Electronic Colloquium on Computational Complexity, Report No. 76 (2019)

The Equivalences of Refutational QRAT

Leroy Chew and Judith Clymo

School of Computing, University of Leeds, UK

Abstract. The solving of Quantified Boolean Formulas (QBF) has been advanced considerably in the last two decades. In response to this, several proof systems have been put forward to universally verify QBF solvers. QRAT by Heule et al. is one such example of this and builds on technology from DRAT, a checking format used in propositional logic. Recent advances have shown conditional optimality results for QBF systems that use extension variables. Since QRAT can simulate Extended Q-Resolution, we know it is strong, but we do not know if QRAT has the strategy extraction property as Extended Q-Resolution does. In this paper, we partially answer this question by showing that QRAT with a restricted reduction rule has strategy extraction (and consequentially is equivalent to Extended Q-Resolution modulo NP). We also extend equivalence to another system, as we show an augmented version of QRAT known as QRAT+, developed by Lonsing and Egly, is in fact equivalent to the basic QRAT. We achieve this by constructing a line-wise simulation of QRAT+ using only steps valid in QRAT.

Keywords: QBF · QRAT · Proof Complexity · Herbrand functions · Certificate

1 Introduction

Quantified Boolean Formulas (QBFs) extend propositional logic with Boolean quantifiers. The languages of true and, symmetrically, of false QBFs are PSPACE-complete, meaning they can capture any problem within the class PSPACE, and that they may allow for more succinct problem representations than propositional logic.

Modern algorithms for solving propositional satisfiability (SAT) problems and deciding the truth of QBFs are able to solve large industrial problems, and may also provide verification by outputting a proof. It is desirable that this "certificate" should be in a standard format, which can be achieved by designing a proof system that is powerful enough to simulate all of the proof systems used in various solving algorithms. A proof can then be easily converted from a solver output into the universal format.

In propositional logic, there is a successful proof checking format known as DRAT [15] that has since been used as a basis for a QBF proof checking format known as QRAT [7]. A further extension of QRAT, QRAT+ [13] has also recently been developed.

ISSN 1433-8092

These proof systems are designed to capture practical QBF solving and involve some complex sub-procedures between the lines. In contrast, proof systems developed for theory tend to be based on simple ideas. Some theoretical results about QRAT and QRAT+ are known: firstly, they were shown to simulate known QBF techniques such as pre-processing [7] and long distance resolution [9] in QBFs; secondly, Skolem functions that certify true QBFs are known to be extractable from QRAT [6] and QRAT+ [13]. However Herbrand functions that certify false QBFs have not been shown to have efficient extraction.

One possible way to get a better handle on these systems from a theoretical point of view is to show that they are equivalent to some simpler system. For propositional logic, DRAT was recently shown to be equivalent to Extended Resolution [10] so there is some hope that QRAT is equivalent to some QBF analogue of Extended Resolution.

In Section 4 we demonstrate such an equivalence partially. If we simplify one of the rules in QRAT in a natural way we get that Herbrand functions can be extracted efficiently. Using a relaxed framework of simulation by Beyersdorff et al. [2] we can say that Extended Q-Resolution (a QBF analogue of extended resolution [8]) simulates, and thus is equivalent in this model to, this restricted version of QRAT.

We also prove an unconditional equivalence between QRAT and QRAT+ for false QBFs which we show in Section 5.

2 Preliminaries

2.1 Proof Complexity.

Formally, a *proof system* [4] for a language $\mathcal{L}$ over alphabet Γ is a polynomial-time computable partial function $f : \Gamma^\star \rightharpoondown \Gamma^\star$ with $rng(f) = \mathcal{L}$, where rng denotes the range. A proof system maps *proofs* to *theorems*. A *refutation* system is a proof system where the language $\mathcal{L}$ is of contradictions.

The partial function f actually gives a proof checking function. Soundness is given by $rng(f) \subseteq \mathcal{L}$ and completeness is given by $rng(f) \supseteq \mathcal{L}$. The polynomial-time computability is an indication of feasibility, relying on the complexity notion that an algorithm that runs in polynomial-time is considered feasible.

Proof size is given by the number of characters appearing in a proof. A proof system f is *polynomially bounded* if there is a polynomial p such that every theorem of size n has a proof in f of size at most $p(n)$.

Proof systems are compared by simulations. We say that a proof system f *simulates* a proof system g ($g \leq f$) if there exists a polynomial p such that for every g-proof π_g there is an f-proof π_f with $f(\pi_f) = g(\pi_g)$ and $|\pi_f| \leq p(|\pi_g|)$. If in addition π_f can be constructed from π_g in polynomial-time, then we say that f *p-simulates* g ($g \leq_p f$). Two proof systems f and g are *(p-)equivalent* ($g \equiv_{(p)} f$) if they mutually (p-)simulate each other.

In propositional logic a literal is a variable (x) or its negation ($\neg x$), a clause is a disjunction of literals and a formula in conjunctive normal form (CNF) is a conjunction of clauses. For a literal l, if $l = x$ then $\bar{l} = \neg x$, and if $l = \neg x$ then $\bar{l} = x$. An assignment τ for formula A over n variables is a partial function from the variables of A to $\{0,1\}^n$. For clause C, $\tau(C)$ is the result of evaluating C under assignment τ. For formula (or circuit) A, we define $A[b/x]$ so that all instances of variable x in A are replaced with $b \in \{0,1\}$.

Several kinds of inferences can be made on formulas in conjunctive normal form. *Unit propagation* simplifies a formula Φ in conjunctive normal form by building a partial assignment and applying it to Φ. It builds the assignment by satisfying any literal that appears in a singleton (unit) clause. Doing so may negate opposite literals in other clauses and result in them effectively being removed from that clause. In this way, unit propagation can create more unit clauses and can keep on propagating until no more unit clauses are left.

Algorithm 1 Formal definition of unit propagation on CNF ϕ.

Unit(ϕ)
for each clause $C \in \phi$ **do**
  if literal 1 occurs in C **then**
    delete C from ϕ
  end if
  delete every literal 0 in C
end for
if there is an empty clause in ϕ **then**
  return $\perp$
else if there is a clause $C \in \phi$ with exactly one literal c **then**
  return Unit($\phi[1/c]$)
else
  return ϕ
end if

The advantage of unit propagation is that it reaches fix-point in time that is polynomial in the number of clauses, however unit propagation is not a complete procedure and so is not a refutational proof system for propositional logic.

An example of a proof system, and also in fact a refutation system, is *Resolution* (Res). It works on formulas in CNF. Two clauses $(C \vee x)$ and $(D \vee \neg x)$ can be merged removing the pivot variable x to get $(C \vee D)$. An enhanced version of Resolution known as *Extended Resolution* allows new variables known as extension variables to be introduced. Extension variables represent functions of existing variables, these variables are introduced alongside clauses, known as extension clauses, that define this function. In Figure 1 we define these systems in the more general logic of QBF (to understand Figure 1 in the context of SAT treat all quantifiers as existential).

2.2 Quantified Boolean Formulas

Quantified Boolean Formulas (QBF) extend propositional logic with quantifiers $\forall, \exists$ that work on propositional atoms [11]. The standard QBF semantics is that $\forall x \Psi$ is satisfied by the same truth assignments as $\Psi[0/x] \wedge \Psi[1/x]$ and $\exists x \Psi$ is satisfied by the same truth assignments as $\Psi[0/x] \vee \Psi[1/x]$.

A prenex QBF is a QBF where all quantification is done outside of the propositional connectives. A prenex QBF Ψ therefore consists of a propositional part Φ called the matrix and a prefix of quantifiers Π and can be written as $\Psi = \Pi\Phi$. Starting from left to right we give each bound variable x a numerical level, denoted $\mathrm{lv}(x)$, starting from 1 and increasing by one each time the quantifier changes (it stays the same whenever the quantifier is not changed). For literal l, $\mathrm{var}(l) = x$ if $l = x$ or $l = \neg x$ and $\mathrm{lv}(l) = \mathrm{lv}(x)$. We write $\mathrm{lv}(x) <_\Pi \mathrm{lv}(y)$ to indicate that x appears in an earlier quantifier level than y in Π (though the subscript can be omitted if the context is clear). When the propositional matrix of a prenex QBF is in conjunctive normal form then we have a PCNF. A QBF of arbitrary structure can be transformed in polynomial time in to a PCNF. If no variables appear in the QBF without quantification then the QBF is closed.

It is natural to understand a PCNF as a set of clauses, and a clause as a set of literals. As such we will use notation $C \in \Phi$ to indicate that QBF $\Psi = \Pi\Phi$ has the clause C in its matrix. Similarly $l \in C$ indicates that clause C contains literal l.

Set notation is also used to define sub-clauses and sub-formulas, for example we can define the sub-clause of C containing only literals bound at level i or earlier by $\{l \in C \mid \mathrm{lv}(l) \leq i\}$.

A closed prenex QBF may be thought of as a game between two players. One player is responsible for assigning values to the existentially quantified variables, and the other responsible for the universally quantified variables. The existential player wins the game if the formula evaluates to true once all assignments have been made, the universal player wins if the formula evaluates to false. The players take turns to make assignments according to the quantifier prefix, so the levels of the prefix correspond to turns in the game. A strategy for the universal player on QBF $\Pi\Phi$ is a method for choosing assignments for each universal u that depends only on variables earlier than u in Π. If this strategy ensures the universal player always wins games on $\Pi\Phi$ (however the existential player makes assignments), then we say the universal player has a winning strategy. A QBF is false if and only if the universal player has a winning strategy. Strategies for the existential player are defined analogously and the QBF is true if and only if the existential player has a winning strategy.

For a universal variable u in QBF $\Pi\Phi$, a function σ_u that acts on assignments to the existential variables prior to u in Π and has Boolean output, is called a Herbrand function. The collection of σ_u for all universal variables u is a strategy for $\Pi\Phi$ and is denoted σ. Given $\tau_\exists$, an assignment to the existentially quantified variables prior to u, and Herbrand function σ_u, we can extend this assignment to u by evaluating $\sigma_u(\tau_\exists)$. We say this is an extension of the partial assignment $\tau_\exists$ that is consistent with σ_u.

A proof system is said to admit strategy extraction if and only if it is possible to efficiently (i.e. in polynomial time in the size of the proof) construct a circuit representing a winning strategy for the universal player from a refutation of a QBF.

An example of a QBF proof system which admits strategy extraction is the refutation system Q-Resolution ($\mathsf{Q\text{-}Res}$) by Kleine-Büning et al. [12]. It combines the Resolution rule with a QBF rule known as universal reduction ($\forall\mathsf{red}$). Universal reduction allows you to locally set the value of a universal literal within a clause, but only under the condition that no other literal in that clause appears to the right of it in the prefix. In a clause, only one choice of value does not satisfy the clause. By assumption universal reduction set the value so that it does not satisfy the clause.

Like Resolution, Q-Resolution can be augmented with extension variables to get Extended Q-Resolution. In Figure 1 we detail each proof system based on which rules we have. Note that for propositional proof systems Resolution and Extended Resolution that we assume the prefix is purely existential as this is equivalent to the propositional setting.

$$\frac{}{C}\ (\mathrm{Ax}) \qquad\qquad \frac{}{(\neg x \vee \neg y \vee \neg v), (x \vee v), (y \vee v)}\ (\mathrm{Ext})$$

Ax: C is a clause in the propositional matrix.
Ext: x, y are variables already in the formula, v is a fresh variable, v is inserted into prefix as existentially quantified, after x and y in the prefix.

$$\frac{C \vee x \qquad D \vee \neg x}{C \vee D}\ (\mathrm{Res}) \qquad\qquad \frac{C \vee l}{C}\ (\forall\mathsf{red})$$

Res: variable x is existential.
$\forall$*red*: literal l has variable u, which is universal, and all other existential variables $x \in C$ are left of u in the quantifier prefix. Literal $\bar{l}$ does not appear in C.

Fig. 1. Rules of our resolution systems in the language of QBF, Resolution is given by (Ax)+(Res), Extended Resolution is given by (Ax)+(Ext)+(Res), Q-Resolution is given by (Ax)+(Res)+($\forall\mathsf{red}$) and Extended Q-Resolution is given by (Ax)+(Ext)+(Res)+($\forall\mathsf{red}$) [12].

This propositional case is important for QBFs. Any logical propositional implication is also valid in QBF (provided it is model preserving) and we can use this to make important steps in QBF inference. For a QBF $\Pi\Phi$, a full abstraction $\mathsf{Abs}(\Pi)$ returns an identical prefix with all variables of Π, except that every variable is quantified existentially. We will use a more general version of abstraction in Section 5.

Definition 1 (NP Oracle derivations [2]). *For QBF refutational system g, a g^{NP} proof of a QBF Ψ is a derivation of the empty clause by any of the g rules or an* NP*-derivation rule:*

$$\frac{C_1, \dots, C_l}{D} \text{ (}\mathsf{NP}\text{-}\textit{derivation)}$$

For any l, where $\mathsf{Abs}(\Pi) \bigwedge_{i=1}^{l} C_i \models \mathsf{Abs}(\Pi)(\bigwedge_{i=1}^{l} C_i) \wedge D$. $C_1, \dots, C_l$ and D must be permitted in the system g.

An NP-derivation rule can infer ΠD from $\Pi \bigwedge_{i=1}^{l} C_i$ whenever $\bigwedge_{i=1}^{l} C_i \models D$. When we add D we do not change the prefix Π. Hence g^{NP} augments QBF proof system g with all propositional inferences.

Notice that g^{NP} is not a proof system unless we can check the NP-derivation in polynomial-time. This cannot be done unless $\mathsf{P} = \mathsf{NP}$.

Definition 2. *Let P, Q be QBF proof systems, then we write $f \equiv^{\mathsf{NP}} g$ whenever f^{NP} and g^{NP} mutually p-simulate each other.*

In [3] Extended Q-Res with an NP derivation rule (Extended $\mathsf{Q\text{-}Res}^{\mathsf{NP}}$) was investigated. In the NP-derivation rule the input arguments $C_1, \dots, C_l$ and the output argument D are all clauses as Extended Q-Res works on clauses. D is added to the CNF. $\mathrm{QRAT}^{\mathsf{NP}}$ is defined in the same way, adding clauses logically implied by existing clauses.

We can combine our understanding of NP-oracle derivations and strategy extraction to show a weak optimality result for Extended Q-Res.

Theorem 1 (Chew 18 [3]). *If a refutational QBF proof system has strategy extraction it can be simulated by Extended $\mathsf{Q\text{-}Res}^{\mathsf{NP}}$.*

As we can see, Extended Q-Res is very strong and so it simulates most of the known QBF proof systems. It is therefore important to understand its relation to other strong systems such as the systems QRAT and QRAT+ which we will study in this paper.

3 QRAT

QRAT was introduced as a universal proof checking format for QBF. It simulates many QBF preprocessing techniques and proof systems. QRAT is based on the propositional DRAT format which is an advancement of blocked clause addition. Blocked literals generalise extension variables and so DRAT simulates extended resolution, likewise QRAT p-simulates Extended Q-Res.

QRAT works by having a PCNF QBF $\Pi\Phi$ that is edited throughout the proof by a number of satisfiability preserving rules. In contrast to line-based systems like Extended Resolution, QRAT does not just accumulate lines based on other lines. New conjuncts can be added and clauses can be altered or even deleted and rules are usually based globally around the current status of $\Pi\Phi$.

In [7] six rules were listed for QRAT. These were named ATA, ATE, QRATA, QRATE, QRATU and EUR. However QRAT has two modes that make it a proof system: refutation and satisfaction. In this work we focus only on refutation, which uses only ATA, QRATA, QRATU, EUR, and a general deletion rule. We will state each of these rules. Their correctness is proved in [7], but the strategy extraction arguments we make in Section 4 double as arguments for correctness.

If C is a clause, then $\bar{C}$ is the conjunction of the negation of the literals in C. We denote that the clause D can be derived by unit propagation applied to $\Pi\Phi$ by $\Pi\Phi \vdash_1 D$. Unit propagation is used because it is a polynomial-time procedure.

Definition 3 (Asymmetric Tautology Addition (ATA)). *Let $\Pi\Phi$ be a closed PCNF with prefix Π and CNF matrix Φ. Let C be a clause not in Φ. Let Π' be a prefix including the variables of C and Φ, Π is a sub-prefix of Π' containing the variables of Φ only.*

Suppose $\Pi'\Phi \wedge \bar{C} \vdash_1 \bot$. Then we can make the following inference

$$\frac{\Pi\Phi}{\Pi'\Phi \wedge C}\ (ATA)$$

Notice that in the way that we define QRAT, $\Pi'\Phi \wedge C$ replaces $\Pi\Phi$.

Definition 4 (Outer Clause, Outer Resolvent). *Let $\Pi\Phi$ be a PCNF with closed prefix Π and CNF matrix Φ. Let C be a clause not in Φ. Let Π' be a prefix including the variables of C and Φ, Π is a sub-prefix of Π' containing the variables of Φ only.*

Suppose C contains a literal l. Consider all clauses D in Φ with $\bar{l} \in D$. The outer clause O_D *of D is $\{k \in D \mid \mathrm{lv}(k) \leq_\Pi \mathrm{lv}(l), k \neq \bar{l}\}$. The* outer resolvent *$\mathcal{R}(C, D, \Pi, l)$ is defined as $C \vee O_D$.*[1]

Definition 5 (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' be a prefix including the variables of C and Φ, Π is a sub-prefix of Π' containing the variables of Φ only.*

If C contains an existential literal l such that for every $D \in \Phi$ with $\bar{l} \in D$, $\Pi\Phi \wedge \bar{C} \wedge \bar{O}_D \vdash_1 \bot$ (or equivalently $\Pi\Phi \wedge \bar{\mathcal{R}}(C, D, \Pi, l) \vdash_1 \bot$) then we can derive

$$\frac{\Pi\Phi}{\Pi'\Phi \wedge C}\ (QRATA\ w.r.t.\ l)$$

Definition 6 (Quantified Resolution Asymmetric Tautology Universal (QRATU)). *Let $\Pi\Phi$ be a PCNF with closed prefix Π and CNF matrix Φ. Let $C \vee l$ be a clause with universal literal l.*

If for every $D \in \Phi$ with $\bar{l} \in D$, $\Pi\Phi \wedge \bar{C} \wedge \bar{O}_D \vdash_1 \bot$ then we can derive

$$\frac{\Pi\Phi \wedge (C \vee l)}{\Pi\Phi \wedge C}\ (QRATU\ w.r.t.\ l)$$

[1] Other authors have made two separate definitions of outer resolvents based on existential or universal l. In order to simplify, we only use the existential definition here and factor in the necessary changes in the description of our rules.

Definition 7 (Extended Universal Reduction (EUR)). *Given a clause $C \vee u$ with universal literal u, consider extending C by*

$C := C \cup \{k \in D \mid \mathrm{lv}(k) >_{\Pi} \mathrm{lv}(u) \text{ or } k = \bar{u}\}$,

where $D \in \Phi$ is any clause with some $p : \mathrm{lv}(p) >_{\Pi} \mathrm{lv}(u)$, $p \in C$ and $\bar{p} \in D$,

until we reach a fixed point denoted ε. If $\bar{u} \notin \varepsilon$ then we can perform the following rule.

$$\frac{\Pi\Phi \wedge (C \vee u)}{\Pi'\Phi \wedge C} \; (EUR)$$

EUR encompasses the important reduction rule used in Q-Resolution. Along with the other rules, this allows us to simulate Extended Q-Res and thus we have refutational completeness. EUR is strictly stronger than the standard universal reduction (UR), because it uses a dependency scheme from [14]. While refutational QRAT works perfectly fine with a standard universal reduction rule, extended universal reduction was used because it allows QRAT to easily simulate expansion steps used in QBF preprocessing. Note that this was the *only* preprocessing rule that required the extended universal reduction rule. In theory QRAT could be augmented with any sound dependency scheme and that is used as the basis of its reduction rule.

Definition 8 (Clause Deletion). *In refutational QRAT clauses may be arbitrarily deleted without checking if they conform to a rule, note this is not the case in satisfaction QRAT.*

$$\frac{\Pi\Phi \wedge C}{\Pi\Phi}$$

It is important to note that because some of the rules work by looking at every clause contained in Φ, clause deletion may not be superficial. A clause may need to be deleted in order for EUR to be performed. The situation is true also for QRATA and QRATU with respect to l where our property needs to check all other clauses that have the complimentary literal $\bar{l}$.

4 Strategy Extraction and Simulations

In [6] it was shown that satisfaction QRAT has strategy extraction. The equivalent, however, was not proved for refutational QRAT. The problem is the asymmetry of the two systems as EUR is not needed in satisfaction QRAT. EUR causes particular problems which we were not able to get around. It reduces a universal variable u in a clause C even when there are variables in C to the right of u in the prefix. Now this is also true for QRATU however in that case we have the QRAT framework to work with, which is not similar to the dependency framework of EUR.

We have avoided this issue by removing EUR altogether. EUR is not essential to the underlying techniques of QRAT and QRAT can still simulate the main preprocessing techniques except $\forall$-Expansion. In addition EUR is not required to simulate Extended Q-Resolution as this can be done with a simpler reduction rule, as extension clauses can be added with QRATA wrt the extension variable, Resolution uses ATA and reduction can be performed with simple universal reduction.

Let QRAT(X) be QRAT with the EUR rule replaced with reduction rule X. This means that the standard QRAT is given by QRAT(EUR). An alternative would be to use the ∀-reduction rule from Q-Res, which allows

$$\frac{\Pi\Phi \wedge (C \vee u)}{\Pi\Phi \wedge C}$$

whenever $\mathrm{lv}(u)$ is greater than $\mathrm{lv}(x)$ for all existentially quantified x in C.

We call this simplest version QRAT(UR).

In order to show that QRAT(UR) has polynomial-time strategy extraction on false QBFs we will inductively compute a winning universal strategy for formulas at each step of the QRAT proof. For the inductive step we need to construct a winning strategy σ for the formula prior to some proof step, from a known winning strategy σ' for the formula after that proof step. We prove that this is possible for each derivation rule in refutational QRAT. The strategy σ is composed of Herbrand functions σ_u for each universal variable u.

Lemma 1. *If $\Pi'\Phi \wedge C$ is derived from $\Pi\Phi$ by ATA, and σ' is a winning universal strategy for $\Pi'\Phi \wedge C$, then we can can construct a winning universal strategy for $\Pi\Phi$.*

Proof. If clause C is added by ATA then $\Phi \wedge \bar{C} \vdash_1 \bot$ and we derive:

$$\frac{\Pi\Phi}{\Pi'\Phi \wedge C}\ (\text{ATA})$$

Because $\Phi \wedge \bar{C} \vdash_1 \bot$, any assignment that falsifies C also falsifies Φ. Therefore if σ' is a strategy that falsifies $\Pi'\Phi \wedge C$, σ' must falsify Φ. Let $\sigma = \sigma'$ except that if $\Pi \neq \Pi'$ then any existential input variable that does not appear in Φ can be restricted in σ to 0 or 1 arbitrarily (there is a winning strategy for either assignment, so we just pick one). If a universal variable is in Π' but not Π we do not need a strategy for it as it will have no effect on the outcome of the game for $\Pi\Phi$. □

Lemma 2. *If $\Pi'\Phi \wedge C$ is derived from $\Pi\Phi$ by QRATA, and σ' is a winning universal strategy for $\Pi'\Phi \wedge C$, then we can can construct a winning universal strategy for $\Pi\Phi$.*

Proof. C contains some existential literal l such that for every $D \in \Phi$ with $\bar{l} \in D$ and outer clause O_D, $\Pi'\Phi \wedge \bar{C} \wedge \bar{O}_D \vdash_1 \bot$. For notational convenience, let $A = \{k \in C \mid k \neq l, \mathrm{lv}(k) \leq_\Pi \mathrm{lv}(l)\}$, and $B = \{k \in C \mid \mathrm{lv}(k) >_\Pi \mathrm{lv}(l)\}$ so that $C = (A \vee l \vee B)$ and $\Pi'\Phi \wedge \bar{A} \wedge \bar{l} \wedge \bar{B} \wedge \bar{O}_D \vdash_1 \bot$. We derive:

$$\frac{\Pi\Phi}{\Pi'\Phi \wedge (A \vee l \vee B)}\ (\text{QRATA wrt. } l)$$

Initially we assume $\Pi = \Pi'$. Let u be a universal variable in Π'. If $\mathrm{lv}(u) <_{\Pi'} \mathrm{lv}(l)$ then $\sigma'_u = \sigma_u$. If $\mathrm{lv}(u) >_{\Pi'} \mathrm{lv}(l)$, we proceed by a case distinction. Let $\tau_\exists$ be an assignment to the existential variables x with $\mathrm{lv}(x) <_{\Pi'} \mathrm{lv}(u)$, and τ its extension consistent with the Herbrand functions of universal variables y with $\mathrm{lv}(y) <_{\Pi'} \mathrm{lv}(u)$.

$$\sigma_u(\tau_\exists) = \begin{cases} \sigma'_u(\tau'_\exists) & \tau(A \vee l) = \bot\text{, but for every clause } D \text{ with } \bar{l} \in D \\ & \text{if } O \text{ is the outer clause of } D \text{ then } \tau(O) = \top \\ & \text{where } \tau'_\exists \text{ differs from } \tau_\exists \text{ only on variable } l \\ & \text{such that } l \text{ is satisfied in } \tau'_\exists, \\ \sigma'_u(\tau_\exists) & \text{otherwise.} \end{cases}$$

We have to show that σ actually falsifies $\Pi\Phi$. Assume we reach an assignment τ by playing according to σ.

Suppose τ satisfies A then $\tau(A \vee l) = \top$ so for all u, $\sigma_u(\tau_\exists) = \sigma'_u(\tau_\exists)$. τ is consistent with σ' so falsifies $\Pi'\Phi \wedge (A \vee l \vee B)$. It cannot falsify $(A \vee l \vee B)$ so τ falsifies $\Pi\Phi$.

Suppose τ falsifies A but satisfies the outer clauses of every D with $\bar{l} \in D$. If $\tau_\exists$ were modified so l is true then σ' yields τ' which satisfies $(A \vee l \vee B)$ so must falsify Φ. Changing l to be false in τ' cannot satisfy any additional clauses since the outer clauses of all D with $\bar{l} \in D$ are already satisfied (by construction τ and τ' are identical prior to l). Under σ all universal variables right of l are played according to σ' as if l were made true, this will falsify some clause in Φ regardless of how l is actually set.

If τ falsifies A but also falsifies the outer clause O of some clause D with $\bar{l} \in D$ then we know that the responses from σ here are defined to be consistent with the original σ'. This means that either τ falsifies Φ or τ falsifies $A \vee l \vee B$. We know that $\Pi'\Phi \wedge \bar{A} \wedge \bar{l} \wedge \bar{B} \wedge \bar{O}_D \vdash_1 \bot$. If τ falsifies $A \vee l \vee B$ then also $\tau(O_D \vee A \vee l \vee B) = \bot$, and so Φ is also falsified by τ.

If in fact $\Pi \neq \Pi'$ then Π' contains more variables than Π. First construct the strategies as above, assuming prefix Π', then fix these as in Lemma 1 to not include the variables missing from Π. Universal variables not in Π simply have their strategies removed from σ. Existential variables not in Π are restricted in σ to 0 or 1 arbitrarily. □

Example 1. Consider clause $(a \vee l \vee x \vee \neg y)$ and QBF $\exists ablx \forall y \phi$ where
$\phi = (a \vee b \vee \neg y) \wedge (\neg b \vee y) \wedge (b \vee l \vee y) \wedge (b \vee \neg l \vee y)$.

The only clause that contains $\neg l$ in ϕ is $(b \vee \neg l \vee y)$, so the only outer clause we need to consider is (b). $\exists ablx \forall y \phi \wedge \neg b \wedge \neg a \wedge \neg l \wedge \neg x \wedge y \vdash_1 \bot$ so QRATA is possible with respect to l.

A winning strategy for the universal player after the new clause is added is to play y to 1 if and only if a, l and x are all 0. In our strategy extraction we derive the strategy prior to when QRATA is used.

If any of a, l, x are 1, we can continue to set y to 0 and falsify some clause in ϕ not containing $\neg y$.

If a, b, l, x are all 0, we falsify the only outer clause (b) thus we know via unit propagation some other clause (here $(a \vee b \vee \neg y)$) will be falsified if we continue to falsify $(a \vee l \vee x \vee \neg y)$ only. We keep playing the old strategy for this reason.

If a, l, x are all 0 and b is 1, setting y to 1 no longer works as it only falsifies the added clause, we instead see what happens when l is flipped to 1, and set y to 0 according to the strategy, falsifying clause $(\neg b \vee y)$.

Lemma 3. *If $\Pi'\Phi \wedge C$ is derived from $\Pi\Phi \wedge (C \vee l)$ by QRATU, and σ' is a winning universal strategy for $\Pi'\Phi \wedge C$, then we can construct a winning universal strategy for $\Pi\Phi \wedge (C \vee l)$.*

Proof. The rule QRATU reduces universal l from $C \vee l$. As before, it is useful to have notation for subclauses of C having variables to the left or right of l in the prefix. Let $A = \{k \in C \mid k \neq l, \mathrm{lv}(k) \leq_\Pi \mathrm{lv}(l)\}$, and $B = \{k \in C \mid \mathrm{lv}(k) >_\Pi \mathrm{lv}(l)\}$ so that $(C \vee l) = (A \vee l \vee B)$.

For every clause $D \in \Phi$ with $\bar{l} \in D$ and outer clause O_D, $\Pi'\Phi \wedge \bar{A} \wedge \bar{B} \wedge \bar{O}_D \vdash_1 \bot$.

$$\frac{\Pi\Phi \wedge (A \vee l \vee B)}{\Pi'\Phi \wedge (A \vee B)} \text{ (QRATU wrt. } l)$$

Let u be a universal variable in Π. If $\mathrm{var}(l) \neq u$ then $\sigma_u = \sigma'_u$. If $u = \mathrm{var}(l)$ does not appear in Π' then $\sigma_u = c$ where $c \in \{0, 1\}$ falsifies l.

Otherwise, if $\mathrm{var}(l) = u$, we define σ_u by

$$\sigma_u(\tau_\exists) = \begin{cases} c & \tau(A) = \bot, \text{ but for every clause } D \text{ with } \bar{l} \in D \\ & \text{if } O \text{ is the outer clause of } D \text{ then } \tau(O) = \top \\ & \text{where } c \in \{0,1\} \text{ is such that setting } u \text{ to } c \text{ falsifies } l, \\ \sigma'_u(\tau_\exists) & \text{otherwise.} \end{cases}$$

We have to show that σ actually falsifies $\Pi\Phi \wedge (A \vee l \vee B)$. Let assignment τ be consistent with σ. If τ satisfies A then τ is also consistent with the original σ' so it must falsify $\Phi \wedge (A \vee B)$, but since it satisfies A there is some clause in Φ that τ falsifies.

If τ falsifies A but satisfies the outer clauses of all D with $\bar{l}$ in D then it may be that τ is not consistent with σ'. We observe that there is another assignment τ' consistent with σ' which is identical to τ except possibly on the variable of l where τ sets literal l to 0. Therefore τ' satisfies all the outer clauses and modifying τ' so l is false (i.e. to τ) cannot satisfy any additional clauses in $\Pi'\Phi \wedge (A \vee B)$. τ' falsifies $\Pi'\Phi \wedge (A \vee B)$. If τ' falsifies a clause in Φ then τ falsifies the same clause. If τ' falsifies $(A \vee B)$ then τ falsifies $(A \vee l \vee B)$.

If τ falsifies A and some outer clause O of D with $\bar{l} \in D$ then τ is consistent with σ', so it falsifies either Φ or $(A \vee B)$. If τ falsifies $(A \vee B)$ then τ falsifies $O \vee (A \vee B)$ so it must also falsify Φ since $\Pi'\Phi \wedge \bar{A} \wedge \bar{B} \wedge \bar{O} \vdash_1 \bot$. □

Lemma 4 (Balabanov, Jiang [1]). *If $\Pi'\Phi \wedge C$ is derived from $\Pi\Phi \wedge (C \vee l)$ by UR, and σ' is a winning universal strategy for $\Pi'\Phi \wedge C$, then we can can construct a winning universal strategy for $\Pi\Phi \wedge (C \vee l)$.*

Proof. The reduction rule UR removes l from clause $C \vee l$ where $\mathrm{lv}(k) \leq_{\Pi} \mathrm{lv}(l)$ for every literal $k \in C$.

$$\frac{\Pi\Phi \wedge (C \vee l)}{\Pi'\Phi \wedge C}\ (\forall\text{-red})$$

Let $\sigma_u = \sigma'_u$ when $\mathrm{var}(l) \neq u$. If $u = \mathrm{var}(l)$ does not appear in Π' then $\sigma_u = c$ where $c \in \{0, 1\}$ falsifies l. Otherwise, if $\mathrm{var}(l) = u$ then, given an assignment $\tau_\exists$ to all existential variables left of u, τ its extension consistent with the Herbrand functions for universal variables y with $\mathrm{lv}(y) \leq_{\Pi} \mathrm{lv}(u)$.

$$\sigma_u(\tau_\exists) = \begin{cases} c & \tau(C) = \bot \\ & \text{where } c \in \{0,1\} \text{ is such that setting } u \text{ to } c \text{ falsifies } l, \\ \sigma'_u(\tau_\exists) & \text{otherwise} \end{cases}$$

This deviates from σ' exactly when C is falsified and always falsifies $C \vee l$ in that case. So $\Phi \wedge (C \vee l)$ is always falsified under σ. ☐

Lemma 5. *If $\Pi'\Phi$ is derived from $\Pi\Phi \wedge C$ by clause deletion, and σ' is a winning universal strategy for $\Pi'\Phi$, then we can can construct a winning universal strategy for $\Pi\Phi \wedge C$.*

Proof. Clause deletion allows to derive

$$\frac{\Pi\Phi \wedge C}{\Pi'\Phi}$$

We simply let $\sigma'_u = \sigma_u$. Since σ' always falsifies a clause from Φ then also σ' will falsify a clause from Φ. ☐

Theorem 2. *QRAT(UR) has polynomial-time strategy extraction on false QBFs.*

Proof. We inductively show that we can compute a winning strategy σ on the current formula $\Pi\Phi$ during our steps of QRAT. σ can be constructed as a circuit with polynomial size in the length of the QRAT proof of $\Pi\Phi$ and consists of Herbrand functions σ_u for each of the universal variables u in the formula, based on existential variables of lower level.

We work backwards in the proof. Initially (i.e. at the end of the proof) we have the empty clause so in the base case our universal strategy is to set all u to 0.

For the inductive steps we construct a new strategy σ for $\Pi\Phi$ based on σ' for $\Pi'\Phi'$, which is possible by the Lemmas above. The circuits σ_u constructed for ATA and clause deletion steps are no larger than σ'_u. For UR we have one copy of σ'_u and a circuit to check whether C is satisfied. For QRATU we need a circuit to determine if A and each of the outer clauses are satisfied, and this result with the output of σ'_u determines the final value for u. QRATA is the least obvious case when transforming into a polynomial size circuit. A new circuit is added to decide whether $(A \vee l)$ and the outer clauses are satisfied. The output of this is used to possibly change the input to σ'_u, which can be achieved with a small sub-circuit. Crucially, only one copy of σ'_u is needed.

Eventually we get to our first line and thus provide a strategy for the initial formula which can be constructed as a polynomial circuit, giving us strategy extraction. □

5 Equivalence with QRAT+

In [13] Lonsing and Egly took the QRAT framework and improved on it by relaxing the properties required to add or delete a clause. In QRAT (and the original DRAT) we used the fact that $\Phi \wedge \bar{C}$ is a contradiction that can be checked by unit propagation in order to add C, this is known as Reverse Unit Propagation (RUP) [5]. This works nicely because unit propagation is a polynomial-time procedure and it is done in practice between the main inference steps in a propositional solver. In fact in Conflict-Driven Clause Learning (CDCL) solving it is used exactly in this way, to confirm a conflict whose negation is added as a clause.

However, in the QBF CDCL setting universal reduction is also done in between steps. So Lonsing's and Egly's definition of QRAT+ changed $\vdash_1$ to $\vdash_{1\forall}$, where $\vdash_{1\forall}$ is an inference using unit propagation *and* universal reduction.

In order to make this sound for all the QRAT rules, only certain $\forall$-reduction steps are allowed. To achieve this, an *abstraction* of the quantifier prefix Π is taken.

Definition 9. *Let i be the maximum level of all variables in clause C. Then $\mathsf{Abs}(\Pi', C)$ is the quantifier prefix obtained from Π' by setting all quantifiers with level $\leq_{\Pi'} i$ to existential.*

C is a *Quantified* Asymmetric Tautology (QAT) in QBF $\Pi\Phi$ when $\mathsf{Abs}(\Pi', C)\Phi \wedge \bar{C} \vdash_{1\forall} \bot$. We use Π' instead of Π because C could contain variables not in Π.

The asymmetric tautology property used for ATA is replaced by QAT. The property QRAT used for QRATA and QRATU is similarly updated to QRAT+. Recall clause C has QRAT with respect to literal l in QBF $\Pi\Phi$ if and only if for all $D \in \Phi$ with $\bar{l} \in D$

$$\Phi \wedge \bar{C} \wedge \bar{O}_D \vdash_1 \bot$$

where O_D is the outer clause of D with respect to l.

Clause C has QRAT+ with respect to literal l in QBF $\Pi\Phi$ if and only if for all $D \in \Phi$ with $\bar{l} \in D$

$$\mathsf{Abs}(\Pi', C \vee l)\Phi \wedge \bar{C} \wedge \bar{O}_D \vdash_{1\forall} \bot$$

QATA allows C to be added to QBF $\Pi\Phi$ when C is a Quantified Asymmetric Tautology in $\Pi\Phi$. QRATA+ allows addition of C to $\Pi\Phi$ given there is an existential literal $l \in C$ such that C has QRAT+ with respect to l in $\Pi\Phi$. QRATU+ allows removal of l from $C \vee l$ if C has QRAT+ with respect to l in $\Pi\Phi$. The extended universal reduction rule is not changed in QRAT+.

Lonsing and Egly do not differentiate between a refutational and satisfaction QRAT+. Here we focus on the refutational case where arbitrary clauses can be deleted. Refutational QRAT+ therefore consists of QATA, QRATA+, QRATU+, EUR and clause deletion.

Proposition 1. *Refutationally, QRAT is p-equivalent to QRAT+.*

The key here is that in QRAT we derive the some of the intermediate steps used in QRAT+, in particular the clauses that are to be $\forall$-reduced in the $\vdash_{1\forall}$ derivation. From there we can use the universal reduction rule in QRAT to reduce it and continue to use it to get our desired clause using only the rules in QRAT.

Example 2. We use an example from [13] to give a simple illustration of how this method works. Let $\Phi = \forall u_1, u_2 \exists x_1, x_2 \forall u_3 \exists x_3 \bigwedge_{i=0}^{7} C_i$ where

$$\begin{aligned} C_0 &= (\neg u_2 \vee \neg x_1 \vee \neg x_2) & C_3 &= (u_2 \vee x_1 \vee x_2) & C_6 &= (\neg x_1 \vee x_2 \vee \neg x_3) \\ C_1 &= (\neg u_1 \vee \neg x_1 \vee x_2) & C_4 &= (\neg x_1 \vee \neg x_2 \vee x_3) & C_7 &= (\neg u_3 \vee x_3) \\ C_2 &= (u_1 \vee x_1 \vee \neg x_2) & C_5 &= (u_3 \vee \neg x_3) \end{aligned}$$

Both u_1 and u_2 can be removed by QRATU+ but not by QRATU. In Φ, C_0 has QRAT+ with respect to $\neg u_2$. The only clause containing u_2 is C_3. Unit propagation of x_1 and x_2 allows us to derive x_3 (from C_4) then we use this to derive u_3 (from C_5) which is then removed by $\forall$-reduction. Therefore we can replace C_0 by $C_0 \backslash \{\neg u_2\}$. This is not possible by QRATU, however it is possible to derive $C = \neg x_1 \vee \neg x_2 \vee u_3$ by ATA since $\Phi \wedge \bar{C} \vdash_1 \bot$. We have unit propagation on x_1 and x_2 as before, then use x_3 to derive u_3 which does not need to be removed by $\forall$-reduction because we can use unit propagation with $\neg u_3$ instead to derive the empty clause. Once C is derived we can safely remove u_3 by $\forall$-reduction to reach the target clause.

Lemma 6. *The QATA step is p-simulated by steps in refutational QRAT(UR).*

Proof. Let Φ be a CNF and let C be a clause not in Φ. Let Π' be a prefix including the variables of C and Φ, and Π a sub-prefix of Π' containing the variables of Φ.

$$\frac{\Pi \Phi}{\Pi' \Phi \wedge C}$$

is possible whenever C has QAT on $\Pi\Phi$. This means $\mathsf{Abs}(\Pi', C)\Phi \wedge \bar{C} \vdash_{1\forall} \bot$, where i is the maximum level of literals in C. However it may not be the case that $\Pi'\Phi \wedge \bar{C} \vdash_1 \bot$ because we may have used a universal reduction step (potentially multiple times).

The idea is that we break the single QAT step into several steps based on what can be done with unit propagation versus what uses $\forall$-reduction. We deal with unit propagation by using ATA steps and then use UR or EUR to do the reduction steps.

We single out the clauses that we perform reduction on during the QAT procedure and label them L_i with the literal p_i being reduced (in order, so the first one derived is L_1). So $\Pi'\Phi \wedge \bar{C} \vdash_1 L_1$, and for $i > 1$, $\Pi'\Phi \wedge \bigwedge_{j<i} L_j \backslash \{p_j\} \wedge \bar{C} \vdash_1 L_i$. The condition on reduction is that p_j must be the greatest level literal in L_j, but it must also be at a greater level than any literal in C because we are using the modified prefix so that every variable at a lower level than any variable in C is existentially quantified.

Induction Hypothesis: We can learn $C \vee L_i \backslash \{p_i\}$ in a short proof using only ATA and $\forall$-reduction steps.

Base Case: We need to add $C \vee L_1$ via ATA, we know $\Pi'\Phi \wedge \bar{C} \vdash_1 L_1$ so $\Pi'\Phi \wedge \bar{C} \wedge \bar{L}_1 \vdash_1 \bot$

Inductive Step: From assuming our induction hypothesis we have $\Pi'\Phi \wedge \bigwedge_{j<i}(C \vee L_j \backslash \{p_j\})$ now we need to add $C \vee L_i$ via ATA. For each $j < i$, $(C \vee L_j \backslash \{p_j\}) \wedge \bar{C} \vdash_1 L_j \backslash \{p_j\}$, and $\Phi \wedge \bigwedge_{j<i} L_j \backslash \{p_j\} \wedge \bar{C} \vdash_1 L_i$ so we can join these unit propagation inferences together to get that $\Pi\Phi \wedge \bigwedge_{j<i}(C \vee L_j \backslash \{p_j\}) \wedge \bar{C} \wedge \bar{L}_i \vdash_1 \bot$. We learn $C \vee L_i$ but we now need to remove p_i, which we can do via reduction. Note that per the rules of QRAT+, $\mathrm{lv}(p_i) \geq_{\Pi'} \mathrm{lv}(x)$ for any literal $x \in L_i$ and any x in C. Thus we can reduce p_i.

So we learn all the clauses $C \vee L_i \backslash \{p_i\}$. These new clauses allow us to derive C via ATA without using reduction since those steps are now available.

From $\Pi\Phi \wedge \bigwedge_j (C \vee L_j \backslash \{p_j\}) \wedge \bar{C}$ we can derive each $L_j \backslash \{p_j\}$ which gives us every reduced clause we need to derive $\bot$ from $\Phi \wedge \bar{C}$. Hence $\Pi\Phi \wedge \bigwedge_j (C \vee L_j \backslash \{p_j\}) \wedge \bar{C} \vdash_1 \bot$. Thus we can add C via ATA. □

We can take this framework and use it to simulate other rules in the QRAT+ refutation system. We prove this essential lemma about the property QRAT+.

Lemma 7. *If C has QRAT+ in $\Pi\Phi$ with respect to l then for every clause $D \in \Phi$ with $\bar{l} \in D$, the outer resolvent $C \cup \{k \in D \mid \mathrm{lv}(k) \leq_\Pi \mathrm{lv}(l), k \neq \bar{l}\}$ can be added to $\Pi\Phi$ via a sequence of polynomial size iterations of ATA and $\forall$-reduction.*

Proof. Let R_D denote $C \cup \{k \in D \mid \mathrm{lv}(k) \leq_\Pi \mathrm{lv}(l), k \neq \bar{l}\}$. The property of QRAT+ can be stated as $\mathsf{Abs}(\Pi', C)\Phi \wedge \bar{R}_D \vdash_{1\forall} \bot$ for every $D \in \Phi$ with $\bar{l} \in D$.

Let us fix some D and prove we can derive R_D via ATA and $\forall$-reduction steps. Since D is now fixed we drop the subscript for convenience and let $R = R_D$.

As we did to simulate QATA, we label the clauses that we perform reduction on during the $\vdash_{1\forall}$ procedure as L_i with the literal p_i being reduced (in order, so the first one derived is L_1), we perform induction on this index i.

Induction Hypothesis: We can learn $R \vee L_i \backslash \{p_i\}$ in a short proof using only ATA and $\forall$-reduction steps.

Base Case: We need to add $R \vee L_1$ via ATA, we know $\Phi \wedge \bar{R} \vdash_1 L_1$ so $\Pi'\Phi \wedge \bar{R} \wedge \bar{L}_1 \vdash_1 \bot$. Thus we add $R \vee L_1$. We now reduce p_1 which can be done because $\mathrm{lv}(p_i) \geq_{\Pi'} \mathrm{lv}(x)$ for any literal $x \in L_i$ and any x in C, and since R only adds outer clause literals it works for any $x \in R$ as well.

Inductive Step: From assuming our induction hypothesis we have $\Pi'\Phi \wedge \bigwedge_{j<i}(R \vee L_j \backslash \{p_j\})$ now we need to add $R \vee L_i$ via ATA.

For each $j < i$, $(R \vee L_j \backslash \{p_j\}) \wedge \bar{R} \vdash_1 L_j \backslash \{p_j\}$, and $\Phi \wedge \bigwedge_{j<i} L_j \backslash \{p_j\} \wedge \bar{R} \vdash_1 L_i$ so we can join these unit propagation inferences together to get that $\Pi'\Phi \wedge \bigwedge_{j<i}(R \vee L_j \backslash \{p_j\}) \wedge \bar{R} \wedge \bar{L}_i \vdash_1 \bot$. We learn $R \vee L_i$ but we now need to remove p_i, which we can do via reduction. Note that per the rules of QRAT+, $\text{lv}(p_i) \geq_{\Pi'} \text{lv}(x)$ for any literal $x \in L_i$ and any x in C, and since R only adds outer clause literal it works for any $x \in R$ as well. Thus we can reduce p_i.

Putting this proof together we get all the $(R \vee L_j \backslash \{p_j\})$ clauses we need. $\Pi'\Phi \wedge \bigwedge_{j<i}(R \vee L_j \backslash \{p_j\}) \wedge \bar{R} \vdash_1 L_i$ for every i, thus we also get $\Pi'\Phi \wedge \bigwedge_{j}(R \vee L_j \backslash \{p_j\}) \wedge \bar{R} \vdash_1 \bot$ and can add R via ATA.

We can then remove the intermediate clauses $R \vee L_j \backslash \{p_j\}$ if necessary. □

Lemma 8. *The QRATA+ step is p-simulated by steps in refutational QRAT(UR).*

Proof. Let Φ be a CNF and let C be a clause not in Φ. Let Π be a prefix including the variables of Φ and Π' be a prefix including the variables of C and Φ. Π is a sub-prefix of Π'. The QRATA+ step is given as

$$\frac{\Pi\Phi}{\Pi'\Phi \wedge C}$$

is possible whenever C has QRAT+ with respect to l on $\Pi\Phi$.

Let $\Omega = \{O_D \mid O_D \text{ is the outer clause of some clause } D \in \Phi \text{ with } \bar{l} \in D\}$.

The aim here is to add each $C \vee O_D$ via a short proof using ATA and UR rules. This can be done directly via Lemma 7 since $C \vee O_D$ is an outer resolvent and we have QRATA+. Note that we are allowed to continue adding outer resolvents by this method to Φ even after one or more has already been added. The rules ATA and UR that we use in Lemma 7 are not prohibited by the presence of additional clauses.

Now that we have all $C \vee O$ for every $O \in \Omega$, we need to derive C. We can do this via QRATA with respect to l. This is simple, we need to show for each O_D that $\Pi\Phi \wedge \bigwedge_{O\in\Omega}(C \vee O) \wedge \bar{C} \wedge \bar{O}_D \vdash_1 \bot$. We know this is true since we can directly refute the clause $C \vee O_D$ using $\bar{C} \wedge \bar{O}_D$ in each case.

Once we have derived C we can freely delete all clauses from $\bigwedge_{O\in\Omega}(C \vee O)$ if we require to. Note that refutational QRAT does not require you to use QRATE to remove clauses (see Section 6.2 in [7]). □

Lemma 9. *The QRATU+ step is p-simulated in refutational QRAT(UR).*

Proof. Let Φ be a CNF and let $C \vee l$ be a clause not in Φ. Let Π be a prefix including the variables of $C \vee l$ and Φ. The QRATU+ step is given as

$$\frac{\Pi\Phi \wedge (C \vee l)}{\Pi\Phi \wedge C}$$

whenever C has QRAT+ with respect to universal literal l on $\Pi\Phi$.

Let $\Omega = \{O_D \mid O_D$ be the outer clause of some clause $D \in \Phi$ with $\bar{l} \in K\}$.

The aim is to add $O \cup C$ for every $O \in \Omega$ and we can do that directly using Lemma 7 as $O \cup C$ is the outer resolvent $\mathcal{R}(C \vee l, D, \Pi, l)$ for some clause D and we have QRAT+.

Now we have to show $\Pi\Phi \wedge (\bigwedge_{O\in\Omega} C \vee O) \wedge \bar{C} \wedge \bar{O}_D \vdash \bot$. This is straightforward, as in each case a $C \vee O$ clause is refuted directly from all the units. This means we can perform QRATU to replace $\Pi\Phi \wedge \bigwedge(C \vee O) \wedge C \vee l$ with $\Pi\Phi \wedge \bigwedge(C \vee O) \wedge C$. We can then freely delete all clauses from $\bigwedge C \vee O$ if we require to. □

Proposition 1 *Refutationally, QRAT is p-equivalent to QRAT+.*

Proof. A QRAT+ proof is a sequence of QATA, QRATA+, QRATU+, EUR and deletion rules. We claim this can be simulated by a QRAT proof, in other words a sequence of ATA, QRATA, QRATU, EUR and deletion steps. EUR and deletion rules remain the same in both systems. In Lemma 6 we showed that QATA steps are simulated by ATA and UR steps, in Lemma 8 we showed that QRATA+ steps are simulated by ATA, QRATA and UR steps and in Lemma 9 we showed that QRATU+ steps are simulated by ATA, QRATU and UR steps. We can simulate UR steps by EUR so we can do all this in refutational QRAT.

QRAT is simulated by QRAT+ also, so they are equivalent. □

Theorem 3. *Extended* Q-Res*, refutational QRAT(UR) and refutational QRAT(UR)+ are all equivalent QBF proof systems (modulo NP).*

Proof. Refutational QRAT(UR) simulates refutational QRAT+(UR). This can be seen by putting Lemmas 6, 8 and 9 together to simulate the rules QATA, QRATA+ and QRATU+ using ATA, QRATA, QRATU and UR. Clause deletion and UR are present in both systems.

Extended Q-Res is equivalent QRAT(UR) when both systems have the assistance of an NP oracle. Hence all these systems are equivalent modulo NP. □

Note that the "modulo NP" here is important because we do not know whether Extended Resolution can easily prove the reflection principle of QRAT(UR). If it can (as argued in [3]) then they are indeed all equivalent unconditionally. This would seem likely as Extended Resolution was shown to simulate DRAT in [10].

6 Conclusion

We have collapsed QRAT and QRAT+ into the same refutation system. We have also examined the relationships of these systems to Extended Q-Resolution under the framework of an NP oracle and restricting the reduction rule in QRAT.

The NP oracle probably does not make a difference here as Extended Resolution is very powerful and is likely to provide the propositional implications we need anyway, especially when it already simulates DRAT.

Dealing with EUR, in its full power, may prove more tricky, as it will deal with understanding the relationship between strategy extraction and the dependency scheme of [14]. In addition, EUR is used in QRAT to simulate universal expansion and the relationship between strategy extraction and expansion is also opaque at the moment.

We also wish to note that despite the equivalences we show here, it is still our estimation that the QRAT and QRAT+ formats provide practical advantages in QBF solving not covered by complexity theory, just as DRAT continues to be used despite its equivalence to Extended Resolution. It should also be noted that we say nothing on the equivalence of satisfiability proofs for these systems.

Acknowledgements

Research supported by a Postdoctoral Prize Fellowship from EPSRC (1st author).

References

1. Balabanov, V., Jiang, J.H.R.: Unified QBF certification and its applications. Formal Methods in System Design **41**(1), 45–65 (2012)
2. Beyersdorff, O., Hinde, L., Pich, J.: Reasons for hardness in QBF proof systems. Electronic Colloquium on Computational Complexity (ECCC) **24**, 44 (2017), `https://eccc.weizmann.ac.il/report/2017/044`
3. Chew, L.: Hardness and optimality in QBF proof systems modulo NP. Electronic Colloquium on Computational Complexity (ECCC) **25**, 178 (2018), `https://eccc.weizmann.ac.il/report/2018/178`
4. Cook, S.A., Reckhow, R.A.: The relative efficiency of propositional proof systems. The Journal of Symbolic Logic **44**(1), 36–50 (1979)
5. Goldberg, E., Novikov, Y.: Verification of proofs of unsatisfiability for cnf formulas. In: Proceedings of the Conference on Design, Automation and Test in Europe - Volume 1. pp. 10886–. DATE '03, IEEE Computer Society, Washington, DC, USA (2003), `http://dl.acm.org/citation.cfm?id=789083.1022836`
6. Heule, M., Seidl, M., Biere, A.: Efficient extraction of skolem functions from QRAT proofs. In: Formal Methods in Computer-Aided Design, FMCAD 2014, Lausanne, Switzerland, October 21-24, 2014. pp. 107–114 (2014). https://doi.org/10.1109/FMCAD.2014.6987602, `https://doi.org/10.1109/FMCAD.2014.6987602`
7. Heule, M., Seidl, M., Biere, A.: A unified proof system for QBF preprocessing. In: Automated Reasoning - 7th International Joint Conference, IJCAR 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 19-22, 2014. Proceedings. vol. 8562, pp. 91–106. Springer (2014). https://doi.org/10.1007/978-3-319-08587-6_7, `https://doi.org/10.1007/978-3-319-08587-6\_7`
8. Jussila, T., Biere, A., Sinz, C., Kröning, D., Wintersteiger, C.M.: A first step towards a unified proof checker for QBF. In: Marques-Silva, J., Sakallah, K.A. (eds.) SAT. vol. 4501, pp. 201–214. Springer (2007)

9. Kiesl, B., Heule, M.J.H., Seidl, M.: A little blocked literal goes a long way. In: Theory and Applications of Satisfiability Testing - SAT 2017 - 20th International Conference, Melbourne, VIC, Australia, August 28 - September 1, 2017, Proceedings. pp. 281–297 (2017). https://doi.org/10.1007/978-3-319-66263-3_18, `https://doi.org/10.1007/978-3-319-66263-3\_18`
10. Kiesl, B., Rebola-Pardo, A., Heule, M.J.H.: Extended resolution simulates DRAT. In: 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. pp. 516–531 (2018). https://doi.org/10.1007/978-3-319-94205-6_34, `https://doi.org/10.1007/978-3-319-94205-6\_34`
11. Kleine Büning, H., Bubeck, U.: Theory of quantified Boolean formulas. In: Biere, A., Heule, M., van Maaren, H., Walsh, T. (eds.) Handbook of Satisfiability, Frontiers in Artificial Intelligence and Applications, vol. 185, pp. 735–760. IOS Press (2009)
12. Kleine Büning, H., Karpinski, M., Flögel, A.: Resolution for quantified Boolean formulas. Inf. Comput. **117**(1), 12–18 (1995)
13. Lonsing, F., Egly, U.: QRAT+: generalizing QRAT by a more powerful QBF redundancy property. CoRR **abs/1804.02908** (2018), `http://arxiv.org/abs/1804.02908`
14. Slivovsky, F., Szeider, S.: Variable dependencies and Q-resolution. In: Sinz, C., Egly, U. (eds.) 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. vol. 8561, pp. 269–284. Springer (2014). https://doi.org/10.1007/978-3-319-09284-3_21, `http://dx.doi.org/10.1007/978-3-319-09284-3\_21`
15. Wetzler, N., Heule, M.J.H., Hunt, W.A.: Drat-trim: Efficient checking and trimming using expressive clausal proofs. In: Sinz, C., Egly, U. (eds.) Theory and Applications of Satisfiability Testing – SAT 2014. pp. 422–429. Springer International Publishing, Cham (2014)

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il