Better Extension Variables in DQBF via Independence

Leroy Chew ✉ 🏠 🆔
TU Wien, Austria

Tomáš Peitl ✉ 🏠 🆔
TU Wien, Austria

—— **Abstract** ——

We show that extension variables in (D)QBF can be generalised by conditioning on universal assignments. The benefit of this is that the dependency sets of such conditioned extension variables can be made smaller to allow easier refutations. This simple modification instantly solves many challenges in p-simulating the QBF expansion rule, which cannot be p-simulated in proof systems that have strategy extraction [13]. Simulating expansion is even more crucial in DQBF, where other methods are incomplete. In this paper we provide an overview of the strength of this new independent extension rule. We find that a new version of Extended Frege called IndExtFrege + ∀red can p-simulate a multitude of difficult QBF and DQBF techniques, even techniques that are difficult to approach with eFrege + ∀red. We show five p-simulations, that IndExtFrege + ∀red p-simulates QRAT, DQBF-IR-calc, IR($\mathcal{D}^{\text{rrs}}$)-calc, Fork-Resolution and DQRAT which together underpin most DQBF solving and preprocessing techniques. The p-simulations work despite these systems using complicated rules and our new extension rule being relatively simple. Moreover, unlike recent p-simulations by eFrege + ∀red we can simulate the proof rules line by line, which allows us to mix QBF rules more easily with other inference steps.

1 Introduction

Proof systems that allow extension variables are very powerful [17]. We know that in propositional logic, extended resolution can p-simulate a multitude of disparate techniques, and lower bounds to extended resolution remain an open problem. This is remarkable because resolution itself is a weak proof system with various lower bounds, but simply adding the extension rule upgrades it to be amongst the most powerful propositional proof systems and is equivalent to the checking format DRAT [31]. An extension rule for variable v takes the form:

$v \leftrightarrow b(X) : b$ is a Boolean function, X is a set of existing variables

Quantified Boolean Formulas (QBF) also use extension variables [19] and these too can be very powerful. QBFs list all variables in a quantifier prefix. This defines whether a variable is existential (∃) or universal (∀). The order defines dependencies so variables only depend on other variables to their left. When used in refutations, extension variables must

28th International Conference on Theory and Applications of Satisfiability Testing (SAT 2025).
Editors: Jeremias Berg and Jakob Nordström; Article No. 11; pp. 11:1–11:24
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

be existential, otherwise it is too easy to violate extension clauses. The dependency set of the new variables (which is the same as the quantification order in QBF) must be careful not to introduce falsity into the formula. In fact one of the earliest approaches was to say that every new variable is quantified rightmost to conservatively give it the entire dependency set. The drawback with this approach is that it limits the power of inferences from using extension variables [8]. In the current alternative, sometimes known as "strong extension", we place $\exists v$ immediately after all the variables X used in b. eFrege $+ \forall$red uses this strong notion, and it is so powerful no unconditional lower bound can be found, unless either a long standing proof complexity or circuit complexity open problem is solved [7]. Some problems do emerge when looking at conditional lower bounds, though. The Select family of formulas can be shown by the QBF proof system QRAT to be equivalent to the law of non-contradiction on QBFs, but do not have short proofs in QBF Extended Frege unless PSPACE $\subseteq$ P/poly [13]. QRAT manages this through a combination of extension variables and an explicit rule that calculates so-called spurious dependencies in quantified variables, when nominal dependencies can be ignored when making inferences. Note how different this is to propositional logic, in most cases where we would take a basic proof system and add even a complicated rule we would still be simulated by extended resolution. Based on our observations in this paper we find that we can do better with extension variables. Instead of extensions being pure definitions, they are now under conditions.

$$\alpha \to (v \leftrightarrow b(X)) : \alpha \text{ is a partial assignment of the universal variables}$$

The utility of this is not immediately obvious because v is weaker than it could be, however precisely because this is weaker now the dependency of v on the variables of α is no longer necessary for soundness. Because we can remove arbitrary dependencies, the natural class for this proof system is for Dependency QBF (DQBF). But we can still use this rule for QBF, in fact it adds substantial clarity to existing QBF proof systems.

    Our main contribution is that we propose a line based proof system, that each new line addition preserves satisfiability, so when arriving at the falsum symbol 0 we know we started with an unsatisfiable DQBF. Unlike many QBF systems we are not able to construct countermodels from following the proof steps (unless P = PSPACE). We then show how it can p-simulate existing QBF and DQBF proof system rules. Figures 1 and 2 shows the known p-simulations in DQBF and QBF proof systems *after* considering the work in this paper[1]. New results are presented in **bold**. The existing results are from papers that span over decades worth of work, and transitively show our new proof system p-simulates all systems in both figures [1, 2, 3, 4, 6, 7, 9, 10, 12, 14, 16, 15, 18, 22, 23, 24, 25, 26, 28, 30].
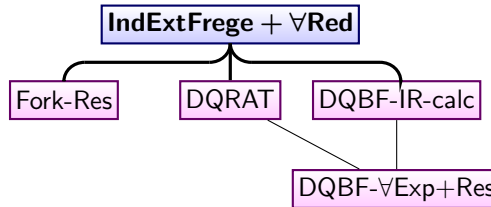
Figure 1 The p-simulation structure of refutational S-form DQBF proof systems.
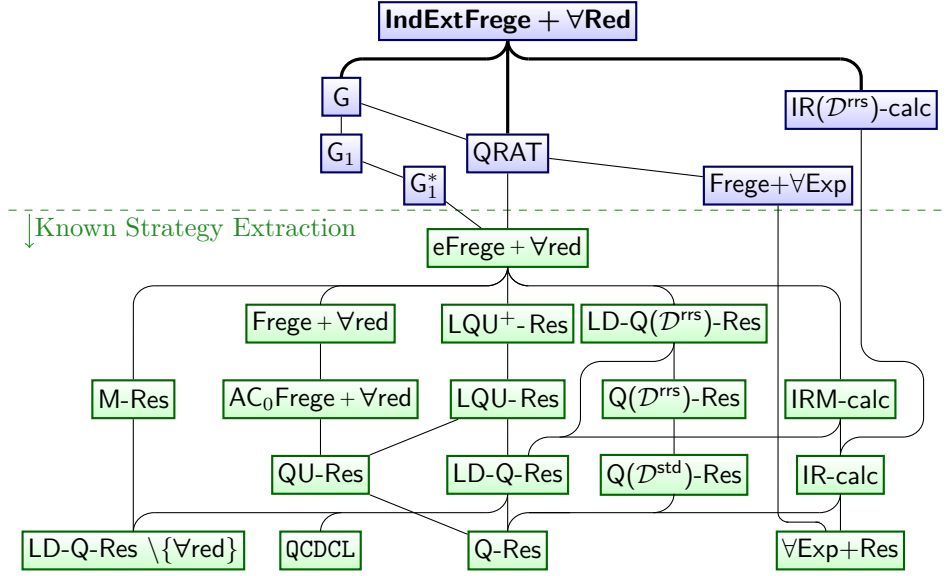
Figure 2 The p-simulation structure of refutational QBF proof systems.

1.1 Related Work

The original Extended Q-Resolution was defined by Jussila et al. [19] and discussed the weak and strong extension variables. QRAT by Biere, Heule and Seidl generalises extension variables in its clause additions. In the original paper [18] and later in Kiesl and Seidl's paper [22], they use similar clause additions to our independent extension definitions. Unlike our definitions, the new variables have a larger dependency set, but get round this by detecting spurious dependencies with an Extended Universal Reduction rule (EUR). These QRAT extension variables can capture the annotated variables used in the QBF proof system ∀Exp+Res, but there has not been success making the same method work for the more general QBF proof system IR-calc [11]. Our approach works backwards, instead of using the presence of resolution path schemes to derive expansions, we use expansions to show the validity of resolution path schemes.

Blinkhorn proposed a DQBF proof system based on generalising the QBF proof system QRAT to DQBF [10]. This system too allows the addition of new variables v, but when using the RAT addition rules to make definition clauses the RAT addition rule only allows the variables in the dependency set of v to be used unlike in our case. Therefore we suspect their proof system is weaker. Rabe proposed a DQBF proof system called Fork-Resolution that has extension variables but only in the case of clause-splitting [26].

Chew and Clymo showed that QRAT's strongest rule violated the property of strategy extraction unless P = PSPACE [13]. Later, Chew and Heule showed that the QBF sequent calculus G p-simulates QRAT [14]. The sequent calculus G [24] creates quantified variables of QBF witnesses so it has a witnessing scheme that works to simulate EUR. We use the same witnessing scheme in our work to simulate EUR with our DQBF system, hence showing strategy extraction, even for the QBF fragment, is impossible unless P = PSPACE.

The discussion around new variables comes after Reichl and Slivovsky's successful DQBF solver `Pedant` [27] introduces the similar *arbiter variables*, with a few more limitations.

1.2   Organisation

We define the necessary preliminaries of QBF and DQBF in Section 2. As an example we include the DQBF complete refutation system DQBF-IR-calc. In Section 3 we define our new extension rule and show its soundness, firstly we integrate it into a full proof Frege system that we call IndExtFrege + ∀red in Section 3.2, and later an equivalent resolution system in Section 3.3. In order to show completeness we p-simulate the complete DQBF-IR-calc proof system. Section 4 demonstrates the advantages of using IndExtFrege + ∀red on (D)QBFs, with more p-simulations. In Section 4.1 we give a definition of the clausal proof system Fork-Resolution for DQBF and show how to p-simulate that in IndExtFrege + ∀red. In Section 4.2 we give a definition of the QBF interference based proof system QRAT and show how IndExtFrege + ∀red can p-simulate it. In Section 4.3 we do the same for the QBF proof system IR($\mathcal{D}^{\text{rrs}}$)-calc which uses the reflexive resolution path dependency scheme.

2   Preliminaries

We assume a countably infinite set $\mathcal{V}$ of propositional *variables* is given. A *literal* is either a variable $x \in \mathcal{V}$ or its negation ($\neg x$), also written as $\bar{x}$, whereby $\bar{\bar{x}} = x$. A *(propositional) formula* is defined recursively: (1) literals are formulas; and (2) if ϕ and ψ are formulas, then $\phi \wedge \psi$, $\phi \vee \psi$, $\phi \to \psi$, $\phi \leftrightarrow \psi$ and $\bar{\phi}$ are also formulas. A *circuit* is like a formula, but its recursive structure is a DAG (directed acyclic graph) instead of a tree: subformulas can be reused. Every formula is a circuit, but transforming a circuit into a formula involves potentially exponential duplication of nodes. The set of variables of a circuit or formula is defined recursively as $\text{var}(\phi \circ \psi) = \text{var}(\phi) \cup \text{var}(\psi)$ for any operator $\circ$, $\text{var}(\bar{\phi}) = \text{var}(\phi)$, and $\text{var}(x) = x$ if x is a variable. We use $\sqcup$ to denote the disjoint union, i.e. the union of two sets that are known to be disjoint. A *clause* is a finite set of literals, semantically interpreted as their disjunction (equivalently, a formula consisting only of literals and "or" connectives). A clause C is *tautological*, if $\{x, \bar{x}\} \in C$ for some variable x. A clause is *unit* if only contains a single literal, a clause is the *empty clause* if it contains no literals. The negation $\bar{C}$ of a clause C can be viewed as a conjunction of unit clauses $\bigwedge_{l \in C}\{\bar{l}\}$.

A *(partial) assignment* to a circuit ϕ is a mapping $\alpha : V \subseteq \text{var}(\phi) \to \{0, 1\}$. A partial assignment α is *complete* if it assigns every variable, i.e. $V = \text{var}(\phi)$. We can write an assignment α as a function, a conjunction of literals or sequence of literals i.e. $x\bar{y}\bar{z}$. As a function an assignment is extended to circuits by $\alpha(\phi \wedge \psi) = \alpha(\phi) \cdot \alpha(\psi)$, $\alpha(\phi \vee \psi) = \max(\alpha(\phi), \alpha(\psi))$ and $\alpha(\bar{\phi}) = 1 - \alpha(\phi)$. α *satisfies* (is a *model* of) a circuit ϕ if $\alpha(\phi) = 1$. A circuit is *satisfiable* if it has a model, and *unsatisfiable* otherwise. Circuit ϕ is a *tautology* if $\bar{\phi}$ is unsatisfiable. Circuit ϕ *entails* a circuit ψ, written $\phi \models \psi$, if every complete assignment $\alpha : \text{var}(\phi) \to \{0, 1\}$ that satisfies ϕ also satisfies ψ. Two circuits ϕ and ψ are *(logically) equivalent*, written $\phi \equiv \psi$, if $\text{var}(\phi) = \text{var}(\psi)$, and $\phi \models \psi$ and $\psi \models \phi$. Two circuits ϕ and ψ are *equisatisfiable*, written $\phi \cong \psi$, if they are both satisfiable or both unsatisfiable.

A formula is in *conjunctive normal form (CNF)* if it is a (finite) conjunction of clauses. Any circuit ϕ can be transformed into a logically equivalent CNF using distributivity and De Morgan's rules, but the resulting size may be exponential. Allowing *extension variables*, i.e. variables that do not occur in ϕ, it is possible to transform any circuit ϕ into $\text{CNF}(\phi)$, an equisatisfiable CNF in linear time [29]. This translation has the additional property that if α is a model of $\text{CNF}(\phi)$, then $\alpha|_{\text{var}(\phi)}$ is a model of ϕ and vice versa, if α is a model of ϕ, then there exists a model β of $\text{CNF}(\phi)$ such that $\beta(x) = \alpha(x)$ for all $x \in \text{var}(\phi)$.

2.1 Proof systems

A *proof system* as defined by Cook and Reckhow [17] for some non-empty language $\mathcal{L}$ is a polynomial-time computable function on strings whose range is exactly $\mathcal{L}$. Intuitively f maps proofs to valid theorems, non-proofs can be mapped to some arbitrary known element in $\mathcal{L}$. Soundness comes from well-definition, and completeness is from surjectivity. Many proof systems are line-based where a finite set of rules govern the derivation of valid inferences in the language until a conclusive line is derived.

Proof complexity measures the *sizes* of proofs, i.e. number of characters in the proof string. In a line based proof system, where we can distinguish individual lines we can measure the proof *length*- the number of lines. Where lines are clauses, the clause *width* is the number of literals in the clause. Given a proof system g for language $\mathcal{L}$ we say that proof system f *p-simulates* g if there is a polynomial time procedure p that maps g proofs to f proofs such that $f(p(\pi)) = g(\pi)$ for all g-proofs. We do not require f to necessarily be a proof system for $\mathcal{L}$, but a proof system for $\mathcal{L}' \supseteq \mathcal{L}$. In this paper we do this when we p-simulate QBF proof systems with DQBF proof systems.

Frege systems are "text-book" style line-based proof systems for propositional logic. They consist of a finite, sound and complete set of axioms and rules where any variable can be substituted with any formula. Fig. 3 gives an example of a Frege system.

$$\overline{1} \qquad \overline{x_1 \to (x_2 \to x_1)} \qquad \overline{((x_1 \to 0) \to 0) \to x_1}$$

$$\overline{(x_1 \to (x_2 \to x_3)) \to ((x_1 \to x_2) \to (x_1 \to x_3))} \qquad \frac{x_1 \qquad x_1 \to x_2}{x_2}$$

$$\overline{(x_1 \to 0) \to \neg x_1} \qquad \overline{\neg x_1 \to (x_1 \to 0)} \qquad \overline{(x_1 \vee x_2) \to (x_2 \vee x_1)}$$

$$\overline{x_1 \to (x_1 \vee x_2)} \qquad \overline{(x_1 \vee x_2) \to (\neg x_1 \to x_2)}$$

$$\overline{(x_1 \to x_2) \to (\neg x_1 \vee x_2)} \qquad \overline{(x_1 \wedge x_2) \to x_1}$$

$$\overline{(x_1 \wedge x_2) \to x_2} \qquad \overline{x_1 \to (x_2 \to x_1 \wedge x_2)}$$

Figure 3 A Frege system for connectives $0, 1, \neg, \to \vee, \wedge$.

The rules will depend on the connectives included, but Cook and Reckhow [17] showed all Frege systems are p-equivalent. For example a Frege system that uses $\vee, \neg, \wedge$ can adopt the following laws without changing the proof complexity:

$$\frac{C}{C \vee p} \text{ (Weak)} \qquad \frac{C \vee p \qquad C \vee \neg p}{C} \text{ (Res)} \qquad \frac{C \vee p \qquad C \vee q}{C \vee (p \wedge q)} \text{ (Distr)}$$

2.2 (Dependency) Quantified Boolean Formulas

A *quantified Boolean formula (QBF)* is a propositional formula equipped with Boolean quantifiers: $\forall$ and $\exists$. $\forall x \phi(x) \equiv \phi(0) \wedge \phi(1)$ and $\exists x \phi(x) \equiv \phi(0) \vee \phi(1)$. A QBF in prenex form $\Pi \phi$ contains a propositional *matrix* ϕ which is quantifier-free and a *prefix* $\Pi = \mathcal{Q}_1 x_1 \ldots \mathcal{Q}_k x_k$

where $\mathcal{Q}_i \in \{\forall, \exists\}$ for $1 \leq i \leq k$. A closed QBF requires every variable to be bound to some quantifier in the prefix. We will mainly work with closed prenex QBFs in this paper. For variable x we use $x \in \exists$ to denote that x is existentially quantified somewhere in the prefix, or $x \in \forall$ to denote that x is universally quantified in the prefix. The quantifier order matters, we say that existential variable x depends on u if u in quantified left on x in the prefix. In this way we can build a dependency set D_x of each existentially quantified variable x containing exactly the universal variables that are left of x.

A *Skolem function* for existential variable x is a Boolean function $f_x : D_x \to \{0, 1\}$. This allows us to use alternative semantics to define the truth of a QBF: that a closed prefix QBF is true if and only if there is a set of Skolem functions, one for each existential variable x, such that for every complete assignment to all the universal variables the universal assignment completed with the values of the Skolem functions under that assignment, form a satisfying assignment to the propositional matrix. We call such a set of Skolem functions *winning*. Dually, a closed QBF is false if and only if there is a set of *Herbrand functions*, one for each universal variable y, such that for every complete assignment to all the existential variables the Herbrand functions falsify the propositional matrix.

A Dependency Quantified Boolean Formula (QBF) can be defined and an S-form DQBF uses this notion of Skolem functions as its main semantics. An S-form DQBF $\Pi\phi$ has a prefix $\Pi = \forall u_1 \ldots u_p \exists x_1(D_{x_1}) \ldots x_q(D_{x_q})$, here the quantifier order does not matter as the dependency sets are explicitly given. Each D_x can be any arbitrary subset of $\{u_1 \ldots u_p\}$. A DQBF is true if and only if there is a set of Skolem functions, one for each existential variable x, such that for every complete assignment to all the universal variables the universal assignment completed with the values of the Skolem functions under that assignment, form a satisfying assignment to the proposition matrix. We sometimes write $\forall U \exists E \phi$ for an arbitrary S-form DQBF, where U is the set of universal variables, E the set of existential variables each with their own unspecified dependency set and ϕ a propositional matrix containing no quantifiers. We define D_u for a universal variable to be $\{u\}$. We can also define the dependency set of a clause C as $(\bigcup_{\text{var}(y) \in C} D_y)$ including universal literals.

QBF is a PSPACE-complete language and DQBF is NEXPTIME-complete. We can demonstrate a DQBF is true by exhibiting its Skolem functions as circuits and showing the matrix with the Skolem functions substituted in is a propositional tautology. To show a DQBF is false we can use a DQBF proof system such as DQBF-IR-calc [3] given in Figure 4. We will define a new refutational proof system for DQBF in the next section.

3   IndExtFrege + ∀red, an S-form DQBF proof system

3.1   Independent Extension

Consider the refutational proof system Frege + ∀red in QBF [7], composed of Frege rules and a ∀red rule. Frege rules allow for propositional line based inference and the reduction rule allows a universal variable u to be replaced by a constant 0 or 1, given below:

$$\frac{\Pi\phi \wedge L(u)}{\Pi\phi \wedge L(u) \wedge L(0)} \qquad \frac{\Pi\phi \wedge L(u)}{\Pi\phi \wedge L(u) \wedge L(1)}$$

No variable can appear to the right of u in L. A sound DQBF version exists.

▶ **Lemma 1** (∀red soundness). *Suppose $\Pi\phi \wedge L(u)$ is a true S-form DQBF, and L contains no existential variables x such that u is in the dependency set of x. Then the S-form DQBF $\Pi\phi \wedge L(u) \wedge L(0)$ is true and the S-form DQBF $\Pi\phi \wedge L(u) \wedge L(1)$ is also true.*

Proof. An S-form DQBF is true if and only if it has a set of satisfying Skolem functions, a function σ_x for each of its existential variables x. Suppose $\Pi\phi \wedge L(u)$ is satisfied by the set $\{\sigma_x \mid x \in \exists\}$. We will show that $\Pi\phi \wedge L(u) \wedge L(0)$ is satisfied by the same functions. Consider an arbitrary universal assignment α, ϕ is satisfied by assumption and so is $L(u)$. If $\alpha(u)$ is 0 then $L(u) = L(0)$ is satisfied. Otherwise consider β which is identical to α except on u. $\alpha(u) = 1$ and $\beta(u) = 0$, but the outputs of Skolem functions of the existential variables in L remain unaffected by changing between α and β, only the variable u is affected and so $L(0)$ is satisfied. Therefore $\Pi\phi \wedge L(u) \wedge L(0)$ is satisfied by the same set of Skolem functions as $\Pi\phi \wedge L(u)$. The case with $L(1)$ is symmetrical. ◀

Universal reduction in $L(u)$ is blocked when there are existential variables in L that depend on u. Extension variables are also existential and can end up blocking reduction through excessive dependency. We define extension variables that *conditionally* represent Boolean circuits, for smaller dependency sets. We give two versions, one for conjunction and one for disjunction. We could instead use a single rule based on a functionally complete connective such as NAND (which we do in Section 3.3), but our definitions fit more nicely into the proofs of this paper. Let α be a conjunction of universal literals, and Y is a set of literals, both existential and universal.

$$\frac{\Pi\phi}{\Pi\exists v(D_v)\phi \wedge (\alpha \to (v \leftrightarrow \bigwedge_{y \in Y} y))} \qquad \frac{\Pi\phi}{\Pi\exists v(D_v)\phi \wedge (\alpha \to (v \leftrightarrow \bigvee_{y \in Y} y))}$$

The extension variable v is a new variable not appearing in Π, nor in ϕ. D_v is calculated as the union over all $D_{\mathrm{var}(y)}$ for $y \in Y$ and we then subtract the domain of α. This means v is independent of every variable in α and even if some variable x in Y does depend on some variable u, that dependence will be removed if $u \in \alpha$. In the earlier extension rule [19, 7], the variables that extension variable was defined on coincided with its dependency set. In our new rule v can be defined on variables and not receive its full dependency set. Having a smaller dependency set means that v prevents fewer reduction steps. For why this is permissible, we can think of the equational part of the definition only applying once α is already set, therefore there is no scenario of the α variables where v is required to consider a different input value for these variables, other than in the situation where it must consider α.

The downside of this definition is that substituting a Boolean function b for its extension variable v adds condition α. But because we subtract the domain of α, v no longer blocks the reduction of variables from α so in many cases we can reduce these variables.

▶ **Lemma 2.** *Suppose $\Pi\phi$ is a true S-form DQBF, and $\Pi\exists v(D_v)\phi \wedge (\alpha \to (v \leftrightarrow b(Y)))$ is constructed according to the Independent Extension rule, where b is a Boolean function. Then $\Pi\exists v(D_v)\phi \wedge (\alpha \to (v \leftrightarrow b(Y)))$ is a true S-form DQBF.*

Proof. $\Pi\phi$ is a true S-form DQBF, therefore it has Skolem functions σ_x for each existential variable x. This set of Skolem functions satisfies all lines in ϕ, but $\alpha \to (v \leftrightarrow b(Y))$ may or may not be satisfied. To make sure it is satisfied we use a Skolem function σ_v for v. We apply substitution of α to Y to get $\alpha(Y)$ which assigns some variables to constants. Notice those variables in the domain of α are now constant and no longer in the domain of $\alpha(Y)$. The free variables of $\alpha(Y)$ are those in Y but not α. We define $\sigma_v(D_v) = b(\alpha(Y))$, which works because $b(\alpha(Y))$'s dependency set is $(\bigcup_{y \in Y} D_y) \setminus D_\alpha$. Note that v does not appear in ϕ, so ϕ remains satisfied.

If α is not satisfied then $\alpha \to (v \leftrightarrow b(Y))$ is automatically satisfied. If α is satisfied then $Y = \alpha(Y)$. If $b(Y)$ is true then $b(\alpha(Y))$ is true and so σ_v is true which makes v true, so $v \leftrightarrow b(Y)$ is satisfied. Likewise if $b(Y)$ is false, then $b(\alpha(Y))$ is false so σ_v and thus v are false, therefore $v \leftrightarrow b(Y)$ is satisfied. ◀

> **Figure 4** Proof rules of DQBF-IR-calc [3] .

3.2   A Sound and Complete Proof System

We define our new proof system IndExtFrege + ∀red in Figure 5. The proof system works as a refutational proof typically does, starting with a DQBF $\forall U \exists E \phi$. ϕ is a propositional formula. For QBFs, IndExtFrege + ∀red automatically generalises eFrege + ∀red, but since we also desire DQBF completeness only p-simulation of a complete DQBF proof system suffices, here we choose DQBF-IR-calc given in Figure 4. DQBF-IR-calc works by removing all universal literals from the formula and replacing each existential literal with an annotated literal, the annotations are partial universal assignments. The idea is that you can remove universal quantifiers by expansion, but you create multiple copies of the inner existential variables, so the annotations track which expansions have led to this literal. A p-simulation works because these annotated variables can be defined by independent extension, where they drop the dependence on universal variables that have been expanded on.

▶ **Theorem 3.** *There is an $O(w^2 l)$ IndExtFrege + ∀red p-simulation of DQBF-IR-calc. Where l is the number of lines in π, w is the size of the largest clause $w = \max_{C \in \pi}(\sum_{l^\alpha \in C} 1 + |\alpha|)$ (π being the DQBF-IR-calc proof).*

Proof. DQBF-IR-calc uses existential annotated variables. For each annotated variable x^α appearing in the DQBF-IR-calc proof we introduce definition clauses $\bar{\alpha} \vee x^\alpha \vee \bar{x}$ and $\bar{\alpha} \vee \bar{x}^\alpha \vee x$ based on $\alpha \rightarrow (x^\alpha \leftrightarrow x)$. We therefore add $O(wl)$ many clauses each of width $O(w)$.
Axiom: in DQBF-IR-calc an axiom involves some instantiation of a clause C. τ is the partial assignment that contradicts the universal literals in C. We replace each existential literal l with $l^{\tau|_{D_l}}$ where $\tau|_{D_l}$ restricts τ's domain to variables in $D_{\text{var}(l)}$. We obtain this by resolving with $\bar{\tau}|_{D_l} \vee l^{\tau|_{D_l}} \vee \bar{l}$. We accumulate universal literals from $\bar{\tau}$ in our axiom, but these can be reduced now there are no existential literals that block these literals.
Res: The resolution step is easy to p-simulate as Frege p-simulates resolution.
Inst: The final rule allows us to instantiate to increase the universal annotation uniformly everywhere in a clause. One $\mathsf{inst}(C, \beta)$ of size $O(w)$ can be simulated by lines that total size $O(w^2)$. Instantiation may replace literal x^α with $x^{\alpha \sqcup \beta'}$ for some $\beta' \subseteq \beta$. We already have definitions $\bar{\alpha} \vee \bar{x}^\alpha \vee x$, $\bar{\alpha} \vee \bar{\beta}' \vee x^{\alpha \sqcup \beta'} \vee \bar{x}$. Resolving over x gets us $\bar{\alpha} \vee \bar{\beta}' \vee x^{\alpha \sqcup \beta'} \vee \bar{x}^\alpha$. Neither x^α nor $x^{\alpha \sqcup \beta'}$ have α's variables in its dependency set, so we can now reduce the literals of $\bar{\alpha}$ to get $\bar{\beta}' \vee x^{\alpha \sqcup \beta'} \vee \bar{x}^\alpha$.

All rules and axioms, for any Frege system of choice.

$$\frac{}{L} \text{ (Axiom)}$$

L is a conjunct in the propositional matrix ϕ.

$$\frac{}{(\alpha \rightarrow (v \leftrightarrow \bigwedge_{y \in Y} y))} \text{ (IndExt-}\wedge\text{)} \qquad \frac{}{(\alpha \rightarrow (v \leftrightarrow \bigvee_{y \in Y} y))} \text{ (IndExt-}\vee\text{)}$$

v is a fresh $\exists$ variable, α is a conjunction of $\forall$ literals. $D_v = (\bigcup_{y \in Y} D_y) \setminus D_\alpha$.

$$\frac{L(u)}{L(0)} \text{ (0-red)} \qquad \frac{L(u)}{L(1)} \text{ (1-red)}$$

u is a $\forall$ variable. There is no $\exists$ variable x in $\text{var}(L)$ such that $u \in D_x$.
As an additional rule, the prefix Π may be weakened to Π' to add a new variable that does not appear in the matrix.

Figure 5 Proof rules of IndExtFrege $+ \forall$red.

We use these clauses to instantiate via resolving $O(w)$ many times. β' is necessarily a sub assignment of β, but we may accumulate any of the literals of $\bar{\beta}$ in our clause. Instantiation means that there will be no existential variable remaining that will depend on the variables of β. And so all universal literals that accumulate can be reduced. The lines are of $O(w)$ size and we involve $O(w)$ many of them to simulate this line.   ◀

▶ **Example 4.** Let our DQBF prefix be $\forall uvw\exists a(u,v)b(w)$. Suppose we have the instantiation step $\text{inst}(a^u \vee b, v\bar{w})$. In a p-simulated DQBF-IR-calc proof we already have some clauses that define annotated variables a^u, a^{uv}, $b^{\bar{w}}$. We can resolve $(\bar{u} \vee \bar{a}^u \vee a)$ and $(\bar{u} \vee \bar{v} \vee a^{uv} \vee \bar{a})$ to get $(\bar{u} \vee \bar{v} \vee \bar{a}^u \vee a^{uv})$, now we reduce $u = 1$ to get $(\bar{v} \vee \bar{a}^u \vee a^{uv})$. Using $(\bar{v} \vee \bar{a}^u \vee a^{uv})$ and $a^u \vee b$ we get $\bar{v} \vee a^{uv} \vee b$, we resolve again with $(w \vee b^{\bar{w}} \vee \bar{b})$ to get $\bar{v} \vee w \vee a^{uv} \vee b^{\bar{w}}$. We then reduce with $v = 1, w = 0$ to get $a^{uv} \vee b^{\bar{w}}$ which is exactly what $\text{inst}(a^u \vee b, v\bar{w})$ becomes under our prefix.

Notice that unlike previous simulations of IR-calc such as the simulation of IR-calc by eFrege $+ \forall$red [16], we are not formalising the strategy, but going line-by-line and replicating each line and its original semantic meaning. In QBF IR-calc relies on the base propositional inference rule being as weak as resolution for strategy extraction to be possible, but here we can use stronger forms of inference on instantiated clauses.

▶ **Corollary 5.** IndExtFrege $+ \forall$red *is refutationally complete for S-form DQBFs.*

Proof. Technically, simulating DQBF-IR-calc only shows this for DQBF with CNF matrices. Any propositional formula can be transformed into a logically equivalent CNF through enumerating all falsifying assignments to ϕ. This can be done with Frege rules and the refutation can then proceed by simulating DQBF-IR-calc.   ◀

▶ **Theorem 6.** IndExtFrege $+ \forall$red *is a sound refutational proof system.*

Proof. We claim that if DQBF $\forall U \exists E \phi$ is true and a number of lines $L_1 \ldots L_n$ are derived by IndExtFrege $+ \forall$red from $\forall U \exists E \phi$, then $\forall U' \exists E' \phi \wedge L_1 \wedge \cdots \wedge L_n$ is also a true DQBF, where E' and U' extend the prefix only to include the new variables added by prefix weakening or

the Independent Extension rules from $L_1 \ldots L_n$. We can prove that if $\forall U' \exists E' \phi \wedge L_1 \wedge \cdots \wedge L_n$ has a set of Skolem functions that satisfies all conjuncts then $\forall U'' \exists E'' \phi \wedge L_1 \wedge \cdots \wedge L_{n+1}$ has a set of Skolem functions that satisfies all conjuncts (where E'' and U'' extend E' and U', respectively to include the new variables of L_{n+1}). Axiom and Frege rules and reduction rules preserve winning Skolem functions. For Axiom and Frege this is easy to see. Since these rules preserve models in propositional logic they preserve whether a Skolem function satisfies all lines. For reduction, if every line is satisfied by the Skolem functions, this includes $L(u)$, and this does not change under both values of u and factoring in these values does not necessitate updating any of the Skolem functions of the variables of L because they do not depend on u (Lemma 1). The IndExt axioms preserve DQBF truth by Lemma 2.   ◄

In the following example, we demonstrate that IndExtFrege $+ \forall$red is conditionally strictly stronger than eFrege $+ \forall$red.

▶ **Example 7.** Let X be the set of variables $\{x_1, \ldots, x_{2n}\}$ and $\phi(X)$ a CNF in the variables of X. Then $\Pi\phi(X)$ with prefix $\Pi = \forall x_1 \exists x_2 \forall x_3 \ldots \exists x_{2n}$ is a closed PCNF. We also define a second set of mirrored variables $X' = \{x_1', \ldots, x_{2n}'\}$. We can define two false QBF families:

Duality$(\Pi\phi) = \exists x_1' \forall x_1 \exists x_2 \forall x_2' \ldots \exists x_{2n-1}' \forall x_{2n-1} \exists x_{2n} \forall x_{2n}' \phi(X) \wedge \neg\phi(X')$,

Select$(\Pi\phi) = \forall u \exists x_1' \forall x_1 \exists x_2 \forall x_2' \ldots \exists x_{2n} \forall x_{2n}' (\phi(X) \vee u) \wedge (\neg\phi(X') \vee \neg u)$.

Due to the prefix ordering, Duality has an easy strategy for each variable, but Select has a PSPACE-hard strategy for its first variable u. It was shown [13] that using the easy strategy Duality always has a short proof in eFrege $+ \forall$red. Because of strategy extraction, Select cannot have short proofs in eFrege $+ \forall$red unless PSPACE $\subset$ P/poly, but they have short proofs in IndExtFrege $+ \forall$red.

For each variable x_i we create new variable y_i, if x_i is existential in Π then we use definition $\neg u \to (y_i \leftrightarrow x_i)$ if x_i is universal then we use definition $u \to (y_i \leftrightarrow x_i')$. These variables have a dependency subset of x_i and x_i', respectively. Importantly, they are independent of u, much like in the Duality formula where there is no u variable. Now we replace all existential variables x_i and x_i' in $(\phi(X) \vee u) \wedge (\neg\phi(X') \vee \neg u)$ with y_i variables. This gives us two conjuncts $(\phi(Y) \vee u)$ and $(\neg\phi(Y') \vee \neg u)$ where $Y = \{y_i \mid \exists x_i \in \Pi\} \cup \{x_i \mid \forall x_i \in \Pi\}$ and $Y' = \{y_i \mid \exists x_i' \in \Pi'\} \cup \{x_i' \mid \forall x_i' \in \Pi'\}$. The conditional part of the definition is absorbed by the $\vee u$ and $\vee \neg u$ part, respectively. In each conjunct, u can be reduced due to independence from all the existential variables. Therefore we have $\phi(Y) \wedge \neg\phi(Y')$, and structurally we have the Duality$(\Pi\phi)$ formula, so we simply proceed with the short eFrege $+ \forall$red proof to get a short proof. Because these proofs are short and uniform, IndExtFrege $+ \forall$red cannot have polynomial time strategy extraction unless P $=$ PSPACE and so eFrege $+ \forall$red cannot p-simulate IndExtFrege $+ \forall$red unless P $=$ PSPACE.

3.3 A Resolution Version

We can better demonstrate the simplicity of the new rule by defining an equivalent resolution system that uses only four clausal rules. We give the proof system in Figure 6.

▶ **Theorem 8.** IndExtQURes *and* IndExtFrege $+ \forall$red *are p-equivalent.*

Proof. ($\Leftarrow$) We p-simulate each individual rule. (Ax) in IndExtQURes is a straightforward applications of (Ax) in IndExtFrege $+ \forall$red. (Red) in IndExtQURes can be p-simulated using $(0 - \text{red})$ or $(1 - \text{red})$, depending on whether the reduced literal was positive of negative, respectively. Using Frege we can say a literal equal to 0 or $\neg 1$ is equivalent to it being removed. It is well-known that the resolution rule is p-simulated by Frege. For the (IndExt) rule we us the (IndExt-$\vee$) rule in Figure 5 on $Y = \{\bar{y}_1, \bar{y}_2\}$. Then each of the three clauses

$$\frac{}{L} \ (\text{Ax}) \qquad\qquad \frac{C \vee u}{C} \ (\text{Red}) \qquad\qquad \frac{E \vee \neg x \qquad F \vee x}{E \vee F} \ (\text{Res})$$

L is a clause in the propositional matrix ϕ. u is a $\forall$ literal. There is no $\exists$ literal l in C such that $\text{var}(u) \in D_{\text{var}(l)}$, and there is no $\bar{u} \in C$.

$$\frac{}{(\bar{\alpha} \vee v \vee y_1), (\bar{\alpha} \vee v \vee y_2), (\bar{\alpha} \vee \bar{v} \vee \bar{y}_1 \vee \bar{y}_2)} \ (\text{IndExt})$$

v is a fresh $\exists$ variable, α is a conjunction of $\forall$ literals. $D_v = (D_{y_1} \vee D_{y_2}) \setminus D_\alpha$.

As an additional rule, the prefix Π may be weakened to Π' to add a new variable that does not appear in the matrix.

Figure 6 Proof rules of IndExtQURes.

for (IndExt) in Figure 6 is a propositional implicant of the formula derived by (IndExt-$\vee$) from Figure 5 and can be derived using a short Frege proof. Finally whenever the empty clause $\perp$ is derived in IndExtQURes, the final steps either use a resolution step or a reduction step. In either case we can p-simulate and derive 0 instead.

($\Rightarrow$) we interpret every line L_i in IndExtFrege + $\forall$red as an extension variable l_i that is built as the circuit for the formula in the line. All Frege rules and axioms can be p-simulated by the well know p-simulation of Ext Frege by Ext Res. The axiom rule for IndExtFrege + $\forall$red will technically have to be p-simulated by deriving a singleton clause with the variable l representing the disjunction, but every literal x in the axiom clause will be used in a definition clause $\neg x \vee l$, and so we resolve away the literals until we get singleton l.

For (IndExt-$\wedge$) and (IndExt-$\vee$), consider Figure 5 where we define a new variable v we represent the $\bigwedge_{y \in Y} y$ and $\bigvee_{y \in Y} y$ formulas from Figure 5 with an extension variable p, we take the variables y_1 and y_2 from Figure 6 to both be $\bar{p}$ and then we define v using (IndExt) (using the same α). Once we define extension variable q with $l \leftrightarrow (\alpha \rightarrow (v \leftrightarrow w))$ we can resolve to get singleton l.

Suppose we perform a reduction from $L(u)$ to $L(0)$ in IndExtFrege + $\forall$red. Let us label $L(u)$ as p and $L(0)$ as q. $L(u) \wedge \bar{u} \rightarrow L(0)$ is an obvious propositional tautology and we use the p-simulation of Ext Frege by Ext Res to derive clause $\bar{p} \vee u \vee q$, as we do not have weakening in IndExtQURes, we may obtain a stronger clause, which is just as useful. Once we obtain singleton p we resolve it to get $q \vee u$. u is not blocked by q and so we reduce to get q, or even the empty clause (which saves us from all subsequent lines). This works symmetrically for $(1 - \text{red})$ as it does for $(0 - \text{red})$

And finally if we derive l which represents the empty clause from IndExtFrege + $\forall$red, we simply resolve with $\neg l$ which is part of the definition of l to get the empty clause in IndExtQURes. ◀

4 P-simulations

In order to show completeness we have already shown that IndExtFrege + $\forall$red p-simulates DQBF-IR-calc. We can show more p-simulations that demonstrate the power of this new extension rule, including Fork Resolution, QRAT, and $\mathcal{D}^{\text{rrs}}$ based systems. With these p-simulations we show that IndExtFrege + $\forall$red can capture the vast majority of (D)QBF solving and preprocessing techniques.

These p-simulations also demonstrate that $\mathsf{IndExtFrege} + \forall\mathsf{red}$ is powerful. A proof system is roughly as powerful as the most expressive object it can "cut" on. Resolution cuts on literals, Frege cuts on propositional formulas, bounded-depth Frege cuts on bounded depth formulas and G cuts on QBFs. The extension variables in eFrege and Ext Res cut on propositional circuits, as is the case in $\mathsf{eFrege} + \forall\mathsf{red}$. $\mathsf{IndExtFrege} + \forall\mathsf{red}$ allows extension variables to be more expressive and by combining them we can express PSPACE-hard objects, and we make use of this in the p-simulations of this section.

4.1 P-Simulation of Fork Resolution

In this section we show how $\mathsf{IndExtFrege} + \forall\mathsf{red}$ can p-simulate another DQBF proof system that has a different notion of extension variables.

The Fork Resolution proof system is sound and complete for DQBFs that have a CNF matrix. Its main novelty is a Fork Extension rule [26], which is used for splitting clauses.

$$\frac{C_1 \cup C_2}{e \vee C_1 \quad \bar{e} \vee C_2} \text{ (F-Ext)} \qquad \frac{x \vee C_1 \quad \bar{x} \vee C_2}{C_1 \cup C_2} \text{ (Res)} \qquad \frac{C \vee u}{C} \text{ ($\forall$red)}$$

e is a new $\exists$-variable and has dependency set $(\bigcup_{\mathrm{var}(y)\in C_1} D_y) \cap (\bigcup_{\mathrm{var}(y)\in C_2} D_y)$. Fork Resolution also uses a resolution rule and a reduction rule: provided $\bar{u}$ is not in C and $\mathrm{var}(u)$ is not in the dependency set of C.

Given an instance of the Fork Extension on $C_1 \cup C_2$, the idea is to get an extension variable x that is equivalent to $\forall\{u \in U \mid u \in (\bigcup_{y\in C_2} D_y), u \notin \bigcup_{y\in C_1} D_y\}C_2$. In other words x is true if C_2 is true under all assignments to the specific universal variables that govern C_2 but not C_1.

It should be clear why this should mean $C_1 \cup C_2 \to \bar{x} \vee C_2$, because x is a stronger version of C_2. $C_1 \cup C_2$ also implies $x \vee C_1$ is true because if a Skolem function always satisfies $C_1 \cup C_2$, then C_2 must be satisfied, whenever C_1 is falsified regardless of the values of any remaining dependencies of C_2.

▶ **Lemma 9.** $\mathsf{IndExtFrege} + \forall\mathsf{red}$ *can p-simulate the Fork Extension rule (F-Ext).*

Proof. We order all universal variables in the *marginal* dependency set of C_2, $\{u_1, \ldots, u_k\} = \{u \in U \mid u \in (\bigcup_{y\in C_2} D_y), u \notin (\bigcup_{y\in C_1} D_y)\}$. We define:

$$e_0 \leftrightarrow C_2, \;\; u_{i+1} \to (e_i^{u_{i+1}} \leftrightarrow e_i), \;\; \bar{u}_{i+1} \to (x_i^{\bar{u}_{i+1}} \leftrightarrow e_i), \;\; e_{i+1} \leftrightarrow (e_i^{u_{i+1}} \wedge e_i^{\bar{u}_{i+1}})$$

And we can use the extension clauses that give these definitions. We make the induction hypothesis that there is a short proof of $e_i \vee C_1$ and of $\bar{e}_i \vee C_2$ in $\mathsf{IndExtFrege} + \forall\mathsf{red}$. Starting from $i = 0$ and incrementing until $i = k$.

Base Case. We resolve off all the C_2 literals in $C_1 \vee C_2$ to get $C_1 \vee e_0$. $\bar{e}_0 \vee C_2$ is part of the definition of $e_0 = C_2$.

Inductive Step. We start with $C_1 \vee e_i$ and $\bar{e}_i \vee C_2$. Resolving $\bar{u}_{i+1} \vee e_i \vee \bar{e}_i^{u_{i+1}}$ with $u_{i+1} \vee e_i \vee \bar{e}_i^{\bar{u}_{i+1}}$ gets us $e_i \vee \bar{e}_i^{u_{i+1}} \vee \bar{e}_i^{\bar{u}_{i+1}}$ and then with $\bar{e}_{i+1} \vee e_i^{u_{i+1}}$ and $\bar{e}_{i+1} \vee e_i^{\bar{u}_{i+1}}$, we get that $e_i \vee \bar{x}_{i+1}$. We can then, with a resolution step, get $\bar{e}_{i+1} \vee C_2$

In $C_1 \vee e_i$ we replace literal e_i with $\bar{u}_{i+1} \vee e_i^{u_{i+1}}$. Since $\bar{u}_{i+1}$ is not blocked by any literal in C_1 and not blocked by $e_i^{u_{i+1}}$ it can be reduced to derive $C_1 \vee e_i^{u_{i+1}}$, $C_1 \vee x_i^{\bar{u}_{i+1}}$ is derived in the same way. We resolve both with $e_{i+1} \vee \bar{e}_i^{u_{i+1}} \vee \bar{e}_i^{\bar{u}_{i+1}}$ to get $C_1 \vee e_{i+1}$

This DAG-like process is linear in the number of induction steps, potentially multiplied by the width of $C_1 \cup C_2$. To finalise, we take $e = e_k$, and necessarily it is not dependent on variables not in the dependency set of C_2 because of the initial definition of e_0 and any variable not in the dependency set of C_1 is removed by the time we get to e_k. ◀

▶ **Corollary 10.** IndExtFrege + ∀red *p-simulates Fork Resolution.*

Unlike in eFrege + ∀red where new variables are essentially propositional circuits, here we show how a new variable can be efficiently constructed to resemble a QBF properly. IndExtFrege + ∀red shares the ability to cut over a QBF with G, which goes some way to explain why IndExtFrege + ∀red is so powerful.

4.2 P-simulation of QRAT

In this section with show our DQBF proof system IndExtFrege + ∀red can p-simulate the QBF proof system QRAT [18], which governs many preprocessing steps in QBF.

4.2.1 Definition of QRAT

In propositional logic, interference based proof systems modify a CNF into another equisatis-fiable CNF without preserving models. Resolution, is not an interference based proof system because it preserves logical equivalence. An example of an interference based proof system is DRAT [31], where clause addition can destroy an existing model, but only when there is a guarantee that another model exists and is preserved. QRAT has similar rules to DRAT, but since QRAT is a QBF proof system, quantifier order must be respected. Since QRAT works in QBF, the notion of preserving individual models, is understood as preserving Skolem functions. QRAT does not preserve Skolem functions.

▶ **Definition 11** ([10])**.** *Fix a QBF prefix Π. Now consider a clause D and a literal l (not necessarily in D) we define, $O_D = \{k \in D \mid \mathrm{lv}(k) \le \mathrm{lv}(l), \mathrm{var}(k) \ne \mathrm{var}(l)\}$, where $\mathrm{lv}(k) \le \mathrm{lv}(l)$ indicates that k has a lower quantifier level than l, i.e k is left of l or, because we include equality, both are part of the same quantifier block (i.e. same dependency set). O_D is called the* outer clause*. For a DQBF we define the outer clause O_D differently for existential and universal literals l.*

When l is existential: $O_D := \{x \in D \mid x \in \exists, D_{\mathrm{var}(x)} \subseteq D_{\mathrm{var}(l)}, \mathrm{var}(x) \ne \mathrm{var}(l)\} \cup \{x \in D \mid x \in \forall, \mathrm{var}(x) \in D_{\mathrm{var}(l)}\}$

When l is universal we define a set of existential variables $\mathcal{S} = \{x \in \exists \mid \mathrm{var}(l) \in D_x\}$ from that we define a set of universal variables $T = \bigcap_{x \in \mathcal{S}} D_x \setminus \{\mathrm{var}(l)\}$, and we can then define the outer clause of D: $O_D := \{x \in D \mid x \in \exists, D_{\mathrm{var}(x)} \subseteq T\} \cup \{x \in D \mid x \in \forall, \mathrm{var}(x) \in T\}$.

The outer clause is a concept used to evaluate whether new clauses can be added, as it is the critical part that decides soundness. This is in the same vein as how extension clauses rely on the outer variables of the definition.

For some rules in QRAT, they can only be applied soundly if some semantic implication is known, but for a proof system we need to know if these implications are true in polynomial time. To do this we use the sound but incomplete unit propagation procedure.

▶ **Definition 12.** $\phi \vdash_1 \bot$ *means we arrive at the empty clause through the following process: For every clause in ϕ of size one (in other words a unit clause), $\{l\}$ we add l to a partial assignment α ($\mathrm{var}(l)$ is assigned the polarity of literal l). Now we remove literals from clauses in ϕ whenever they conflict with α. This may produce more unit clauses, or even an empty clause, we repeat this process until no more unit clauses are produced.*

When it comes to simulate QRAT, fortunately unit propagation is something that can be easy to p-simulate in a sequence of rules. QRAT's first rule uses unit propagation very directly to infer clauses propositionally, in this rule we do not have to consider quantifier type or order.

▶ **Definition 13** (Asymmetric Tautology Addition (ATA)). *Let ϕ be a CNF with Π a prefix. Let C be a clause not in ϕ. Let Π' be a prefix including the variables of C and ϕ, $\Pi \subset \Pi'$. Suppose $\phi \wedge \bar{C} \vdash_1 \bot$. Then we can infer $\Pi' \phi \wedge C$ from $\Pi\phi$.*

The next rule, QRATA deals with adding or removing a clause, but this time the Skolem function for a particular existential literal l changes as a result of this rule. This means that QRATA preserves truth but does not necessarily preserve the strategies. QRATA is a generalisation of the extension rule in $\mathsf{eFrege} + \forall\mathsf{red}$, and as such we have to respect the quantification order.

▶ **Definition 14** (Quantified Resolution Asymmetric Tautology Addition (QRATA)). *Let $\Pi\phi$ be a PCNF with closed prefix Π and CNF matrix ϕ. Let C be a clause not in ϕ. Let Π_1 and Π_2 be disjoint prefixes and x a variable such that $\Pi \subseteq \Pi_1 \exists x \Pi_2$ and a literal l with $\mathrm{var}(l) = x$. The difference in prefix is simply to allow new variables coming from $C \vee l$. For every clause $D \in \phi$ with $\bar{l} \in D$ if $\phi \wedge \bar{C} \wedge \bar{l} \wedge \bar{O}_D \vdash_1 \bot$, then we can derive $\Pi_1 \exists x \Pi_2 \phi \wedge (C \vee l)$ from $\Pi\phi$.*

The next two rules QRATU and EUR remove a universal literal from a clause, with side conditions which we will soon introduce.

$$\frac{\Pi_1 \forall u \Pi_2 \phi \wedge (C \vee l)}{\Pi_1 \forall u \Pi_2 \phi \wedge C} \text{ (QRATU)} \qquad \frac{\Pi_1 \forall u \Pi_2 \phi \wedge (C \vee l)}{\Pi_1 \forall u \Pi_2 (\phi \wedge C)} \text{ (EUR)}$$

For QRATU the condition is similar to that of QRATA but instead of adding a blocked clause over an existential literal, it removes a blocked universal literal. EUR has a different condition based on resolution paths.

▶ **Definition 15** (Quantified Resolution Asymmetric Tautology Universal (QRATU)). *Let $\Pi_1 \forall u \Pi_2 \phi$ be a PCNF with closed prefix $\Pi_1 \forall u \Pi_2$ and CNF matrix ϕ. Let $C \vee l$ be a clause with universal literal l, with $\mathrm{var}(l) = u$. If for every $D \in \phi$ with $\bar{l} \in D$, $\phi \wedge \bar{C} \wedge \bar{O}_D \vdash_1 \bot$, then we can derive $\Pi_1 \forall u \Pi_2 \phi \wedge C$ from $\Pi_1 \forall u \Pi_2 \phi \wedge (C \vee l)$.*

For EUR, we consider potential resolution steps connecting clauses to one another. Clauses being in different connected components indicate independence from one another and we can expand on this idea to calculate when a universal literal is locally pure and can only be considered one polarity.

▶ **Definition 16.** *Consider a CNF ϕ and subset χ of clauses in ϕ and a subset $\mathcal{S}$ of variables. $\mathfrak{L}(\phi, \chi, \mathcal{S})$ lists the $\mathcal{S}$-literals on the resolutions paths from χ and $\mathfrak{C}(\phi, \chi, \mathcal{S})$ lists the clauses on the resolution paths from χ. These are found using an iterative procedure until reaching a fix-point.*
Initialisation. *We start with the clauses in χ and the $\mathcal{S}$ literals in those clauses. $\mathfrak{L}(\phi, \chi, \mathcal{S}) \leftarrow \{l \mid$ there is $C \in \chi$ s.t. $l \in C, \mathrm{var}(l) \in \mathcal{S}\}$ and $\mathfrak{C}(\phi, \chi, \mathcal{S}) \leftarrow \chi$.*
Adding a clause. *If there if some $D \in \phi$ such that $\bar{p} \in D$ and $p \in \mathfrak{L}(\phi, \chi, \mathcal{S})$, then we can update $\mathfrak{L}(\phi, \chi, \mathcal{S})$ and $\mathfrak{C}(\phi, \chi, \mathcal{S})$. $\mathfrak{L}(\phi, \chi, \mathcal{S}) \leftarrow \mathfrak{L}(\phi, \chi, \mathcal{S}) \cup \{q \in D \mid q \neq \bar{p}, \mathrm{var}(q) \in \mathcal{S}\}$ and $\mathfrak{C}(\phi, \chi, \mathcal{S}) \leftarrow \mathfrak{C}(\phi, \chi, \mathcal{S}) \cup \{D\}$. We continue this until we reach fix-point, in other words for all $p \in \mathfrak{L}(\phi, \chi, \mathcal{S})$ if $D \in \phi$ and $\bar{p} \in D$, then $\{q \in D \mid q \neq \bar{p}, \mathrm{var}(q) \in \mathcal{S}\} \subset \mathfrak{L}(\phi, \chi, \mathcal{S})$ and $D \in \mathfrak{C}(\phi, \chi, \mathcal{S})$. Fix-point is reached in polynomial time.*

▶ **Definition 17.** *Let $\Pi_1 \forall u \Pi_2 \phi$ be a PCNF with closed prefix $\Pi_1 \forall u \Pi_2$ and CNF matrix ϕ. Let $C \vee l$ be a clause with universal literal l, with $\mathrm{var}(l) = u$.*

If the resolution path $\mathfrak{C}(\phi \wedge C, C, \mathcal{S})$ contains no clause D such that $\bar{l} \in D$, when $\mathcal{S}$ is the set of existential variables right of l in the prefix (i.e. in Π_2), then we can derive $\Pi_1 \forall u \Pi_2 (\phi \wedge C)$ from $\Pi_1 \forall u \Pi_2 \phi \wedge (C \vee l)$.

The refutational system QRAT includes ATA, QRATA, QRATU, EUR rules as well as arbitrary clause deletion and that the prefix allows for new variables to be added.

4.2.2 Interference-based Reasoning via Independent Extension Variables

Our p-simulation is somewhat surprising, as QRAT includes a complicated Extended Universal Reduction rule that performs reductions according to global resolution path connectivity between clauses. On the surface level IndExtFrege $+\forall$red would seem to be ill-equipped to talk globally about a formula, as it does not have any global conditions. In fact IndExtFrege $+\forall$red is monotonic in the sense that no deletion rule is necessary. The way we p-simulate clause deletion is simply to ignore clauses that would have been in the QRAT proof. This has a small technicality, QRAT can eliminate a variable and then introduce it again with a different meaning, but in IndExtFrege $+\forall$red we have to take care to create new variables when we refresh the meanings of variables. In fact when we use an interference based rule such as QRATA, QRATU or EUR we create a new copy of the formula.

We begin with p-simulations of DQBF generalisations of ATA and QRATA, in fact we do not need the independence power of the new Independent Extension rule for this.

▶ **Lemma 18** ([21]). *Suppose we have a CNF ϕ and a clause $C \notin \phi$. If $\phi \wedge \bar{C} \vdash_1 \bot$ then there is a short resolution + weakening proof starting from ϕ to $\phi \wedge C$.*

Proof. Any simplification of a clause can be derived via resolution. If we have a unit clause x and clause $C \vee \bar{x}$, then the clause C can be derived in a single step. The number of literals removed before the empty clause is bounded above by the total number of individual literals. Hence the short proof. ◀

▶ **Lemma 19.** *Suppose we have a DQBF: $\forall U \exists E \phi$. Let l be a literal of some existential variable in E and C be a clause not in ϕ, where C contains literal l.*

With respect to existential literal/variable l, recall the DQBF definition of an outer clause O_K of some clause K, according to Definition 11. If for every clause $D \in \phi$ with $\bar{l} \in D$ satisfies: $\phi \wedge \bar{C} \wedge \bar{O}_D \wedge \bar{l} \vdash_1 \bot$, then using a polynomial bounded (in the size of $\forall U \exists E \psi$) number of rules of IndExtFrege $+\forall$red we can extend $\exists E$ to $\exists E'$, which includes a new variable l' such that $D_{l'} \subseteq D_{\mathrm{var}(l)}$ and derive the clauses of $(\phi \wedge C)'$ which is a copy of $\phi \wedge C$ where every l literal in S is replaced by l' and every $\bar{l}$ literal in S is replaced by $\bar{l'}$.

Proof. The aim is to replace l with the substitution $l' = l \vee \bigwedge_{D \in \phi}^{\bar{l} \in D} O_D$ and this will respect all existing clauses and add C'. l' has the same dependency set as l because of the care taken when selecting literals for the outer clause. We can prove the clauses of ϕ' by cases:

1. Clauses $K \in \phi$ that do not contain l nor $\bar{l}$ can remain unchanged in ϕ'.
2. Clauses $K \in \phi$ that contain l but not $\bar{l}$ can be weakened to $K \vee \bigwedge_{D \in \phi}^{\bar{l} \in D} O_D$ and the replacement of $l \vee \bigwedge_{D \in \phi}^{\bar{l} \in D} O_D$ with l' gains us $K' = K \setminus \{l\} \cup \{l'\}$.
3. Clauses $K \in \phi$ that contain $\bar{l}$ but not l have the property that the tautology $O_K \vee \bar{O}_K$ implies $K \setminus \{\bar{l}\} \vee \bigvee_{D \in \phi}^{\bar{l} \in D} \bar{O}_D$. Combine with a conjunction of K we get $K \setminus \{\bar{l}\} \vee (\bar{l} \wedge \bigvee_{D \in \phi}^{\bar{l} \in D} \bar{O}_D)$ and now can replace $(\bar{l} \wedge \bigvee_{D \in \phi}^{\bar{l} \in D} \bar{O}_D)$ with $\bar{l'}$ to get $K' = (K \setminus \{\bar{l}\}) \cup \{\bar{l'}\}$.
4. Clauses $K \in \phi$ that contain both l and $\bar{l}$ then since $l' \vee \bar{l'}$ is also tautological we can derive $K' = K \setminus \{l, \bar{l}\} \cup \{l', \bar{l'}\}$.

We use the short proof of $\phi \to C \vee l \vee \bigwedge_{D \in \phi}^{\bar{u} \in D} O_D$ from unit propagation to derive $C \vee l \vee \bigwedge_{D \in \phi}^{\bar{u} \in D} O_D$ and then replace $l \vee \bigwedge_{D \in \phi}^{\bar{u} \in D} O_D$ with l' to get $C' \vee l'$. ◀

Instead of p-simulating EUR and QRATU separately, what we will show here is a simulation of a powerful mixed reduction rule that acts like a combination of QRATU and EUR and even works for DQBF. QRATU and EUR will both be special cases when the prefix gives a QBF.

▶ **Lemma 20.** *Suppose we have a DQBF: $\forall U \exists E \psi$. Let u be a literal of some universal variable in U and $C \vee u$ be a clause in ψ, i.e $\psi = \phi \wedge (C \vee u)$. Recall Definition 11 for the definition of $\mathcal{S}$ Let $\chi_0 = \mathfrak{C}(\psi, C \vee u, \mathcal{S})$, the set of clauses reachable from $C \vee u$ via resolution path in $\mathcal{S}$.*

If for every clause $D \in \chi_0$ with $\bar{u} \in D$ satisfies: $\phi \wedge \bar{C} \wedge \bar{O}_D \vdash_1 \bot$, then using a polynomial bounded (in the size of $\forall U \exists E \psi$) number of rules of IndExtFrege *we can extend $\exists E$ to $\exists E'$, which includes for every variable x in $\mathcal{S}$ a variable x' such that $D_{x'} \subseteq D_x$ and derive the clauses of $(\phi \wedge C)'$ which is a copy of $\phi \wedge C$ where every x in $\mathcal{S}$ is replaced by x'.*

As specified in the lemma the proof is done by replacing literals with different extension variables. The origins of the replacement scheme used here trace back to the simulation of QRAT by G [14], although this simulation here simulates a more powerful rule and works in the DQBF setting. This is despite being seemingly handicapped by only being able to create definitions that are existential variables, whereas G can use complicated functions to witness universal variables. To overcome this we hide functions that previously witness universal variables into new definitions of the existential variables that depend on them.

Proof. We start by introducing the extensions $\bar{u} \to (x^{\bar{u}} \leftrightarrow x)$ and $u \to (x^u \leftrightarrow x)$ for every $x \in \mathcal{S}$. Let $\mathcal{O} = \bigvee_{D \in \chi_0}^{\bar{u} \in D} \bar{O}_D$. For each literal in $L = \mathfrak{L}(\psi, C \vee u, \mathcal{S})$ and their complement we now define $l^* = \left(\bar{u} \vee \bar{\mathcal{O}} \to l^{\bar{u}}\right) \wedge (u \wedge \mathcal{O} \to l^u)$. In other words, l^* is $l^{\bar{u}}$ except in the circumstance that u is true and some outer clause is falsified, and in that case and only that case we have l^* takes the value from l^u. The definition is consistent with negation (i.e. $\neg l^* = (\neg l)^*$). The dependency set of l^* is no greater than that of l.

One may notice that if there were Skolem functions that satisfied $\forall U \exists E \psi$, these Skolem functions extended to the new $*$-variables would also satisfy χ_0^*, which replaces all x variables with x^* in χ_0. This is because if u is false every l^* plays as $l^{\bar{u}}$ which is how l plays anyway. If u is true and some O_D for $D \in \chi_0, \bar{u} \in D$ is falsified, all l^* variables play consistently as l^u which is consistent with l and u and thus satisfies all clauses. If u is true and every O_D for $D \in \chi_0, \bar{u} \in D$ is satisfied, l^* plays as $l^{\bar{u}}$ which will still satisfy every clause in χ_0 except these D, but each of them is already satisfied from their outer clause.

We have to actually construct χ_0^* using rules from IndExtFrege. There are four types of clauses K in χ_0^*. For each we have to construct K^* which replaces every $\mathcal{S}$-literal l with l^* in K: **1.** K contains no u literal. **2.** K contains a positive u literal only. **3.** K that contains a negative u literal only. **4.** K contains a tautology over u.

1. For each $\mathcal{S}$-literal l of K we can use definition $\bar{u} \to (l^{\bar{u}} \leftrightarrow l)$. For K we replace each of these literals l for $l^{\bar{u}} \vee u$ via resolving with the definitions. At the end of this process we can denote this clause as $K^{\bar{u}} \vee u$. There is no literal that blocks u from reduction here so we can derive $K^{\bar{u}}$. We do this similarly with definitions $u \to (l^u \leftrightarrow l)$ to create K^u. $(u \wedge \mathcal{O}) \vee K^* \vee \bar{K}^{\bar{u}}$ is a tautology once the definition of l^* is considered which can be proven in a linear size Frege proof. Likewise $\bar{u} \vee \bar{\mathcal{O}} \vee K^* \vee \bar{K}^u$ is a tautology which can be proven in a linear size Frege proof. And by resolving disjuncts together we are left with K^*.

$$\frac{K^{\bar{u}} \vee u}{K^{\bar{u}}} \qquad (u \wedge \mathcal{O}) \vee K^* \vee \bar{K}^{\bar{u}} \qquad \frac{K^u \vee \bar{u}}{K^u} \qquad \bar{u} \vee \bar{\mathcal{O}} \vee K^* \vee \bar{K}^u$$
$$\frac{(u \wedge \mathcal{O}) \vee K^*}{\qquad \qquad \bar{u} \vee \bar{\mathcal{O}} \vee K^*}$$
$$K^*$$

2. For K, we replace each of these literals l for $l^{\bar{u}} \vee u$. At the end of this process we can denote this clause as $K^{\bar{u}}$, we do not have to include the extra u literal as that is already included. We do this similarly with definitions $u \to (l^u \leftrightarrow l)$ to create $K^u \vee \bar{u}$. The $\bar{u}$ need not be reduced as it will be absorbed into the $\bar{u}$ disjunct later. $(u \wedge \mathcal{O}) \vee K^* \vee \bar{K}^{\bar{u}}$ is a tautology once the definition of l^* is considered which can be proven in a linear size Frege proof. Likewise $(u \vee \bar{\mathcal{O}}) \vee K^* \vee \bar{K}^u$ is a tautology which can be proven in a linear size Frege proof. And by resolving disjuncts together we are left with K^*.

3. We can derive $(u \wedge \bigvee_{D \in \chi_0}^{\bar{u} \in D} \bar{O}_D) \vee K^*$ simply because $\bar{O}_K \vee O_K$ and $u \vee \bar{u}$ are tautological clauses and that K^* which replaces every $\mathcal{S}$-literal l with l^*, contains $\bar{u}$ and O_K as subclauses. The clause is derived using distributivity.

   We replace each of these literals l for $l^u \vee \bar{u}$. At the end of this process we can denote this clause as K^u we do not have to include the extra $\bar{u}$ literal as that is already included. $u \vee \bar{\mathcal{O}} \vee K^* \vee \bar{K}^u$ is a tautology which can be proven in a linear size Frege proof. K^* can be derived by resolving disjuncts.

$$\bar{O}_K \vee O_K \qquad \frac{u \vee \bar{u}}{u \vee K^*} \qquad \frac{K^u \qquad \bar{u} \vee \bar{\mathcal{O}} \vee K^* \vee \bar{K}^{\bar{u}}}{\bar{u} \vee \bar{\mathcal{O}} \vee K^*}$$
$$\frac{\mathcal{O} \vee K^* \qquad \qquad \bar{\mathcal{O}} \vee K^*}{K^*}$$

4. K^* is a tautological clause.

We are not yet done, because we have only derived clauses for χ_0 not the whole of ϕ. To do this we need a slightly better replacement scheme than l to l^*. To start with, for $\mathcal{S}$ variables x where neither x nor $\bar{x}$ appear in L, we can use trivial replacement $x' = x$. Now consider $L = \mathfrak{L}(\psi, C \vee u, \mathcal{S})$. For any literal l such that $l, \bar{l} \in L$ then we use replacement l to l^*. For any literal $l \in L$ such that $\bar{l} \notin L$, we use replacement l to l' where $l' = l^* \vee l$. Note that l' and l have the same dependency set.

Consider χ_0^* and weaken every l^* literal such that $l \in L$ and $\bar{l} \notin L$ to $l \vee l^*$. Now consider all clauses in $\chi_1 = \mathfrak{C}(\psi, \phi \setminus \chi_0, \mathcal{S})$ and for each clause weaken every l literal such that $l \in L$ and $\bar{l} \notin L$ to $l \vee l^*$. It remains to replace $\bar{l} \notin L$ when $l \in L$. We will have to use $\bar{l}' = \bar{l}^* \wedge \bar{l}$ to match with l'. This can be done by distributivity in Frege, if every clause $K \vee \bar{l}$ we have a copy of $K' \vee \bar{l}$ and $K' \vee \bar{l}^*$ where K' is a copy of K with every literal $l \in \mathcal{S}$ replaced by l'.

This turns out to be basically the case because K can only contain literals k such that $k \in L$ and $\bar{k} \notin L$. $k \in L$ because we can extend the path from $C \vee u$ to l through $\bar{l}$. If $\bar{k} \in L$ then $\bar{l}$ would be in L as well, against our assumption. If $K \vee \bar{l} \in \chi_0 \cap \chi_1$ we already have $K' \vee \bar{l}^*$ and $K' \vee \bar{l}$. It is possible that $K \vee \bar{l} \in \chi_0$ but not χ_1 in which case we create $K' \vee \bar{l}$ by weakening each $\mathcal{S}$ literal in $K \vee \bar{l}$ that is not $\bar{l}$.

Finally we need clause C' which is the subclause C where every $\mathcal{S}$-literal l is replaced by l'. Note that all $\mathcal{S}$-literals in C are by definition in L, so first we can derive C^* and then weaken to get C'. To do this we need the unit propagation proofs of $\phi \wedge \bar{C} \wedge \bar{O}_D \vdash_1 \bot$ for each $D \in \chi_0$ and $\bar{l} \in D$. We prove $\phi \wedge \bar{C} \wedge \bar{O}_D \to \bot$ in $O(|\phi|)$ many Frege steps. We can then derive $C \vee \bigwedge_{D \in \chi_0}^{\bar{u} \in D} O_D$ from ϕ next. Now we can replace all $\mathcal{S}$-literals l in C with $l^u \vee \bar{u}$ where $u \to (l^u \leftrightarrow l)$. This gets us $\bar{u} \vee \bigwedge_{D \in \chi_0}^{\bar{u} \in D} O_D \vee C^u$ i.e $\bar{u} \vee \bar{\mathcal{O}} \vee C^u$. We also use $\bar{u} \to (l^{\bar{u}} \leftrightarrow l)$ and $C \vee u$ to replace every $\mathcal{S}$-literal l with $l^{\bar{u}}$ and get $C^{\bar{u}} \vee u$. The u literal can be reduced, (since this rule generalises the original universal reduction this is crucial). $(u \wedge \mathcal{O}) \vee C^* \vee \bar{C}^{\bar{u}}$ is a tautology once the definition of l^* is considered which can be proven in a linear size Frege proof. Likewise $\bar{u} \vee \bar{\mathcal{O}} \vee C^* \vee \bar{C}^u$ is a tautology which can be proven in a linear size Frege proof. And by resolving disjuncts together we are left with C^*. We can get always get C' through weakening C^*.

$$\cfrac{\cfrac{C^{\bar{u}} \qquad (u \wedge \mathcal{O}) \vee C^* \vee \bar{C}^{\bar{u}}}{(u \wedge \mathcal{O}) \vee C^*} \qquad \cfrac{\bar{u} \vee \bar{\mathcal{O}} \vee C^u \qquad \bar{u} \vee \bar{\mathcal{O}} \vee C^* \vee \bar{C}^u}{\bar{u} \vee \bar{\mathcal{O}} \vee C^*}}{\cfrac{C^*}{C'}} \qquad\qquad \blacktriangleleft$$

▶ **Example 21.** Consider the QBF with the prefix $\exists x \forall u \exists y, z, a$ and the matrix

$$\psi = (\overbrace{u \vee z}^{C \vee u}) \wedge (\overbrace{x \vee \bar{u} \vee y \vee \bar{z}}^{C_1}) \wedge (\overbrace{\bar{y} \vee z \vee a}^{C_2}) \wedge (\overbrace{x \vee \bar{a}}^{C_3}) \wedge (\overbrace{x \vee y \vee z}^{C_4}) \wedge (\overbrace{\bar{y} \vee \bar{z}}^{C_5}) \wedge (\overbrace{\bar{x} \vee \bar{u} \vee a}^{C_6})$$

We have $\mathcal{S} = \{y, z, a\}$, $\chi_0 = \mathfrak{C}(\psi, C \vee u, \mathcal{S}) = \{C \vee u, C_1, \ldots, C_5\}$, and $L = \mathfrak{L}(\psi, C \vee u, \mathcal{S}) = \{y, \bar{y}, z, \bar{z}, a\}$. We are looking at outer clauses of those clauses in χ_0 that contain $\bar{u}$, which is only C_1, with $O_{C_1} = \{x\}$. Notice that there is also the clause C_6 with $\bar{u}$ in it, but $C_6 \notin \chi_0$ because the only possible predecessor to C_6 is C_3 and it cannot be both entered and exited via its only $\mathcal{S}$-literal $\bar{a}$. We need to verify whether $C_1 \wedge \cdots \wedge C_6 \wedge \bar{C} \wedge \bar{O}_{C_1} \vdash_1 \bot$. Setting $\bar{C} \wedge \bar{O}_{C_1} = \bar{x} \wedge \bar{z}$ propagates $\bar{a}$ from C_3, then $\bar{y}$ from C_2, and finally $\bot$ from C_4. The preconditions of Lemma 20 are therefore satisfied, and C can be soundly obtained by reduction from $C \vee u$.

Let us now perform the substitutions from the proof of Lemma 20. First, we introduce the extensions $y^{\bar{u}}, y^u, z^{\bar{u}}, z^u, a^{\bar{u}}, a^u$. Next, we define the *-variables using $\mathcal{O} = \bigvee_{D \in \chi_0}^{\bar{u} \in D} \bar{O}_D = \bar{x}$:

$$y^* = (\bar{u} \vee x \to y^{\bar{u}}) \wedge (u \wedge \bar{x} \to y^u) \qquad\qquad z^* = (\bar{u} \vee x \to z^{\bar{u}}) \wedge (u \wedge \bar{x} \to z^u)$$
$$a^* = (\bar{u} \vee x \to a^{\bar{u}}) \wedge (u \wedge \bar{x} \to a^u)$$

We have four types of clauses in χ_0 depending on which literals of u they contain. Let us start with the clauses that do not contain any u literal, namely C_2, C_3, C_4, C_5. By resolution with the extension definitions and subsequent reduction of u, we obtain

$$C_2^u = \bar{y}^u \vee z^u \vee a^u \qquad C_3^u = x \vee \bar{a}^u \qquad C_4^u = x \vee y^u \vee z^u \qquad C_5^u = \bar{y}^u \vee \bar{z}^u$$
$$C_2^{\bar{u}} = \bar{y}^{\bar{u}} \vee z^{\bar{u}} \vee a^{\bar{u}} \qquad C_3^{\bar{u}} = x \vee \bar{a}^{\bar{u}} \qquad C_4^{\bar{u}} = x \vee y^{\bar{u}} \vee z^{\bar{u}} \qquad C_5^{\bar{u}} = \bar{y}^{\bar{u}} \vee \bar{z}^{\bar{u}}$$

For the sake of brevity, we shall demonstrate the next step only on C_3. Now $\bar{u} \vee \bar{\mathcal{O}} \vee C_3^* \vee \bar{C}_3^{\bar{u}} = \bar{u} \vee x \vee a^* \vee \bar{a}^u$ can be simplified to $\bar{u} \vee x \vee a^u \vee \bar{a}^u$, which is obviously a tautology. It can be derived as follows: from the definition of a^*, we obtain $u \wedge \bar{x} \to a^* = a^u$, and thus $u \wedge \bar{x} \wedge \bar{a}^u \to \bar{a}^*$, or in other words, the clause above. Other *-replacements for the clauses C_1, C_2, C_4, C_5 are performed in a similar fashion.

For the y, z the variables y^*, z^* are the final replacements, but for the variable a we need to create the variable $a' = a^* \vee a$. We have $\chi_1 = \mathfrak{C}(\psi, \{C_6\}, \mathcal{S}) = \{C_3, C_6\}$. We take $C_3^* = x \vee \bar{a}^*$ and $C_3 = x \vee \bar{a}$ and apply distributivity to obtain $C_3' = x \vee (\bar{a}^* \wedge \bar{a})$. For other clauses that contain a positively, we simply weaken to obtain C_i'. Note that if C_3 contained, say a z literal, we would not be able to apply distributivity, however a literal like z would also cause a path from C to C_6.

Finally, we need to prove $C \vee \bar{\mathcal{O}} = x \vee z$, which we get by resolving the clauses on the unit propagation path: C_4, C_3, and C_2. This will give us $x \vee z$. By resolving with the extension definitions for z^u and $z^{\bar{u}}$, we obtain $u \vee x \vee z^{\bar{u}}$ and $\bar{u} \vee x \vee z^u$. Because z^u and $z^{\bar{u}}$ do not depend on u, we can apply universal reduction to get the clauses $C^u = z^u$ and $C^{\bar{u}} = z^{\bar{u}}$. Finally, from the definition of z^* we have $z^{\bar{u}} \to (z^* \leftrightarrow (u \wedge \bar{x} \to z^u))$, and thus $z^{\bar{u}} \wedge z^u \to z^*$, and we obtain the desired Frege proof of $C^* = \{z^*\}$.

We have demonstrated ability to replace a conjunctive normal form with another and preserve DQBF satisfiability, by replacing variables with new variables from the Independent Extension rule. Changing the formula, but preserving satisfiability is a key part of preprocessing, and this gives us a key connections to interference proof systems like DRAT, QRAT and DQRAT, whose introduction was to certify preprocessing steps.

▶ **Lemma 22.** *Both QRATU and EUR are special cases of the rule p-simulated in Lemma 20.*

Proof.

QRATU. Suppose $C \lor u$ reduces to C via QRATU. We know for every clause $D \in \phi$ with $\bar{u} \in D$, $\phi \land \bar{O}_D \land \bar{C} \vdash_1 \bot$. Therefore no matter how many of these D belong to χ_0 in Lemma 20, every one satisfies the condition.

EUR. Suppose $C \lor u$ reduces to C via EUR. Via the side condition of EUR, χ_0 does not contain any clauses $D \in \phi$ with $\bar{u} \in D$ therefore the side condition of the rule in Lemma 20. ◀

With the lemmas we have shown, we now have the p-simulation of QRAT.

▶ **Corollary 23.** IndExtFrege + ∀red *can p-simulate QRAT for false QBFs.*

Proof. When adding a clause through ATA. We simply use Lemma 18 to add the clause. Instead of deleting a clause, we simply ignore it. IndExtFrege + ∀red does not have any global conditions for any of its rules, except that a new extension rule require a new name for the variable, but this is immaterial to the proof complexity. For simulating QRATA, QRATU and EUR we create a new copy of the entire formula and then forget about clauses and variables outside of this. For QRATA we use Lemma 19 and for QRATU and EUR we use different special cases of Lemma 20 which is allowed by Lemma 22. ◀

4.3 P-simulation of the $\mathcal{D}^{\mathrm{rrs}}$ Rule

The EUR rule uses resolution paths to calculate when it is safe to drop a single isolated literal. But in QBF and DQBF a more general approach can be taken, where resolution paths can be used to calculate that the dependency between a universal literal and a potential blocking existential literals is always spurious, in all clauses and throughout all stages of the proof.

The $\mathcal{D}^{\mathrm{rrs}}$ relation [28] calculates (u, x) only if there is a resolution path from u to $\bar{u}$ through existential x. $\mathcal{D}^{\mathrm{rrs}}$ is a sound way of modifying a DQBF prefix. We can add this to sound DQBF proof systems, in fact a number of QBF and DQBF proof system are already constructed in this way.

4.3.1 Definition of IR($\mathcal{D}^{\mathrm{rrs}}$)-calc

Q($\mathcal{D}^{\mathrm{rrs}}$)-Res [28] is a QBF proof system using the $\mathcal{D}^{\mathrm{rrs}}$ rule, we can extend this to IR($\mathcal{D}^{\mathrm{rrs}}$)-calc [3] which uses DQBF-IR-calc as its proof system after modifying the proof system via $\mathcal{D}^{\mathrm{rrs}}$. The rules can be seen in Figure 7.

4.3.2 Definition of DQRAT

Another proof system that uses $\mathcal{D}^{\mathrm{rrs}}$ is DQRAT. Here the $\mathcal{D}^{\mathrm{rrs}}$ rule is a prefix modification rule used in conjunction with rules similar to QRAT. The rules of DQRAT are given in full in [10] and we provide a simplified overview here.
ATA: Add a clause C if $\phi \land \bar{C} \vdash_1 \bot$.

Axiom rule: $\mathsf{axiom}(\phi)$

$$\frac{}{\{a^{\tau|_{D_a^{\mathsf{rrs}}}} \mid a \in C,\ \mathrm{var}(a) \in \exists\}}$$

- C is a clause in the matrix ϕ.
- $\tau|_{D_a^{\mathsf{rrs}}} := \{\neg l \mid l \in C, \mathrm{var}(l) \in \forall,$
  $\qquad\qquad (\mathrm{var}(l), \mathrm{var}(a)) \in D^{\mathsf{rrs}}\}$.

Instantiation rule: $\mathsf{inst}(C, \beta)$

$$\frac{C}{\{a^{(\alpha \circ \beta)|_{D_a^{\mathsf{rrs}}}} \mid a^{\alpha} \in C\}}$$

- β is a partial assignment to universal variables.
- $\alpha \circ \beta$ is $\{l \mid (l \in \alpha) \vee (l \in \beta \wedge l \notin \alpha)\}$
- $(\alpha \circ \beta)(a) := \{l \in \alpha \circ \beta \mid$
  $\qquad\qquad (\mathrm{var}(l), \mathrm{var}(a)) \in D^{\mathsf{rrs}}\}$.

Resolution rule: $\mathsf{res}(C_1, C_2, x^{\tau})$

$$\cdot\ \frac{C_1 \qquad C_2}{(C_1 \cup C_2) \setminus \{x^{\tau}, \neg x^{\tau}\}}$$

- $x^{\tau} \in C_1$ and $\neg x^{\tau} \in C_2$.

■ **Figure 7** Proof rules of $\mathsf{IR}(\mathcal{D}^{\mathsf{rrs}})$-calc [3] .

DQRAT$_\exists$: Add a clause C if l is an existential literal in C and if for all $D \in \phi$ with $\bar{l} \in D$ that $\phi \wedge \bar{C} \wedge O_D \vdash_1 \bot$. Where the outer clause O_D with respect to l is the set of all universal literals u in D, that are in $D_{\mathrm{var}(l)}$, the dependency set of l and all existential literals x in D, whose dependency set is no larger than l's, i.e. $D_{\mathrm{var}(x)} \subset D_{\mathrm{var}(l)}$.

UR: Modify a clause $C \vee u$ to C if u is a universal literal and not in the dependency set of C.

DQRAT$_\forall$: Modify a clause $C \vee u$ to C, if u is a universal literal and if for all $D \in \phi$ with $\bar{u} \in D$ that $\phi \wedge \bar{C} \wedge O_D \vdash_1 \bot$. Where the outer clause O_D with respect to u is the set of all existential literals x in D, that have variables in the set S and the set of universal literals v in D that are in the dependency set of every variables in S with $\mathrm{var}(v) \neq \mathrm{var}(u)$. S is given as the set of all existential variables that depend on u.

BPM: A new variable is added to the prefix. When $\exists$, an arbitrary dependency set is used.

DRRS: The dependency set of every existential variable can be modified to remove any spurious dependencies calculated by the Reflexive Resolution Path dependency scheme ($\mathcal{D}^{\mathsf{rrs}}$).

DEL: Delete a clause.

4.3.3 P-simulations using $\mathcal{D}^{\mathsf{rrs}}$

▶ **Lemma 24.** *Suppose we have a DQBF $\forall U \exists E \psi$. We can calculate the relation $\mathcal{D}^{\mathrm{rrs}}$ on $\forall U \exists E \psi$ between universal variables and existential variables. Suppose there is a universal variable u and we let $\mathcal{S}$ define the set of these variables that depend on u, $\mathcal{S} = \{x \in X \mid u \in D_x\}$. Using $\mathsf{IndExtFrege} + \forall\mathsf{red}$ we can derive the clauses of ψ' that replaces each variable $x \in \mathcal{S}$ such that $(u, x) \notin \mathcal{D}^{\mathrm{rrs}}$ with x' such that $D_{x'} \subset D_x \setminus u$.*

Proof. We focus on the variables $x \in \mathcal{S}$ that are declared independent to u via $\mathcal{D}^{\mathrm{rrs}}$ and we need to replace it and its negation in all clauses. If there is resolution path from u to x there is no resolution path from $\bar{u}$ to $\bar{x}$ and if there is a resolution path from $\bar{u}$ to x there is no resolution path from u to $\bar{x}$. We can write this formally, let χ_u be the subset of ψ which contains every clause with a u literal and let $\chi_{\bar{u}}$ be the subset of ψ which contains every clause with a $\bar{u}$ literal. Define $L_u = \mathfrak{L}(\psi, \chi_u, \mathcal{S})$ and $L_{\bar{u}} = \mathfrak{L}(\psi, \chi_{\bar{u}}, \mathcal{S})$, then for $(u, x) \notin \mathcal{D}^{\mathrm{rrs}}$, $(x \notin L_u \vee \bar{x} \notin L_{\bar{u}}) \wedge (x \notin L_{\bar{u}} \vee \bar{x} \notin L_u)$. This leaves the following possibilities (up to some symmetries):

1. $x \notin L_u$, $\bar{x} \notin L_u$, $x \notin L_{\bar{u}}$ and $\bar{x} \notin L_{\bar{u}}$ 3. $x \in L_u$, $\bar{x} \notin L_u$, $x \in L_{\bar{u}}$ and $\bar{x} \notin L_{\bar{u}}$
2. $x \in L_u$, $\bar{x} \notin L_u$, $x \notin L_{\bar{u}}$ and $\bar{x} \notin L_{\bar{u}}$ 4. $x \in L_u$, $\bar{x} \in L_u$, $x \notin L_{\bar{u}}$ and $\bar{x} \notin L_{\bar{u}}$

We also need the symmetric cases for $\bar{u}$ in case 3 using $u \to (x^u \leftrightarrow x)$. In cases 1, 2 and 3, we use both $\bar{u} \to (x^{\bar{u}} \leftrightarrow x)$ and $u \to (x^u \leftrightarrow x)$ to get clauses $u \vee \bar{x} \vee x^{\bar{u}}$ and $\bar{u} \vee \bar{x} \vee x^u$. Now resolve them to get $\bar{x} \vee x^{\bar{u}} \vee x^u$. Replace every x literal with $x^{\bar{u}} \vee x^u$ via resolution. We consider $x' = x^{\bar{u}} \vee x^u$. Now consider the presence of $\bar{x}$ in some clause $D \in \psi$. Every other $\mathcal{S}$-literal $l \in D$ cannot have $\bar{l} \in L_u$ nor $\bar{l} \in L_{\bar{u}}$, because it means $\bar{x} \in L_u$ or $\bar{x} \in L_{\bar{u}}$ which cannot happen outside of case 4. We should have a copy of D where every $\mathcal{S}$-literal l is replaced by $l^u \vee l^{\bar{u}}$ except $\bar{x}$. We duplicate this and in one copy replace $\bar{x}$ with $u \vee \bar{x}^{\bar{u}}$, the u literal can be reduced. In the other copy replace $\bar{x}$ with $\bar{u} \vee \bar{x}^u$ and here the $\bar{u}$ literal can be reduced. Now by taking a conjunction we effectively make the final S-literal replacement of this clause, which is to replace $\bar{x}$ with $\bar{x}^{\bar{u}} \wedge \bar{x}^u$.

In Case 4 use definition $\bar{u} \to (x^{\bar{u}} \leftrightarrow x)$ to replace every x literal with $u \vee x^{\bar{u}}$ and every $\bar{x}$ literal with $u \vee \bar{x}^{\bar{u}}$. We have one remaining issue. Clauses K with variables from case 4, may have an additional universal literal. Firstly we claim that both u and $\bar{u}$ literals are not present, this can only be introduced with another case 4 literal for $\bar{u}$ and this can only happen if there was a path from $\bar{u}$ to a literal k in K and a path from $\bar{u}$ to $\bar{k}$, but since $u \vee x^{\bar{u}}$, there would also be a path from u to k meaning k is not case 4. We also claim that you can always remove this universal literal u with Lemma 20, specifically the DQBF version of EUR. Suppose after the replacement there is a resolution path from K to some clause with $\bar{u}$ in it, this $\bar{u}$ comes from either a clause originally with $\bar{u}$ in it, or a clause with a case 4 variable in it of the opposite polarity. Either way, there would be a resolution path from $\bar{u}$ to literal x contradicting our assumptions in case 4. Therefore $x' = x^{\bar{u}}$ suffices.   ◀

▶ **Corollary 25.** IndExtFrege $+ \forall$red *can p-simulate IR($\mathcal{D}^{\mathsf{rrs}}$)-calc for false QBFs.*

▶ **Corollary 26.** IndExtFrege $+ \forall$red *can p-simulate DQRAT for false DQBFs.*

5   Conclusion

We have introduced a powerful new proof rule that p-simulates most (D)QBF pre-processing and solving rules. Some proof complexity questions remain, in particular whether IndExtFrege $+ \forall$red is reciprocally p-simulated by some of these proof systems. We have not provided any direct p-simulations that work for long-distance Q resolution and its variations in QBF proof systems. A major obstacle is that long-distance resolution in DQBF is unsound. It will still be possible to represent these rules with side conditions that force the prefix to be ordered and like a QBF. Technically a p-simulation is already known because QRAT p-simulates long-distance Q-resolution via blocked literal eliminations (using QRATU) [20]. It may be possible that other simulations work via the methods we have used. In the full preprint of this paper [15] we provide a proof that IndExtFrege $+ \forall$red p-simulates G.

Other techniques that may have simulations include other dependency scheme rules such as the tautology free dependency scheme [5]. One difficulty in simulating this currently is that not much is yet known about the Skolem transformations which have helped us find the right definitions for replacement variables.

References

1 Valeriy Balabanov and Jie-Hong R. Jiang. Unified QBF certification and its applications. *Formal Methods in System Design*, 41(1):45–65, 2012. `doi:10.1007/s10703-012-0152-6`.

2 Valeriy Balabanov, Magdalena Widl, and Jie-Hong R. Jiang. QBF resolution systems and their proof complexities. In Carsten Sinz and Uwe Egly, editors, *SAT 2014*, volume 8561 of *Lecture Notes in Computer Science*, pages 154–169. Springer, 2014. `doi:10.1007/978-3-319-09284-3_12`.

3 Olaf Beyersdorff, Joshua Blinkhorn, Leroy Chew, Renate A. Schmidt, and Martin Suda. Reinterpreting dependency schemes: Soundness meets incompleteness in DQBF. *Journal of Automated Reasoning*, 63(3):597–623, 2019. `doi:10.1007/s10817-018-9482-4`.

4 Olaf Beyersdorff, Joshua Blinkhorn, and Meena Mahajan. Building strategies into QBF proofs. In Rolf Niedermeier and Christophe Paul, editors, *36th International Symposium on Theoretical Aspects of Computer Science, STACS 2019, March 13-16, 2019, Berlin, Germany*, volume 126 of *LIPIcs*, pages 14:1–14:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. `doi:10.4230/LIPIcs.STACS.2019.14`.

5 Olaf Beyersdorff, Joshua Blinkhorn, and Tomáš Peitl. Strong (D)QBF dependency schemes via tautology-free resolution paths. In Luca Pulina and Martina Seidl, editors, *Theory and Applications of Satisfiability Testing – SAT 2020*, volume 12178 of *Lecture Notes in Computer Science*, pages 394–411, Cham, 2020. Springer International Publishing. `doi:10.1007/978-3-030-51825-7_28`.

6 Olaf Beyersdorff and Benjamin Böhm. Understanding the relative strength of QBF CDCL solvers and QBF resolution. In James R. Lee, editor, *12th Innovations in Theoretical Computer Science Conference, ITCS 2021, January 6-8, 2021, Virtual Conference*, volume 185 of *LIPIcs*, pages 12:1–12:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. `doi:10.4230/LIPIcs.ITCS.2021.12`.

7 Olaf Beyersdorff, Ilario Bonacina, Leroy Chew, and Jan Pich. Frege systems for quantified Boolean logic. *J. ACM*, 67(2), April 2020. `doi:10.1145/3381881`.

8 Olaf Beyersdorff, Leroy Chew, and Mikoláš Janota. Extension variables in QBF resolution. In Adnan Darwiche, editor, *Beyond NP, Papers from the 2016 AAAI Workshop*, volume WS-16-05 of *AAAI Technical Report*. AAAI Press, 2016. URL: `http://www.aaai.org/ocs/index.php/WS/AAAIW16/paper/view/12612`.

9 Olaf Beyersdorff, Leroy Chew, and Mikoláš Janota. New resolution-based QBF calculi and their proof complexity. *ACM Trans. Comput. Theory*, 11(4):26:1–26:42, 2019. `doi:10.1145/3352155`.

10 Joshua Blinkhorn. Simulating DQBF preprocessing techniques with resolution asymmetric tautologies. *Electron. Colloquium Comput. Complex.*, TR20, 2020. URL: `https://eccc.weizmann.ac.il/report/2020/112`.

11 Sravanthi Chede and Anil Shukla. Does QRAT simulate IR-calc? QRAT simulation algorithm for ∀ exp+ res cannot be lifted to IR-calc. *arXiv preprint arXiv:2107.04547*, abs/2107.04547, 2021. `doi:10.48550/arXiv.2107.04547`.

12 Leroy Chew. Proof simulation via round-based strategy extraction for QBF. In Toby Walsh, Julie Shah, and Zico Kolter, editors, *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pages 11176–11184. AAAI Press, 2025. `doi:10.1609/aaai.v39i11.33215`.

13 Leroy Chew and Judith Clymo. How QBF expansion makes strategy extraction hard. In Nicolas Peltier and Viorica Sofronie-Stokkermans, editors, *Automated Reasoning - 10th International Joint Conference, IJCAR 2020, Paris, France, July 1-4, 2020, Proceedings, Part I*, volume 12166 of *Lecture Notes in Computer Science*, pages 66–82. Springer, 2020. `doi:10.1007/978-3-030-51074-9_5`.

14 Leroy Chew and Marijn J. H. Heule. Relating existing powerful proof systems for QBF. In Kuldeep S. Meel and Ofer Strichman, editors, *25th International Conference on Theory and Applications of Satisfiability Testing, SAT 2022, August 2-5, 2022, Haifa, Israel*, volume 236 of *LIPIcs*, pages 10:1–10:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. `doi:10.4230/LIPIcs.SAT.2022.10`.

15 Leroy Chew and Tomáš Peitl. Better extension variables in dqbf via independence, 2025. `doi:10.48550/arXiv.2505.20069`.

16 Leroy Chew and Friedrich Slivovsky. Towards uniform certification in QBF. *Log. Methods Comput. Sci.*, 20(1), 2024. `doi:10.46298/lmcs-20(1:14)2024`.

17 Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *The Journal of Symbolic Logic*, 44(1):36–50, 1979. `doi:10.2307/2273702`.

18 Marijn Heule, Martina Seidl, and Armin Biere. A unified proof system for QBF preprocessing. In Stéphane Demri, Deepak Kapur, and Christoph Weidenbach, editors, *7th International Joint Conference on Automated Reasoning (IJCAR)*, Lecture Notes in Computer Science, pages 91–106, 2014. `doi:10.1007/978-3-319-08587-6_7`.

19 Toni Jussila, Armin Biere, Carsten Sinz, Daniel Kröning, and Christoph M. Wintersteiger. A first step towards a unified proof checker for QBF. In João Marques-Silva and Karem A. Sakallah, editors, *SAT 2007*, volume 4501 of *Lecture Notes in Computer Science*, pages 201–214. Springer, 2007. `doi:10.1007/978-3-540-72788-0_21`.

20 Benjamin Kiesl, Marijn J. H. Heule, and Martina Seidl. A little blocked literal goes a long way. In Serge Gaspers and Toby Walsh, editors, *SAT 2017*, volume 10491 of *Lecture Notes in Computer Science*, pages 281–297. Springer, 2017. `doi:10.1007/978-3-319-66263-3_18`.

21 Benjamin Kiesl, Adrián Rebola-Pardo, and Marijn JH Heule. Extended resolution simulates drat. In Didier Galmiche, Stephan Schulz, and Roberto Sebastiani, editors, *Automated Reasoning: 9th International Joint Conference, IJCAR 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 14-17, 2018, Proceedings*, volume 10900 of *Lecture Notes in Computer Science*, pages 516–531. Springer, Springer, 2018. `doi:10.1007/978-3-319-94205-6_34`.

22 Benjamin Kiesl and Martina Seidl. QRAT polynomially simulates ∀Exp+Res. In Mikolás Janota and Inês Lynce, editors, *Theory and Applications of Satisfiability Testing - SAT 2019 - 22nd International Conference, SAT 2019, Lisbon, Portugal, July 9-12, 2019, Proceedings*, volume 11628 of *Lecture Notes in Computer Science*, pages 193–202. Springer, 2019. `doi:10.1007/978-3-030-24258-9_13`.

23 Hans Kleine Büning, Marek Karpinski, and Andreas Flögel. Resolution for quantified Boolean formulas. *Inf. Comput.*, 117(1):12–18, 1995. `doi:10.1006/inco.1995.1025`.

24 Jan Krajíček and Pavel Pudlák. Quantified propositional calculi and fragments of bounded arithmetic. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 36:29–46, 1990. `doi:10.1002/malq.19900360106`.

25 Tomás Peitl, Friedrich Slivovsky, and Stefan Szeider. Long-distance Q-Resolution with dependency schemes. *J. Autom. Reason.*, 63(1):127–155, 2019. `doi:10.1007/s10817-018-9467-3`.

26 Markus N. Rabe. A resolution-style proof system for DQBF. In Serge Gaspers and Toby Walsh, editors, *Theory and Applications of Satisfiability Testing – SAT 2017*, pages 314–325, Cham, 2017. Springer International Publishing. `doi:10.1007/978-3-319-66263-3_20`.

27 Franz-Xaver Reichl and Friedrich Slivovsky. Pedant: A certifying DQBF solver. In Kuldeep S. Meel and Ofer Strichman, editors, *25th International Conference on Theory and Applications of Satisfiability Testing (SAT 2022)*, volume 236 of *LIPIcs*, pages 20:1–20:10. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. `doi:10.4230/LIPIcs.SAT.2022.20`.

28 Friedrich Slivovsky and Stefan Szeider. Variable dependencies and Q-resolution. In Carsten Sinz and Uwe Egly, editors, *Theory and Applications of Satisfiability Testing - SAT 2014 - 17th International Conference, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 14-17, 2014. Proceedings*, volume 8561 of *Lecture Notes in Computer Science*, pages 269–284. Springer, 2014. `doi:10.1007/978-3-319-09284-3_21`.

29 Grigorii Samuilovich Tseitin. On the complexity of proof in prepositional calculus. *Zapiski Nauchnykh Seminarov POMI*, 8:234–259, 1968.

30   Allen Van Gelder. Variable independence and resolution paths for quantified Boolean formulas. In *Principles and Practice of Constraint Programming - CP 2011 - 17th International Conference, CP 2011, Perugia, Italy, September 12-16, 2011. Proceedings*, volume 6876, pages 789–803. Springer, 2011. `doi:10.1007/978-3-642-23786-7_59`.

31   Nathan Wetzler, Marijn Heule, and Warren A. Hunt Jr. DRAT-trim: Efficient checking and trimming using expressive clausal proofs. In Carsten Sinz and Uwe Egly, editors, *Theory and Applications of Satisfiability Testing - SAT 2014 - 17th International Conference, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 14-17, 2014. Proceedings*, volume 8561 of *Lecture Notes in Computer Science*, pages 422–429. Springer, 2014. `doi:10.1007/978-3-319-09284-3_31`.