$\dfrac{C}{\bigvee_{l^\tau\in C} l^{\tau\circ\sigma}}$	$\dfrac{C'}{\bigvee_{l^\tau\in C'} l^{\tau\circ\sigma}}$	$f_D(\mathsf{inst}(\sigma, l^\tau))$ $= \mathsf{inst}(\sigma, f_C(l^\tau))$
M1	$\dfrac{C = K \vee l^\tau \vee l^\sigma}{K \vee l^\xi}$	$\dfrac{K' \vee l^{\tau'} \vee l^{\sigma'}}{K' \vee l^{\xi'}}$ when $f_C(l^{\tau'}) = l^\tau$ and $f_C(l^{\sigma'}) = l^\sigma$	$f_D(x) = \begin{cases} f_C(x), & x \in K' \\ l^\xi, & x = l^{\xi'} \end{cases}$
M2		$\dfrac{C' = K' \vee l^{\tau'}}{K' \vee l^{\tau'}}$ when $f_C(l^{\tau'}) = l^\tau$ and $\forall l^{\sigma'} \in C', f_C(l^{\sigma'}) \neq l^\sigma$	$f_D(x) = \begin{cases} f_C(x), & x \in K' \\ l^\xi, & x = l^{\tau'} \end{cases}$
M3		$\dfrac{C'}{C'}$ $\forall l^{\sigma'} \in C', l^\tau \neq f_C(l^{\sigma'}) \neq l^\sigma$	$f_D(x) = f_C(x)$
R1	$\dfrac{C_1 \qquad C_2}{\mathsf{inst}(\xi, K_1) \vee \mathsf{inst}(\sigma, K_2)}$ $C_1 = K_1 \vee x^{\tau\circ\sigma}$ $C_2 = K_2 \vee \neg x^{\tau\circ\xi}$ τ, σ, ξ pairwise disjoint $\mathsf{rng}(\tau) = \{0, 1\}$	$\dfrac{K_1' \vee x^{\tau\circ\sigma'} \quad K_2' \vee \neg x^{\tau\circ\xi'}}{\mathsf{inst}(\xi', K_1') \vee \mathsf{inst}(\sigma', K_2')}$ when $f_{C_1}(x^{\tau\circ\sigma'}) = x^{\tau\circ\sigma}$ and $f_{C_2}(\neg x^{\tau\circ\xi'}) = \neg x^{\tau\circ\xi}$	$f_D(\mathsf{inst}(\beta, l^\alpha)) =$ $\begin{cases} \mathsf{inst}(\beta, f_{C_1}(l^\alpha)) & l^\alpha \in C_1' \\ \mathsf{inst}(\beta, f_{C_2}(l^\alpha)) & l^\alpha \in C_2' \end{cases}$
R2		$\dfrac{C_1' \quad C_2'}{\mathsf{inst}(\xi', C_1')}$ $\forall l^\alpha \in C_1', f_{C_1}(l^\alpha) \neq x^{\tau\circ\sigma}$ $\xi'(u) = \begin{cases} 0 & \xi(u) = * \\ \xi(u) & \text{else} \end{cases}$	$f_D(\mathsf{inst}(\xi', l^\alpha))$ $= \mathsf{inst}(\xi, f_{C_1}(l^\alpha))$
R3		$\dfrac{K_1' \vee x^{\tau\circ\sigma'} \qquad \top}{\top}$ $f_{C_1}(x^{\tau\circ\sigma'}) = x^{\tau\circ\sigma}$	

Fig. 8. Transformation of IRM-calc proof steps under the restriction ϵ with conditions and construction of the injection $f_D : D' \longrightarrow D$ used in Lemma 11, where D is the clause derived in our line. Each rule also has tautological cases where the premises are tautologies, which are omitted here. $l \in \exists$ denotes that l is existential.

for every $c/u \in \sigma'$ there is exactly one $d/u \in \sigma$ where $d = c$ or $d = *$. If C' is tautologous then C is satisfied by ϵ.

The purpose of this injection is to ensure that in the restricted proof the clauses have at most the number of literals in the original proof. This along with the condition on tautological clauses ensures that the restricted proof ends in $\bot$. The condition on f_C regarding annotations allows resolution to occur whenever the pivot literals are present.

Base case. In the axiom step (A), the clause in the restricted proof either has some literals removed, or is set to $\top$ when ϵ satisfies the clause. We detail the first case in Figure 8 and define the injection f by mapping identical literals.

Instantiation. An instantiation step (I) keeps the number of literals exactly the same, giving rise to a bijection between literals in C and $\mathsf{inst}(\sigma, C)$. This allows us to carry over the same injection f to the next line.

If D' is tautologous then C' is as well. By the induction hypothesis ϵ satisfies C and since instantiation does not change the existential literals then ϵ satisfies D.

Merging. Consider the first merge case (M1) in Figure 8. In the restricted clause the literals l^τ and l^σ become $l^{\tau'}$ and $l^{\sigma'}$, respectively. (M1) is the case where both these literals are present. Some $*$ in the annotations τ, σ might be 0 or 1 in τ' or σ'. However, the domains of τ, τ', σ, and σ' are all equal. The restricted proof will contain a merge of $l^{\tau'}$ and $l^{\sigma'}$.

In order to satisfy the annotation condition, we note that wherever a $*/u$ appears in ξ', a $*/u$ will appear in ξ, since it either comes from a $*$ itself or a $0/1$ conflict. Either the $0/1$ conflict is present in the original merge or there is a $*$ there by the condition from the induction hypothesis. This means our condition is satisfied.

Now consider the second merge case (M2) in Figure 8. If the restricted clause contains only one of the literals, say $l^{\tau'}$, the merge step is not performed in π_ϵ. The injection is constructed straightforwardly: the literal $l^{\tau'}$ in the new proof is mapped to the merged literal in the original proof and all other literals remain the same. The third case (M3) is similar to the second, but simpler.

Merging does not affect the presence of a literal that is satisfied by ϵ. If D' is tautologous then C' is as well. By the induction hypothesis ϵ satisfies C and since instantiation does not change which existential literals are present then ϵ satisfies D.

Contraction. Contraction is a special case of merging, so the same argument applies.

Resolution. Consider the first resolution case (R1) in Figure 8, this is the case where both pivot literals are present so we perform resolution again. In Figure 8 the way the rule is defined is slightly different to its definition in Figure 5, using $\circ$ instead of $\cup$ to make it clear that the annotations in τ takes precedent. The induction hypothesis gives us that τ, σ', ξ' also are disjoint so this is indeed the same as Figure 5 once we prove the induction hypothesis.

To show this inductive step for (R1), we can easily see that the function f_D is an injection as it maps literals based on whether they are from C_1' or C_2' with $f_{C_1'}$ and $f_{C_2'}$ respectively, which were injections via the induction hypothesis. As we do not perform contraction in this step each literal is genuinely only from one of the parents. We need to show the required properties of our inductive claim hold, that the annotation domains for literals remain the same under f, and furthermore that every c/u in the annotation of l in the restricted proof becomes d/u in the annotation of $f_D(l)$ where $d = c$ or $d = *$. Suppose, without loss of generality, $l^{\alpha'} \in K_1'$ and $f_{C_1}(l^{\alpha'}) = l^\alpha$. Since $\mathrm{dom}(\alpha) = \mathrm{dom}(\alpha')$ and $\mathrm{dom}(\xi) = \mathrm{dom}(\xi')$, then $\mathrm{dom}(\alpha \circ \xi) = \mathrm{dom}(\alpha) \cup \mathrm{dom}(\xi) = \mathrm{dom}(\alpha') \cup \mathrm{dom}(\xi') = \mathrm{dom}(\alpha' \circ \xi')$. This shows that the domains remain the same under f_D, which is one part of the condition. For the remaining part we consider two cases; either $u \in \mathrm{dom}(\alpha)$ or $u \notin \mathrm{dom}(\alpha)$, and we let $c \in \{0, 1\}$.

For $u \in \mathrm{dom}(\alpha)$ we have

$$c/u \in \alpha \Rightarrow c/u \in \alpha' \Rightarrow c/u \in \alpha' \circ \xi' \qquad \text{and} \qquad */u \in \alpha' \Rightarrow */u \in \alpha \Rightarrow */u \in \alpha \circ \xi.$$

Now suppose $u \in \mathrm{dom}(\alpha \circ \xi)$ and $u \notin \mathrm{dom}(\alpha)$. Then

$$c/u \in \xi \Rightarrow c/u \in \xi' \Rightarrow c/u \in \alpha' \circ \xi' \qquad \text{and} \qquad */u \in \xi' \Rightarrow */u \in \xi \Rightarrow */u \in \alpha \circ \xi.$$

Hence the inductive claim holds.

Consider now the second case (R2). Here C_1' is simply instantiated, crucially the pivot literal is missing from C_1', so f_D is an injection. In order to show the inductive condition, we suppose again that $l^{\alpha'} \in K_1'$ and $f_{C_1}(l^{\alpha'}) = l^{\alpha}$. Since $\mathrm{dom}(\alpha) = \mathrm{dom}(\alpha')$ and $\mathrm{dom}(\xi) = \mathrm{dom}(\xi')$, then $\mathrm{dom}(\alpha \circ \xi) = \mathrm{dom}(\alpha' \circ \xi')$. Now let $c \in \{0,1\}$ and suppose $u \in \mathrm{dom}(\alpha)$. As in (R1) we have $c/u \in \alpha \Rightarrow c/u \in \alpha' \Rightarrow c/u \in \alpha' \circ \xi'$ and $*/u \in \alpha' \Rightarrow */u \in \alpha \Rightarrow */u \in \alpha \circ \xi$.

Now suppose $u \in \mathrm{dom}(\alpha \circ \xi)$ and $u \notin \mathrm{dom}(\alpha)$. $*/u$ cannot be in ξ' because $*$ annotations become 0. All 0/1 annotations remain the same. This satisfies the inductive condition.

Finally, consider the third case (R3), which has a tautological resolvent. By induction hypothesis ϵ satisfies C_2, but ϵ cannot satisfy $\neg x$ since this would prevent any x literal from appearing anywhere in the restricted proof, such as the one we require for $f_{C_1}(x^{\tau\circ\sigma'}) = x^{\tau\circ\sigma}$. Hence ϵ satisfies K_2 and thus satisfies D.

If ϵ satisfies C_1 and C_2 then ϵ satisfy D. □

We now proceed with property 2 of our general approach to strategy extraction, i.e., we show that the strategy for the universal player on U can be read off from π_ϵ. In fact, we show a slightly more general statement for arbitrary IR-calc proofs.

Lemma 12. *Let π be an IR-calc refutation of a QBF starting with a block of universally quantified variables U. Consider the set of annotations μ on variables U that appear anywhere in π. Then μ is non-contradictory.*

Proof. Without loss of generality, we can assume that the proof π is connected. The proof proceeds by induction on the derivation depth. Let μ_C denote the set of annotations to variables in U appearing anywhere in the derivation of C (i.e., we only consider the connected component of the proof dag with sink C). The induction hypothesis states:

(i) The set μ_C is non-contradictory.
(ii) For every literal $l^\sigma \in C$, it holds that $\mu_C \subseteq \sigma$.

Base case. Condition (i) is satisfied by the axioms, because we are assuming that there are no complementary literals in clauses in the matrix. Condition (ii) is satisfied because all existential literals are at a higher level than the variables of U.

Instantiation. Let $u \in U$ and $C = \mathrm{inst}(c/u, C')$ in the proof π (note that we can split up all instantiation steps into steps with a single variable). By induction hypothesis, u either appears in the annotations of all the literals l^ξ in C' or it does not appear in any of them. In the first case, the instantiation step is ineffective. Condition (i) holds since we make no change of μ_C from the previous case. Condition (ii) holds because no new annotations appear either.

In the second case, c/u is added to all literals in C. By the induction hypothesis, u does not appear in any annotation of any clause in the sub-proof deriving C' so Condition (i) holds once we add c/u to μ_C. Since $\mu_{C'} \subseteq \xi$ then $\mu_C \subseteq \xi \cup \{c/u\}$ satisfying Condition (ii).

Resolution. Let C be derived by resolving $x^\tau \vee C_1$ and $\neg x^\tau \vee C_2$.

No new annotations are introduced in Resolution. μ_C is now the union of μ_{C_1} and μ_{C_2}. Since both μ_{C_1} and μ_{C_2} are subsets of τ by the induction hypothesis, they are not contradictory with each other, hence Condition (i) holds.

Let $u \in U$. Consider the following cases:

Case 1. $c/u \in \tau$. By induction hypothesis, c/u appears in all annotations of C_1, C_2 and hence in all annotations of the resolvent.

Case 2. $u \notin \mathrm{dom}(\tau)$. Then u does not appear as annotation anywhere in the derivation of either of the antecedents and neither it will appear in the resolvent nor μ_C.

Hence we show Condition (ii) holds. □

Corollary 13. *We can show that Lemma 12 also holds for IRM-calc.*

Proof. To do this we add a third condition to the inductive claim.

(iii) $*/u \notin \mu_C$, for any $u \in U$.

Since $*$ is only introduced by the merging rule, we argue that we do not obtain any annotations involving $*/u$ for $u \in U$. This holds because by condition (i) there are never contradictory annotations on u. □

We are now in a position to efficiently extract winning strategies.

Theorem 14. *The systems IR-calc and IRM-calc have efficient strategy extraction, i.e., from a refutation of a formula we can extract a winning strategy for the universal player in polynomial time.*

Proof. Given a QBF $\exists E \forall U. \Phi$, an IR-calc (resp. IRM-calc) refutation π of it, and an assignment ϵ to the leftmost existential block E, we use Lemma 10 (resp. Lemma 11) to construct a proof π_ϵ of $\forall U. \Phi|_\epsilon$. From this we obtain an assignment to the universal variables U by collecting all the assignments μ to U appearing in annotations in π_ϵ (or instantiations when we have the empty clause instantiated immediately); any variable not appearing in π_ϵ is given an arbitrary value.

To obtain $\pi_{\epsilon,\mu}$, remove occurrences of U-variables from the annotations in the proof π_ϵ. This yields a valid refutation because by Lemma 12 (respectively Corollary 13) for each variable in U only a single-value constant annotation can appear in the entire proof π_ϵ.

For any QBF $\Gamma = \exists E \forall U. \Phi$ and assignment ϵ to E, the above construction provides an IR-calc (IRM-calc) refutation $\pi_{\epsilon,\mu}$ of $\Phi|_{\epsilon \cup \mu}$. This process is iterated until no universal variables are left in the formula. Hence we get an IR-calc (IRM-calc) refutation of whatever was left from the matrix of Γ. Since an IR-calc (IRM-calc) refutation of a formula with no universal variables is in fact a classical propositional resolution refutation, we are left with an unsatisfiable formula, i.e. a formula with no winning move for the existential player. Hence, all the considered assignments correspond to a game won by the universal player. Since this process works for any assignment made by the existential player, it yields a winning strategy for the universal player. □

The soundness of IR-calc (IRM-calc) follows directly from Theorem 14.

Corollary 15. *The calculi IR-calc and IRM-calc are sound.*

We remark that an alternative proof of soundness of IR-calc by simulation has been given in [33].

3.4 Completeness and simulations of known QBF systems

In this section we prove that IR-calc and IRM-calc simulate the main existing resolution-based QBF proof systems. As a by-product, this also shows completeness of our proof systems IR-calc and IRM-calc. We start by simulating Q-resolution, which is even possible with our simpler calculus IR-calc.

Theorem 16. *IR-calc p-simulates Q-Res.*

Proof. Let $C_1, \dots, C_k$ be a Q-Res proof. We translate the clauses into $D_1, \dots, D_k$, which will form the skeleton of a proof in IR-calc.

- For an axiom C_i in Q-Res, D_i is obtained from C_i by the axiom rule of IR-calc, i.e., we remove all universal variables and add annotations.
- If C_i is obtained via $\forall$-reduction from C_j, then $D_i = D_j$.
- Consider now the case that C_i is derived by resolving C_j and C_k with pivot variable x. Then $D_j = x^\tau \vee K_j$ and $D_k = \neg x^\sigma \vee K_k$. We instantiate to get $D'_j = \mathsf{inst}(\sigma, D_j)$ and $D'_k = \mathsf{inst}(\tau, D_k)$. Define D'_i as the resolvent of D'_j and D'_k. In order to obtain D_i we must ensure that there are no identical literals with different annotations. For this consider the set $\zeta = \left\{c/u \mid c/u \in \mu, l^\mu \in D'_i\right\}$ and define $D_i = \mathsf{inst}(\zeta, D'_i)$. This guarantees that D_i does not contain more literals than C_i.

It remains to show that the resolution steps are valid, i.e., we need to show the following:

Claim. In the last item above, τ and σ are not contradictory and ζ does not contain contradictory annotations.

We will prove inductively on the number of lines the following:

Induction hypothesis. For all existential literals l we have $l \in C_i$ if and only if $l^{\mu} \in D_i$ for some annotation μ. Additionally, if $0/u \in \mu$ for a literal u, then $u \in C_i$ (where for a variable x, we equivalently denote the annotation $1/x$ by $0/\neg x$).

Before proving the induction we argue that this yields the claim above. Assume for a contradiction that τ contradicts σ. This means that for some universal variable u, both u and $\neg u$ appear in C_i, which is not allowed; similarly if ζ contains contradictory annotations.

We now show the inductive claim by induction on the proof length.

Base case. This concerns the axiom step from the first bullet point above. $l^{\mu} \in D_i$ if and only if $l \in C_i$ by definition. As annotations falsify all universal literals in the original clause, $0/u \in \mu$ for literal u implies $u \in C_i$.

Inductive step. Axioms are handled as in the base case.

$\forall$-reduction. Suppose C_i is obtained via universal reduction from C_j by reducing the universal literal u. Then u has the highest level among all literals in C_j. Using the inductive hypothesis, u therefore does not appear in any annotations of D_j. As $D_i = D_j$, u also does not appear in annotations in D_j and the inductive claim carries over from the inductive hypothesis on D_j.

Resolution. Suppose that C_i is derived by resolving C_j and C_k over variable x, and $D_j = x^{\tau} \vee K_j$ and $D_k = \neg x^{\sigma} \vee K_k$. Then $l \in C_i$ if and only if $l \in C_j \setminus \{x\} \cup C_k \setminus \{\neg x\}$. By construction and inductive hypothesis, the latter is equivalent to $l^{\mu} \in K_j \cup K_k$ for some annotation μ, which is equivalent to $l^{\mu'} \in D_i$ for some potentially different annotation μ' obtained by instantiation in the construction of D_i.

For the second inductive claim, if $0/u \in \mu$ then there is some literal $l^{\mu' \circ \sigma} \in D_i'$ (with $l^{\mu'} \in D_j$) such that $0/u \in \mu' \circ \sigma$. If $0/u \in \mu'$ then $u \in C_j$ by inductive hypothesis, and if $0/u \in \sigma$ then $u \in C_k$, again by inductive hypothesis; hence $u \in C_i$. □

Corollary 17. *The system IR-calc is refutationally complete.*

Despite its simplicity, IR-calc is powerful enough to also simulate the expansion proof system ∀Exp+Res from [48].

Theorem 18. *IR-calc p-simulates ∀Exp+Res.*

Proof. Let $C_1, \dots, C_k$ be an ∀Exp+Res proof. Transform it into an IR-calc proof $D_1, \dots, D_k$ as follows. If C_i is an axiom from clause C and assignment τ, construct D_i by taking the IR-calc axiom rule for C and then the instantiation $\mathsf{inst}(\tau, C)$. If C_i is a resolvent of C_j, C_k over variable x^{τ}, derive D_i by resolving D_j, D_k over variable x^{τ}. This yields a valid IR-calc proof because $l^{\mu} \in D_i$ iff $l^{\mu} \in C_i$, which is preserved under applications of both rules. □

We now come to the simulation of a more powerful system than Q-resolution, namely LQ-Res from [3]. We show that this system is simulated by IRM-calc. Compared to Theorem 16, the proof uses a similar, but more involved technique.

Theorem 19. *IRM-calc p-simulates LQ-Res.*

Proof. Consider an LQ-Res refutation $C_1, \dots, C_n$. We construct clauses $D_1, \dots, D_n$, which will form the skeleton of the IRM-calc proof. The construction will preserve the following four invariants for $i = 1, \dots, n$.

(1) For an existential literal l, it holds that $l \in C_i$ iff $l^{\mu} \in D_i$ for some μ.

(2) The clause D_i has no literals l^{μ_1} and l^{μ_2} such that $\mu_1 \neq \mu_2$.
(3) If $l^\mu \in D_i$ with $0/u \in \mu$, then $u \in C_i$ or $u^* \in C_i$, likewise if $l^\mu \in D_i$ with $1/u \in \mu$, then $\neg u \in C_i$ or $u^* \in C_i$.
(4) If $l^\mu \in D_i$ with $*/u \in \mu$, then $u^* \in C_i$.

The actual construction proceeds as follows. If C_i is an axiom, D_i is constructed by the axiom rule of IRM-calc from C_i. If C_i is a $\forall$-Red step or $\forall$-Red* step of C_j with $j < i$, then we set D_i equal to D_j. If C_i is obtained by a resolution step from C_j and C_k with $j < k < i$, the clause D_i is obtained by a resolution step from D_j and D_k, yielding clause K, and by performing some additional steps on K. Firstly, we let $\theta = \{c/u \mid c \in \{0,1\}, c/u \in \mu, l^\mu \in K\} \cup \{0/u \mid */u \in \mu, l^\mu \in K\}$ and perform instantiation on K by substitutions in θ, in any order, to derive K'. After this all annotations in K' have the same domain. We merge all pairs of literals $l^\sigma, l^\tau \in K'$ with $\tau \neq \sigma$ (in any order) to derive D_i.

To show that this construction yields a valid IRM-calc refutation, we first need to prove the invariants above. This proceeds by induction on i.

Base case. Because we do not remove or add any existential literals in the axiom case, condition (1) holds. Likewise we do not create duplicates, so (2) holds. Any $0/1$ annotation corresponds exactly to the opposite literal appearing in the clause, by definition of the axiom rule, hence (3) holds. As we do not obtain any $*$ annotations from axioms, (4) holds.

Inductive step. Again, axioms are handled as in the base case.

Reduction. Consider a $\forall$-Red step (resp. $\forall$-Red* step) from C_j to C_i on universal variable u (resp. u^*). Because we do not alter the existential literals in a reduction step and the corresponding clause D_i in the IRM-calc proof remains unchanged, conditions (1) and (2) are satisfied by induction hypothesis. For conditions (3) and (4) we note that D_j cannot contain any annotations involving u (resp. u^*). This holds because u (resp. u^*) would only appear as annotation on existential literals with level higher than u. These cannot exist as they would be blocking the reduction by (1).

Resolution step. For this consider C_j, C_k being resolved in LQ-Res to obtain C_i. Only the pivot variable is removed from C_j and C_k. By condition (2) of the inductive hypothesis for D_j and D_k, the pivot is also completely removed when resolving D_j and D_k. Hence D_i fulfils (1).

We now argue for (2). By induction hypothesis we know that there can be at most two copies of each variable when we derive K. Their annotations have the same domain in K', because instantiation by θ applies the entire domain of all annotations in the clause to all its literals. It then follows that all copies of identical literals are merged into one literal in D_i. Therefore (2) holds for D_i.

To prove (3) consider the case where $l^\mu \in D_i$ with $0/u \in \mu$. The case with $1/u \in \mu$ is analogous. We know that $0/u$ appearing in D_i means that $0/u$ must appear in K' as merging cannot produce a new annotation $0/u$. Existence of $0/u$ in K' means that either $*/u$ appears in K or $0/u$ appears in K. No new annotations are created in a resolution step, so either $*/u$ or $0/u$ must appear in one or more of D_j, D_k. By induction hypothesis this means that u or u^* appears in $C_j \cup C_k$, hence also in C_i.

To show condition (4), let $l^\mu \in D_i$ with $*/u \in \mu$. Then either $*/u$ is present in K', or $0/u$ and $1/u$ are present in K' and will be merged. In the first case it is clear that some $*/u$ annotation appears in K and thus in D_j or in D_k, in which case from (4) of the induction hypothesis u^* must appear in C_i. In the second case it is possible that $0/u$ in K' was obtained from $*/u$ in K. Thus as already argued, u^* must appear in C_i. If instead $1/u$, $0/u$ are both present in K then they must come from the original clauses D_j, D_k. If they both appear in the same clause D_j, then by condition (3) it must be the case that u^* appears in C_j and thus in C_i. If, however, they appear in different clauses, then by (3) either of the clauses C_j, C_k contains u^* or they contain literals over u of opposite polarity. Both situations merge the literals to $u^* \in C_i$.

We now show that these invariants imply that we indeed obtain a valid IRM-calc proof. We only need to consider the resolution steps. Suppose $x^{\mu_1} \in D_j$ and $\neg x^{\mu_2} \in D_k$ where C_j and C_k are resolved on x to get C_i in the LQ-Res proof. To perform the resolution step between D_j and D_k we need to ensure that we do not have $c/u \in \mu_1$, $d/u \in \mu_2$ where $c \neq d$ or $c = d = *$. Assume on the contrary that $*/u \in \mu_1$ and $c/u \in \mu_2$. By (4) we have $u^* \in C_j$, and by (3) some literal of u is in C_k. But as $\text{lv}(u) < \text{lv}(x)$ the LD-resolution of C_j and C_k on variable x is forbidden, giving a contradiction. Similarly, if there is $0/u \in \mu_1$ and $1/u \in \mu_2$, then either we get the same situation or we have two opposite literals of u in the different clauses C_j, C_k. In either case the resolution of C_j, C_k is forbidden. Hence the IRM-calc proof is correct.

By the inductive claim, the construction above yields a valid IRM-calc proof. As the clauses D_i in this proof always have at most the same number of literals as C_i, we end with the emtpy clause and hence obtain a refutation. As all steps of the construction can be performed in polynomial time, we obtain a p-simulation. □

Corollary 20. *The system IRM-calc is refutationally complete.*

4 THE STRATEGY EXTRACTION TECHNIQUE: LOWER BOUNDS FOR CDCL RESOLUTION CALCULI

We now proceed towards a proof-complexity analysis of the lattice of QBF resolution systems. Our first aim is to find formulas easy for expansion systems, but hard for CDCL systems. To do this we show a new and conceptually interesting lower bound for QU-Res (and thus for Q-Res). This lower bound constitutes in fact a new lower bound technique that is widely applicable (cf. Section 1.2). We illustrate this technique here with an exponential lower bound for parity formulas in QU-Res (Section 4.1). This provides a separation between QU-Res and ∀Exp+Res. We then lift this lower bound to the long-distance systems LQ-Res and LQU$^+$-Res (Section 4.2).

4.1 Lower bounds for Q-Res and QU-Res via strategy extraction

The lower bound argument hinges on strategy extraction, which is a widely used paradigm in QBF solving and proof systems. We recall that QU-Res admits strategy extraction via a computationally very restricted model, namely decision lists.

Definition 21 (decision list [63]). *A* decision list $D = (t_1, c_1), \dots, (t_n, c_n)$ *is a finite sequence of pairs where t_i is a term and $c_i \in \{0, 1\}$ is a Boolean constant. Additionally, the last term is the empty term (equivalent to true). For an assignment μ, a decision list D evaluates to c_i if i is the least index such that $\mu \models t_i$, in such case we say that (t_i, c_i)* triggers *under μ.*

Winning strategies in form of decision lists can be efficiently extracted from QU-Res proofs:

Theorem 22 (Balabanov, Jiang, Widl [3, 5]). *Given a Q-Res or QU-Res refutation π of QBF ϕ, there exists a winning strategy for the universal player for ϕ, such that each of its strategies for the universal variables is computable by a decision list of size polynomial in $|\pi|$.*

Balabanov et al. use a different form than decision lists, but it is semantically equivalent. We deem decision lists as more intuitive for our purposes. Note that under our definition, a strategy for a universal variable may take as input the outputs of strategy functions with a smaller index (similarly as in the strategy construction by Goultiaeva et al. [42]).

The general idea behind our lower bound technique is as follows. First, we observe that we can define a family of QBFs Φ_f such that every winning strategy of the universal player *must* compute a unique Boolean function f. If we know that strategy extraction is possible by a weak computational model, say AC^0, we can carefully choose the Boolean formula Φ_f such that the unique winning strategy f cannot be computed by AC^0 circuits. As the extracted strategy is polynomial in the proof, this implies a lower bound on the proof size. Thus we immediately turn circuit lower bounds to lower bounds for the proof size.

We will now implement this idea for the *parity function* $\textsc{Parity}(x_1, \dots, x_n) = x_1 \oplus \dots \oplus x_n$. The parity function determines if the number of true variables is odd ($\oplus$ denotes exclusive-or). This is the classical example of a function not computable in AC^0.

Theorem 23 (Furst, Saxe, Sipser [36], Håstad [43]). *$\textsc{Parity} \notin \mathsf{AC}^0$. In fact, every family of bounded-depth circuits computing $\textsc{Parity}$ is of exponential size.*

To obtain parity formulas, consider the QBF $\Phi = \exists X \forall z \exists T.\ (F^+ \land F^-)$ where F^+ is a CNF encoding of $z \lor \textsc{Parity}(X)$ and F^- encodes $\neg z \lor \neg\textsc{Parity}(X)$. Both F^+ and F^- use additional variables in T. More precisely, for $N > 1$ define QParity_N as follows.

Definition 24 (QParity formulas). *Let* $\text{xor}(o_1, o_2, o)$ *be the set of clauses*

$$\{\neg o_1 \lor \neg o_2 \lor \neg o,\ o_1 \lor o_2 \lor \neg o,\ \neg o_1 \lor o_2 \lor o,\ o_1 \lor \neg o_2 \lor o\},$$

which defines o to be $o_1 \oplus o_2$. Define QParity_N as

$$\exists x_1, \dots, x_N\, \forall z\, \exists t_2, \dots, t_N.\ \text{xor}(x_1, x_2, t_2) \cup \bigcup_{i=3}^{N} \text{xor}(t_{i-1}, x_i, t_i) \cup \{z \lor t_N, \neg z \lor \neg t_N\}.$$

Note that since we want to encode parity in CNF, i.e. a bounded-depth formula, and $\textsc{Parity} \notin \mathsf{AC}^0$, we need to use further existential variables (recall that existential AC^0 characterises all of NP). Choosing existential variables t_i to encode the prefix sums $x_1 \oplus \dots \oplus x_i$ of the parity $x_1 \oplus \dots \oplus x_N$ provides the canonical CNF formulation of parity. This corresponds to a Tseitin encoding of parity.[6]

To use the lower bound of Theorem 23 we need to verify that QU-Res enables strategy extraction in AC^0. This holds as decision lists can be turned into bounded-depth circuits.

Lemma 25. *If f_D can be represented as a polynomial-size decision list D, then $f_D \in \mathsf{AC}^0$.*

Proof. Let $S = \{i \mid (t_i, 1) \in D\}$ be the indices of all pairs in D with 1 as the second component. Observe that f_D evaluates to 1 under μ iff one of the t_i with $i \in S$ triggers under μ. For each t_i with $i \in S$ construct a function $f_i = t_i \land \bigwedge_{l=1}^{i-1} \neg t_l$. Construct a circuit for the function $\bigvee_{i \in S} f_i$, which is equivalent to f_D and is computable in AC^0 as all t_i are just terms. □

We can now put everything together and turn the circuit lower bound of Theorem 23 into a lower bound for proof size in QU-Res.

Theorem 26. *Any QU-Res refutation of QParity_N is of exponential size in N.*

Proof. The strategy for the universal variable z may only depend on the variables $x_1, \dots, x_N$ and it must be so that the matrix evaluates to false under the given assignment μ to the x_i variables. By inspecting the matrix, z must be set equal to $x_1 \oplus \dots \oplus x_N$, i.e. there is a unique strategy for the variable z in QParity_N, which is the $\textsc{Parity}$ function on N variables. From Theorem 22, there is a polynomial-time algorithm for constructing a decision list D_N from any QU-Res refutation of QParity_N. Such decision list can be converted in polynomial time into a circuit with bounded depth by Lemma 25. Hence, the decision list and therefore the proof must be of exponential size in N due to Theorem 23. □

[6]While we here aim for hard formulas, we remark that the hardness is sensitive to the choice of the encoding and in particular the quantifier prefix. In [13] it is shown that the use of additional extension variables, which are placed as leftmost as possible in the quantifier prefix, results in short proofs.

In contrast to this lower bound, the QParity formulas are easy in ∀Exp+Res.

Lemma 27. *The formulas QParity$_N$ have polynomial-size ∀Exp+Res refutations.*

Proof. Instantiate all clauses in both polarities of z, which generates the clauses $\operatorname{xor}(x_1, x_2, t_2^{0/z}) \cup \bigcup_{i=3}^{N} \operatorname{xor}(t_{i-1}^{0/z}, x_i, t_i^{0/z}) \cup \{t_N^{0/z}\}$ and $\operatorname{xor}(x_1, x_2, t_2^{1/z}) \cup \bigcup_{i=3}^{N} \operatorname{xor}(t_{i-1}^{1/z}, x_i, t_i^{1/z}) \cup \{\neg t_N^{1/z}\}$.

Inductively, for $i = 2, \ldots, N$ derive clauses representing $t_i^{0/z} \Leftrightarrow t_i^{1/z}$. This lets us derive a contradiction using the clauses $t_N^{0/z}$ and $\neg t_N^{1/z}$. □

Theorem 26 together with Lemma 27 immediately give the following separations.

Corollary 28. *Q-Res and QU-Res do not simulate ∀Exp+Res, IR-calc, IRM-calc.*

This also has consequences for the complexity of strategy extraction in ∀Exp+Res.

Corollary 29. *Winning strategies for ∀Exp+Res cannot be computed in* AC^0. *This even holds when the system ∀Exp+Res is restricted to formulas with constant quantifier complexity.*

Proof. The formulas QParity$_N$ have polynomial-size ∀Exp+Res refutations by Lemma 27. Hence we cannot extract strategies in AC^0 as these would compute parity. □

Note, however, that strategy extraction for ∀Exp+Res and in fact even IRM-calc is in P due to Theorem 14.

4.2 Extending the lower bound to LQ-Res and LQU$^+$-Res

We now aim to extend the lower bound from the previous section to stronger QBF proof systems using long-distance resolution. For this we cannot directly use the strategy extraction method from the last section. However, we will slightly modify the QParity formulas and then reduce the hardness of those in the stronger systems to the hardness of QParity in Q-Res. As the modified formulas remain easy for ∀Exp+Res, these lower bounds imply many new separations between the proof systems involved.

We start by extending the lower bound to LQ-Res, which will provide a separation of LQ-Res and ∀Exp+Res. While we cannot use the original QParity formulas (which have short proof in LQ-Res, cf. [29]), we will use a variant of these parity formulas.

Definition 30 (LQParity formulas). *Let* $\operatorname{xor}_l(o_1, o_2, o, z)$ *be the set of clauses* $\{z \vee \neg o_1 \vee \neg o_2 \vee \neg o,\ z \vee o_1 \vee o_2 \vee \neg o,\ z \vee \neg o_1 \vee o_2 \vee o,\ z \vee o_1 \vee \neg o_2 \vee o\}$ *(*xor_l *defines* o *to be equal to* $o_1 \oplus o_2$ *if* $z = 0$*). The formulas LQParity$_N$ are constructed from QParity$_N$ by replacing each occurrence of* $\operatorname{xor}(\ldots)$ *by two copies* $\operatorname{xor}_l(\ldots, z)$ *and* $\operatorname{xor}_l(\ldots, \neg z)$*, yielding*

$$\exists x_1, \ldots, x_N \forall z \exists t_2, \ldots, t_N.\ \operatorname{xor}_l(x_1, x_2, t_2, z) \cup \bigcup_{i=3}^{N} \operatorname{xor}_l(t_{i-1}, x_i, t_i, z)$$

$$\cup \operatorname{xor}_l(x_1, x_2, t_2, \neg z) \cup \bigcup_{i=3}^{N} \operatorname{xor}_l(t_{i-1}, x_i, t_i, \neg z) \cup \{z \vee t_N, \neg z \vee \neg t_N\}.$$

It is easy to verify that the same arguments as for QParity in Section 4.1 also apply to LQParity, yielding:

Proposition 31. *The formulas LQParity$_N$ have polynomial-size ∀Exp+Res refutations, but require exponential-size QU-Res refutations.*

We now want to show that LQParity is hard for LQ-Res by arguing that long-distance steps do not help to refute these formulas. In the next two lemmas we will show that this actually applies to all QBFs Φ meeting the following condition.

Definition 32. *We say that z is* completely blocked in a QBF Φ, *if all clauses of Φ contain the universal variable z and some existential literal l such that* $\mathrm{lv}(z) < \mathrm{lv}(l)$.

Lemma 33. *Let Φ be a QBF and z be completely blocked in Φ. Let further C be a clause derived from Φ by LQ-Res. If C contains some existential literal l such that* $\mathrm{lv}(z) < \mathrm{lv}(l)$, *then $z \in C$, $\neg z \in C$, or $z^* \in C$.*

Proof. We prove the lemma by induction on the derivation depth.

Base case. This is established by the clauses in the matrix of Φ due to the condition that z must be in all matrix clauses and also that all these clauses contain some existential literal that blocks z.

Reduction. The hypothesis is preserved by $\forall$-reduction because a literal over z cannot be $\forall$-reduced if the clause contains an existential literal l with $\mathrm{lv}(z) < \mathrm{lv}(l)$.

Resolution. Consider now two clauses $C_1 = D_1 \vee x$ and $C_2 = D_2 \vee \neg x$ resolved into the clause C_3. If C_3 contains some literal l such that $\mathrm{lv}(z) < \mathrm{lv}(l)$, then one of C_1, C_2 must contain l and from induction hypothesis it must also contain the variable z, which then appears in C_3. □

Lemma 34. *Let Φ be a QBF such that z is completely blocked in Φ and let π be a refutation of Φ such that the variable z is $\forall$-reduced as early as possible. Then the derivation of the empty clause in π does not contain z^* in any of its clauses.*

Proof. Assume that we have a clause C in π that contains z^*. We will argue that C is not necessary to derive the empty clause $\bot$, i.e., there is no path in π from C to $\bot$. Since z^* does not appear in any of the matrix clauses, there must be a resolution step where it is introduced. Consider any such two clauses $C_1 = D_1 \vee x \vee z$ and $C_2 = D_2 \vee \neg x \vee \neg z$ resolved into $C = D_3 \vee z^*$. From the assumption that $\forall$-reductions are carried out as soon as possible, in both clauses C_1 and C_2 there must be some literals that block z and $\neg z$, respectively. From the conditions on LQ-Res, x or $\neg x$ cannot be the blocking literal (it must be that $\mathrm{lv}(x) < \mathrm{lv}(z)$ upon merging). This means that C contains at least one literal b that blocks z^*.

Now we argue that b cannot be resolved away. For contradiction assume that there is a resolution step of some C' and D on b where there is a path from C to C'. Moreover, let that be the first resolution step on b, i.e., b appears in all clauses on the path between C and C'. From Lemma 33, the clause D must contain a literal on the variable z. But this contradicts the conditions of LQ-Res because resolution steps are not permitted on literals b with $\mathrm{lv}(z) < \mathrm{lv}(b)$ if one of the antecedents contains a merged literal z^* and the other contains some literal on z. This means that C does not participate in the derivation of the empty clause $\bot$. □

This enables us to prove the hardness of LQParity in LQ-Res.

Theorem 35. *Any refutation of LQParity$_N$ in LQ-Res has size exponential in N.*

Proof. Any LQ-Res refutation π can be translated in polynomial time into a refutation π' such that $\forall$-reductions are carried out as soon as possible (such a refutation has clauses that are equal to the clauses of π or some universal literals are missing). From Lemma 34, the derivation of $\bot$ in π' contains no occurrences of the merged literal z^*, hence any such clauses can be removed from the refutation. Therefore π' is in fact also a Q-Res refutation. Hence, π must be exponential in N due to Proposition 31. □

From LQParity$_N$, we can get back the clauses of QParity$_N$ by resolving over the universal pivot z using the xor$_l$ clauses. This gives then, together with the short LQ-Res refutations [29], short LQU$^+$-Res refutations of LQParity$_N$.

Our next goal is to extend the lower bound for the parity formulas to the system LQU$^+$-Res, which enables both long-distance and universal resolution. For this we again modify the formula QParity, using a similar technique as in [5]. The trick is essentially to double the universal literals so they form tautological clauses when resolved. This way resolution on universal variables does not give any advantage.

Definition 36 (QUParity formulas). *We define formulas QUParity$_N$ from LQParity$_N$ as follows: replace the universal quantifier $\forall z$ by two new quantifiers $\forall z_1 \forall z_2$ and replace all occurrences of the literal z by $z_1 \vee z_2$ and likewise of $\neg z$ by $\neg z_1 \vee \neg z_2$. This gives the formulas QUParity$_N$*

$$\exists x_1, \ldots, x_N \forall z_1, z_2 \exists t_2, \ldots, t_N . \mathrm{xor}_u(x_1, x_2, t_2, z_1, z_2) \cup \bigcup_{i=3}^{N} \mathrm{xor}_u(t_{i-1}, x_i, t_i, z_1, z_2) \cup$$

$$\mathrm{xor}_u(x_1, x_2, t_2, \neg z_1, \neg z_2) \cup \bigcup_{i=3}^{N} \mathrm{xor}_u(t_{i-1}, x_i, t_i, \neg z_1, \neg z_2) \cup \{z_1 \vee z_2 \vee t_N, \neg z_1 \vee \neg z_2 \vee \neg t_N\},$$

where $\mathrm{xor}_u(o_1, o_2, o, l_1, l_2)$ *is the set of clauses*

$$\{l_1 \vee l_2 \vee \neg o_1 \vee \neg o_2 \vee \neg o,\ l_1 \vee l_2 \vee o_1 \vee o_2 \vee \neg o,\ l_1 \vee l_2 \vee \neg o_1 \vee o_2 \vee o,\ l_1 \vee l_2 \vee o_1 \vee \neg o_2 \vee o\}.$$

It is clear that these formulas are false as the universal player should play both z_1 and z_2 as they would z in LQParity.

We will now assume that in an LQU$^+$-Res refutation we $\forall$-reduce immediately. It is easy to verify that this does not increase proof size (cf. also Proposition 1 in [5]). For QUParity we now show an analogous result to Lemma 33.

Lemma 37. *Let C be a clause in an LQU$^+$-Res refutation of QUParity$_N$ where $\forall$-reduction steps are performed as soon as possible. If C contains some existential literal l such that $\mathrm{lv}(z_2) < \mathrm{lv}(l)$, then either $z_1, z_2 \in C$, or $\neg z_1, \neg z_2 \in C$, or $z_2^* \in C$.*

Proof. The proof is the same as for Lemma 33, except for the possibility of universal resolution steps. As $\forall$-reductions are required to happen as soon as possible, in our induction hypothesis we know that a z_1 literal can only occur together with the corresponding z_2 literal. Therefore resolving on z_1 removes this variable and merges the complementary z_2 literals; hence we get the z_2^* literal.

The merged literals cannot be pivots. Neither can z_2. This holds because we know by induction hypothesis that when z_2 appears unmerged, then also z_1 appears unmerged with the same polarities. Hence resolving with z_2 as the pivot would merge z_1, which is illegal due to the index restriction. □

We now argue that neither long-distance resolution steps nor resolution over universal pivots help to refute QUParity.

Lemma 38. *Any LQU$^+$-Res refutation of QUParity$_N$ is a Q-Res refutation.*

Proof. We first argue for z_2^*. Let $z_2^* \in C$. As we assume that $\forall$-reductions are performed immediately, the literal z_2^* is blocked by an existential literal l when z_2^* is created in C by a long-distance resolution step. Then l cannot be removed from C by resolution as any clause with $\neg l$ in it contains a literal over z_2 by Lemma 37. Also z_2^* cannot be removed via universal resolution. So the empty clause can never be derived from any clause containing z_2^*.

Let us now argue for z_1^* and assume $z_1^* \in C$. If z_1^* is introduced into C by resolving clauses D_1 and D_2, the literals z_1 and $\neg z_1$ in D_1 and D_2, respectively, must be blocked by existential literals.

Therefore by Lemma 37, the clauses D_1 and D_2 also contain z_2 and $\neg z_2$, respectively. Hence also z_2^* is introduced into C and we get back to the previous case.

Finally, universal resolution steps cannot be performed when deriving the empty clause. For universal resolution on z_1, using again Lemma 37 together with the assumption of performing $\forall$-reductions as early as possible leads to the introduction of z_2^*, and we again get back to the case above.

No resolution on z_2 is possible as from Lemma 37 it would cause both literals of z_1 to be merged, which is illegal due to the index restriction in long-distance resolution over universal variables. □

This immediately implies the hardness of QUParity for LQU$^+$-Res because by the previous lemma any LQU$^+$-Res refutation of QUParity$_N$ is a Q-Res refutation, which by Theorem 26 is exponential in size.

Theorem 39. *QUParity$_N$ require exponential-size refutations in LQU$^+$-Res.*

As QUParity$_N$ still remains easy for $\forall$Exp+Res in a proof similar to Lemma 27 we get the following separations.

Corollary 40. *LQU$^+$-Res does not simulate $\forall$Exp+Res, IR-calc, and IRM-calc.*

5 LOWER BOUNDS FOR EXPANSION PROOF SYSTEMS

While the strategy extraction technique from the last section is very effective for CDCL systems, it does not yield lower bounds in expansion systems. Therefore, to obtain such lower bounds we need to use different techniques — and in fact different types of formulas. We first show a lower bound for IR-calc (Section 5.1) and then extend this bound to even IRM-calc (Section 5.2).

5.1 A lower bound in IR-calc for the formulas of Kleine Büning et al.

We present a proof complexity analysis of a well-known family of formulas KBKF(t) first defined by Kleine Büning et al. [51]. Here we prove that the KBKF(t) formulas are hard for IR-calc, which is stronger than Q-Res (Corollary 28). This provides the first non-trivial lower bound for IR-calc, and further even separates the system from LQ-Res.

Definition 41 (Kleine Büning, Karpinski and Flögel [51]). *The formula* KBKF(t) *has prefix* $\exists y_0, y_{1,0}, y_{1,1} \forall x_1 \exists y_{2,0}, y_{2,1} \forall x_2 \dots \forall x_{t-1} \exists y_{t,0}, y_{t,1} \forall x_t \exists f_1 \dots f_t$ *and matrix clauses*

$$\begin{array}{llll}
C_- = \{\neg y_0\} & C_0 = \{y_0, \neg y_{1,0}, \neg y_{1,1}\} & \\
C_i^0 = \{y_{i,0}, x_i, \neg y_{i+1,0}, \neg y_{i+1,1}\} & C_i^1 = \{y_{i,1}, \neg x_i, \neg y_{i+1,0}, \neg y_{i+1,1}\} & \text{for } i \in [t-1] \\
C_t^0 = \{y_{t,0}, x_t, \neg f_1, \dots, \neg f_t\} & C_t^1 = \{y_{t,1}, \neg x_t, \neg f_1, \dots, \neg f_t\} & \\
C_{t+i}^0 = \{x_i, f_i\} & C_{t+i}^1 = \{\neg x_i, f_i\} & \text{for } i \in [t].
\end{array}$$

Let us verify that the KBKF(t) formulas are indeed false QBFs and — at the same time — provide some intuition about them. The existential player starts by playing $y_0 = 0$ because of clause C_-. Clause C_0 forces the existential player to set one of $y_{1,0}, y_{1,1}$ to 0. Assume the existential chooses $y_{1,0} = 0$ and $y_{1,1} = 1$. If the universal player tries to win, he will counter with $x_1 = 0$, thus forcing the existential player again to set one of $y_{2,0}, y_{2,1}$ to 0. This continues for t rounds, leaving in each round a choice of $y_{i,0} = 0$ or $y_{i,1} = 0$ to the existential player, to which the universal counters by setting x_i accordingly. Finally, the existential player is forced to set one of $f_1, \dots, f_t$ to 0. This will contradict one of the clauses $C_{t+1}^0, C_{t+1}^1, \dots, C_{2t}^0, C_{2t}^1$, and the universal player wins.

It is clear from this explanation, that the existential player has exponentially many choices and the universal player likewise needs to uniquely counter all these choices to win. The aim of this section is to show that IR-calc and therefore Q-Res in some sense need to go through all these exponentially many options in order to refute the formula, thus forcing IR-calc and Q-Res proofs of exponential size.

A similar family $\mathrm{KBKF}_{\mathrm{lq}}(t)$ was developed in [5]. These defeat the main advantage LQ-Res has over the original $\mathrm{KBKF}(t)$. In fact these can be shown to have exponential size proofs in LQ-Res.

Definition 42 (Balabanov, Widl, Jiang [5]). *The formula* $\mathrm{KBKF}_{\mathrm{lq}}(t)$ *has quantifier prefix* $\exists y_0, y_{1,0}, y_{1,1} \forall x_1 \exists y_{2,0}, y_{2,1} \forall x_2 \dots \forall x_{t-1} \exists y_{t,0}, y_{t,1} \forall x_t \exists f_1 \dots f_t$ *and matrix clauses*

$$\begin{aligned}
C_- &= \{\neg y_0\}\\
C_0 &= \{y_0, \neg y_{1,0}, \neg y_{1,1}, \neg f_1, \dots, \neg f_t\}\\
C_i^0 &= \{y_{i,0}, x_i, \neg y_{i+1,0}, \neg y_{i+1,1}, \neg f_1, \dots, \neg f_t\} \qquad \text{for } i \in [t-1]\\
C_i^1 &= \{y_{i,1}, \neg x_i, \neg y_{i+1,0}, \neg y_{i+1,1}, \neg f_1, \dots, \neg f_t\} \qquad \text{for } i \in [t-1]
\end{aligned}$$

$$\begin{aligned}
C_t^0 &= \{y_{t,0}, x_t, \neg f_1, \dots, \neg f_t\} \quad & C_t^1 &= \{y_{t,1}, \neg x_t, \neg f_1, \dots, \neg f_t\}\\
F_i^0 &= \{x_i, f_i, \neg f_{i+1}, \dots, \neg f_t\} \quad & F_i^1 &= \{\neg x_i, f_i, \neg f_{i+1}, \dots, \neg f_t\} \qquad \text{for } i \in [t].
\end{aligned}$$

Syntactically, $\mathrm{KBKF}(t)$ and $\mathrm{KBKF}_{\mathrm{lq}}(t)$ are *existential Horn formulas*, i.e., they contain at most one positive existential literal per clause. In fact, they even have a stronger property: C_- is the only clause without a head (a positive existential literal). We will strengthen this in the next lemma by a simple modification such that now all clauses have a head. This invariant will be useful in our proof of hardness as it allows us to keep track of the annotation of the head of each derived clause. The reader may also note that other authors have simplified these formulas by restricting it with $y_0 = 0$. We keep the original definition of $\mathrm{KBKF}(t)$ in order to give our considered formulas such invariants as the one discussed.

Lemma 43. *We can transform every IR-calc refutation π of* $\mathrm{KBKF}(t)$ *(or* $\mathrm{KBKF}_{\mathrm{lq}}(t)$*) into an IR-calc proof π' of y_0 from $G(t) = \mathrm{KBKF}(t) \setminus \{\neg y_0\}$ (or $G(t) = \mathrm{KBKF}_{\mathrm{lq}}(t) \setminus \{\neg y_0\}$). We perform this by:*

1. *deleting every instance of the axiom $\{\neg y_0\}$, and removing the steps where y_0 is a pivot;*
2. *replicating all other steps, using the same rules, same pivots and same annotations when instantiating.*

Proof. The proof is immediate, observing that y_0 cannot gain annotations and the only rules applicable on y_0 are resolution rules between the axiom $\{\neg y_0\}$ and clauses containing y_0. Thus instead of the empty clause we derive $\{y_0\}$. □

After this transformation, which preserves proof length, we can focus on proofs of y_0 from $G(t) = \mathrm{KBKF}(t) \setminus \{\neg y_0\}$ (or $G(t) = \mathrm{KBKF}_{\mathrm{lq}}(t) \setminus \{\neg y_0\}$). Exploiting that all axioms now contain exactly one positive literal we show six invariants, which hold for all clauses in all IR-calc proofs of the formulas (in both the KBKF and $\mathrm{KBKF}_{\mathrm{lq}}$ setting).

Let C be an annotated clause in an IR-calc proof of y_0 from $G(t)$. Then the following invariants hold for C. All invariants are shown by induction on the structure of the directed acyclic graph of the proof. We can assume the induction hypothesis is the statement of invariant in each proof.

Invariant 1. *C has exactly one positive literal $y_{h,a}^{\alpha}$ or f_h^{α} for $h \leq t$, $a \in \{0,1\}$ (or y_0 with no annotation). We call this unique literal the* head *of C and use the indices h and a also in the following invariants to denote its position as well as α for its annotation.*

Proof. We use induction on the number of proof steps.

Base case. Invariant 1 holds when inspecting the axioms of $G(t)$.

For the inductive step we only need to consider the resolution case as annotations do not affect the polarities of the literals.

Resolution. Suppose C is derived from resolving D_1 and D_2 with pivot z. Without loss of generality let z be positive in D_2. Then Invariant 1 holds for C as there are exactly two positive literals between D_1 and D_2, but the head z of D_2 gets removed by resolution. □

The next invariant states that when we order the literals in the clause by the prefix, the head appears leftmost.

Invariant 2. *Let $j \in \{1, \dots, t\}$, $b \in \{0, 1\}$, and β be some annotation.*

(1) If $y^{\alpha}_{h,a}$ is the head of C and $\neg y^{\beta}_{j,b} \in C$ then $j > h$.

(2) If f^{α}_{h} is the head of C then there is no $\neg y^{\beta}_{j,b} \in C$.

(3) If f^{α}_{h} is the head of C and $\neg f^{\beta}_{j} \in C$ then $j > h$.

Proof. Again we show this by induction.

Base case. Invariant 2 holds in all axioms of $G(t)$.

For the inductive step we only need to consider the resolution case as annotations do not affect the indices of the literals. Let C be derived from resolving D_1 and D_2.

Resolution. Let C be derived from resolving D_1 and D_2. Assume first that the resolved variable is $y^{\epsilon}_{k,e}$ in D_2. By induction hypothesis we know that $y^{\alpha}_{h,a}$ or y_0 of C is the head of D_1. We can ignore the case of y_0, since it already appears leftmost and will not be resolved away. If we have a negative literal $\neg y^{\beta}_{j,b} \in C$, then $\neg y^{\beta}_{j,b} \in D_1$ or $\neg y^{\beta}_{j,b} \in D_2$. If in D_1 then $j > h$ by Invariant 2 for D_1. If in D_2 then $j > k$ by Invariant 2 for D_2. As $\neg y^{\epsilon}_{k,e} \in D_1$ we get $k > h$ again by Invariant 2 for D_1. Therefore $j > h$ and Invariant 2 holds for C.

If the resolved variable is $f^{\epsilon}_{k} \in D_2$ then by induction hypothesis we only add negative literals $\neg f^{\beta}_{b}$ from D_1 with $b > k$. In order to falsify the invariant we would need $f^{\alpha}_{h} \in D_1$ as the head, but then we know $h < k < b$, satisfying the invariant. □

The next two invariants explain how the annotation of the head determines all further annotations in the clause.

Invariant 3. *If positive literal $f_i^{\tau} \in C$ then the following two conditions hold: $x_i \in \mathrm{dom}(\tau)$ and all literals in C have exactly the same annotations.*

Proof. By induction.

Base case. This is true in all axioms, and these annotations cannot be removed in the inductive step. We can only alter the annotations uniformly by instantiation.

Resolution. Suppose we have that C is the resolvent of D_1 and D_2. Suppose D_1 contributes our head $f_i^{\tau} \in C$. Using Invariant 2, any resolved variable is $\neg f_j^{\epsilon} \in D_1$ and must resolve with a clause where all literals have the same annotations as well, hence C has the same annotation in every literal.

(Note that when $G(t) = \mathrm{KBKF}(t) \setminus \{y_0\}$ we only ever have one literal in a clause with head $f_i^{\tau} \in C$, the invariant that the annotation is the same for all literals is mostly relevant for when $G(t) = \mathrm{KBKF}_{\mathrm{lq}}(t) \setminus \{y_0\}$.) □

Invariant 4. *If $y^{\alpha}_{h,a}$ is the head of C and if $\neg y^{\beta}_{j,b} \in C$ (or $\neg f^{\beta}_{j} \in C$), then $\alpha \cup \{a/x_h\} \subseteq \beta$, where all extra annotations in β are of the form c_k/x_k for $k > h$. In other words the head literal $y^{\alpha}_{h,a}$ determines all annotations up to x_h.*

Proof. By induction.

Base case. For the base case we only need to consider the axioms C_i^c with negative existential literals. The head of C_i^c is $y_{i,c}$ and there are no universal variables in the clause of lower level than the head, hence its annotation $\alpha = \emptyset$. The axiom is instantiated so that c/x_i is added to the negative literals, hence we satisfy the invariant.

Instantiation. For the inductive step, suppose first that C is derived from instantiation of clause D. By Invariant 2 we know that $y^{\alpha}_{h,a}$ is the lowest level literal in the clause D. Any annotation involving x_l with $l < h$ is therefore both added to α and to the annotations of all other literals.

Resolution. Now suppose C is derived from resolving D_1 and D_2. Without loss of generality the pivot appears positively in D_2. We again use the fact that the head $y^{\alpha}_{h,a}$ of C is also the head of D_1. If negative literal $\neg z^{\beta} \in C$ comes from D_1 then $\alpha \cup \{a/x_h\} \subseteq \beta$ by Invariant 4 for D_1. If, on the other hand, negative literal $\neg z^{\beta} \in C$ comes from D_2, then we consider the nature of the resolved variable.

We first let the resolved variable be $y^{\epsilon}_{k,e}$ in D_2. Then $\epsilon \cup \{e/x_k\} \subseteq \beta$ by Invariant 4 for D_2. However, as $\neg y^{\epsilon}_{k,e} \in D_1$ then $\alpha \cup \{a/x_h\} \subseteq \epsilon \subseteq \epsilon \cup \{e/x_k\} \subseteq \beta$. Likewise for an annotation in β of level lower than $\mathrm{lv}(y_{h,a})$, it must be in ϵ and hence in α.

If the resolved variable is f^{ϵ}_k then $\epsilon = \beta$ by Invariant 3, but since $\alpha \cup \{a/x_h\} \subseteq \epsilon$ we are done. □

Invariant 5. *If $\neg y^{\beta}_{j,b} \in C$ then for all k, $h \le k < j$ there is $c_k \in \{0, 1\}$ such that $c_k/x_k \in \beta$.*

Proof. By induction.

Base case and instantiation. Invariant 5 holds in the axiom case and is unaffected by instantiation.

Resolution. Now suppose C is derived from resolving D_1 and D_2. Without loss of generality the resolved variable is z^{ϵ} in D_2. All literals in C that come from D_1 already fulfil Invariant 5. Now we can branch on the nature of variable z.

If $z^{\epsilon} = y^{\epsilon}_{k,e}$ then for the literals coming from D_2 we use Invariant 4. Since $\neg y^{\epsilon}_{k,e} \in D_1$ the annotation ϵ already contains all the necessary assignments for head $y^{\alpha}_{h,a}$. But since $\epsilon \cup \{e/x_k\} \subseteq \beta$ for any $\neg y^{\beta}_{j,b} \in D_2$, then together with Invariant 5 for D_2, these literals have the required annotations for the new head $y^{\alpha}_{h,a}$.

If the resolved variable is f^{ϵ}_k, then we do not add or remove any y literals. Hence the induction hypothesis is preserved. □

Invariant 6. *If $\neg y^{\beta}_{j,b} \in C$ with $j \le t$, then there is no $\neg y^{\delta}_{k,d} \in C$ such that $\beta \cup \{b/x_j\} \subseteq \delta$.*

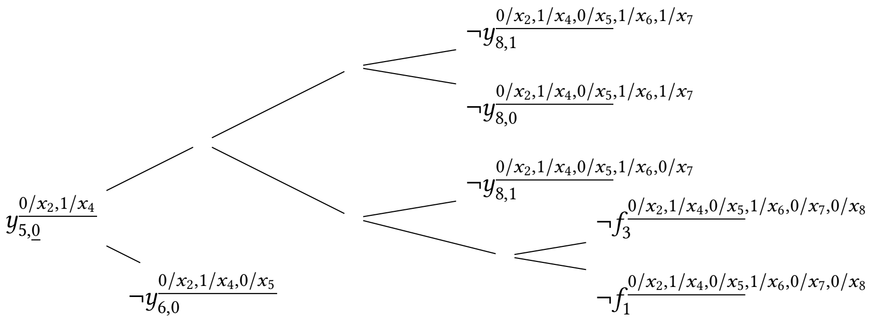
Proof. By induction.

Base case. We start with the base case for which we consider the axioms. Invariant 6 holds, because C_0 and all C^c_j with $j \le t$ are the only clauses with negative existential y literals, these are all of the same level.

Instantiation. Suppose first that C is derived by instantiation of clause D. We know from Invariants 2 and 5 and induction hypothesis that for any $\neg y^{\beta}_{j,b} \in D$ and $\neg y^{\epsilon}_{k,e} \in D$ with $k > j$, where j and k are indices in $\{0, \dots t\}$, there is some $c/x_l \in \beta \cup \{b/x_j\}$ such that $(1 - c)/x_l \in \epsilon$ and this conflict does not change by instantiation.

Resolution. Now suppose C is derived from resolving D_1 and D_2. The important case is when the resolved variable is $y^{\epsilon}_{k,e}$ in D_2. We need to check that if $\neg y^{\beta}_{j,b} \in D_1$ and $\neg y^{\delta}_{k,d} \in D_2$, then there is some conflict in the annotations (by using Invariants 2 and 5 for C). By Invariant 6 for D_1 we know that there is some $c/x_l \in \beta \cup \{b/x_j\}$ such that $(1 - c)/x_l \in \epsilon$. By Invariant 4 we have $\epsilon \cup \{e/x_k\} \subseteq \delta$, hence $(1 - c)/x_l \in \delta$.

The case where the resolved variable is f^{ϵ}_k is simpler. We do not add or remove any y literals so the induction hypothesis is preserved. □



$\neg y_{8,1}^{\underline{0/x_2,1/x_4,0/x_5,1/x_6,1/x_7}}$
$\neg y_{8,0}^{\underline{0/x_2,1/x_4,0/x_5,1/x_6,1/x_7}}$
$\neg y_{8,1}^{\underline{0/x_2,1/x_4,0/x_5,1/x_6,0/x_7}}$
$y_{5,\underline{0}}^{\underline{0/x_2,1/x_4}}$
$\neg f_3^{\underline{0/x_2,1/x_4,0/x_5,}1/x_6,0/x_7,0/x_8}$
$\neg y_{6,0}^{\underline{0/x_2,1/x_4,0/x_5}}$
$\neg f_1^{\underline{0/x_2,1/x_4,0/x_5,}1/x_6,0/x_7,0/x_8}$

Fig. 9. Structure of an example clause in an IR-calc refutation of KBKF(8). The literals are labelling a tree that branches on the variable index and annotations. Invariant 4 is highlighted by the underlined annotations and indices.

In order to provide some intuition on the proofs of $\mathrm{KBKF}(t)\backslash\{\neg y_0\}$, we will illustrate how the invariants allow us to study clauses as binary trees.

Example 44. *Consider Figure 9. By Invariant 1 there is only one head of the clause and we place it at the root of the tree. We branch by increasing the index i of the literals $y_{i,c}$, which will make all negative literals descendants of the root by Invariant 2. We underline to highlight Invariant 4. The remaining annotations must be present by Invariant 5, but we have two choices for each annotation, plus a choice of c for $y_{i,c}$. We use these choices to construct a binary tree. Notice that none of the internal nodes produces a literal: this is prevented by Invariant 6.*

Hence the positive literal is the root and the negative literals are the leaves. We can imagine instantiation of clauses to increase the bold part of the annotations and resolution on y literals to merge two different trees on leaf and root to form a new tree.

We will now give the ***overall idea of our lower bound argument***. For a clause C we define a set $\Sigma(C)$ of annotations associated with C. Our lower bound argument then rests on counting the set $\Sigma(C)$ as we progress through the proof. More precisely, we show that axioms have empty Σ (Lemma 49) and that instantiation steps do not add any new elements to Σ at all (Lemma 52). In a resolution step $\frac{D_1 \quad D_2}{C}$, the set $\Sigma(C)$ either equals $\Sigma(D_1) \cup \Sigma(D_2)$ or grows by exactly one new element (Lemma 53). In some sense, we only make progress in the proof in the latter case, and we need exponentially many resolution steps of this kind. Putting everything together, we find that by the end of the proof we must have collected all the exponentially many annotations in $\Sigma(y_0)$, implying an exponential lower bound to the proof length (Theorem 54).

We now give the details of the formal argument. We start with the definition of Σ, which we will first define for annotated variables and then for clauses.

Definition 45 (Σ for variables).

(1) We define $\Sigma(y_0)$ as the set of all complete assignments to variables $x_1, \ldots, x_t$.

(2) For $y_{j,b}^{\beta}$ with $\mathrm{dom}(\beta) = \{x_i \mid i < j\}$ we define $\Sigma(y_{j,b}^{\beta}) = \{\sigma \in \Sigma(y_0) \mid \beta \cup \{b/x_j\} \subseteq \sigma\}$. Invariant 6 ensures that if $\neg y_{j,b}^{B}$ and $\neg y_{k,d}^{D}$ appear together in a clause, then $\Sigma(y_{j,b}^{B}) \cap \Sigma(y_{k,d}^{D}) = \emptyset$.

(3) For f_j^B with a complete assignment β, we define $\Sigma(f_j^{\beta}) = \{\beta\}$, otherwise if β is not complete we set $\Sigma(f_j^{\beta}) = \emptyset$.

We now extend this definition to clauses, which we first classify into three types.

Definition 46. *We class C a clause in the proof of $G(t)$ as follows:*

- *Type 1 clause: The head of the clause is f_i^α.*
- *Type 2 clause: The head of the clause is $y_{h,a}^\alpha$ and there is some x_j with $j < h$ where $x_j \notin \mathrm{dom}(\alpha)$.*
- *Type 3 clause: Any other clause. These will have head y_0 or $y_{h,a}^\alpha$ where for all $j < h$, $x_j \in \mathrm{dom}(\alpha)$.*

Definition 47 (Σ for clauses). *Let C be a clause in an IR-calc proof of y_0 from $G(t)$.*

(1) When C is a type-1 or type-2 clause we set $\Sigma(C) = \emptyset$.
(2) For a type-3 clause C with head y we set $\Sigma(C) = \Sigma(y) \setminus (\bigcup_{\neg l \in C} \Sigma(l))$.

Remark 48. *For a type-3 clause C we have the following properties of $\Sigma(C)$:*

- *An annotation is only in $(\bigcup_{\neg l \in C}, \Sigma(l))$ (and thus removed from $\Sigma(C)$) (or $\Sigma(y_0)$) when it was originally included in $\Sigma(y_{h,a}^\alpha)$ from the presence of the head $y_{h,a}^\alpha$ (or y_0). This is true by Invariant 4.*
- *For different $\neg l_1, \neg l_2 \in C$ $\Sigma(l_1)$ and $\Sigma(l_2)$ are disjoint provided that l_1 and l_2 are both y variables. In other words unless for some $0 \leq j \leq t$ we have $\neg f_j^\sigma \in C$, we only remove an annotation σ at most once from our construction of $\Sigma(C)$. This holds by Invariant 6. If $\neg f_j^\sigma \in C$ for $0 \leq j \leq t$, then σ can be removed in our construction of Σ up to t times in total by $\neg f_l^\sigma \in C$.*

An important fact is that for axioms we get empty Σ as we verify in the next lemma.

Lemma 49. *For each clause $C \in \mathrm{KBKF}(t) \setminus \{\neg y_0\}$, instantiated as an IR-calc axiom C^β, we get $\Sigma(C^\beta) = \emptyset$.*

Proof. For axiom C_0 we first add all annotations to Σ, but then due to the presence of $\neg y_{1,0}$ and $\neg y_{1,0}$ remove all annotations starting with $0/x_1$ and $1/x_1$, respectively. This results in $\Sigma(C_0) = \emptyset$.

Using $C_1^0 = \{y_{1,0}, x_1, \neg y_{2,0}, \neg y_{2,1}\}$ as an IR-calc axiom results in $\{y_{1,0}, \neg y_{2,0}^{0/x_1}, \neg y_{2,1}^{0/x_1}\}$. Computing Σ of this clause, we first add all annotations starting with $0/x_1$, but then remove all annotations starting with $(0/x_1, 0/x_2)$ and $(0/x_1, 1/x_2)$, yielding again empty Σ. Analogous reasoning applies to C_1^1.

When using clauses C_i^0, C_i^1 with $2 \leq i \leq t$ as IR-calc axioms, we obtain type-2 clauses, which have empty Σ by definition. The remaining clauses C_{t+i}^0, C_{t+i}^1 give rise to type-1 clauses, again with empty Σ. ◻

It will be crucial for our lower bound argument to understand how $\Sigma(C)$ changes when we go through the clauses C in the proof. For this we first need two technical lemmas on the structure of IR-calc proofs of $\mathrm{KBKF}(t) \setminus \{\neg y_0\}$.

Lemma 50. *In an IR-calc proof from $G(t)$, a type-2 clause cannot be resolved with a type-3 clause.*

Proof. Suppose we have a resolution step between a type-2 and a type-3 clause. We can deduce that the resolved variable is $y_{k,e}^\epsilon$, otherwise one of the clauses is a type-1 clause.

The resolved variable has a complete annotation as, by Invariants 4 and 5, type-3 clauses have them for literals $\neg y_{j,e}^\epsilon$. However, this means that, by Invariants 4 and 5, the head of the type-2 clause also must have a complete annotation, contradicting our assumption. ◻

Lemma 51. *Let C be a type-2 or type-3 clause in an IR-calc proof from $G(t)$. If $\neg f_j^\beta \in C$ and $\neg f_l^\beta \notin C$ with $j, l > 0$, then there is an annotation of x_l in β.*

Proof. We proceed by induction on the number of lines to derive C.

Base case. This is vacuously true as all $\neg f_j$ literals get introduced in the same axioms.

Instantiation. If C is derived by instantiation of D we do not lose any annotation, so the hypothesis remains true.

Resolution. If C is derived by resolving D_1 and D_2, the claim still holds if the resolved variable is different from any f_l^β. If we resolve on f_l^β then D_2 is a type-3 clause and by Invariant 3, x_l appears in β and any other annotation of a literal introduced by D_2. □

The next two lemmas are the key to our lower bound. We show first that the set Σ does not gain elements in instantiation steps. In Lemma 53 we will then analyse how Σ changes in a resolution step.

Lemma 52. *Suppose in an IR-calc proof from $G(t)$ we instantiate clause D to get clause D'. Then $\Sigma(D) \supseteq \Sigma(D')$.*

Proof. If D is a type-1 clause it remains a type-1 clause under instantiation. Hence $\Sigma(D')$ remains empty.

If D is a type-3 clause it has complete annotations in its head, and this is unchanged by instantiation. Moreover, by Invariant 5, every $\neg y_{j,b}^\beta \in D$ has a complete annotation, which again is unaffected by instantiation. The only possible effect of the instantiation on type-3 clauses is therefore that negative literals $\neg f_j^\beta \in D$ receive a complete annotation, which means that the assignment β is deleted from $\Sigma(D')$, thus $\Sigma(D') \subseteq \Sigma(D)$.

If D is a type-2 clause then the only problem arises when instantiation makes the head annotation complete, i.e., the clause turns into a type-3 clause. In this case we will show that $\Sigma(D')$ is empty, hence $\Sigma(D') = \Sigma(D) = \emptyset$. We can show that $\Sigma(D') = \Sigma(D) = \emptyset$ by induction over the dag-structure of the proof starting at the axioms and working our way to D.

Induction hypothesis. Let D be a type-2 clause with head $y_{h,a}^\alpha$ that can be instantiated by σ to get a type-3 clause D'. Then $\Sigma(D')$ is empty.

Base case. We observe that instantiating any type-2 axiom always gives empty Σ by Lemma 49.

Instantiation. For the inductive argument we can ignore the case where D is derived by instantiation as that instantiation could have been incorporated into σ.

Resolution. Let D now be derived by resolving two clauses D_1 and D_2. Assume without loss of generality that D_1 is a type-2 clause. By Lemma 50 the other clause D_2 must be type 1 or type 2.

Suppose first that D_2 is a type-1 clause with head f_j^β. Let D_1' be the instantiation of D_1 by σ. Since σ gives a complete assignment to the annotations of $\neg y_{j,b}^\beta$ in D it must also do so in D_1, since it gives an annotation to the head of D which is the head of D_1. We also observe that $\neg f_j^\beta \in D_1$ becomes $\neg f_j^\xi \in D_1'$ under σ.

Suppose ξ is not complete then $\Sigma(D_1')$ is the same as $\Sigma(D_1' \backslash \{\neg f_j^\xi\})$ therefore as D' contains the same positive literals and no fewer negative literals than $D_1' \backslash \{\neg f_j^\xi\}$.By inductive hypothesis we have $\Sigma(D_1') = \emptyset$, $\Sigma(D') \subset \Sigma(D_1' \backslash \{\neg f_j^\xi\}) = \Sigma(D_1') = \emptyset$. So we now consider the case where ξ is complete.

When considering the change from $\Sigma(D_1)$ to $\Sigma(D)$ the only result of the resolution step with D_2 is the removal of the literal $\neg f_j^\beta$ and its replacement with all the negative literals of D_2. By Invariants 2 and 3 these negative literals are of the form $\neg f_k^\beta$ for $k > j$. If D_2 contains any negative literals at all then $\Sigma(D') = \Sigma(D_1 \backslash \{f_j^\xi\}) \backslash \Sigma(f_k^\xi)$. Because $\Sigma(f_j^\xi) = \{\xi\} = \Sigma(f_k^\xi)$ this would mean that $\Sigma(D') = \Sigma(D_1) \cup \{\xi\} \backslash \{\xi\} = \Sigma(D_1)$ which by our induction hypothesis was empty. Furthermore if D_1 has any other negative literals of the form $\neg f_l^\beta$,$l \neq j$ then this will have the same effect, $\Sigma(D')$ will not gain the annotation ξ due to f_j^ξ's absence. This is also true if $\neg y_{l,u}^\delta \in D_1$ such that $\delta \subset \beta$.

So, in order for this to have any effect on the inductive claim, there can be no other $\neg f_l^\beta$ (for $1 \leq l \leq t, l \neq j$) in D_1 or D_2, nor can there be $\neg y_{l,u}^\delta \in D'$ such that $\delta \subset \beta$. Hence, if we are to refute our induction claim at this stage, then by Lemma 51, β contains annotations for all x_l with $1 \leq l \leq t, l \neq j$. Further β contains an annotation of x_j as D_2 must contain such an annotation by Invariant 3. Therefore β is a complete assignment to $x_1, \ldots, x_n$. By Invariant 4 this can only happen when D_1 is a type-3 clause rather than a type-2 clause.

We have argued for when D_2 is a type-1 clause, now suppose instead, D_2 is a type-2 clause, and without loss of generality the resolved variable $y_{k,e}^\epsilon$ is positive in D_2, i.e., it is the head of D_2 and the resolved variable. By the induction hypothesis if we instantiate D_2 into a type-3 clause D_2' then $\Sigma(D_2')$ is empty. This can only happen if D_2' contains a negative literal so we can safely assume D_2 has at least one negative literal.

Let D_2' now be the instantiation of D_2 by σ specifically. Under σ the resolved variable is $y_{k,e}^{\epsilon'}$. Since we choose σ so that it gives a complete assignment to the annotations of the $y_{j,b}$ literals in D it must also do so in D_2, i.e., D_2' is type 3. This holds since D_2 has a negative literal and that negative literal is in D; with Invariant 4 this implies that D_2' is type 3. By inductive hypothesis we therefore have $\Sigma(D_2') = \Sigma(y_k^{\epsilon'}) \setminus \left(\bigcup_{\neg l \in D_2} \Sigma(l)\right) = \emptyset$. This means $\Sigma(y_k^{\epsilon'}) \subseteq \bigcup_{\neg l \in D_2} \Sigma(l)$.

Similarly let D_1' be the instantiation of D_1 by σ. Under σ the head variable is called $y_{h,a}^{\alpha'}$, and D_1' must also be type 3 as this is also the head of D'. Again by our induction hypothesis $\Sigma(D_1') = \Sigma(y_{h,a}^{\alpha'}) \setminus \left(\bigcup_{\neg l \in D_1} \Sigma(l)\right) = \emptyset$.

We have

$$\Sigma(D') = \Sigma(y_{h,a}^{\alpha'}) \setminus \left(\bigcup_{\neg l \in D} \Sigma(l)\right) = \Sigma(y_{h,a}^{\alpha'}) \setminus \left(\bigcup_{\neg l \in D, l \neq y_k^{\epsilon'}} \Sigma(l) \cup \bigcup_{\neg l \in D_2} \Sigma(l)\right),$$

but since $\Sigma(y_k^{\epsilon'}) \subseteq \bigcup_{\neg l \in D_2} \Sigma(l)$, then the removed elements $\left(\bigcup_{\neg l \in D_1} \Sigma(l)\right)$ from $\Sigma(D_1')$, are a subset of the removed elements $\bigcup_{\neg l \in D, l \neq y_k^{\epsilon'}} \Sigma(l) \cup \bigcup_{\neg l \in D_2} \Sigma(l)$ from $\Sigma(D')$ and thus $\Sigma(D') \subseteq \Sigma(D_1') = \emptyset$.

In other words when computing $\Sigma(D')$, the lack of the negative literal on the resolved variable means we may have additional elements only from $\Sigma(y_k^{\epsilon'})$ in $\Sigma(D')$. However, these were exactly the assignments that were added by the head $y_k^{\epsilon'}$ in D_2' and so we know — as $\Sigma(D_2')$ is empty — that we have the sufficient literals in D' to remove all elements in $\Sigma(D')$. We do not lose any $\neg f_j^\beta$ literals in this case, so the inductive claim holds. □

The next lemma is crucially important for the lower bound, it explains the conditions of new elements being added to Σ.

Lemma 53. *Let $\sqcup$ denote the union of two disjoint sets. Suppose in an IR-calc proof from $G(t)$ we resolve D_1 with D_2 to get clause D, where the resolved variable is positive in D_2.*

If D_1 is a type-3 clause that is resolved with the type-1 clause D_2 with head f_j^β for $j > 0$ and there is no $k > 0, k \neq j$ such that $\neg f_k^\beta \in D_1$ nor $\neg y_{k,e}^\epsilon \in D_1$ such that $\epsilon \subseteq \beta$, then $\Sigma(D) = \Sigma(D_1) \sqcup \Sigma(D_2) \sqcup \{\beta\} = \Sigma(D_1) \sqcup \{\beta\}$. Otherwise $\Sigma(D) = \Sigma(D_1) \sqcup \Sigma(D_2)$.

Proof. In the first case, D_1 is a type-3 clause and D_2 is a type-1 clause. Then the resolution step removes $\neg f_j^\beta$ from the clause D_1. The resolvent D must be a type-3 clause as all annotations remain. Because there is no $k > 0, k \neq j$ such that $\neg f_k^\beta \in D_1$, we infer by Lemma 51 that β is complete and $\beta \in \Sigma(D)$. All other annotations remain in $\Sigma(D)$.

If we otherwise resolve a type-3 clause D_1 with a type-1 clause D_2, but there is another $\neg f_k^\beta \in D_1$, $\neg y_{k,e}^\epsilon \in D_1$ such that $\epsilon \subseteq \beta$, or $\neg f_k^\beta \in D_2$ (by Invariant 3 this will necessarily occur if D_2 is not a singleton), then the same assignments are added and deleted in $\Sigma(D)$ as in $\Sigma(D_1)$ hence $\Sigma(D) = \Sigma(D_1) = \Sigma(D_1) \sqcup \Sigma(D_2)$.

Consider now the remaining cases. If we resolve a type-1 clause with a type-2 clause, then we obtain a type-2 clause, hence Σ remains empty. Likewise, resolving two type-2 clauses results in a type-2 clause and therefore again empty Σ. By Lemma 50, we cannot resolve type-2 with type-3 clauses.

Therefore the last case is when two type-3 clauses are resolved. Let D_2 provide the positive resolved literal $y_{k,e}^\epsilon$. Because $y_{k,e}^\epsilon$ is the head of D_2, every annotation $\sigma \in \Sigma(D_2)$ has $\epsilon \cup \{e/x_k\} \subseteq \sigma$. As $\neg y_{k,e}^\epsilon \in D_1$, the sets of assignments $\Sigma(D_1)$ and $\Sigma(D_2)$ are disjoint. But also there is no annotation $\xi \in \Sigma(D_1)$ with $\epsilon \cup \{e/x_k\} \subseteq \xi$ because of the presence of $\neg y_{k,e}^\epsilon$ in D_1 and Invariant 6.

We can show that $\Sigma(D)$ is the union of $\Sigma(D_1)$ and $\Sigma(D_2)$. We start by showing that $\Sigma(D) \subseteq \Sigma(D_1) \sqcup \Sigma(D_2)$ Suppose we have an assignment σ in $\Sigma(D)$ then it is necessarily in $\Sigma(y_{h,a}^\alpha)$, where $y_{h,a}^\alpha$ is the head of D. $y_{h,a}^\alpha$ must also be the head of D_1.σ cannot be in $\Sigma(l)$ for any negative literal $\neg l$ in D, hence σ cannot be in $\Sigma(l)$ for any negative literal in D_2. Therefore if $\sigma \in \Sigma(y_{k,e}^\epsilon)$ it must be in $\sigma(D_1)$. Now if $\sigma \notin \Sigma(y_{k,e}^\epsilon)$ then σ cannot be in any $\Sigma(l)$ for any negative literal $\neg l$ in D_1 and thus $\sigma \in \Sigma(D_1)$.

To show $\Sigma(D_1) \sqcup \Sigma(D_2) \subseteq \Sigma(D)$ we suppose we have an assignment ξ in $\Sigma(D_1)$, then per Definition 47, ξ is in $\Sigma(y_{h,a}^k)$ and cannot be in any $\Sigma(l)$ for any negative literal $\neg l$ in D_1. ξ is also not in $\Sigma(D_2)$ so not in $\Sigma(y_{k,e}^\epsilon)$, since all remaining negative literals in D come from D_2 this means via Invariant 4 that for all the literals in $\neg l$ in D, $\xi \notin \Sigma(l)$ thus it must be in $\Sigma(D)$. Now suppose we have an assignment ξ in $\Sigma(D_2)$, this assignment must be in $\Sigma(y_{k,e}^\epsilon)$ and thus by Invariant 4 in $\Sigma(y_{h,a}^\alpha)$. It will not be in $\Sigma(l)$ for any $\neg l$ in D_2. By Invariant 6 it will also not be in $\Sigma(l)$ for any $\neg l$ in D_1 except for $\Sigma(y_{k,e}^\epsilon)$. Therefore ξ is in $\Sigma(D)$. □

We can now deduce that all proofs of KBKF(t) in IR-calc are of at least exponential size.

THEOREM 54. *All proofs of* KBKF(t) *in IR-calc have length at least* 2^t.

PROOF. We will show that all IR-calc proofs of y_0 from KBKF$(t) \setminus \{\neg y_0\}$ are of exponential size. By Lemma 43 each refutation of KBKF(t) can be transformed into one of these while preserving size. Hence each refutation of KBKF(t) must be of exponential size.

Consider now an IR-calc proof $\pi = (D_1, D_2, \dots, D_m)$ of y_0 from $G(t)$ and define $s_i = |\bigcup_{j=1}^i \Sigma(D_j)|$. By Lemma 49, the axioms all have empty Σ, hence $s_1 = 0$. By Definition 47, the set $\Sigma(y_0)$ contains all 2^t complete annotations, therefore $s_m = 2^t$. Progressing in the proof from the axioms to y_0, we therefore build up the set Σ from an empty to an exponential-size set. If the clause D_{i+1} is an axiom or derived by instantiation, then $s_i = s_{i+1}$ by Lemmas 49 and 52. For a resolution step, we have $s_{i+1} \leq s_i + 1$ by Lemma 53. Therefore the proof π contains at least 2^t resolution steps. □

Since IR-calc simulates Q-Res (Theorem 16), we get as a corollary the hardness of KBKF(t) for Q-Res as already stated in [51].

COROLLARY 55. *All proofs of* KBKF(t) *in Q-Res are of at least exponential size in* t.

As the formulas KBKF(t) are easy for long-distance and universal resolution [35, 71] we obtain the following exponential separations.

COROLLARY 56. *IR-calc neither simulates LQ-Res nor QU-Res.*

Proof. The formulas $\mathrm{KBKF}(t)$ admit polynomial-size proofs in LQ-Res [35] and QU-Res [71], and therefore by the known simulations (including those shown here) also in LQU-Res, LQU$^+$-Res, and IRM-calc. □

5.2 Extending the lower bound to IRM-calc

In the previous section we showed that IR-calc neither simulates LQ-Res nor QU-Res. The stronger calculus IRM-calc simulates LQ-Res by Theorem 19 (and thus has short proofs of KBKF). Therefore, in terms of the simulation order (cf. Figure 1) the only question still open by now is whether IRM-calc also simulates QU-Res (We know that QU-Res does not simulate IRM-calc by Corollary 28). We will show here that this simulation does not hold. To do this we need formulas that are hard for IRM-calc, but easy for QU-Res.

For this we need the modification of the $\mathrm{KBKF}(t)$ formulas known as $\mathrm{KBKF}_{\mathrm{lq}}(t)$ from the last section. They are shown to be hard for LQ-Res. We will show here that they are indeed hard for the stronger system IRM-calc.

We first observe that these formulas remain hard for IR-calc.

Lemma 57. $\mathrm{KBKF}_{\mathrm{lq}}(t)$ *require exponential-size (in t) proofs in IR-calc.*

Proof. We define $G(t)$ as $\mathrm{KBKF}_{\mathrm{lq}}(t)\backslash\{\neg y_0\}$ and use exactly the same chain of arguments as in the previous section to show that $\mathrm{KBKF}_{\mathrm{lq}}(t)\backslash\{\neg y_0\}$ require exponential proofs of $\{y_0\}$. We remark that in the previous section the Invariants 1 to 6 and Lemmas 49 to 53 hold for $G(t)$, which was allowed to be $\mathrm{KBKF}_{\mathrm{lq}}(t)\backslash\{\neg y_0\}$.

Consider now an IR-calc proof $\pi = (D_1, D_2, \ldots, D_m)$ of y_0 from $G(t)$ and define $s_i = |\bigcup_{j=1}^{i} \Sigma(D_j)|$. By Lemma 49, the axioms all have empty Σ, hence $s_1 = 0$. By Definition 47, the set $\Sigma(y_0)$ contains all 2^t complete annotations, therefore $s_m = 2^t$. Progressing in the proof from the axioms to y_0, we therefore build up the set Σ from an empty to an exponential-size set.

Lemma 43 showed us that any IR-calc refutation of $\mathrm{KBKF}_{\mathrm{lq}}(t)$ is transformed into an IR-calc proof of of y_0 from $G(t)$ while preserving size. We get that $\mathrm{KBKF}_{\mathrm{lq}}(t)$ must require exponential-size (in t) proofs in IR-calc from the exponential lower bound of Σ. □

Now consider proofs of $\neg y_0$ from $\mathrm{KBKF}_{\mathrm{lq}}(t)\backslash\{\neg y_0\}$ in IRM-calc.

Remark 58. *Note that Invariant 1 from Section 5 still holds as there are never two positive literals. Invariant 3 still holds in IRM-calc since merging does not disrupt annotations as they are all identical. Invariant 4 also holds because the role of the head does not change under merging.*

We also introduce two new invariants specifically for IRM-calc proofs.

For any clause C in an IRM-calc proof of $\mathrm{KBKF}_{\mathrm{lq}}(t)$ the following holds:

Invariant 7. *If positive literal $y_{i,c}^{\tau} \in C$ then for all $j < i$ either $x_j \in \mathrm{dom}(\tau)$ or $\neg f_j^{\sigma} \in C$ for some annotation σ.*

Invariant 8. *If $f_i^{\tau} \in \mathrm{var}(C)$ then for all $j > i$ either $x_j \in \mathrm{dom}(\tau)$ or $\neg f_j^{\sigma} \in C$ with $\mathrm{dom}(\tau) = \mathrm{dom}(\sigma)$ and τ and σ differ only where $*$ appears.*

Proof. We prove these claims by induction on the number of lines.

Base case. It is sufficient to observe that all the axioms have these properties.

Instantiation and merging. For the inductive step, we first consider instantiation and merging steps. These keep literals and elements already in the annotation domains, thus preserving Invariant 7. For Invariant 8, merging may turn a constant value into $*$, but this is allowed and preserves the invariant.

Resolution. We next consider resolution steps. For Invariant 7, assume the head of one of the parent clauses is $y^{\tau}_{i,c}$. We only have to consider the case when f_j with $j < i$ is the pivot. This means that we resolve with a clause that contains a positive literal f^{σ}_j. From Invariant 3 we infer $x_j \in \mathsf{dom}(\sigma)$. During the resolution step the head of the clause $y_{i,c}$ will be instantiated by x_j which will give it an annotation as $j < i$.

For Invariant 8, we only need to consider a loss of an f_j literal for $j > i$, which is the pivot when a literal of variable f^{τ}_i is otherwise present. In this case we must resolve with a clause that contains a positive f^{σ}_j. By Invariant 3 we get $x_j \in \mathsf{dom}(\sigma)$. During the resolution step $\mathrm{var}(f_i)^{\tau}$ will therefore get annotated with the correct value for x_j. □

We can now show the lower bound for IRM-calc.

Theorem 59. $\mathrm{KBKF}_{\mathrm{lq}}(t)$ *require exponential-size proofs in IRM-calc.*

Proof. The proof uses the same technique as in [5]. There they showed that any LQ-Res refutation of $\mathrm{KBKF}_{\mathrm{lq}}$ can be transformed in polynomial time into a Q-Res refutation. Instead we show here that any IRM-calc refutation of $\mathrm{KBKF}_{\mathrm{lq}}$ can be transformed in polynomial time into an IR-calc refutation.

To do this we consider the $*$ annotations and the merge rule. Without applications of the merge rule in IRM-calc we essentially have an IR-calc proof (the slight change in the resolution rule will not matter). We therefore now consider exhaustively all the possible literals that can be produced by merging.

(1) A positive literal l^{τ} with $*/x_j \in \tau$ is introduced.
(2) A negative literal $\neg f^{\tau}_i$ with $*/x_j \in \tau$ is introduced, where $\forall j \geq i, */x_j \notin \tau$.
(3) A negative literal $\neg f^{\tau}_i$ is introduced, where $*/x_i \in \tau$.
(4) A negative literal $\neg f^{\tau}_i$ is introduced, where $\exists j > i, */x_j \in \tau$.
(5) A negative literal $\neg y^{\tau}_{i,c}$ is introduced where $\exists j < i$ with $*/x_j \in \tau$.

We will either show, that these are impossible or are not essential and can be removed in a polynomial number of IR-calc steps.

(1) It is impossible that there is a positive literal l^{τ} with $*/x_j \in \tau$.

It is not possible to introduce a $*$ by merging as there are never two positive literals in a clause. It is also impossible by resolution as we would already need a positive pivot with a $*/x_j$ annotation. Since the proof is a DAG, or sequence of lines, there must be (by the well-ordering of natural numbers) an earliest line in the proof where this occurs. This cannot have gained such a $*/x_j$ annotation via Resolution as it would imply the existence of an earlier line with a $*/x_j$ annotation.

(2) Literals $\neg f^{\tau_1}_i$ and $\neg f^{\tau_2}_i$ are merged in clause C to get $\neg f^{\tau}_i$ where $\forall j \geq i, */x_j \notin \tau$.

This is possible, but here we show that we can transform this step into a number of steps that do not require merging. If we start and repeat from the first time this appears, we can deal with (2) by resolving $\neg f^{\tau_1}_i$ and $\neg f^{\tau_2}_i$ away before the merging. This is done by resolving twice with clause D (which we will construct below) with $f^{\tau_1}_i$ and $f^{\tau_2}_i$ as pivots. The purpose of D is to resolve while only adding literals that are there already and not introducing any new annotations. To construct D we first observe that by Invariant 3, C has head $y^{A}_{h,b}$ (or y_0). For any $j < i$ such that $c/x_j \in \tau_1$ and $(1-c)/x_j \in \tau_2$, either $*/x_j \in A$, which is excluded by case 1, or $h \leq j$. Therefore since one $*$ must be created for merging to occur, $h \leq j < i$. Now consider all $j \geq i$: there may exist c_j such that $c_j/x_j \in \tau$, $c_j \in \{0,1\}$ because we are not merging on these annotations. D is created by repeatedly resolving axiom $F^{c_i}_i$ with the set of axioms $\Delta = \{F^{c_j}_j \mid j > i, c_j/x_j \in \tau\}$, in order of increasing j. After this D will be a clause with a f^{σ}_i head, which will be our pivot and contains some literals

$\neg f_j^{\sigma}$, which will later be merged with literals from C. The fact that $\sigma \subset \tau_1$ and $\sigma \subset \tau_2$ means we can use f_i^{σ} as a pivot literal resolve with $\neg f_i^{\tau_1}$ and $\neg f_i^{\tau_2}$, and in fact we do both by reusing D.

After resolving twice with D we remove $\neg f_i^{\tau_1}$ and $\neg f_i^{\tau_2}$ without instantiating the clause C. There may be some additional literals $\neg f_j^{\tau_1}$ and $\neg f_j^{\tau_2}$ for $j > i$ from resolving with D, but similar literals must have previously existed in C by Invariant 8. These possibly had $*$ annotations in C, but $0, 1$ values from D, so merging can make these the same as they were in C (but we will see below that this is not needed as the remaining cases exclude this possibility). Thus we end up with a subclause of C and thus shorten the remaining proof.

We repeat this until all literals of this form have been removed. We will look at the remaining cases to observe that we actually already have an IR-calc refutation, as all the remaining cases are impossible.

(3) There cannot be a negative literal $\neg f_i^{\tau}$ where $*/x_j \in \tau$ and $j = i$.
(4) There cannot be a negative literal $\neg f_i^{\tau}$ where there is some $j > i$ with $*/x_j \in \tau$.
(5) There cannot be a negative literal $\neg y_{i,c}^{\tau}$ where there is some $j < i$ with $*/x_j \in \tau$.

In all the cases consider that later in the proof we must resolve this negative literal as part of clause C with a clause D to remove that literal. D must now have the positive version of the literal, but it must contain no annotation on x_j.

For case (3), this is impossible as by Invariant 3 it must contain an annotation on x_i. In case (4) we must have $f_j^{\sigma} \in D$ by Invariant 8 and so in the resolvent we have $f_j^{\sigma'}$. But since we instantiate when we resolve and $x_j \notin \mathrm{dom}(\sigma)$, we get $*/x_j \in \sigma'$ which takes us to case (3) again. In case (5) we must have $f_j^{\sigma} \in D$ by Invariant 8 and so in the resolvent we have $f_j^{\sigma'}$. By Invariant 2 we have that $x_j \notin \mathrm{dom}(\sigma)$, hence $*/x_j \in \sigma'$ which takes us to case (3) again.

Therefore every IRM-calc proof of $\mathrm{KBKF}_{\mathrm{lq}}$ can be transformed in polynomial time to an IR-calc proof. Hence IRM-calc inherits an exponential lower bound from Lemma 57. □

In contrast, $\mathrm{KBKF}_{\mathrm{lq}}$ admits polynomial-size refutations in QU-Res as shown in [5]. Therefore we obtain the following separation.

Corollary 60. *IRM-calc does not simulate QU-Res.*

6 CONCLUSION

We have shown new lower bounds for Q-Res, IR-calc, IRM-calc, LQ-Res, and LQU$^+$-Res, and thereby settled the relative complexity of the main resolution-based QBF calculi. This reveals the complete picture of the simulation order of these proof systems (cf. Figure 1). Most importantly, our results show striking separations between all proof systems modelling CDCL-based QBF solving vs. proof systems modelling expansion-based solving. This provides theoretical evidence that these two paradigms for QBF solving are indeed complementary and should enhance the power of the solvers when carefully used in conjunction.

Acknowledgments. We thank the anonymous reviewers for their careful reading and helpful suggestions.

Part of the results of this paper are included (without full proofs) in the proceedings of MFCS'14 [11] and STACS'15 [12].

The work was supported by the European Regional Development Fund under the project AI&Reasoning (reg. no. CZ.02.1.01/0.0/0.0/15 003/0000466), by funds through FCT – Fundao para a Cincia e a Tecnologia with reference UID/CEC/50021/2019, the project INFOCOS with reference PTDC/CCI-COM/32378/2017, grant no. 60842 from the John Templeton Foundation, and a grant from the Carl Zeiss Foundation. The second author is supported by an EPSRC Doctoral Prize Fellowship.

REFERENCES

[1] Sanjeev Arora and Boaz Barak. 2009. *Computational Complexity – A Modern Approach.* Cambridge University Press. I–XXIV, 1–579 pages.

[2] Albert Atserias and Sergi Oliva. 2014. Bounded-width QBF is PSPACE-complete. *J. Comput. Syst. Sci.* 80, 7 (2014), 1415–1429. https://doi.org/10.1016/j.jcss.2014.04.014

[3] Valeriy Balabanov and Jie-Hong R. Jiang. 2012. Unified QBF certification and its applications. *Formal Methods in System Design* 41, 1 (2012), 45–65.

[4] Valeriy Balabanov, Jie-Hong Roland Jiang, Mikoláš Janota, and Magdalena Widl. 2015. Efficient Extraction of QBF (Counter)models from Long-Distance Resolution Proofs. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence.* AAAI Press, 3694–3701.

[5] Valeriy Balabanov, Magdalena Widl, and Jie-Hong R. Jiang. 2014. QBF Resolution Systems and Their Proof Complexities. In *SAT.* Springer, 154–169.

[6] Eli Ben-Sasson and Avi Wigderson. 2001. Short proofs are narrow - resolution made simple. *J. ACM* 48, 2 (2001), 149–169.

[7] Marco Benedetti. 2004. Evaluating QBFs via Symbolic Skolemization. In *LPAR*, Franz Baader and Andrei Voronkov (Eds.), Vol. 3452. Springer, 285–300.

[8] Marco Benedetti and Hratch Mangassarian. 2008. QBF-Based Formal Verification: Experience and Perspectives. *JSAT* 5, 1-4 (2008), 133–191.

[9] Olaf Beyersdorff and Joshua Blinkhorn. 2016. Dependency Schemes in QBF Calculi: Semantics and Soundness. In *Principles and Practice of Constraint Programming - CP.* 96–112.

[10] Olaf Beyersdorff, Ilario Bonacina, and Leroy Chew. 2016. Lower Bounds: From Circuits to QBF Proof Systems. In *Proc. ACM Conference on Innovations in Theoretical Computer Science (ITCS).* ACM, 249–260.

[11] Olaf Beyersdorff, Leroy Chew, and Mikoláš Janota. 2014. On Unification of QBF Resolution-Based Calculi. In *International Symposium on Mathematical Foundations of Computer Science (MFCS).* Springer, 81–93.

[12] Olaf Beyersdorff, Leroy Chew, and Mikoláš Janota. 2015. Proof Complexity of Resolution-based QBF Calculi. In *Proc. Symposium on Theoretical Aspects of Computer Science (STACS).* LIPIcs series, 76–89.

[13] Olaf Beyersdorff, Leroy Chew, and Mikoláš Janota. 2016. Extension Variables in QBF Resolution. In *Beyond NP, Papers from the 2016 AAAI Workshop.*

[14] Olaf Beyersdorff, Leroy Chew, Meena Mahajan, and Anil Shukla. 2017. Feasible Interpolation for QBF Resolution Calculi. *Logical Methods in Computer Science* 13 (2017). Issue 2.

[15] Olaf Beyersdorff, Leroy Chew, Meena Mahajan, and Anil Shukla. 2018. Are Short Proofs Narrow? QBF Resolution is not so Simple. *ACM Transactions on Computational Logic* 19 (2018). Issue 1.

[16] Olaf Beyersdorff, Leroy Chew, Renate A. Schmidt, and Martin Suda. 2016. Lifting QBF Resolution Calculi to DQBF. In *SAT.*

[17] Olaf Beyersdorff, Leroy Chew, and Karteek Sreenivasaiah. 2019. A game characterisation of tree-like Q-Resolution size. *J. Comput. System Sci.* 104 (2019), 82–101.

[18] Olaf Beyersdorff, Nicola Galesi, and Massimo Lauria. 2010. A Lower Bound for the Pigeonhole Principle in Tree-like Resolution by Asymmetric Prover-Delayer Games. *Inform. Process. Lett.* 110, 23 (2010), 1074–1077.

[19] Olaf Beyersdorff, Nicola Galesi, and Massimo Lauria. 2013. A Characterization of Tree-Like Resolution Size. *Inform. Process. Lett.* 113, 18 (2013), 666–671.

[20] Olaf Beyersdorff, Nicola Galesi, and Massimo Lauria. 2013. Parameterized Complexity of DPLL Search Procedures. *ACM Transactions on Computational Logic* 14, 3 (2013).

[21] Olaf Beyersdorff and Ján Pich. 2016. Understanding Gentzen and Frege systems for QBF. In *Proc. ACM/IEEE Symposium on Logic in Computer Science (LICS).*

[22] Armin Biere. 2004. Resolve and Expand. In *SAT.* 238–246.

[23] Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh (Eds.). 2009. *Handbook of Satisfiability.* Frontiers in Artificial Intelligence and Applications, Vol. 185. IOS Press.

[24] Armin Biere, Florian Lonsing, and Martina Seidl. 2011. Blocked Clause Elimination for QBF. In *International Conference on Automated Deduction CADE-23.* 101–115.

[25] Maria Luisa Bonet, Juan Luis Esteban, Nicola Galesi, and Jan Johannsen. 2000. On the Relative Complexity of Resolution Refinements and Cutting Planes Proof Systems. *SIAM J. Comput.* 30, 5 (2000), 1462–1484. https://doi.org/10.1137/S0097539799352474

[26] Uwe Bubeck and Hans Kleine Büning. 2007. Bounded Universal Expansion for Preprocessing QBF. In *Theory and Applications of Satisfiability Testing - SAT.* 244–257.

[27] Samuel R. Buss. 2012. Towards NP-P via proof complexity and search. *Ann. Pure Appl. Logic* 163, 7 (2012), 906–917.

[28] Hubie Chen. 2017. Proof Complexity Modulo the Polynomial Hierarchy: Understanding Alternation as a Source of Hardness. *TOCT* 9, 3 (2017), 15:1–15:20.

[29] Leroy Chew. 2017. *QBF proof complexity.* Ph.D. Dissertation. University of Leeds.

[30] Judith Clymo and Olaf Beyersdorff. 2018. Relating size and width in variants of Q-resolution. *Inf. Process. Lett.* 138 (2018), 1–6.

[31] Stephen A. Cook and Phuong Nguyen. 2010. *Logical Foundations of Proof Complexity.* Cambridge University Press.

[32] Stephen A. Cook and Robert A. Reckhow. 1979. The relative efficiency of propositional proof systems. *The Journal of Symbolic Logic* 44, 1 (1979), 36–50.

[33] Uwe Egly. 2016. On Stronger Calculi for QBFs. In *SAT.*

[34] Uwe Egly, Martin Kronegger, Florian Lonsing, and Andreas Pfandler. 2017. Conformant planning as a case study of incremental QBF solving. *Ann. Math. Artif. Intell.* 80, 1 (2017), 21–45.

[35] Uwe Egly, Florian Lonsing, and Magdalena Widl. 2013. Long-Distance Resolution: Proof Generation and Strategy Extraction in Search-Based QBF Solving, See [57], 291–308.

[36] Merrick L. Furst, James B. Saxe, and Michael Sipser. 1984. Parity, Circuits, and the Polynomial-Time Hierarchy. *Mathematical Systems Theory* 17, 1 (1984), 13–27.

[37] Enrico Giunchiglia, Paolo Marin, and Massimo Narizzano. 2009. Reasoning with Quantified Boolean Formulas. See [23], 761–780.

[38] Enrico Giunchiglia, Paolo Marin, and Massimo Narizzano. 2010. sQueezeBF: An Effective Preprocessor for QBFs Based on Equivalence Reasoning. In *SAT*, Ofer Strichman and Stefan Szeider (Eds.), Vol. 6175. Springer, 85–98.

[39] E. Giunchiglia, M. Narizzano, and A. Tacchella. 2006. Clause/term resolution and learning in the evaluation of quantified Boolean formulas. *Journal of Artificial Intelligence Research* 26, 1 (2006), 371–416.

[40] Alexandra Goultiaeva and Fahiem Bacchus. 2013. Recovering and Utilizing Partial Duality in QBF. In *Theory and Applications of Satisfiability Testing - SAT*, M. Järvisalo and A. Van Gelder (Eds.). Springer, 83–99. https://doi.org/10.1007/978-3-642-39071-5_8

[41] Alexandra Goultiaeva, Martina Seidl, and Armin Biere. 2013. Bridging the gap between dual propagation and CNF-based QBF solving. In *Design, Automation and Test in Europe, DATE.* 811–814.

[42] Alexandra Goultiaeva, Allen Van Gelder, and Fahiem Bacchus. 2011. A Uniform Approach for Generating Proofs and Strategies for Both True and False QBF Formulas. In *IJCAI.* 546–553.

[43] Johan Håstad. 1987. *Computational Limitations of Small-depth Circuits.* MIT Press, Cambridge, MA, USA.

[44] Marijn J. H. Heule, Martina Seidl, and Armin Biere. 2017. Solution Validation and Extraction for QBF Preprocessing. *J. Autom. Reasoning* 58, 1 (2017), 97–125.

[45] Mikoláš Janota. 2017. An Achilles' Heel of Term-Resolution. *CoRR* abs/1704.01071 (2017). https://arxiv.org/abs/1704.01071 https://arxiv.org/abs/1704.01071.

[46] Mikoláš Janota, Radu Grigore, and Joao Marques-Silva. 2013. On QBF Proofs and Preprocessing, See [57], 473–489.

[47] Mikoláš Janota, William Klieber, Joao Marques-Silva, and Edmund Clarke. 2016. Solving QBF with Counterexample Guided Refinement. *Artificial Intelligence* 234 (2016), 1–25. https://doi.org/10.1016/j.artint.2016.01.004

[48] Mikoláš Janota and Joao Marques-Silva. 2015. Expansion-based QBF solving versus Q-resolution. *Theor. Comput. Sci.* 577 (2015), 25–42.

[49] Toni Jussila, Armin Biere, Carsten Sinz, Daniel Kröning, and Christoph M. Wintersteiger. 2007. A First Step Towards a Unified Proof Checker for QBF. In *Theory and Applications of Satisfiability Testing - SAT.* 201–214.

[50] Hans Kleine Büning and Uwe Bubeck. 2009. Theory of Quantified Boolean Formulas. See [23], 735–760.

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

[52] Hans Kleine Büning, K. Subramani, and Xishun Zhao. 2007. Boolean Functions as Models for Quantified Boolean Formulas. *J. Autom. Reasoning* 39, 1 (2007), 49–75.

[53] William Klieber, Samir Sapra, Sicun Gao, and Edmund M. Clarke. 2010. A Non-prenex, Non-clausal QBF Solver with Game-State Learning. In *Theory and Applications of Satisfiability Testing - SAT.* 128–142.

[54] Jan Krajíček. 1997. Interpolation theorems, lower bounds for proof systems and independence results for bounded arithmetic. *The Journal of Symbolic Logic* 62, 2 (1997), 457–486.

[55] Florian Lonsing and Armin Biere. 2010. Integrating Dependency Schemes in Search-Based QBF Solvers. In *Theory and Applications of Satisfiability Testing - SAT.* 158–171.

[56] Florian Lonsing, Uwe Egly, and Allen Van Gelder. 2013. Efficient Clause Learning for Quantified Boolean Formulas via QBF Pseudo Unit Propagation. In *Theory and Applications of Satisfiability Testing - SAT.* 100–115.

[57] Kenneth L. McMillan, Aart Middeldorp, and Andrei Voronkov (Eds.). 2013. *Logic for Programming, Artificial Intelligence, and Reasoning, LPAR.* Springer.

[58] Christos H. Papadimitriou. 1994. *Computational Complexity.* Addison-Wesley.

[59] Tomás Peitl, Friedrich Slivovsky, and Stefan Szeider. 2016. Long Distance Q-Resolution with Dependency Schemes. In *SAT.* 500–518.

[60] Pavel Pudlák. 1997. Lower bounds for resolution and cutting planes proofs and monotone computations. *The Journal of Symbolic Logic* 62, 3 (1997), 981–998.

[61] Pavel Pudlák and Russell Impagliazzo. 2000. A lower bound for DLL algorithms for SAT. In *Proc. 11th Symposium on Discrete Algorithms*. 128–136.

[62] Jussi Rintanen. 2007. Asymptotically Optimal Encodings of Conformant Planning in QBF. In *AAAI*. AAAI Press, 1045–1050.

[63] Ronald L. Rivest. 1987. Learning Decision Lists. *Machine Learning* 2, 3 (1987), 229–246.

[64] John Alan Robinson. 1965. A Machine-Oriented Logic Based on the Resolution Principle. *J. ACM* 12, 1 (1965), 23–41.

[65] Marko Samer. 2008. Variable Dependencies of Quantified CSPs. In *Logic for Programming, Artificial Intelligence, and Reasoning - LPAR*. 512–527.

[66] Marko Samer and Stefan Szeider. 2009. Backdoor Sets of Quantified Boolean Formulas. *J. Autom. Reasoning* 42, 1 (2009), 77–97.

[67] Horst Samulowitz and Fahiem Bacchus. 2006. Binary Clause Reasoning in QBF. In *SAT*, Armin Biere and Carla P. Gomes (Eds.), Vol. 4121. Springer, 353–367.

[68] Nathan Segerlind. 2007. The Complexity of Propositional Proofs. *Bulletin of Symbolic Logic* 13, 4 (2007), 417–481.

[69] Friedrich Slivovsky and Stefan Szeider. 2016. Soundness of Q-resolution with dependency schemes. *Theor. Comput. Sci.* 612 (2016), 83–101.

[70] Allen Van Gelder. 2011. Variable Independence and Resolution Paths for Quantified Boolean Formulas. In *Principles and Practice of Constraint Programming - CP*. 789–803.

[71] Allen Van Gelder. 2012. Contributions to the Theory of Practical Quantified Boolean Formula Solving. In *CP*, Michela Milano (Ed.), Vol. 7514. Springer, 647–663.

[72] Allen Van Gelder. 2013. Primal and Dual Encoding from Applications into Quantified Boolean Formulas. In *CP*. 694–707. https://doi.org/10.1007/978-3-642-40627-0_51

[73] H. Vollmer. 1999. *Introduction to Circuit Complexity – A Uniform Approach.* Springer Verlag, Berlin Heidelberg.

[74] Lintao Zhang and Sharad Malik. 2002. Conflict Driven Learning in a Quantified Boolean Satisfiability Solver. In *ICCAD*. 442–449.

[75] Lintao Zhang and Sharad Malik. 2002. Towards a Symmetric Treatment of Satisfaction and Conflicts in Quantified Boolean Formula Evaluation. In *Principles and Practice of Constraint Programming - CP*. 200–215. http://link.springer.de/link/service/series/0558/bibs/2470/24700200.htm