Electronic Colloquium on Computational Complexity, Report No. 104 (2021)

Does QRAT simulate IR-calc? QRAT simulation algorithm for ∀Exp+Res cannot be lifted to IR-calc

Sravanthi Chede[a], Anil Shukla[a]

[a]*Department of Computer Science and Engineering, IIT Ropar, India.*
{sravanthi.20csz0001,anilshukla}@iitrpr.ac.in

Abstract

We show that the QRAT simulation algorithm of ∀Exp+Res from [B. Kiesl and M. Seidl, 2019] cannot be lifted to IR-calc.

Keywords: Quantified Boolean Formulas (QBF), proof complexity, simulation

1. Introduction

Quantified Boolean formulas (QBFs) extend propositional formulas by adding quantification ∃ (there exists) and ∀ (for all) to the variables. Several QBF proof systems like Q-Res [1], LD-Q-Res [2], ∀Exp+Res [3], IR-calc [4] have been developed. However, these proof systems are unable to simulate the preprocessing steps used by several QBF-solvers. To overcome this a new proof system Quantified Resolution Asymmetric Tautologies (QRAT) [5] has been developed and also shown that it is capable of simulating all the existing preprocessing steps used by the current QBF-solvers [5].

Recently it has been shown that QRAT can even simulate ∀Exp+Res [6] and LD-Q-Res [7]. We know that IR-calc and LD-Q-Res are incomparable [8] and since QRAT can simulates LD-Q-Res, it implies that IR-calc cannot simulate QRAT. But, it is still open whether QRAT can simulate IR-calc?

Since IR-calc is an extension of ∀Exp+Res, it is very natural to use the QRAT simulation algorithm of ∀Exp+Res for the IR-calc. In this note we show that this is not possible. That is, we cannot lift the QRAT simulation algorithm of ∀Exp+Res for IR-calc in general. For proving the same we consider an important family of false QBFs ϕ_n from [3], which is known to be easy for IR-calc and hard for ∀Exp+Res and we show that IR-calc proof of ϕ_n

ISSN 1433-8092

cannot be simulated by the proposed modified algorithm (Section 4) which is the only approach to lift the existing simulation algorithm.

2. Definitions

In this note we assume that QBFs are in closed prenex form i.e., we consider the form $Q_1X_1...Q_kX_k.\psi$, where X_i are pairwise disjoint sets of variables; $Q_i \in \{\exists, \forall\}$ and $Q_i \neq Q_{i+1}$. The propositional part ψ of a QBF is called the matrix which should be in CNF (Conjunctive Normal Form) and the rest is the prefix Q. A clause is a disjunction of literals and a CNF is a conjunction of clauses. We denote the empty clause by $\bot$. If a variable x is in the set X_i, we say that x is at level i and write $lv(x) = i$. Note that only variables are in prefix, so given a literal l if x=var(l),then quantifier of l is same as that of x. Given a literal ℓ with quantifier Q_i and a literal k with quantifier Q_j, we write $\ell \leq_Q k$ if $i \leq j$ (we say that ℓ occurs left of k).

Informally, a proof system is a function f which maps proofs to theorems (or contradictions). A proof system f simulates another proof system g (i.e., $f \leq_p g$) if every g-proof of a theorem (or contradiction) can be efficiently translated into an f-proof of the same theorem. Proof systems f and g are said to be incomparable, if none of them can simulate the other.

One of the main approach to QBF-solving is through expansion of quantifiers. Several expansion-based QBF proof systems have been developed, for example, ∀Exp+Res [3]. This calculus downloads the axioms by dropping all universal literals in a clause and annotating the existential literals by an assignment to all universal variables which occur to left of that variable. It also allows the following resolution step: $\frac{(C_1 \vee x^\tau) \qquad (C_2 \vee \overline{x}^\tau))}{(C_1 \vee C_2)}$, where C_1 and C_2 are clauses, x^τ is a literal, and $(C_1 \vee C_2)$ is the resolvent.

The IR-calc proof system [4] has been developed as an extension of the ∀Exp+Res. Here, in the axiom steps, the existential variables are only annotated with ∀ variables which are on its left and belong to the same clause. The following instantiation step is also introduced: $inst(\sigma, C) = \{x^{\tau[\sigma]} \mid x^\tau \in C\}$ where, σ is a partial assignment to the universal variables and for every ∀ variable ℓ to the left of x, $\tau[\sigma]$ returns $\tau(\ell)$ if $\ell \in dom(\tau)$ else $\sigma(\ell)$ if $\ell \in dom(\sigma)$. The resolution step remains the same.

QRAT Proof System [5]: We need the following definitions:

Definition 1 ([5]). Clause C is an **Asymmetric Tautology** (AT) w.r.t. to CNF ψ iff $\psi \vdash_1 C$. (Alternatively can be checked if $\bot \in$ unit-propagation($\psi \cup \overline{C}$)). Unit propagation($\vdash_1$) simplies a CNF by repeating the following: If there is a unit clause (ℓ) then remove all clauses that contain the literal ℓ and remove the literal $\overline{\ell}$ from all clauses.

Given two clauses $(C \vee \ell), (D \vee \overline{\ell})$ of a QBF $Q.\psi$, the outer resolvent OR(Q,C,D,ℓ) is the clause consisting of all literals in C together with those literals of D that occur to the left of ℓ, i.e $C \cup \{k \mid k \in D, k \leq_Q \ell\}$. A clause $C \vee \ell$ is QRAT-clause w.r.t a QBF $Q.\psi$ if for every $D \vee \overline{\ell} \in \psi$ the OR(Q,C,D,ℓ) is implied by unit propagation. We say that ℓ is the QRAT-literal.

If a clause C contains an existential QRAT literal, it has been shown in [5] that C can be removed or added without effecting the satisfiability. Also, if a clause C contains a universal QRAT literal ℓ then dropping ℓ from C is also a satisfiability preserving step. Note that a clause which is AT is also a QRAT-clause on any literal. Additionally, QRAT allows elimination of any clause at any point in the proof. The only remaining rule of QRAT system is the Extended Universal Reduction (EUR) rule. We need the following:

Definition 2 ([6])**.** Given a QBF ϕ=$Q.\psi$, a universal literal u and an existential literal e_n to the right of u, we say that ϕ contains a **resolution path** from u to e_n if there exists a sequence $C_1, ..., C_n$ of clauses in ψ such that $u \in C_1$, $e_n \in C_n$ along with a sequence $e_1, ..., e_{n-1}$ of existential literals which occur to the right of u, where $e_i \in C_i$, $\overline{e_i} \in C_{i+1}$ and $var(e_i) \neq var(e_{i+1})$.

The **reflexive-resolution-path dependency** scheme (D^{rrs}) defines that an existential literal e depends on a universal literal u iff one of the following conditions holds: (1) There exist resolution paths from u to e and from $\overline{u}$ to $\overline{e}$. (2) There exist resolution paths from u to $\overline{e}$ and from $\overline{u}$ to e.

The EUR rule of QRAT allows to remove a universal literal u from a clause $(C \vee u) \in \psi$ where all $\ell \in C$ are independent of u according to D^{rrs}.

3. A brief recap of the QRAT simulation of ∀Exp+Res from [6]

The algorithm from [6] starts with a QBF $Q.\psi$ and a ∀Exp+Res proof π of $Q.\psi$. Then constructs a QRAT proof Π of $Q.\psi$ as follows:

Step 1 (Introduction of definitions): For each annotated variable x^τ in π, introduce the definition clauses $(\overline{x}^\tau \vee x)$ and $(x^\tau \vee \overline{x})$ in this order and place x^τ in the same quantifier block as x. Denote the resulting accumulated formula by $Q'.\psi_1$. Observe that this step is valid: the definition clause $(\overline{x}^\tau \vee x)$ is a QRAT since x^τ is new. Then, the definition $(x^\tau \vee \overline{x})$ is a QRAT since the only outer-resolvent upon x^τ is the tautology $(\overline{x} \vee x)$.

Step 2 (Introduction of annotated clauses): For each clause $C^\tau \in \pi$

that was obtained from clause $C \in \psi$ by axiom rule, add a clause $C^{\tau} \vee u_1 \vee ... \vee u_k$ (where $u_1, ..., u_k$ are universal literals in C). Denote resulting accumulated formula by $Q^{'}.\psi_2$. Observe that this new clause is AT w.r.t ψ_1.
Step 3 (Elimination of Input Clauses and Definitions): $Q^{'}.\psi_3 = \psi_2 - \psi_1$.
Step 4 (Removal of all universal literals): Apply EUR rule in annotated clauses for dropping all $\forall$ variables from right to left in the prefix $Q^{'}$. This completes the axiom steps. This step is valid because of the following Lemma.

Lemma 3 ([6]). *If $Q^{'}.\psi_3$ contains a resolution path from u to e, then e must be an annotated literal of the form l^{τ} where the assignment τ falsifies u.*

This lemma being true implies that D^{rrs} can never be found between u and e as the path sees the literals u,e and $\overline{e}$ but never includes $\overline{u}$.This being true for any such e implies u can be safely dropped by EUR rule.
Step 5 (Resolution proof): Simulate the remaining resolution steps.

3.1. Problems with direct usage of above Algorithm for IR-calc proofs

In order to simulate an IR-calc proof, one also need to simulate the instantiation steps. We observe that if we do not delete all the definition clauses in Step 3, we can simulate the instantiation steps as well. Note that we may need to introduce more definition clauses for fresh variables when introduced.

Lemma 4. *Instantiation step can always be simulated when retaining all the definition clauses in the QBF.*

Proof. Suppose we have a clause $C_i = ({x_1}^{\tau_1} \vee {x_2}^{\tau_2} \vee {e_i}^{\tau})$ and the IR-calc proof applies an instantiation step on C_i, i.e, $inst(\sigma, C_i)$. Let the step return $C_i^{'} = ({x_1}^{\tau_1} \vee {x_2}^{\tau_2[\sigma]} \vee {e_i}^{\tau[\sigma]})$ and say annotations of x_1 have not changed but those of x_2 have changed and resulted in a new variable $({x_2}^{\tau_2[\sigma]})$ not present in the QBF currently and that we wanted to resolve on e_i so it's annotations have changed but the new variable is already existing in the QBF.

${x_2}^{\tau_2[\sigma]}$ being a new variable, we add its definitions $(\overline{x_2} \vee {x_2}^{\tau_2[\sigma]}) \wedge (x_2 \vee \overline{x_2}^{\tau_2[\sigma]})$ and because ${x_2}^{\tau_2}$,${e_i}^{\tau}$,${e_i}^{\tau[\sigma]}$ are already existing we would have $(x_2 \vee \overline{x_2}^{\tau_2})$, $(e_i \vee \overline{e_i}^{\tau})$, $(\overline{e_i} \vee {e_i}^{\tau[\sigma]})$ clauses already present in the QBF. Observe that a series of resolving steps on these clauses derives the clause $C_i^{'}$. Therefore, it is an AT and hence a QRAT-clause and can be added. □

However, if we retain all the definition clauses, then Lemma 3 may not always hold, implying that we cannot simulate the axiom download steps. Also we cannot add definition clauses of existing variables after axiom step as they are no longer QRAT clauses. The only way to lift the algorithm for IR-calc is to retain all the **important** definition clauses and delete the unimportant ones. We say that a definition clause is important if we need the same for the simulation of an instantiation step later in the IR-calc proof.

This motivates us to design a two pass algorithm. In the first pass, the algorithm marks all the important definition clauses and deletes the unimportant ones. The algorithm then checks whether Lemma 3 holds with all the important clauses present. If no, the algorithm stops. Otherwise, in the second pass the algorithm continues with the successful simulation of the IR-calc proof. Next, we present the algorithm in detail.

4. Modified QRAT simulation algorithm for IR-calc

The Modified algorithm starts with a QBF $Q.\psi$ and an IR-calc proof π of $Q.\psi$, an constructs a QRAT proof Π of $Q.\psi$ as follows:

Step 1 (Introduction of definition clauses): In the first pass, we add definition clauses of all annotated variables in the Axiom clauses as defined in the above algorithm. Denote the resulting accumulated formula by $Q^{'}.\psi_1$. Also give labels to all definition clauses, say $D_1, ..., D_{2k}$.

Step 2 (Introduction of annotated clauses): Exactly as defined in the Step 2 of above algorithm. Denote the resulting formula by $Q^{'}.\psi_2$.

Step 3 (Elimination of input clauses): We only drop the input clauses from ψ in this step. Denote the resulting accumulated formula by $Q^{'}.\psi_3$.

Step 4 (Find all important definition clauses): Assume that the axiom downloads have been performed and go ahead in the IR-calc proof π scanning for instantiation steps. Let $C_i = ({x_1}^{\tau_1} \vee {x_2}^{\tau_2} \vee {x_3}^{\tau_3})$ be any derived clause in π and $C_{i+1} = \text{inst}(\sigma,C_i)$ be an instantiation step. Say $C_{i+1} = ({x_1}^{\tau_1[\sigma]} \vee {x_2}^{\tau_2[\sigma]} \vee {x_3}^{\tau_3})$, where $x_1^{\tau_1[\sigma]}$ is the literal with modified annotations which already exists in the QBF, $x_2^{\tau_2[\sigma]}$ is a new variable not yet present in the QBF and $x_3^{\tau_3}$ is the literal whose annotations did not change after the instantiation step (zero or more of each type of variables are allowed in the clause C_i).

For each existing changed literal (i.e x_1) we note the clause needed for this change in the annotations of x_1 i.e $(\overline{x_1}^{\tau_1} \vee {x_1}^{\tau_1[\sigma]})$. Re-write these clauses in terms of definition clauses i.e $(x_1 \vee \overline{x_1}^{\tau_1}) \wedge (\overline{x_1} \vee {x_1}^{\tau_1[\sigma]})$. We mark these definition clauses as important. Then, for each new changed literal (i.e x_2), we mark the definition clause $(x_2 \vee \overline{x_2}^{\tau_2})$ as important. Note that the other clause needed in this case will be added at the time of actual simulation.

Step 5 (Drop all unimportant definition clauses added in Step 1).

Step 6 (Find resolution paths): At this point we know that all instantiation steps of π can be simulated. The algorithm now checks whether the

EUR steps are still applicable to complete the simulation of the axiom steps. We check the same as follows: for every universal variable (u) going from right to left order in the prefix $Q^{'}$, check if there exists a resolution path that starts from a clause containing u to one containing $\overline{u}$ or vice-versa. If found, halt and declare that the given IR-calc proof cannot be simulated by the algorithm. If no such resolution paths exist, continue to **Step 7**.

Step 7 (Drop universal literals): Drop all universal literals from all the annotated clauses introduced in Step 2. This completes simulating the axiom download steps of IR-calc.

Step 8 (Simulate resolution and instantiation steps): In the second pass of this algorithm, simulate the resolution and instantiation steps in order as they occur in π. That is, for every resolution step, add the resolvent clause. Since all the important definition clauses are present, every instantiation steps can be simulated by Lemma 4. The completes the algorithm.

Let us quickly understand the algorithm with an example.

Example 5. Consider the following QBF and an IR-calc proof of the same in Figure 1a. Apply the modified algorithm on the same.

$$\Psi_0 = \forall u_1 \exists e_2 \forall u_3 \exists e_4, e_5.\ (\overline{u_1} \vee \overline{e_2} \vee \overline{u_3} \vee e_5) \wedge (\overline{u_1} \vee \overline{u_3} \vee \overline{e_4}) \\ \wedge (e_2 \vee \overline{u_3} \vee e_4) \wedge (u_1 \vee \overline{e_2}) \wedge (\overline{u_1} \vee \overline{e_2} \vee \overline{e_5})$$

Step 1: Add definitions for all the annotated literals in $C_1, .., C_5$ (Fig. 1a). There will be a total of 12 definition clauses added. QBF is now $Q^{'}.\Psi_1$

Step 2: Labels

$$\Psi_2 = \Psi_1 \wedge (\overline{u_1} \vee \overline{e_2}^{u_1} \vee \overline{u_3} \vee {e_5}^{u_1 u_3}) \wedge (\overline{u_1} \vee \overline{u_3} \vee \overline{e_4}^{u_1 u_3}) \qquad (C^{'}_1, C^{'}_2)$$

$$\wedge (e_2 \vee \overline{u_3} \vee {e_4}^{u_3}) \wedge (u_1 \vee \overline{e_2}^{\overline{u_1}}) \wedge (\overline{u_1} \vee \overline{e_2}^{u_1} \vee \overline{e_5}^{u_1}) \qquad (C^{'}_3, C^{'}_4, C^{'}_5)$$

Step 3: $\Psi_3 = \Psi_2 - \Psi_0$

Step 4:(from IR-calc proof in Figure 1a)

1: $C_6 = inst(u_1, C_3)$: Required clauses $= (\overline{e_4}^{u_3} \vee {e_4}^{u_1 u_3}) \wedge (\overline{e_2} \vee {e_2}^{u_1})$
Imp. Def clauses $= (e_4 \vee \overline{e_4}^{u_3}) \wedge (\overline{e_4} \vee {e_4}^{u_1 u_3}) \wedge (\overline{e_2} \vee {e_2}^{u_1})$

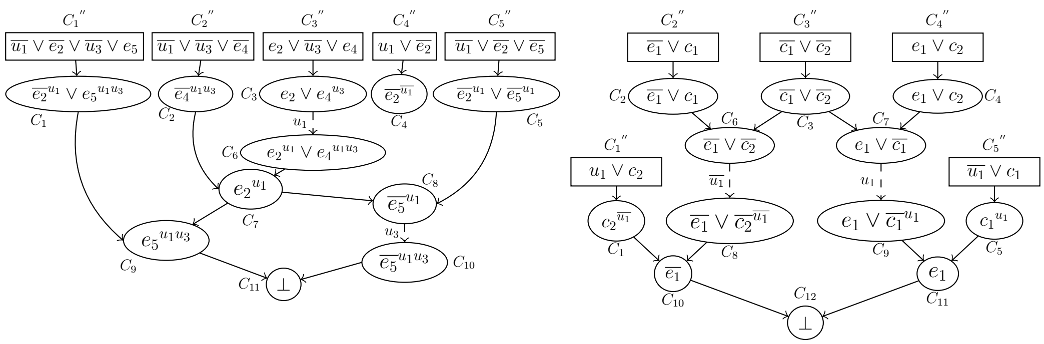
2: $C_{10} = inst(u_3, C_8)$: Required clauses $= ({e_5}^{u_1} \vee \overline{e_5}^{u_1 u_3})$
Imp. Def clauses $= (\overline{e_5} \vee {e_5}^{u_1}) \wedge (e_5 \vee \overline{e_5}^{u_1 u_3})$

Step 5: Drop non-important definitions, now formula will be:

$$\Psi_4 = \Psi_3 - \Psi_1 + \{(e_4 \vee \overline{e_4}^{u_3}) \wedge (\overline{e_4} \vee {e_4}^{u_1 u_3}) \wedge (\overline{e_2} \vee {e_2}^{u_1}) \wedge (\overline{e_5} \vee {e_5}^{u_1}) \wedge (e_5 \vee \overline{e_5}^{u_1 u_3})\}$$

Step 6:

Rightmost $\forall$ variable $= u_3$: But no opposite literals in any clause pairs.



(a) IR-calc proof of Ψ_0 (Example 5) (b) IR-calc proof of ϕ_1

Figure 1: Example IR-calc proofs. (Dashed arrow correspond to the instantiation steps).

Next rightmost $\forall$ variable $= u_1$: No paths found in Ψ_4.
Step 7 & 8: Drop all $\forall$ variables from Ψ_4. Now, resolvent and instantiated clauses of IR-calc can be directly added in order since they are AT w.r.t the QBF at that point. This completes the simulation.

5. Counter-Example:

We show that the proposed two pass algorithm cannot simulate every IR-calc proof. Consider the following false family of QBFs ϕ_n from [3] and an IR-calc proof for ϕ_1 in Figure 1b. Apply the modified algorithm on ϕ_1.

$$\phi_n \equiv \exists e_1 \forall u_1 \exists c_1 c_2 \ldots \exists e_n \forall u_n \exists c_{2n-1} c_{2n}.$$
$$\bigwedge_{i \in [n]} \{(\overline{e_i} \vee c_{2i-1}) \wedge (\overline{u_i} \vee c_{2i-1}) \wedge (e_i \vee c_{2i}) \wedge (u_i \vee c_{2i})\} \wedge (\bigvee_{i \in [2n]} \overline{c_i})$$

$$Q.\Phi_0 = \phi_1 = \exists e_1 \forall u_1 \exists c_1 c_2 .\ (u_1 \vee c_2) \wedge (\overline{e_1} \vee c_1) \wedge (\overline{c_1} \vee \overline{c_2}) \wedge (e_1 \vee c_2) \wedge (\overline{u_1} \vee c_1)$$

Step 1:This QBF will need a total of 4 definition clauses. QBF is now $Q'.\Phi_1$.
Step 2: Labels

$$\Phi_2 = \Phi_1 \wedge (u_1 \vee c_2^{\overline{u_1}}) \wedge (\overline{e_1} \vee c_1) \wedge (\overline{c_1} \vee \overline{c_2}) \qquad (C_1', C_2', C_3')$$
$$\wedge (e_1 \vee c_2) \wedge (\overline{u_1} \vee c_1^{u_1}) \qquad (C_4', C_5')$$

Step 3: $\Phi_3 = \Phi_2 - \Phi_0$
Step 4:(From the IR-calc proof example in Fig. 1b)
1: $C_8 = inst(\overline{u_1}, C_6)$: Required clauses $= (c_2 \vee \overline{c_2}^{\overline{u_1}}) =$ Imp. Def clauses,
2: $C_9 = inst(u_1, C_7)$: Required clauses $= (c_1 \vee \overline{c_1}^{u_1}) =$ Imp. Def clauses.

Step 5: $\Phi_4 = \Phi_3 - \Phi_1 + \{(c_2 \vee \overline{c_2}^{\overline{u_1}}) \wedge (c_1 \vee \overline{c_1}^{u_1})\}$ (D_1, D_2)
Step 6: Rightmost $\forall$ variable $= u_1$: $C_5^{'}$ has $\overline{u_1}$ and $C_1^{'}$ has u_1
Resolution path: $C_5^{'}, D_2, C_3^{'}, D_1, C_1^{'}$
We have a path where every clause is important so algorithm fails and halts.
Discussions and conclusions: In this short note, we show that the QRAT simulation algorithm for ∀Exp+Res cannot be lifted to IR-calc. The only approach to lift this algorithm is similar to the two pass algorithm defined in Section 4. We showed that the modified algorithm cannot simulate the IR-calc proof of the formula ϕ_n, which is known to be easy for IR-calc but hard for ∀Exp+Res. Whether this is always the case is unclear. That is, does the algorithm always fail to simulate the IR-calc proof of QBFs which are hard for ∀Exp+Res? In closing, it is still open 'whether QRAT can simulate IR-calc?'

References

[1] H. Kleine Büning, M. Karpinski, A. Flögel, Resolution for quantified Boolean formulas, Information and Computation 117 (1) (1995) 12–18.

[2] V. Balabanov, J.-H. R. Jiang, Unified QBF certification and its applications, Formal Methods in System Design 41 (1) (2012) 45–65.

[3] M. Janota, J. Marques-Silva, Expansion-based QBF solving versus Q-resolution, Theoretical Computer Science 577 (2015) 25–42.

[4] O. Beyersdorff, L. Chew, M. Janota, On unification of QBF resolution-based calculi, in: Mathematical Foundations of Computer Science (MFCS), 2014, pp. 81–93.

[5] M. J. H. Heule, M. Seidl, A. Biere, Solution validation and extraction for QBF preprocessing, J. Autom. Reason. 58 (1) (2017) 97–125.

[6] B. Kiesl, M. Seidl, QRAT polynomially simulates ∀-Exp+Res, in: Theory and Applications of Satisfiability Testing - SAT 2019, 2019, pp. 193–202.

[7] B. Kiesl, M. J. H. Heule, M. Seidl, A little blocked literal goes a long way, in: Theory and Applications of Satisfiability Testing - SAT 2017, Vol. 10491, Springer, 2017, pp. 281–297.

[8] O. Beyersdorff, L. Chew, M. Janota, Proof complexity of resolution-based QBF calculi, in: STACS 2015, Vol. 30 of LIPIcs, 2015, pp. 76–89.

ECCC ISSN 1433-8092
https://eccc.weizmann.ac.il