Expansion-based QBF Solving versus Q-Resolution

Mikoláš Janota[a,*], Joao Marques-Silva[a,b]

[a] *IST/INESC-ID, Lisbon, Portugal*
[b] *University College Dublin, Ireland*

Abstract

This article introduces and studies a proof system ∀Exp+Res that enables us to refute quantified Boolean formulas (QBFs). The system ∀Exp+Res operates in two stages: it expands all universal variables through conjunctions and refutes the result by propositional resolution. This approach contrasts with the Q-resolution calculus, which enables refuting QBFs by rules similar to propositional resolution. In practice, Q-resolution enables producing proofs from conflict-driven DPLL-based QBF solvers. The system ∀Exp+Res can on the other hand certify certain expansion-based solvers. So a natural question is to ask which of the systems, Q-resolution and ∀Exp+Res, is more powerful? The article gives several partial responses to this question. On the positive side, we show that ∀Exp+Res can p-simulate tree Q-resolution. On the negative side, we show that ∀Exp+Res does not p-simulate unrestricted Q-resolution. In the favor of ∀Exp+Res we show that ∀Exp+Res is more powerful than a certain fragment of Q-resolution, which is important for DPLL-based QBF solving.

Keywords: quantified Boolean formula, proof theory, expansion, Q-resolution, SAT

1. Introduction

Semantically, *Quantified Boolean Formulas (QBFs)* can be seen as propositional formulas since quantifiers can be rewritten through conjunctions and disjunctions. Such rewriting, however, leads to an exponential blowup of the

*Corresponding author.
Email addresses: mikolas@sat.inesc-id.pt (Mikoláš Janota), jpms@ist.utl.pt (Joao Marques-Silva)

formula. Consequently, QBFs and QBF solving are fundamentally different from propositional formulas and *satisfiability solving (SAT)*. While SAT is NP-complete, deciding whether a given QBF is true or not is PSPACE-complete. Further complexity differences exist. For instance, QBF remains PSPACE-complete even for bounded tree-width [1] whereas SAT becomes tractable in such case [2].

This article follows the line of research on proof systems for propositional and Quantified Boolean Formulas (QBFs). This research is motivated by complexity theory and more recently by the objective to develop and certify QBF solvers [3, 4, 5, 6]. Proof systems for QBF come in different styles and flavors. Krajíček and Pudlák propose a Gentzen-style calculus *KP* for QBF [4]. Büning et al. propose a refutation calculus *Q-resolution* [5], an extension of propositional resolution. Giunchiglia et al. extend the work of Büning et al. into *term resolution* for proofs of true formulas [6] . Certain separation results were shown between KP and Q-resolution recently by Egly [7]. Van Gelder introduces a generalization of Q-resolution, called *QU-resolution* [8].

While many QBF solvers are based on the DPLL procedure and conflict-driven learning [9, 10, 11, 12, 13], other solvers tackle the given formula by *expanding* out quantifiers until a single quantifier type is left. At that point, this formula is handed to a SAT solver [14, 15, 16, 17]. Experimental results show that expansion-based QBF solvers can outperform DPLL-based solvers on a number of families of practical instances. Also, expansion can be used in QBF preprocessing [18, 19].

This practical importance of expansion motivates the theoretical study carried out in this article. We define a proof system ∀Exp+Res, which eliminates universal quantification from the given *false* QBF and then applies propositional resolution to refute the remainder. We show several results on how ∀Exp+Res compares to Q-resolution.

The article is organized as follows. Section 2 introduces concepts and notation used throughout the paper. Section 3 introduces the proof system ∀Exp+Res. Section 4 shows that *tree* Q-resolution is polynomially simulated by ∀Exp+Res. Section 5 investigates the relation in the opposite direction and shows that Q-resolution polynomially simulates a certain fragment of ∀Exp+Res. Section 6 shows that the result shown in Section 4 "cannot be improved", i.e. that unrestricted Q-resolution is *not* polynomially simulated by ∀Exp+Res. Section 7 shows a somewhat weaker negative result in the opposite direction; it shows that a certain restriction of Q-resolution does not

polynomially simulate ∀Exp+Res (we discuss why this result is important for QBF solving). Finally, Section 8 summarizes and concludes the article and suggests directions for future work.

This article is based on authors' SAT '13 paper [20], which served as the bases for Sections 3–5. The separation result shown in Section 6 was presented at the International QBF Workshop '13 [21].

2. Preliminaries

A *literal* is a Boolean variable or its negation. The literal complementary to a literal l is denoted as $\bar{l}$, i.e. $\bar{x} = \neg x$, $\overline{\neg x} = x$. A *clause* is a disjunction of zero or more noncomplementary literals. A formula in *conjunctive normal form* (CNF) is a conjunction of clauses. Whenever convenient, a clause is treated as a set of literals and a CNF formula as a set of sets of literals. For a literal $l = x$ or $l = \bar{x}$, we write $\mathsf{var}(l)$ for x. For a clause C, we write $\mathsf{var}(C)$ to denote $\{\mathsf{var}(l) \mid l \in C\}$ and for a CNF ψ, $\mathsf{var}(\psi)$ denotes $\{l \mid l \in \mathsf{var}(\psi), C \in \psi\}$

Substitutions are denoted as $\psi_1/x_1, \ldots, \psi_n/x_n$, with $x_i \neq x_j$ for $i \neq j$ and ϕ_i arbitrary formulas. The set of variables $x_1, \ldots, x_n$ is called the *domain* of the substitution, denoted by $\mathsf{dom}()$. An *application* of a substitution on a formula ϕ is denoted as $\phi[\psi_1/x_1, \ldots, \psi_n/x_n]$ meaning that variables x_i are simultaneously substituted by corresponding ψ_i in ϕ.

An *assignment* is a mapping from Boolean variables to the values 0, 1. An assignment is called *total*, or *complete*, for a set of variables X if each $x \in X$ is in the domain of the assignment. Assignments will be denoted as in this example $[x{\to}0, y{\to}1]$. For assignments τ_1 and τ_2 with disjoint domains we write $\tau_1 \cup \tau_2$ for the assignment that assigns $\tau_1(x)$ to x if $x \in \mathsf{dom}(\tau_1)$ and assigns $\tau_2(x)$ to x if $x \in \mathsf{dom}(\tau_2)$.

For an assignment τ and a formula ϕ we write $\phi\tau$ for the application of τ to ϕ. That is, $x\tau = \tau(x)$ if $x \in \mathsf{dom}(\tau)$ and $x\tau = x$ otherwise; $(\neg\phi)\tau = 1 - c$ if $\phi\tau = c$ for $c \in \{0, 1\}$ and $(\neg\phi)\tau = \neg(\phi\tau)$ otherwise; $(\phi_0 \wedge \phi_1)\tau = 0$ if $\phi_i\tau = 0$ for $i \in 0..1$; $(\phi_0 \wedge \phi_1)\tau = \phi_{1-i}$ if $\phi_i\tau = 1$ for $i \in 0..1$; $(\phi \wedge \psi)\tau = 0$ if $\phi\tau = 0$ or $\psi\tau = 0$; and $(\phi \wedge \psi)\tau = (\phi\tau \wedge \psi\tau)$ otherwise. Similarly for other connectives. If ϕ is in CNF, then the application of an assignment τ additionally removes any clauses that evaluate to 1.

The *resolution* rule is defined for clauses $C_1 \vee x$ and $C_2 \vee \neg x$ s.t. there are no complementary literals in $C_1 \cup C_2$. The result of resolution is called the *resolvent* and is the clause $C_1 \vee C_2$. For a CNF ϕ, a *resolution proof* of

a clause C is a sequence of clauses $C_1, \ldots, C_n$ where $C_n = C$ and any C_i in the sequence is part of the formula ϕ or it is a resolvent for some pair of the preceding clauses. A resolution proof is called a *refutation* iff C is the empty clause, denoted $\bot$.

2.1. Quantified Boolean Formulas

Quantified Boolean Formulas (QBFs) [22] extend propositional logic by enabling quantification over Boolean variables. Any propositional formula ϕ is also a QBF with all variables *free*. If Φ is a QBF with a free variable x, the formulas $\exists x.\, \Phi$ and $\forall x.\, \Phi$ are QBFs with x *bound*, i.e. not free. Note that we disallow expressions such as $\exists x. \exists x.\, x$. Whenever possible, we write $\exists x_1 \ldots x_k$ instead of $\exists x_1 \ldots \exists x_k$; analogously for $\forall$. For a QBF $\Phi = \forall x.\, \Psi$ we say that x is *universal* in Φ and is existential in $\exists x.\, \Psi$. Analogously, a literal l is universal (resp. existential) if $\mathsf{var}(l)$ is universal (resp. existential).

The application of an assignment τ is defined for a QBF Φ if all variables of $\mathsf{dom}(\tau)$ are free in Φ, and, it is defined as $(\mathcal{Q}x.\, \Phi)\tau = \Phi\tau$ for $\mathcal{Q} \in \{\exists, \forall\}$. QBFs can be seen as compact representations of propositional formulas. In particular, the formula $\forall x.\, \Psi$ is satisfied by the same truth assignments as $\Psi[x{\to}0] \wedge \Psi[x{\to}1]$ and $\exists x.\, \Psi$ by $\Psi[x{\to}0] \vee \Psi[x{\to}1]$. Since $\forall x \forall y.\, \Phi$ and $\forall y \forall x.\, \Phi$ are semantically equivalent, we allow writing $\forall X$ for a set of variables X; analogously for $\exists$. A QBF with no free variables is *false* (resp. *true*), iff it is semantically equivalent to the constant 0 (resp. 1).

A QBF is *closed* if it does not contain any free variables. A QBF is in *prenex form* if it is of the form $\mathcal{Q}_1 X_1 \ldots \mathcal{Q}_k X_k.\, \phi$, where $\mathcal{Q}_i \in \{\exists, \forall\}$, $\mathcal{Q}_i \neq \mathcal{Q}_{i+1}$, and ϕ is propositional. The propositional part ϕ is called the *matrix* and the rest the *prefix*. If a variable x is in the set X_i, we say that x is at *level* i and write $\mathsf{lv}(x) = i$; we write $\mathsf{lv}(l)$ for $\mathsf{lv}(\mathsf{var}(l))$.

We write QCNF for the class of QBF in prenex form where the matrix is in CNF. Unless specified otherwise, QBFs are assumed to be closed.

2.2. Q-resolution

Q-resolution [5] is an extension of propositional resolution enabling refuting formulas in QCNF.

For a clause C, a universal literal $l \in C$ is *blocked* by an existential literal $k \in C$ iff $\mathsf{lv}(l) < \mathsf{lv}(k)$. $\forall$*-reduction* is the operation of removing from a clause C all universal literals that are *not* blocked by some literal.

Q-resolution is defined for two clauses $x \vee C_1$ and $\bar{x} \vee C_2$ that satisfy the following conditions: x is existential and there is no literal k s.t. $k, \bar{k} \in$

$C_1 \cup C_2$. The result of Q-resolution is the clause $C_1 \vee C_2$, which we call the *Q-resolvent*. We say simply resolvent whenever the context permits.

For a QCNF $\mathcal{P} \,.\, \phi$, a *Q-resolution proof* of a clause C is a sequence of clauses $C_1, \ldots, C_n$ where $C_n = C$ and any C_i in the sequence is part of the given matrix ϕ; or was obtained from one of the preceding clauses by $\forall$-reduction; or it is a Q-resolvent of some pair of preceding clauses. A Q-resolution proof is called a *refutation* iff C is the empty clause, denoted $\bot$.

In this article Q-resolution and plain resolution proofs are treated as connected directed acyclic graphs (DAG) so that each clause C in the proof corresponds to some node p_n labeled with C. More precisely, a resolution of two clauses C_1 and C_2 resulting in the resolvent C_r corresponds to 3 nodes p_1, p_2, p_r labeled by the respective clauses and an edge going from each p_i to p_r for $i \in 1..2$. If a clause C_r resulted from C_1 by $\forall$-reduction, the graph contains two corresponding nodes p_r and p_1 connected by an edge from p_1 to p_r.

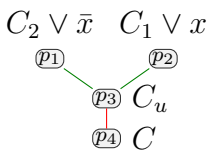


Figure 1: Q-resolution

Any graph representing a resolution or Q-resolution proof has one and only one node with out-degree 0, which we call the *root* (the final clause in the proof). All the nodes with in-degree 0 are labeled with clauses from the original formula and we call them *leafs*. Q-resolution steps are depicted as in Figure 1.

A Q-resolution proof is called *tree* (or *tree-like*), if the corresponding graph forms a tree (rooted in the final clause). It is known that at the propositional level, tree resolution does not p-simulate DAG resolution [23].

Remark 1. *Note that the condition that there are no two complementary literals in the antecedents of a Q-resolution step is there to preserve* soundness. *For instance, for the true formula* $\forall u \exists e.\, (u \vee e) \wedge (\bar{u} \vee \bar{e})$ *removing this condition would enable us to derive a refutation. One would first resolve the two clauses into* $(u \vee \bar{u})$ *and second, apply* $\forall$*-reduction to that clause and thus derive* $\bot$*. Alternatively to this restriction one could instead prohibit* $\forall$*-reductions of complementary literals but that would not give us any advantage.*

2.3. Proof Systems

Throughout the article, we use the term "proof system" in the sense of Cook and Reckhow [3]. That is, for finite alphabets Σ_D, Σ_1, and $L \subseteq \Sigma_D^*$ a

proof system is a function $f : \Sigma_1^* \to L$, where the function f is polynomially computable and it is onto. We say that a proof system $f_2 : \Sigma_2^* \to L$ *p-simulates* the proof system f iff there exists a polynomially computable function $g : \Sigma_1^* \to \Sigma_2^*$ s.t. $f_2(g(x)) = f(x)$ for all x. (Intuitively, the function f produces the formula being proven from its proof and g translates in polynomial time any proof in the system f to a proof of the same formula in the system f_2.)

As usual, we consider refutation-based procedures (such as resolution or Q-resolution) also as proof systems. For such, we can assume that a proof of a QBF is formed by a refutation of its negation. Note that, just as in the case of propositional logic, a QBF that does not have a CNF matrix can be efficiently translated into a QBF with a CNF matrix using *Tseitin transformation* [24] (see e.g. [25, §5] or [26, Sec. 2.2] for further discussion).

3. Expansion

Modern SAT solvers can be readily used in a black-box setting, which suggests a straightforward approach to solving QBFs by expanding variables until only one type of quantifier is left; at that point a SAT solver can be invoked. Further, modern mainstream SAT solvers accept formulas in CNF and enable producing resolution refutations for unsatisfiable formulas. This motivates the proof system developed in this section. For a false QCNF we wish to provide a proof that combines expansion of all but one quantifier and a resolution refutation of the result of the expansion.

Existential quantifier can be expanded by the equivalence $\exists x.\, \Phi = \Phi[x{\to}0] \vee \Phi[x{\to}1]$ and universal quantification by the equivalence $\forall x.\, \Phi = \Phi[x{\to}0] \wedge \Phi[x{\to}1]$. These equivalences reveal two main obstacles to developing a proof system using both expansion and plain resolution (besides the exponential growth). The first obstacle is that the result of an expansion is not in prenex form; this can be overcome by prenexing the expansion. The second obstacle is that the result of expanding the existential quantifier does not yield CNF. Hence, in this paper we focus only on expansion of the universal quantifier. We show that this limitation still leads to a refutation complete calculus with many interesting properties.

Expansion of universal quantifiers enables decreasing the number of quantifiers at the cost of an increase in size. Prenexing enables maintaining prenex normal form at the cost of introducing fresh variables. For instance, expanding $\exists x \forall y \exists z.\, \phi$ yields $\exists x.\, (\exists z.\, \phi[y{\to}0]) \wedge (\exists z.\, \phi[y{\to}1])$. To get back to

prenex form, we add two fresh copies of z, one for the sub-QBF where $y = 0$ and one for the sub-QBF where $y = 1$, thus obtaining the formula $\exists x z^0 z^1.\, \phi[y{\to}0][z^0/z] \wedge \phi[y{\to}1][z^1/z]$.

A significant drawback of expansion is that the formula grows in size exponentially. This effect can be mitigated by observing that only *partial expansions* may be sufficient to show unsatisfiability. For instance, expanding the formula $\forall y \exists x.\, (y \vee x) \wedge (y \vee \bar{x})$ only with the assignment $y{\to}0$ yields $\exists x^0.\, x^0 \wedge \bar{x}^0$ and therefore this expansion is sufficient to show the formula false.

Another source of rapid growth lies in the number of the formula's quantification levels. Expanding y in $\exists x \forall y \exists z \forall u \exists w.\, \phi$ yields $\exists x.\, (\exists z \forall u \exists w.\, \phi[y{\to}0]) \wedge (\exists z \forall u \exists w.\, \phi[y{\to}1])$. We could again prenex all variables but since we are aiming at eventually expanding *all* universal variables, we can expand more carefully by prenexing only z first:

$$\exists x z^0 z^1.\, (\forall u \exists w.\, \phi[y{\to}0][z^0/z]) \wedge (\forall u \exists w.\, \phi[y{\to}1][z^1/z])$$

If for instance now we wish to expand u as 1 in the first sub-formula and 0 in the second sub-formula we obtain the following:

$$\begin{aligned} &\exists x z^0 z^1.\, (\exists w.\, \phi[y{\to}0][z^0/z][u{\to}1]) \wedge (\exists w.\, \phi[y{\to}1][z^1/z][u{\to}0]) \\ &= \exists x z^0 z^1 w^{01} w^{10}.\, \phi[y{\to}0, u{\to}1][z^0/z, w^{01}/w] \wedge \phi[y{\to}1, u{\to}0][z^1/z, w^{10}/w] \end{aligned}$$

Had we started the expansion from higher to lower levels ("inside-out"), this tighter control would not be possible; it would not be possible to expand u differently for each expansion of y.

Consider a general QCNF

$$\Phi = \forall \mathcal{U}_1 \exists \mathcal{E}_2 \ldots \forall \mathcal{U}_{2N-1} \exists \mathcal{E}_{2N}.\, \phi \tag{1}$$

(WLOG we start with a universal quantifier to simplify notation). For succinctness, from now on Φ refers to this formula. An expansion consists of expanding variables $\mathcal{U}_1$ with some values and introducing fresh variables for $\mathcal{E}_2$ variables yielding a sub-QBF for each considered assignment to the $\mathcal{U}_1$ variables. These sub-QBFs are recursively expanded in an analogous fashion. Note that if we expanded from the highest quantification level (innermost level), we would lose the structural information, which is enabling the above-mentioned finer expansion steps. Let us now introduce definitions that formalize this process.

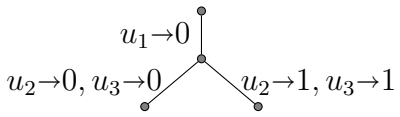


Figure 2: Example ∀-expansion tree.

The following definition introduces a structure that prescribes *how* a given formula should be expanded. Starting from the outermost level, it gives which assignments are considered for the variables $\mathcal{U}_1$ and then it proceeds recursively.

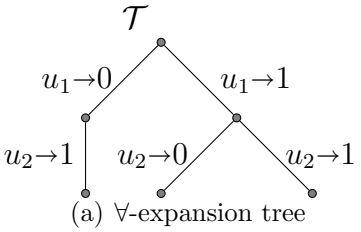
Definition 1 (∀-expansion tree). A ∀-*expansion tree* for Φ is a rooted tree $\mathcal{T}$ such that each path $p_0 \xrightarrow{\tau_1} p_1 \ldots \xrightarrow{\tau_N} p_N$ in $\mathcal{T}$ from the root p_0 to some leaf p_N has exactly N edges and each edge $p_{i-1} \xrightarrow{\tau_i} p_i$ is labeled with a total assignment τ_i to the variables $\mathcal{U}_{2i-1}$, for $i \in 1..N$. Each path in $\mathcal{T}$ is uniquely determined by its labeling.

Example 1. *Consider* $\Phi = \forall u_1 \exists e_1 \forall u_2 \forall u_3 \exists e_2.\, \phi$, *with* $\phi = (u_1 \vee e_1 \vee u_2 \vee u_3 \vee e_2) \wedge (u_1 \vee \bar{e}_1 \vee \bar{u}_2 \vee \bar{u}_3 \vee e_2) \wedge (\bar{e}_2)$. *Figure 2 shows an example expansion tree for this formula. The tree is depicted with the root on top and it first tells us that* u_1 *should be expanded only with the negative polarity. Then, for the variables* u_2, u_3, *two assignments are considered (out of the 4 possible ones). In general, each node can have as many children as there are assignments to the pertaining variables. In particular, if* all *assignments are considered for each set of variables, the number of nodes in the tree are* $2^{|\mathcal{U}_1|} \times \cdots \times 2^{|\mathcal{U}_{2N-1}|} + 1$.

Convention. Since paths from the root in an ∀-expansion tree are uniquely determined by the labeling of the edges, i.e. assignments, we treat paths and the union of the appropriate assignments interchangeably.

Definition 2 (∀-expansion). Let $\mathcal{T}$ be a ∀-expansion tree and $P = p_0 \xrightarrow{\tau_1} p_1 \ldots \xrightarrow{\tau_N} p_N$ be a path from the root p_0 to some leaf p_N.

1. For an existential variable x at level $2k$ with $k \in 1..N$ define $\mathscr{E}_{\mathrm{v}}(P, x) = x^{\tau_1,\ldots,\tau_k}$, where $x^{\tau_1,\ldots,\tau_k}$ is a fresh variable.
2. For a propositional formula ξ define $\mathscr{E}(P, \xi)$ as $\xi[\tau_1 \cup \ldots \cup \tau_N][\sigma_R]$ where $\sigma_R = \{\mathscr{E}_{\mathrm{v}}(P, x)/x \mid x \text{ an existential variable}\}$.



(a) $\forall$-expansion tree

$\Psi = \forall u_1 \exists e_1 \forall u_2 \exists e_2 . \psi$	$\mathscr{E}(\mathcal{T}, \Psi)$
$u_1 \vee e_1 \vee \bar{u}_2 \vee e_2$	$e_1^{u_1 \to 0} \vee e_2^{u_1 \to 0, u_2 \to 1}$
$u_1 \vee e_1 \vee \bar{u}_2 \vee \bar{e}_2$	$e_1^{u_1 \to 0} \vee \bar{e}_2^{u_1 \to 0, u_2 \to 1}$
$u_1 \vee \bar{e}_1$	$\bar{e}_1^{u_1 \to 0}$
$\bar{u}_1 \vee \bar{e}_1 \vee \bar{u}_2 \vee \bar{e}_2$	$\bar{e}_1^{u_1 \to 1} \vee \bar{e}_2^{u_1 \to 1, u_2 \to 1}$
$\bar{u}_1 \vee \bar{e}_1 \vee \bar{u}_2 \vee e_2$	$\bar{e}_1^{u_1 \to 1} \vee e_2^{u_1 \to 1, u_2 \to 1}$
$\bar{u}_1 \vee e_1 \vee u_2 \vee \bar{e}_2$	$e_1^{u_1 \to 1} \vee \bar{e}_2^{u_1 \to 1, u_2 \to 0}$
$\bar{u}_1 \vee e_1 \vee u_2 \vee e_2$	$e_1^{u_1 \to 1} \vee e_2^{u_1 \to 1, u_2 \to 0}$

(b) $\forall$-expansion

Figure 3: Example expansion tree and its application

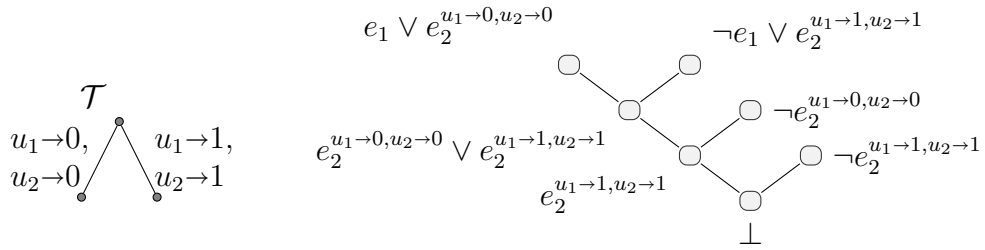
3. Define $\mathscr{E}(\mathcal{T}, \Phi)$ as the union of all $\mathscr{E}(P, \phi)$ for each root-to-leaf path P in $\mathcal{T}$.

Example 2. *Consider again the formula from Example 1 and the expansion tree in Figure 2. The tree has the paths* $P_1 = \{u_1{\to}0\}, \{u_2{\to}0, u_3{\to}0\}$ *and* $P_2 = \{u_1{\to}0\}, \{u_2{\to}1, u_3{\to}1\}$. *It holds that* $\mathscr{E}(P_1, \phi) = (e_1^{u_1 \to 0} \vee e_2^{u_2 \to 0, u_3 \to 0}) \wedge \bar{e}_2^{u_2 \to 0, u_3 \to 0}$ *and* $\mathscr{E}(P_2, \phi) = (\bar{e}_1^{u_1 \to 0} \vee e_2^{u_2 \to 1, u_3 \to 1}) \wedge \bar{e}_2^{u_2 \to 1, u_3 \to 1}$.

Observation 1. *Let P be a root-to-leaf path in an $\forall$-expansion tree $\mathcal{T}$. It holds that $\mathscr{E}(P, \Phi) = \bigcup_{C \in \phi} \mathscr{E}(P, C)$. Consequently, $\mathscr{E}(\mathcal{T}, \Phi)$ can be computed as an expansion of all clauses of the matrix ϕ over all root-to-leaf paths of $\mathcal{T}$.*

Example 3. *Figure 3(a) shows an example of a $\forall$-expansion tree and Figure 3(b) shows a $\forall$-expansion of some formula Ψ based on this tree. The expansion considers both values of u_1 but only the value 1 is considered for u_2 when $u_1 = 0$. The tree has 3 leafs so the formula could potentially grow 3 times. But because the formula is very simple, for each clause C there is only a single path P from the root to some leaf of $\mathcal{T}$ for which $\mathscr{E}(P, C) \neq 1$. Hence, the expansion has the same size as the original formula. Note that there are as many copies of e_2 as there are leafs in the expansion tree ($e_2^{u_1 \to 0, u_2 \to 1}$, $e_2^{u_1 \to 1, u_2 \to 0}$, $e_2^{u_1 \to 1, u_2 \to 1}$) but only two copies of e_1 ($e_1^{u_1 \to 1}$,$e_1^{u_1 \to 1}$); this is because e_1 is at a lower quantification level than e_2.*

Definition 3 ($\forall$Exp+Res). *$\forall$Exp+Res refutation* for Φ is a pair $(\mathcal{T}, \pi)$ where $\mathcal{T}$ is a $\forall$-expansion tree for Φ and π is a resolution refutation for $\mathscr{E}(\mathcal{T}, \Phi)$. A *size* of $(\mathcal{T}, \pi)$, denoted $|(\mathcal{T}, \pi)|$, is the sum of the numbers of nodes in $\mathcal{T}$ and π.



(a) Expansion tree (b) Plain resolution

Figure 4: An example proof in ∀Exp+Res on the prefix $\exists e_1 \forall u_1 u_2 \exists e_2$.

Example 4. *Consider the formula* $\Phi = \exists e_1 \forall u_1 u_2 \exists e_2. (e_1 \vee u_1 \vee u_2 \vee e_2) \wedge (\neg e_1 \vee \neg u_1 \vee \neg u_2 \vee e_2) \wedge \neg e_2$. *Figure 4(a) together with Figure 4(b) form a* ∀*Exp+Res refutation of* Φ.

Remark 2. *In SAT and QBF solving a formula is commonly simplified by the* pure literal rule *[27]. A literal* l *is pure iff* $\overline{l}$ *does not appear in the formula. If a universal literal is pure, then removing all its occurrences from the matrix preserves the truth value of the formula. This is reflected by the* ∀*Exp+Res system. Whenever a formula contains a pure universal literal* l*, it is always sufficient in a* ∀*Exp+Res refutation to consider only one polarity of* $\mathsf{var}(l)$ *(the assignment that sets* l *to 0). Nevertheless, if an expansion tree considers only some assignments, it does* not *mean that there are pure literals. This is illustrated by Example 4. The expansion tree has only 2 branches out of the 4 possible ones. However, none of the universal literals are pure. In fact, any false formula with pure literals can be turned into a formula with no pure literals by adding "dummy" clauses. In contrast, an expansion tree needs to consider only those expansions that are necessary to prove falsity.*

Note that for a ∀-expansion tree $\mathcal{T}$ the size of $\mathscr{E}(\mathcal{T}, \Phi)$ is bounded by the number of leafs of $\mathcal{T}$ times the size of the matrix ϕ. Therefore a ∀Exp+Res refutation can be validated in polynomial time.

Theorem 1. *A formula* Φ *is false iff there exists a* ∀*Exp+Res refutation for* Φ*. Hence, the calculus is sound and refutationally complete.*

Proof. If Φ is false, consider $\mathcal{T}_{\text{full}}$ capturing a full expansion of all of the quantifiers. More precisely, each node p_i of $\mathcal{T}_{\text{full}}$ at depth i (with the root

being at depth 0) has $2^{|\mathcal{U}_{2i+1}|}$ children, each corresponding to a total assignment to variables $\mathcal{U}_{2i+1}$. Since this expansion mirrors semantics of QBF, $\mathscr{E}(\mathcal{T}_{\text{full}}, \Phi)$ is false iff Φ is false.

Throughout the $\forall$-expansion process, (sub-)QBFs $\forall\mathcal{U}\,.\,\Psi$ are replaced with the conjuncts $\Xi = \bigwedge_{\tau\in\omega} \Psi\tau$ for some ω—a set of total assignments to $\mathcal{U}$. Since Ξ is equivalent to $\forall\mathcal{U}\,.\,\Psi$ when ω is the set of *all* assignments, it is weaker if ω is a set of only some total assignments, i.e. $(\forall\mathcal{U}\,.\,\Psi) \to \Xi$. Consequently, $\Phi \to \mathscr{E}(\mathcal{T}, \Phi)$ holds for any $\forall$-expansion tree $\mathcal{T}$. Therefore, if $\mathscr{E}(\mathcal{T}, \Phi)$ is false, then Φ is false. □

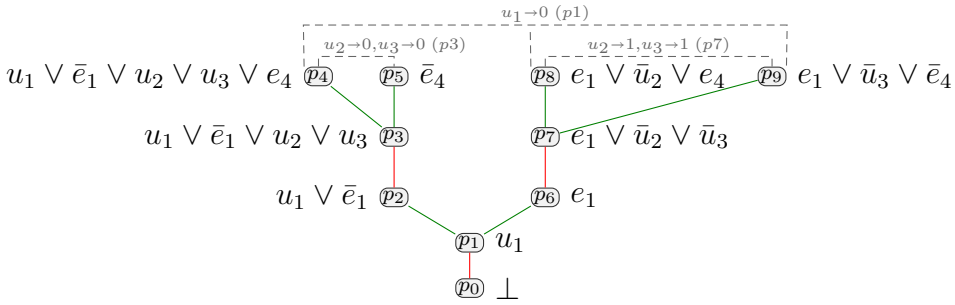
Remark 3. *$\forall$Exp+Res is inspired by the solver* RAReQS *[28, 17], which expands the formula partially, just as $\forall$Exp+Res, and tests whether such partial expansion is sufficient to disprove the formula. Because the solver does not know in advance whether the formula is true or false, it also maintains an expansion of the formula's negation for the case when it is true. In* RAReQS *a partial expansion is referred to as* abstraction*, and, the abstraction is improved by* refinement*. Such refinement corresponds to adding an edge to the expansion tree.* RAReQS *significantly differs from earlier expansion solvers, e.g.* Quantor *[15], which expand in both polarities, i.e. consider the full expansion tree. Consequently, such solvers tend to blowup in memory. In the worst case, however,* RAReQS *may also end up producing the full expansion tree. Nevertheless, experimental analysis show that partial expansion significantly outperforms through expansion on large number of instances [28, 17].*

4. Simulating Tree Q-resolution by $\forall$Exp+Res

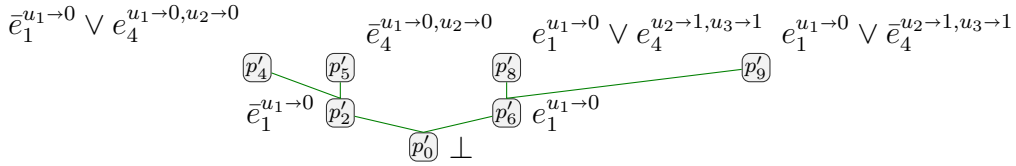
Consider a tree Q-resolution refutation π of Φ. Our objective is to construct a $\forall$Exp+Res refutation $(\mathcal{T}, \pi')$ based on π. We will construct $\mathcal{T}$ and π' so that π' will share its basic structure with π but with universal variables removed and existential variables renamed according to the definition of $\mathscr{E}$.

We observe that if π consists of a single node $\bot$, $\mathcal{T}$ and π' are easily constructed by setting $\mathcal{T}$ to the empty tree and setting π' to $\bot$. Therefore, from now on, we assume that all leafs of π are labeled with nonempty clauses. For the sake of succinctness, in this section, π always refers to the given Q-resolution proof that we wish to translate to a $\forall$Exp+Res refutation $(\mathcal{T}, \pi')$.

We will first look at Figure 5 to illustrate the main ideas of the proof. Figure 5(a) is a tree Q-resolution refutation and Figure 5(b) shows a corresponding $\forall$Exp+Res refutation. Observe that Figure 5(b) maintains the structure of the original proof but the $\forall$-reductions are no longer necessary.



(a) Tree Q-resolution refutation with indicated dependencies on leafs.



(b) ∀Exp+Res refutation with the ∀-expansion tree with branches $\{u_1 \to 0\}, \{u_2 \to 0, u_3 \to 0\}$ and $\{u_1 \to 0\}, \{u_2 \to 1, u_3 \to 1\}$

Figure 5: An example of corresponding proofs on the prefix $\forall u_1 \exists e_1 \forall u_2 u_3 \exists e_4$.

Consider the clause $u_1 \vee \bar{e}_1 \vee u_2 \vee u_3 \vee e_4$. Since the clause contains all universal variables, it immediately tells us that the expansion tree must contain the branch $u_1 \to 0, u_2 \to 0, u_3 \to 0$ since the universal literals of the clause must be assigned to 0. Any other assignment makes the clause true. Once this branch is in the expansion tree, the clause $\bar{e}_4^{u_1 \to 0, u_2 \to 0}$ appears in the expansion and can be used to replicate the resolution step on the node p_2. The clauses p_8 and p_9 require $u_2 \to 1$ and $u_3 \to 1$, respectively. Further, we need to make sure that after the expansion, e_4 has the same annotation in both clauses so that the resolution step on p_7 can be replicated. Hence, there must be a branch containing the assignments $u_2 \to 1, u_3 \to 1$. However, clauses p_8 and p_9 do not give us any hint of how e_1 should be expanded. For such we look at the resolution step on p_1 on the variable e_1. The clause $u_1 \vee \bar{e}_1$ becomes $\bar{e}_1^{u_1 \to 0}$ in the ∀Exp+Res refutation. This means that both p_8 and p_9 must also be expanded with $u_1 \to 0$ so that the resolution step can be replicated.

This example motivates the following approach. For each resolution of clauses $x \vee C_1$ and $\bar{x} \vee C_2$, we need to make sure that the corresponding clauses in the ∀Exp+Res refutation are expanded so that that the variable x is annotated by the same assignment. The literals x and $\bar{x}$ can appear

in the Q-resolution tree π only if they were introduced by some of its leafs. Consequently, the corresponding leafs of the resolution tree π' must contain the same copy of x. So if there are two leafs p_i and p_j that introduce x into the resolution step, we mark down that these leafs must be expanded so that all universal variables with level $< \mathsf{lv}(x)$ are expanded the same. In our example, p_4 and p_5 represent such two leafs due to resolution p_3. The nodes p_8, p_9 are linked due to p_7. The leafs p_4, p_8, p_9 are linked due to the resolution on e_1 in node p_1. These dependencies are shown in Figure 5(a) in dashed lines. Note that the requirements intersect. For instance, the clause $e_1 \vee \bar{u}_2 \vee e_4$ is required to be expanded by $u_3 \to 1$ as well as by $u_1 \to 0$. What we will need to prove is that no such requirements contradict.

This observation motivates the construction. In the first phase of the construction, we identify sets of leafs of π where a certain existential variable must be substituted by the same fresh copy. In the second phase we construct a $\forall$-expansion tree $\mathcal{T}$ that will respect these sets. The $\forall$-expansion tree $\mathcal{T}$ will provide us with the leafs of π'; the resolution steps of π' will correspond to the Q-resolution steps of π. The following definition introduces the concept of a "resolution quadruple", which serves to record the aforementioned dependencies on leafs.

Definition 4 (resolution quadruple). Consider a resolution step in π on some variable x corresponding to nodes p_1 and p_2 with the Q-resolvent node r. Let C_1, C_2, and C_r be the clauses labeling p_1, p_2, and r, respectively. Hence, $C_r = C_1 \cup C_2 \setminus \{x, \bar{x}\}$ (recall that $\forall$-reduction is modeled as a separate step).

Let U be the set of universal literals $l \in C_1 \cup C_2$ such that $\mathsf{lv}(l) < \mathsf{lv}(x)$. Let L be the set of leafs p of π such that there is a path from p to either p_1 or p_2 for which all clauses on the path contain the variable x (including the clause labeling p).

The *resolution quadruple*, corresponding to this resolution step, is the quadruple (r, x, U, L). We write $\mathcal{Q}_\pi$ for the set of all the resolution quadruples corresponding to the resolutions of π.

In the following text we refer to resolution quadruples simply as quadruples (see Examples 5 and 6 for examples of quadruples).

Consider any two leafs p_1, p_2 of π s.t. $p_1, p_2 \in L$ for some $(r, x, U, L) \in \mathcal{Q}_\pi$. Once we ensure that x is replaced with the same fresh copy in the clauses labeling p_1 and p_2, the plain resolution refutation π' is easy to construct. For such construction we assume a mapping M that for each leaf-clause C in π

prescribes a path P of an expansion tree. Using the operator $\mathscr{E}$, we construct a clause $C' = \mathscr{E}(P, C)$, which will be a leaf of π' corresponding to C. Like so we construct all leafs of π'. The rest of π' is constructed by replaying the Q-resolution steps of π as plain-resolution steps. This is formalized in the following proposition. However, the proposition does *not* show that such $\mathcal{T}$ and M *exist*, which will be shown in the rest of the section.

Proposition 1. *Let $\mathcal{T}$ be a $\forall$-expansion tree of Φ and let M be a total mapping from the leafs of π to paths of $\mathcal{T}$. If the following conditions $\mathscr{C}_1$– $\mathscr{C}_3$ hold for $\mathcal{T}$ and M, then there is a resolution refutation π' of $\mathscr{E}(\mathcal{T}, \Phi)$ linear in size of π.*

($\mathscr{C}_1$) *If p is a leaf of π, then $M(p)$ is a path from the root to some leaf in $\mathcal{T}$.*

($\mathscr{C}_2$) *If p is a leaf of π labeled by a clause C, and $M(p) = P$, then P assigns to 0 all universal literals of C.*

($\mathscr{C}_3$) *If leafs p_1, p_2 of π appear in the same L for some quadruple $(r, x, U, L) \in \mathcal{Q}_\pi$, $M(p_1) = P_1$, and $M(p_2) = P_2$, then P_1 and P_2 assign the same values to all universal variables with level $l < \mathsf{lv}(x)$.*

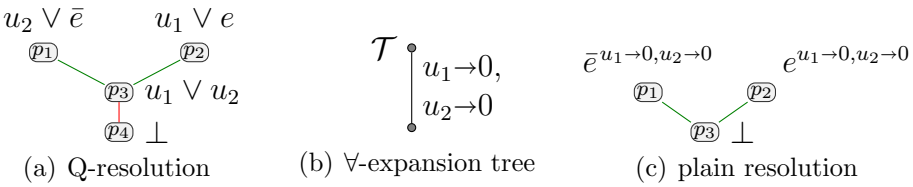
Proof. We construct π' from π in the leaf-to-root direction; during this construction we mark each node of p' in π' as *corresponding* with some node p in π. The construction follows the following rules $\mathscr{R}_l$, $\mathscr{R}_r$, $\mathscr{R}_u$.

($\mathscr{R}_l$) For each leaf p in π labeled with C create a leaf $p' \in \pi'$ labeled with $\mathscr{E}(M(p), C)$; mark p and p' as corresponding.

($\mathscr{R}_r$) Let p_1, p_2, and p_r be nodes that form a resolution step in π over some variable x. That is, C_1 and C_2 label the nodes p_1 and p_2, respectively; $x \in C_1$, $\bar{x} \in C_2$. Consider the corresponding nodes p'_1, p'_2, labeled by C'_1, C'_2, respectively. If there is a variable x^P s.t. $x^P \in C'_1$ and $\bar{x}^P \in C'_2$, create a node r' and label it with the clause $C'_1 \cup C'_2 \setminus \{x^P, \bar{x}^P\}$; mark r' as corresponding to r. (Later we show that there always is such a literal x^P.)

($\mathscr{R}_u$) Let p_1 and p_r be nodes representing a $\forall$-reduction step. That is, p_1 is labeled by some clause C_1 and p_r is labeled by C_r, which is obtained from C_1 by $\forall$-reduction. If p_1 corresponds to p'_1 mark p_r and p'_1 also corresponding.

By induction on resolution depth, we show that the above construction results in a valid resolution tree π'. Additionally we prove, that there is a bijection between the existential literals of corresponding clauses. That is, if C contains an existential literal x, the corresponding clause C' contains



(a) Q-resolution (b) ∀-expansion tree (c) plain resolution

Figure 6: From Q-resolution refutation to ∀Exp+Res refutation

a literal x^P for some P; analogously for literals $\bar{x}$. The consequence of this bijection is that the root of π' must be labeled with the empty clause since the root of π is.

Rule $\mathscr{R}_l$ is well-defined due to conditions ($\mathscr{C}_1$) and ($\mathscr{C}_2$); it establishes the induction hypothesis due to definition of $\mathscr{E}$.

For rule $\mathscr{R}_r$, from induction hypothesis there exists P_1 and P_2 s.t. $x^{P_1} \in C_1'$ and $x^{P_2} \in C_2'$. Since C_1' and C_2' were obtained by valid resolution steps, there must be a path in π' from some leaf p'_{l_1} to p'_1 where all clauses contain the literal x^{P_1}; analogously there a is path in π' from some leaf p'_{l_2} to p'_2 where all clauses contain the literal $\bar{x}^{P_2}$. Both paths correspond to some paths from p_{l_1} to p_1 and p_{l_2} to p_2 in π. Hence, $p_{l_1}, p_{l_2} \in L$ for some $(r, x, U, L) \in \mathcal{Q}_\pi$. Due to condition ($\mathscr{C}_3$), the variable x must be substituted with the same copy in those leafs and therefore also $P_1 = P_2$. Because $x^{P_1} \in C_1'$ and $\bar{x}^{P_1} \in C_2'$, the resolution step on C_1' and C_2' is possible. It remains to be shown that the resolution step does not introduce more than one copy of some literal. Assume that there are literals y^{R_1} and y^{R_2} in C_1' and C_2', respectively. From induction hypothesis, $y \in C_1$ and $y \in C_2$. Consequently, there are some leafs p_{y_1}, p_{y_2} of π s.t. y appears in all clauses on the paths from p_{y_1} to p_1 and from p_{y_2} to p_2. Because π is a refutation proof, y gets eventually resolved away. Therefore there is some $(r_y, y, U_y, L_y) \in \mathcal{Q}_\pi$ for which $p_{y_1}, p_{y_2} \in L_y$ and therefore $R_1 = R_2$ from condition ($\mathscr{C}_3$).

Rule $\mathscr{R}_u$ preserves the induction hypothesis as universal reduction does not modify the set of existential literals. □

Example 5. *Consider* $\forall u_1 u_2 \exists e.\, (u_1 \vee e) \wedge (u_2 \vee \bar{e})$ *with the Q-resolution refutation in Figure 6(a), which induces a single quadruple* $(p_3, e, \{u_1, u_2\}, \{p_1, p_2\})$. *To obtain a ∀Exp+Res refutation, generate the single-branch tree* $\mathcal{T}$ *in Figure 6(b) and mapping* M *with* $M(p_1) = M(p_2) = [u_1 \to 0, u_2 \to 0]$ *yielding the ∀-expansion* $e^{u_1 \to 0, u_2 \to 0} \wedge \bar{e}^{u_1 \to 0, u_2 \to 0}$ *with the corresponding resolution tree Figure 6(c). Observe that conditions* $\mathscr{C}_1$ – $\mathscr{C}_3$ *from Proposition 1 are fulfilled.*

Clauses participating in the Q-resolution step are expanded so that e is replaced with the same copy. The universal literals u_1, u_2 are assigned to 0 by the expansion. Consequently, this Q-resolution step can be reproduced in a plain resolution refutation (Figure 6(c)). Note that universal reduction steps are unnecessary in the resulting plain resolution refutation since expansions remove all universal literals.

4.1. Construction of $\mathcal{T}$ and M

Proposition 1 gives us conditions $\mathscr{C}_1$– $\mathscr{C}_3$ on a $\forall$-expansion tree $\mathcal{T}$ and a mapping M so that any $\mathcal{T}$ and M satisfying these conditions enable us to construct the desired plain-resolution refutation π' for $\mathscr{E}(\mathcal{T}, \Phi)$ from a Q-resolution refutation π. This subsection shows that such $\mathcal{T}$ and M can be constructed for any given tree Q-resolution refutation π. In the following we use some auxiliary concepts introduced by the following definition.

Definition 5. For a quadruple $q = (r, x, U, L) \in \mathcal{Q}_\pi$ we say that q *is at level* $\mathsf{lv}(x)$ and we say that a leaf p of π *is in* q iff $p \in L$.

Recall that if there exists a quadruple $(r, x, U, L) \in \mathcal{Q}_\pi$, the clauses labeling the nodes in L must be expanded so that x is replaced with the same copy in all of them. Further, the assignment used for the expansion must assign to 0 the universal literals in those clauses. This brings about the following question: *If some leaf p of π is in two different quadruples $q_1, q_2 \in \mathcal{Q}_\pi$, how do we ensure that these conditions are not conflicting?* We will be able to show that they are *not* conflicting but for such we will need the assumption that the Q-resolution graph π is a tree.

We say that $(r, x, U, L), (r', x', U', L') \in \mathcal{Q}_\pi$ are *connected* iff $L \cap L' \neq \emptyset$. We say that leafs p_1, p_2 of π *share level* k iff there exists a sequence (with possible repetitions) of quadruples $q_1, \ldots, q_n \subseteq \mathcal{Q}_\pi$, s.t. p_1 is in q_1; p_2 is in q_n; each q_i in the sequence has a level $\geq k$; and each two adjacent quadruples are connected. If two leafs p_1 and p_2 share level k, we write $p_1 \sim_k p_2$.

In the following, we group leafs of π by the relation $\sim_k$. Let us first make a couple of observations about this relation.

Observation 2. *For any $k \in \mathbb{N}$ the relation $\sim_k$ is an equivalence relation on the leafs of π.*

Observation 3. *Let p_1, p_2 be leafs of π. If $p_1 \sim_k p_2$, then $p_1 \sim_l p_2$ for any $l \leq k$. In other words, the equivalence relation $\sim_{k+1}$ is finer than the equivalence relation $\sim_k$.*

Let us look more closely at leafs of π that share some level k. Recall that the given formula Φ has the prefix $\forall\mathcal{U}_1\,\exists\,\mathcal{E}_2\ldots\forall\mathcal{U}_{2N-1}\,\exists\,\mathcal{E}_{2N}$. Consider two connected quadruples $(r,x,U,L),(r',x',U',L')\in\mathcal{Q}_\pi$, both at some level $\geq k$, i.e. $\mathsf{lv}(x)\geq k$ and $\mathsf{lv}(x')\geq k$. Our objective is to build such mapping M that for any two nodes $p_1,p_2\in L$, the paths $M(p_1)$ and $M(p_2)$ share the prefix of length $\mathsf{lv}(x)/2$ corresponding to assignments to variables $\mathcal{U}_1\mathcal{U}_2\ldots\mathcal{U}_{\mathsf{lv}(x)-1}$; this ensures that x is renamed to the same fresh copy in clauses of the leafs L. The same holds for leafs in L'. Since the quadruples are connected, there is some leaf p that belongs to both L and L', i.e. $p\in L\cap L'$. Further, since both x and x' are at a level greater or equal to k, by transitivity, *all* leafs in $L\cup L'$ must be mapped to such paths of the $\forall$-expansion tree $\mathcal{T}$ so that they share their prefixes of length $k/2$, i.e. the prefix that prescribes values to universal variables with lower quantification levels than k. This immediately generalizes to sequences of connected quadruples. If two leafs p_1,p_2 of π share level $k=2l$, then $M(p_1)$ and $M(p_2)$ must have a common prefix of length l, corresponding to assignments to variables $\mathcal{U}_1\mathcal{U}_2\ldots\mathcal{U}_{k-1}$. Further, the assignments $M(p_1)$, $M(p_2)$ must be such that the universal literals with levels $<k$ in the corresponding clauses are all assigned to 0.

This observation motivates Algorithm 1, which is represented as a recursive function. The recursion is initiated by the call $\texttt{Build}(1,2N+1,L_{\text{all}})$ where L_{all} is the set of leafs of π. After this initial call terminates, any root-to-leaf paths with the same labeling in the returned tree are merged to obtain the required $\mathcal{T}$. The function `Build` returns $\mathcal{T}'$, a subtree of the tree $\mathcal{T}$ being constructed, and a mapping M' that maps the given leafs L to paths of $\mathcal{T}'$. The labeling of root-to-leaf paths in $\mathcal{T}'$ are total assignments to variables $\mathcal{U}_k,\mathcal{U}_{k+2},\ldots,\mathcal{U}_{2N-1}$, where k is an odd natural number, i.e. k goes over universal quantification levels.

For the base case of the recursion, i.e. $k=2N+1$, the function creates a single-node tree $\mathcal{T}'$ and maps all given leafs L to an empty path starting and ending in the root of $\mathcal{T}'$.

For the inductive case, the function partitions the given leafs L of π by the relation $\sim_{k+1}$. From the conditions on $\mathcal{T}$, clauses labeling leafs that share level $k+1$ must be expanded such that the universal literals in these clauses with level $\leq k$ are assigned to 0. The function `Build` when invoked as $\texttt{Build}(k,S,L)$, creates assignments to the universal variables at level k. The algorithm visits each partition ρ of the relation $\sim_{k+1}$ and collects quadruples that contain one of the leafs in ρ (line 11). Subsequently, it collects all universal literals at level k that appear in these quadruples and computes

Algorithm 1: Expansion tree construction from $\mathcal{Q}_\pi$

1 **Function** `Build` (k, StopLev, L)
  in : StopLev..base-case level, $k \leq$ StopLev..current level, L..subset of leafs of π
  out: a pair $(\mathcal{T}', M')$, where $\mathcal{T}'$ is an expansion tree for universal variables with level $\geq k$, M' is a mapping from leafs in L to root-to-leaf paths in $\mathcal{T}'$
2 **begin**
3 **if** $k =$ StopLev **then**
4 $\mathcal{T}' \leftarrow$ create a tree with a single node, the root r
5 $M' \leftarrow$ map all nodes in L to the empty path starting in r
6 **return** $(\mathcal{T}', M')$
7 $T' \leftarrow$ a tree with the root node r
8 $M' \leftarrow$ empty mapping
9 $\Xi \leftarrow$ partition nodes L by the relation $\sim_{k+1}$
10 **foreach** $\rho \in \Xi$ **do**
11 $Q_\rho \leftarrow \{q \in \mathcal{Q}_\pi \mid$ there exists $p \in \rho$ in q, q is at level $> k\}$
12 $U_\rho \leftarrow \{l \mid (p, e, U, A) \in Q_\rho, l \in U, \mathsf{lv}(l) = k\}$
13 $\tau_\rho \leftarrow \{u{\rightarrow}0 \mid u \in U_\rho\} \cup \{u{\rightarrow}1 \mid u \notin U_\rho, \mathsf{lv}(u) = k\}$
14 $(\mathcal{T}_\rho, M_\rho) \leftarrow$ `Build`$(k + 2,$ StopLev$, \rho)$
15 connect r to the root of $\mathcal{T}_\rho$ with an edge labeled with τ_ρ
16 if M_ρ maps a leaf $p \in L$ to τ, map p to $\tau_\rho \cup \tau$ in M'
17 **return** $(\mathcal{T}', M')$

an assignment τ_ρ which assigns these literals to 0 and other literals assigns arbitrarily (line 13). Subsequently, a recursive call on the nodes ρ produces a subtree $\mathcal{T}_\rho$. This subtree is integrated into $\mathcal{T}'$ by connecting it to the root with an edge labeled by τ_ρ (line 15). The assignment M_ρ (from L to $\mathcal{T}_\rho$) constructed by the recursive call is augmented by τ_ρ and copied to M' (line 16).

Example 6. *Consider the Q-resolution proof π in Figure 7 with the prefix $\forall u_1 \exists e_2 \forall u_3 \exists e_4$. This tree yields the quadruples depicted on the right hand side of the figure. For $k = 1$, all leafs share level $1 + 1$ and are put into a single partition $\rho = \{p_1, p_2, p_3, p_4\}$ labeled with $[u_1{\rightarrow}0]$. The recursive calls are invoked with $k = 3$. Based on sharing of level $3 + 1$, ρ is split into*

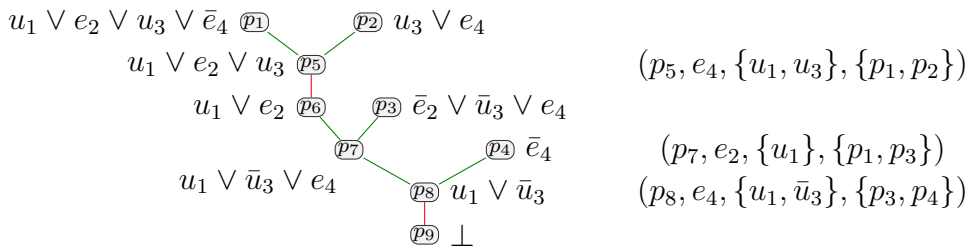


Figure 7: Example Q-resolution and pertaining quadruples $\mathcal{Q}_\pi$

$\{p_1, p_2\}$ *and* $\{p_3, p_4\}$*, which are labeled* $[u_3{\to}0]$ *and* $[u_3{\to}1]$*, respectively. The resulting mapping is* $M(p_1) = M(p_2) = [u_1{\to}0, u_3{\to}0]$ *and* $M(p_3) = M(p_4) = [u_1{\to}0, u_3{\to}1]$.

Let us now focus on the correctness of Algorithm 1. The algorithm is terminating because the set of quadruples $\mathcal{Q}_\pi$ is finite. It needs to be shown that the algorithm constructs mapping M and the tree $\mathcal{T}$ satisfying the conditions $(\mathscr{C}_1)$–$(\mathscr{C}_3)$ of Proposition 1. Observe that the assignment τ_ρ is constructed so that it sets to 0 all the literals in U_ρ. This assignment is then used to annotate the pertaining edge of the $\forall$-expansion tree being constructed. However, for this property to be true, we need to show that the set of literals U_ρ (line 12) does not contain complementary literals—as otherwise such assignment would not exist. For now we assume that this holds and show it later in order to first focus on the overall workings of the algorithm.

The first condition $(\mathscr{C}_1)$ requires that the mapping M assigns to any leaf-node p a path in the constructed expansion tree. Since we are assuming that π has no empty clauses in leafs and all input clauses are $\forall$-reduced, there must exist a quadruple $q \in \mathcal{Q}_\pi$ with p in it. At each level k, quadruples are partitioned so eventually there will be one and only one path P in $\mathcal{T}$ s.t. $M(p) = P$, where P has the length N. Thus satisfying condition $(\mathscr{C}_1)$. (Observe that the recursive sub-calls construct mappings M' to shorter paths, ignoring levels $< k$.)

Condition $(\mathscr{C}_2)$ requires that if a leaf-node p of π is labeled by a clause C, then the assignment $M(p)$ assigns all universal literals of C to 0. If C contains some universal literal l with $\mathsf{lv}(l) = k$, l must be blocked by some existential literal $b \in C$ with $\mathsf{lv}(b) > k$. This literal b is eventually resolved away and therefore there must be a quadruple $q_b = (r, \mathsf{var}(b), U_b, L_b) \in \mathcal{Q}_\pi$ s.t. $p \in L_b$. Since b blocks l on a path from p to some child of r, it also holds that $l \in U_b$.

Hence, once the function `Build` is invoked at level k on a set of leafs L such that $p \in L$, the quadruple q_b will be put in a set Q_ρ, defined on line 11, for the ρ for which $p \in \rho$. Subsequently, l will be placed in the set U_ρ and assigned to 0 by the assignment τ_ρ constructed on line 13. The algorithm places p into a subtree prepended by an edge labeled with τ_ρ thus satisfying condition $(\mathscr{C}_2)$.

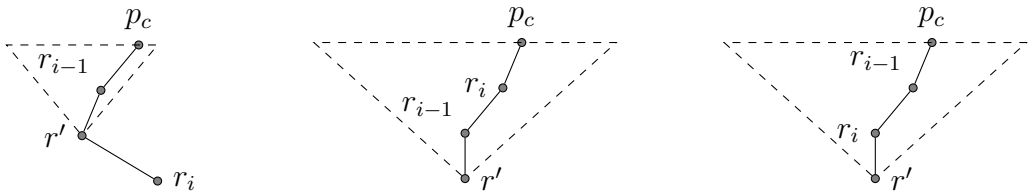
The condition $(\mathscr{C}_3)$ requires that if leafs p_1 and p_2 appear in some quadruple $q = (r, x, U, L)$, the assignments $M(p_1)$, $M(p_2)$ assign the same values to all universal literals with level $< \mathsf{lv}(x)$ (this ensures that x will be replaced with the same copy by $\mathscr{E}$). Since both p_1 and p_2 are in q, they are connected at $\mathsf{lv}(x)$ and therefore automatically they are connected at all levels $\leq \mathsf{lv}(x)$ (Observation 3). In terms of the algorithm, this means that when the algorithm partitions leafs according to $\sim_{k+1}$, the leafs p_1 and p_2 will be put in the same partition ρ for all levels $k < \mathsf{lv}(x)$. And therefore, the paths $M(p_1)$ and $M(p_2)$ will assign the same values to all universal variables with level $< \mathsf{lv}(x)$ thus satisfying condition $(\mathscr{C}_3)$.

As we have established that Algorithm 1 satisfies conditions $(\mathscr{C}_1) - (\mathscr{C}_3)$ of Proposition 1, the tree $\mathcal{T}$ and the mapping M provide us with a $\forall$Exp+Res refutation of the considered formula Φ as long as $\mathcal{T}$ is a valid expansion tree. For such we need to show that the set U_ρ constructed on line 12 does not contain complementary literals. This will be shown in Lemma 6. However, before we reach this lemma, a series of auxiliary lemmas need to be derived.

The following proves hinge on the simple observation that for a quadruple $(r, x, U, L) \in \mathcal{Q}_\pi$ the set U cannot contain complementary literals because Q-resolution does not enable resolving on clauses with more than one complementary literal, and, the set U is defined as the set of universal literals with level $< \mathsf{lv}(x)$ from the union $C_1 \cup C_2$ in the resolution step of $C_1 \vee x$ and $C_2 \vee \bar{x}$.

Observation 4. *For any $(r, x, U, L) \in \mathcal{Q}_\pi$, the literals U are noncontradictory.*

Roughly speaking, in the following we show that for the quadruples grouped in the set Q_ρ, constructed on line 11, there is one quadruple $q_r \in Q_\rho$ that encompasses all the other quadruples in Q_ρ, i.e. q_r corresponds to a resolution step that dominates all the other resolution steps. Further, we show that the set U_r in the quadruple q_r contains all relevant universal literals that appear in Q_ρ, i.e. the set U_ρ constructed on line 12. Observation 4 will enable us to conclude that U_ρ is *not* contradictory.



(a) r_i dom. r' (b) r_{i-1} dom. r_i (c) r_i dom. r_{i-1}

Figure 8: Illustration for the proof of Lemma 3.

We begin by a simple lemma that states that if two quadruples are connected, one of them must correspond to a resolution step dominated by the other one.

Lemma 1. *If any two quadruples* (r_1, x_1, U_1, L_1), $(r_2, x_2, U_2, L_2) \in \mathcal{Q}_\pi$ *are connected, then* r_1 *dominates* r_2*, i.e.* r_2 *is in a subtree of* r_1*, or* r_2 *dominates* r_1*.*

Proof. Since the quadruples are connected, there is some leaf p_c of π s.t. $p_c \in L_1$ and $p_c \in L_2$. At the same time, there is a path from p_c to both r_1 and r_2. If neither r_1 dominated r_2 nor r_2 dominated r_1 there would be an undirected cycle from the root of π to r_1, p_c, r_2, and back to the root. This would be a contradiction with the assumption that π is a tree. □

Lemma 2. *Consider any two quadruples* $(r_1, x_1, U_1, L_1), (r_2, x_2, U_2, L_2) \in \mathcal{Q}_\pi$ *such that* r_1 *dominates* r_2 *and* r_2 *dominates some* $p_l \in L_1$*. Then all the clauses on the path from* r_2 *to* r_1*, except for* r_1*, contain a literal* $b \in \{x_1, \bar{x}_1\}$*.*

Proof. Since the leaf p_l is dominated by both r_1 and r_2, there is a path from p_l through r_2, r_1, and ending in the root of π. Since $p_l \in L_1$, from definition of the quadruples, there is a literal $b \in x_1, \bar{x}_1$ that appears everywhere on the path except for the node r_1. □

The following lemma shows that for any sequence of connected quadruples that are all at some level $\geq k$ there exists one that dominates all in the sequence. Further, its set U contains all relevant universal literals in the sequence.

Lemma 3. *Consider a sequence of quadruples $\gamma = q_1, \ldots, q_n$, such that each $q_i \in \mathcal{Q}_\pi$ in the sequence has a level $\geq k$, and, each two adjacent quadruples are connected. Then, there is a quadruple $(r, x, U, L) \in \gamma$ such that for any quadruple $(r_j, x_j, U_j, L_j) \in \gamma$ the node r dominates r_j, and, each clause on the path from r_j to r, except for r, contains some existential literal with level $\geq k$.*

Proof. Proof by induction on the length of prefix of γ. For the base case choose (r, x, U, L) as q_1. For the inductive case consider $i > 1$ and $q' = (r', x', U', L')$ from the induction hypothesis such that q' satisfies the condition for $q_1, \ldots, q_{i-1}$. Since adjacent quadruples are connected, for $q_i = (r_i, x_i, U_i, L_i)$ and $q_{i-1} = (r_{i-1}, x_{i-1}, U_{i-1}, L_{i-1})$ there is a leaf $p_c \in L_{i-1} \cap L_i$. Split on the following cases.

If q_i is equal to any of the q_j for $j < i$, choose (r, x, U, L) to be q'.

If r_i dominates r' (Figure 8(a)) then invoke Lemma 2 whose preconditions are satisfied because r_i dominates r' and from the induction hypothesis r' dominates p_c (recall that $p_c \in L_{i-1} \cap L_i$). Hence there is a path from r' to one of the antecedents of r_i containing the literal $b \in \{x_i, \bar{x}_i\}$. Note that b does not appear in r_i but it does appear in r'. From induction hypothesis, for any r_j, $j < i$ there is a path from r_j to some antecedent of r' where each clause is blocked by some literal with level $\geq k$. Concatenating the path from r_j to r' with the path r' to r_i satisfies the condition for j. Choose (r, x, U, L) to be q_i.

From Lemma 1, either r_i is dominated by r_{i-1} or r_{i-1} is dominated by r_i. Hence we need to consider only these two remaining cases. If r_{i-1} dominates r_i (Figure 8(b)), then from Lemma 2 there is a $b_{i-1} \in \{x_{i-1}, \bar{x}_{i-1}\}$ that appears on the path from r_i to one of the antecedents of r_{i-1}. From induction hypothesis, there is a path from r_{i-1} to r', excluding r', that contains some existential literals with level $\geq k$. Concatenating this path to the path from r_i to r_{i-1} gives us a path satisfying the required condition for the node r_i.

If r_{i-1} is dominated by r_i (Figure 8(c)) and r_i does not dominate r', then r' must dominate r_i otherwise there would be an unoriented cycle from the root of π to r', r_{i-1}, r_i, and back to the root. From the induction hypothesis, each clause on the path from r_{i-1} to r' contains some existential literal with level $\geq k$. Since r' dominates r_i, which in turn dominates r_{i-1}, the path from r_i to r' is a suffix of the path from r_{i-1} to r' and therefore also satisfies the required condition. Choose (r, x, U, L) to be q'. □

Before we can apply the results derived so far, we have to make a simple but very important observation and that is that any set ρ considered on

line 10 is an equivalence class of $\sim_{k+1}$.

Lemma 4. *For any call* `Build`(k, S, L)*, any set* ρ *considered on line 10 is an equivalence class of* $\sim_{k+1}$ *on the leafs of* π*.*

Proof. Observe that ρ is a subset of an equivalence class of $\sim_{k+1}$ because it is constructed by partitioning the given leafs L by $\sim_{k+1}$.

Proof by induction on k. For $k = 1$, the function `Build` is called with L being the whole set of leafs of π. Hence, any constructed set ρ is an equivalence class of $\sim_{1+1}$.

For the induction step, consider the recursive call `Build`$(k + 2, S, \rho_k)$ for $k \geq 1$. The set ρ_k is partitioned according to $\sim_{k+3}$. From induction hypothesis, ρ_k is an equivalence class of $\sim_{k+1}$. Due to Observation 3, $\sim_{k+1}$ is coarser than $\sim_{k+3}$ and therefore any constructed ρ is an equivalence class of $\sim_{k+3}$. □

Remark 4. *Lemma 4 provides an interesting insight into Algorithm 1. As any relation* $\sim_{k+1}$ *is finer than the relation* $\sim_k$*, one can imagine the equivalence classes forming a tree where the equivalence classes of* $\sim_{k+1}$ *are children of the classes of* $\sim_k$*. The workings of Algorithm 1 can be seen as a traversal of this tree.*

Algorithm 1 defines a set Q^k_ρ, which is parameterized by the set ρ, for which we know that it is an equivalence class of $\sim_{k+1}$ due to Lemma 4. The following lemma shows that Q^k_ρ can be organized in a sequence of connected quadruples. In turn, this will enable us to apply Lemma 3 on the set Q^k_ρ.

Lemma 5. *Consider* ρ *an equivalence class of* $\sim_{k+1}$ *for some odd number* $k > 0$*. Define* $Q^k_\rho \subseteq \mathcal{Q}_\pi$ *as follows.*

$$Q^k_\rho = \{(r, x, U, L) \in \mathcal{Q}_\pi \mid p \in \rho, p \in L, \mathsf{lv}(x) > k\}$$

Then for any $q_a, q_b \in Q^k_\rho$ *there is a sequence of quadruples* $q_1, \ldots, q_m$ *where* $q_a = q_1$*,* $q_b = q_m$*, each* q_i *is at a level* $> k$ *and* $q_i \in Q^k_\rho$*, and each two adjacent* q_i*,*$q_{i+1} \in Q^k_\rho$ *are connected.*

Proof. From definition of Q^k_ρ, there are leafs $p_a, p_b \in \rho$ s.t. $p_a \in q_a$, $p_b \in q_b$. Since ρ is an equivalence class of $\sim_{k+1}$, there is a sequence of connected quadruples $s_1, \ldots, s_n$ such that p_a is in s_1 and p_b is in s_n, and each quadruple in the sequence is at a level $> k$.

Let us denote each s_i as (r_i, x_i, U_i, L_i). Recall that all L_i are nonempty. For each i pick an arbitrary node $p_i \in L_i$. Because all the quadruples s_i in the sequence are connected to one another, $p_a \sim_{k+1} p_i$. Consequently, $p_i \in \rho$ and therefore $s_i \in Q^k_\rho$.

Since q_a and s_1 are connected because of p_a and q_b and q_b are connected because of p_b, constructing the sequence $q_a, s_1, \ldots, s_n, q_b$ yields the required sequence. □

Finally, we can prove the crucial lemma, which shows that the set U_ρ constructed in Algorithm 1 does not contain any complementary literals and therefore provides us with a well-defined assignment τ_ρ.

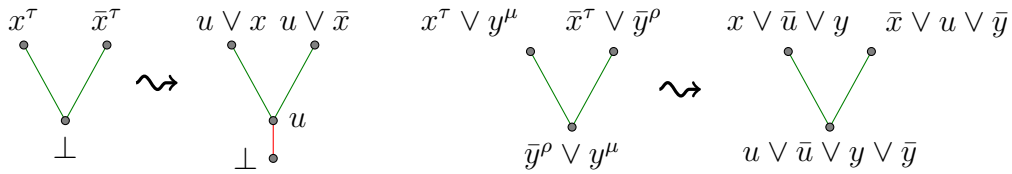
Lemma 6. *Let k, ρ, and Q^k_ρ be defined as in Lemma 5. Define a set of literals U^k_ρ as $U^k_\rho = \{l \mid (r, x, U, L) \in Q^k_\rho, \mathsf{lv}(l) = k, l \in U\}$. The set U^k_ρ does not contain complementary literals.*

Proof. Lemma 5 gives us that Q^k_ρ can be organized into a sequence γ, where each two adjacent quadruples are connected and each $q_i \in \gamma$ is at a level $> k$. From Lemma 3 there is a quadruple $(r_d, x_d, U_d, L_d) \in \gamma$ s.t. for any quadruple $(r_j, x_j, U_j, L_j) \in \gamma$ the node r_d dominates r_j and each of the clauses on the path from r_j to r_d, except for r_d, contains some existential literal with level $> k$. Hence, no universal literals with level $l \leq k$ can be $\forall$-reduced on a path from r_j to r_d in π. Therefore necessarily, U_d contains all literals U_j. Consequently, $U^k_\rho \subseteq U_d$. From Observation 4, the set U_d is noncontradictory and therefore U^k_ρ is also noncontradictory. □

This last lemma concludes the proof of the correctness of Algorithm 1 as it guarantees that the assignment τ_ρ (line 13) is well-defined.

Algorithm 1 operates in time polynomial to the size of π because the size of the set $\mathcal{Q}_\pi$ is linear to the size of π and partitioning by $\sim_{k+1}$ can be computed in polynomial time. This fact, together with Proposition 1 lets us derive the following.

Theorem 2. *For any tree Q-resolution refutation π there exists a $\forall$Exp+Res refutation $(\mathcal{T}, \pi_\mathcal{T})$ s.t. both $\mathcal{T}$ and $\pi_\mathcal{T}$ are polynomial in size of π. This $\forall$Exp+Res refutation can be constructed in time polynomial to π. Hence, $\forall$Exp+Res p-simulates tree Q-resolution.*



(a) Successful (b) Unsuccessful

Figure 9: Reconstruction examples.

Remark 5. *Note that the correctness proof of Algorithm 1 hinges on Observation 4, which tells us that the set of literals U in any quadruple is non-contradictory. Recall that this set is constructed by collecting from the resolvent universal literals with lower levels than the variable being resolved on. Hence, if we consider a modification of Q-resolution that would enable contradictory universal literals in resolvent if their quantification level is higher than the level of the variable resolved on, our proof would still hold. Such extensions of Q-resolution are known and are commonly referred to as* long-distance resolution *[11, 29].*

5. Simulating Restricted ∀Exp+Res by Q-Resolution

A straightforward idea how to obtain a Q-resolution refutation from a ∀Exp+Res refutation is to revert the substitutions made by the expansion operator $\mathscr{E}$. In particular, any variable x^τ appearing in the given ∀Exp+Res refutation would be replaced with its original x and universal literals would be added as in the original clauses of the matrix. This is illustrated by Figure 9(a) where x^τ is replaced with x and the literal u is inserted to the leaf clauses and it is universally reduced at the first opportunity. The reason why this fails in the general case is due to the Q-resolution's restriction disallowing resolving clauses with multiple complementary literals (see Remark 1). What might happen is illustrated by Figure 9(b). Two clauses that are resolved on in a ∀Exp+Res refutation, may contain literals $\bar{y}^\rho$ and y^μ where $\tau \neq \mu$. Here y^ρ and y^μ are distinct variables and therefore the resolvent is *not* tautologous. However, removing the superscript results in a tautology. Similarly, a tautology may arise because of universal literals that are inserted

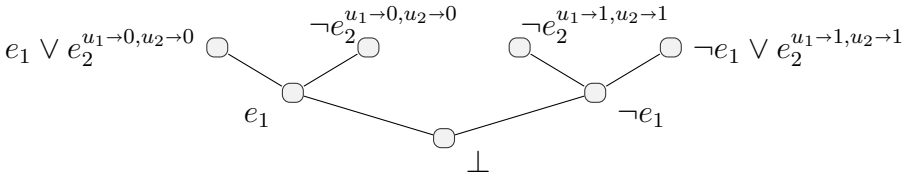


Figure 10: An example of a level-ordered refutation on the prefix $\exists e_1 \forall u_1 u_2 \exists e_2$.

into the leaf clauses (literals u and $\bar{u}$ in the example).[1]

As illustrated by Figure 9, constructing Q-resolution by inverting the $\mathscr{E}$ operation, works only in some cases. This section identifies a certain *fragment* of ∀Exp+Res where it does work. This fragment restricts ∀Exp+Res by allowing resolutions only in a certain order. In particular, it allows only resolutions that follow the order of the quantifier prefix ("inside out"); such refutation will be called level-ordered.

Definition 6 (level-ordered). Let $(\mathcal{T}, \pi)$ be a ∀Exp+Res refutation of a QCNF Φ. We say that $(\mathcal{T}, \pi)$ is *level-ordered* iff the following holds. Let $x^P \vee C_1$ and $\bar{x}^P \vee C_2$ be some clauses resolved in π on x^P, then $\mathsf{lv}(y) \leq \mathsf{lv}(x)$ for any $y^{P_1} \in \mathsf{var}(C_1 \vee C_2)$.

Example 7. *Consider again the formula* $\exists e_1 \forall u_1 u_2 \exists e_2 . (e_1 \vee u_1 \vee u_2 \vee e_2) \wedge (\neg e_1 \vee \neg u_1 \vee \neg u_2 \vee e_2) \wedge \neg e_2$ *from Example 4. The previously shown ∀Exp+Res refutation in Figure 4 is* not *level-ordered because* e_1 *is resolved over before* $e_2^{u_1 \to 1, u_2 \to 1}$ *and* $e_2^{u_1 \to 0, u_2 \to 0}$. *Figure 10 shows an alternative resolution proof, based on the same expansion. This refutation is level-ordered since any* e_2^τ *is resolved away before* e_1.

Note that Definition 6 does *not* impose any order on variables within the same variable block. In particular, any refutation on a single-level or two-level formula $(\exists \mathcal{E}, \forall \mathcal{U} \exists \mathcal{E})$ is level-ordered.

We first observe that in a level-ordered ∀Exp+Res refutation, "branches don't mix", i.e. if a clause contains two variables $x_1^{P_1}$ and $x_2^{P_2}$, the path P_1 must be a prefix of P_2, or the other way around.

[1] While tautologies are always disabled in Q-resolution, tautologies due to universal literals could be considered "worse" because they would lead to unsoundness due to universal reduction, e.g. the clause $\bar{u} \vee u$ is ∀-reduced to $\bot$.

Lemma 7. *Let $(\mathcal{T}, \pi)$ be a level-ordered $\forall$Exp+Res refutation of a QCNF Φ. Let C be some clause in π and $x_1^{P_1}, x_2^{P_2} \in \mathsf{var}(C)$. If $\mathsf{lv}(x_1) \leq \mathsf{lv}(x_2)$, then the path P_1 is a prefix of the path P_2.*

PROOF. By induction on the number of resolution steps that led to C. The condition is true for the leafs of π from the definition of $\mathscr{E}$. For the induction step consider clauses $C_1 \vee \bar{x}_r^P$ and $C_2 \vee x_r^P$ with the resolvent $C = C_1 \vee C_2$. If C is empty or unit, the condition is trivially satisfied. Let $x_1^{P_1}, x_2^{P_2} \in \mathsf{var}(C)$ with $\mathsf{lv}(x_1) \leq \mathsf{lv}(x_2)$. Because π is level-ordered, $\mathsf{lv}(x_1) \leq \mathsf{lv}(x_r)$ and $\mathsf{lv}(x_2) \leq \mathsf{lv}(x_r)$, from which the induction hypothesis gives that both paths P_1 and P_2 are prefixes of the path P. Since $\mathsf{lv}(x_1) \leq \mathsf{lv}(x_2)$, then $|P_1| \leq |P_2|$ from definition of $\mathscr{E}$. Hence the path P_1 is a prefix of the path P_2. □

Lemma 8. *Let $(\mathcal{T}, \pi)$ be a level-ordered $\forall$Exp+Res refutation of a QCNF Φ. Let C be a clause in π and $x^{P_1}, x^{P_2} \in \mathsf{var}(C)$, then $P_1 = P_2$.*

PROOF. Immediate consequence of Lemma 7. □

The following theorem shows that for level-ordered ∀Exp+Res refutation, Q-resolution refutations can be obtained by "removing superscripts".

Theorem 3. *Let $(\mathcal{T}, \pi)$ be a level-ordered $\forall$Exp+Res refutation of $\Phi = \mathcal{P} \,.\, \phi$. Then a Q-resolution refutation of Φ can be constructed in polynomial time with respect to $|(\mathcal{T}, \pi)|$. Hence, Q-resolution p-simulates level-ordered $\forall$Exp+Res.*

PROOF. This proof follows a similar construction as was done for constructing ∀Exp+Res refutations from Q-resolution refutations in Proposition 1 but in the opposite direction.

Construct a Q-resolution refutation π' as follows. For each leaf p in π labeled with a clause C, there exists a path P from the root to some leaf in $\mathcal{T}$ and a clause $C' \in \phi$ such that $\mathscr{E}(P, C') = C$. Replace C with C'. Whenever there is a resolution on some variable x^P in π, perform resolution on x in π'. Add ∀-reduction steps after each resolution step whenever possible.

To show that π' is a valid Q-resolution proof, we prove the following. From construction of π', each leaf or a ∀-reduced clause C' in π' corresponds to some clause C in π and the following conditions are satisfied. (1) An existential literal l' with $\mathsf{var}(l') = x$ is in C' iff there is a literal $l \in C$ with the same polarity as l' and with $\mathsf{var}(l) = x^P$ for some path P of $\mathcal{T}$.

(2) Let $x^P \in \mathsf{var}(C)$, be such that $\mathsf{lv}(x) \geq \mathsf{lv}(y)$ for all $y^{P'} \in \mathsf{var}(C)$. Then for any universal literal $l' \in C'$, $P(l') = 0$.

Proof by induction on the number of resolution steps that led to C'. The conditions are immediately satisfied for the leaf clauses by the definition of $\mathscr{E}$. For the induction step consider clauses C'_1 and C'_2 being resolved on some variable x_r in π'. From construction of π', there exist corresponding clauses C_1 and C_2 in π resolved on some variable x_r^P. Assume WLOG $x_r^P \in C_1$ and $\bar{x}_r^P \in C_2$. First we show that the Q-resolution step is valid for C'_1 and C'_2; second we show that the induction hypothesis is preserved by this Q-resolution step. From induction hypothesis, $x_r \in C'_1$ and $\bar{x}_r \in C'_2$. Therefore C'_1 and C'_2 can be resolved on x_r provided that there is no other complementary literals in them. For contradiction, assume there are two complementary literals $y \in C'_1$ and $\bar{y} \in C'_2$, where $x_r \neq y$.

First assume that y is existential. From induction hypothesis there are literals $y^{R_1} \in C_1$ and $\bar{y}^{R_2} \in C_2$. Since C_1 and C_2 contain only one pair of complementary literals (x_r^P and $\bar{x}_r^P$), it must be that $R_1 \neq R_2$. This is a contradiction with Lemma 8 because their resolvent would contain the variables y^{R_1} and y^{R_2}.

Now assume that y is universal. The contradiction derives from the induction hypothesis that P assigns 0 to all universal literals in C'_1 and C'_2. In particular, it assigns 0 to both the literals y and $\bar{y}$, which is a contradiction as an assignments can give only a single value to a variable. (Note that $\mathsf{lv}(y) < \mathsf{lv}(x_r)$ as otherwise y would be $\forall$-reduced.)

Now we show that conditions (1) and (2) are preserved by resolution steps. Resolving C'_1 and C'_2 yields the clause $C'_u = C'_1 \cup C'_2 \setminus \{x_r, \bar{x}_r\}$, which is subsequently $\forall$-reduced to C'. The clause C' corresponds to $C = C_1 \cup C_2 \setminus \{x_r^P, \bar{x}_r^P\}$ in π.

To show condition (1) we need to find a corresponding literal for each existential literal $l' \in C'$ and conversely, a corresponding literal for each literal $l \in C$. Let $l' \in C'$ be an existential literal where $\mathsf{var}(l') = x_l$. It holds that $x_l \neq x_r$ and l' belongs to one of the clauses C'_1, C'_2. From induction hypothesis, there is a literal l in the corresponding C_1 or C_2 s.t. $\mathsf{var}(l) = x_l^R$ for some R. Hence, l is in C. For the converse, consider a literal $l \in C$ where $\mathsf{var}(l) = x_l^R$. Then $x_l \neq x_r$ due to Lemma 8 and because both x_r^P and $\bar{x}_r^P$ are removed in the resolution step. Since l belongs to one of C_1 or C_2, there is a corresponding l' in the corresponding C'_1 or C'_2 from induction hypothesis.

The condition (2) holds trivially for an empty C'. If C' is nonempty, C is also nonempty and there is a variable $z^R \in \mathsf{var}(C)$ s.t. $\mathsf{lv}(z) \geq \mathsf{lv}(y)$

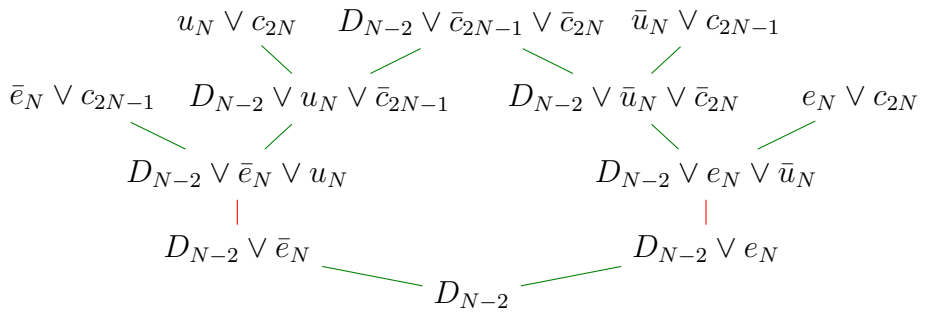


Figure 11: Q-resolution proof fragment (where $D_{n-2} = \bar{c}_1 \vee \cdots \vee \bar{c}_{2n-3} \vee \bar{c}_{2n-2}$)

for all $y^{R'} \in \mathsf{var}(C)$. It must be that R is a prefix of the path P due to Lemma 7 (recall that the resolution is on x^P). From induction hypothesis, P assigns 0 to all universal literals in $C'_1 \cup C'_2$. Hence R assigns 0 to all universal literals from $C'_1 \cup C'_2$ with level $l < \mathsf{lv}(z)$; all the other universal literals are $\forall$-reduced. □

6. ∀Exp+Res Does Not Simulate Q-resolution

Section 4 shows that ∀Exp+Res p-simulates Q-resolution when Q-resolution is restricted to tree refutations. This section shows that this result is "tight", i.e. once a Q-resolution takes general non-tree DAG form, it is *not* p-simulated by ∀Exp+Res. For such we construct the following formula parameterized by a natural number n.[2]

$$\begin{array}{l}\exists e_1 \forall u_1 \exists c_1 c_2 \ldots \exists e_n \forall u_n \exists c_{2n-1} c_{2n}. \\ \bigwedge_{i \in 1..n} (\bar{e}_i \vee c_{2i-1}) \wedge (\bar{u}_i \vee c_{2i-1}) \wedge (e_i \vee c_{2i}) \wedge (u_i \vee c_{2i}) \wedge \\ \left(\bigvee_{i \in 1..2n} \bar{c}_i\right)\end{array} \tag{2}$$

Observe that (2) has size linear in n. In particular, it has $4n{+}1$ clauses and $4n$ variables. Each of the quadruples of clauses encodes the implications $(e_i \vee u_i) \rightarrow c_{2i-1}$ and $(\bar{e}_i \vee \bar{u}_i) \rightarrow c_{2i}$. This means that for either of the values of e_i, the variable u_i can be set so that both c_{2i-1} and c_{2i} must be true. This leads to a contradiction due to the last clause $\bigvee_{i \in 1..2n} \bar{c}_i$. First we show that (2) has a short Q-resolution refutation.

[2] See http://sat.inesc-id.pt/~mikolas/expansion13/ for a formula generator.

Proposition 2. *Formula (2) has a Q-resolution refutation comprising linear number of resolution steps w.r.t. n.*

PROOF (SKETCH). Starting from the clause $\bigvee_{i\in 1..2n} \bar{c}_i$, the literals $\bar{c}_i$ are gradually resolved away from the highest to the lowest level. Figure 11 shows how the variables c_{2n-1}, c_{2n} are resolved away. Subsequently, the variables $c_{2(n-1)-1}, c_{2(n-1)}$ are resolved away in an analogous fashion. This process continues until the empty clause is obtained. □

In the second half of this section we show that there are no short ∀Exp+Res refutations of (2). Moreover, we show that any ∀Exp+Res refutation requires a full expansion of the formula.

Proposition 3. *Any ∀Exp+Res refutation of (2) is exponential in size in n.*

PROOF. Let us look at the expansion of the first universal variable, i.e. the variable u_1. We show that both of the values are needed in the expansion. Expanding u_1 yields the two following subformulas, one for $u_1{\to}1$ and one for $u_1{\to}0$ (for now other variables are left unexpanded).

$$\begin{array}{l}\exists e_2 \forall u_2 \exists c_3 c_4 \ldots \exists e_n \forall u_n \exists c_{2n-1} c_{2n}.\\ (\bar{e}_1 \vee c_1^{u_1\to 1}) \wedge (c_1^{u_1\to 1}) \wedge (e_1 \vee c_2^{u_1\to 1}) \wedge\\ \bigwedge_{i\in 2..n} (\bar{e}_i \vee c_{2i-1}) \wedge (\bar{u}_i \vee c_{2i-1}) \wedge (e_i \vee c_{2i}) \wedge (u_i \vee c_{2i}) \wedge\\ \left(\bar{c}_1^{u_1\to 1} \vee \bar{c}_2^{u_1\to 1} \vee \bigvee_{i\in 3..2n} \bar{c}_i\right)\end{array} \tag{3}$$

$$\begin{array}{l}\exists e_2 \forall u_2 \exists c_3 c_4 \ldots \exists e_n \forall u_n \exists c_{2n-1} c_{2n}.\\ (\bar{e}_1 \vee c_1^{u_1\to 0}) \wedge (c_2^{u_1\to 0}) \wedge (e_1 \vee c_2^{u_1\to 0}) \wedge\\ \bigwedge_{i\in 2..n} (\bar{e}_i \vee c_{2i-1}) \wedge (\bar{u}_i \vee c_{2i-1}) \wedge (e_i \vee c_{2i}) \wedge (u_i \vee c_{2i}) \wedge\\ \left(\bar{c}_1^{u_1\to 0} \vee \bar{c}_2^{u_1\to 0} \vee \bigvee_{i\in 3..2n} \bar{c}_i\right)\end{array} \tag{4}$$

The original formula (2) is equivalent to $\exists e_1 c_1^{u_1\to 0} c_2^{u_1\to 0} c_1^{u_1\to 1} c_2^{u_1\to 1}.\,(4)\wedge(3)$. Or equivalently, (2) is false iff (4) ∧ (3) is unsatisfiable.

The subformula (3) is satisfiable because e_1 can be set to 1, $c_2^{u_1\to 1}$ to 0 and the rest of the existential variables to 1. Similarly, the subformula (4) is satisfiable when e_1 is set to 1. Since both formulas (3) and (4) are satisfiable, both of them are needed to show unsatisfiability. Consequently, any ∀Exp+Res refutation must expand the variable u_1 in both of its values.

Consider expansions ϕ_1 and ϕ_2 of (4) and (3), respectively, such that $\phi_1 \wedge \phi_2$ is unsatisfiable. The only variable ϕ_1 and ϕ_2 share is e_1 because

a_1	$\cdots$	a_1	$\cdots$	a_N	$\cdots$	a_N
b_1	$\cdots$	b_N	$\cdots$	b_1	$\cdots$	b_N

Figure 12: Completion principle

universal variables are expanded away and all existential variables are labeled with $u_1{\to}1$ and $u_1{\to}0$, respectively. From *Craig's interpolation theorem* [30], there is an interpolant I using only variables common to ϕ_1 and ϕ_2, i.e. e_1, and it holds that $\phi_1 \Rightarrow I$ and $\phi_2 \Rightarrow \neg I$. Equivalently, $\phi_1 \wedge \neg I$ and $\phi_2 \wedge I$ are unsatisfiable. Since $\phi_1 \wedge e_1$ is satisfiable, I must be e_1 and $\phi_1 \wedge \bar{e}_1$ must be unsatisfiable. Further, $\phi_1 \wedge \bar{e}_1$ is weaker than (3) $\wedge\, \bar{e}_1$, which can be rewritten as follows.

$$\begin{array}{l}(c_1^{u_1\to 1}) \wedge (c_2^{u_1\to 1}) \wedge \\ \exists e_2 \forall u_2 \exists c_3 c_4 \ldots \exists e_n \forall u_n \exists c_{2n-1} c_{2n}. \\ \bigwedge_{i\in 2..n} (\bar{e}_i \vee c_{2n-1}) \wedge (\bar{u}_i \vee c_{2n-1}) \wedge (e_i \vee c_{2n}) \wedge (u_i \vee c_{2n}) \wedge \\ \left(\bigvee_{i\in 3..2n} \bar{c}_i\right)\end{array} \tag{5}$$

Since $c_1^{u_1\to 1}, c_2^{u_1\to 1}$ appear in only the first part of (5), the second part of the formula has to be false (i.e. unsatisfiable after full expansion). The second part of (5) is in the same form as the initial formula (2) just with 4 fewer variables (the variables e_1, u_1, c_1, and c_2). Hence, by renaming we obtain a formula of the form (2) for $n-1$. Repeating the same arguments as above, (5) must be fully expanded in order to ensure unsatisfiability.

Following the same argument for (4) $\wedge\, e_0$ shows that the proof needs to expand each variable in both polarities at all depths. Thus yielding an exponential expansion of the given formula. □

Hence, Proposition 3 and Proposition 2 give us an exponential separation between Q-resolution and ∀Exp+Res.

Theorem 4. *∀Exp+Res does not p-simulate Q-resolution.*

7. Level-ordered Q-resolution does not p-simulate ∀Exp+Res

This section aims to show certain strength of ∀Exp+Res, which sheds some light on the fact expansion-based solvers perform on certain instances

better than DPLL-based solvers [31]. DPLL-based solvers assign values to variables from left to right in the order of the prefix with the exception of values obtained by unit propagation. However, if there is no unit propagation that "goes over levels", the generated proofs follow the prefix of the formula, inside-out.[3] Hence, we introduce the following restriction on Q-resolution.

Definition 7. Let π be a Q-resolution refutation of a QCNF Φ. We say that π is *level-ordered* iff the following holds. Let $x \vee C_1$ and $\bar{x} \vee C_2$ be some clauses resolved in π, then $\mathsf{lv}(y) \leq \mathsf{lv}(x)$ for any existential variable $y \in \mathsf{var}(C_1 \vee C_2)$.

As in level-ordered ∀Exp+Res refutation (see Definition 6), any Q-resolution is level-ordered for prefixes $\exists \mathcal{E}$ and $\forall \mathcal{U} \exists \mathcal{E}$.

It is well-known that proof may grow exponentially for plain resolution when a certain order is imposed [32]. Hence, one could reuse one such formula and impose a certain order by inserting dummy universal quantifiers (as many as there are original variables) and thus show a separation between level-ordered Q-resolution and general ∀Exp+Res. As this is a very convoluted example, it is not very interesting. We show, instead, that ∀Exp+Res is already more powerful than level-ordered Q-resolution for formulas with only 3 levels of quantification.

Our construction derives from a principle which we name *completion principle*. We consider two sets $A = \{a_1, \ldots, a_n\}$, $B = \{b_1, \ldots, b_n\}$ and their cross-product $A \times B$. Let us visualize the cross-product as in Figure 12. The following game is played. In the first round, the ∃-player deletes one and only one cell from each column. In the second round, the ∀-player chooses one of the two rows. The ∀-player wins if the chosen row contains either the complete set A or the set B. Observe that there is a winning strategy for the ∀-player. If the ∃-player wants to make sure that the bottom row (the b_i row) does *not* contain the complete set B, it must delete at least one element from each of the n copies of B. Hence, for the j-th copy of B there is an element a_j that was not deleted and thus forming the complete set A. Hence, the winning strategy for the ∀-player is to look at the bottom row and see if

[3] If a solver were to propagate across levels on the formula studied here, there would have to be a clause of the form $C = C_x \vee l$, with C_x being literals in the 1^{st} level and $\mathsf{lv}(l) = 3$ (with $z \notin \mathsf{var}(C)$). No such clause is in the original formula and it can be proven by induction that no such clause can be derived by Q-resolution and therefore, it is never learned.

it contains a complete copy of B. If it does, the $\forall$-player selects the bottom row and otherwise he selects the top row.

Let us construct a formula based on the completion principle. For each column i we introduce a variable x_i that determines which cell is deleted by the $\exists$-player in the first round. For the $\forall$-player we introduce a single universal variable z that determines the selected row. Further, we add a clauses that make sure that whenever one of the sets A or B is complete, the formula becomes false.

We construct the formula $\mathtt{CR}_n$ as follows. The prefix is the following. $\exists x_{(1,1)} \dots x_{(n,n)} \forall z \exists a_1 \dots a_n b_1 \dots b_n$ The matrix consists of the conjunction of the two following parts.

$$x_{ij} \vee z \vee a_i,\ i,j \in 1..N \tag{6}$$

$$\bar{x}_{ij} \vee \bar{z} \vee b_j,\ i,j \in 1..N \tag{7}$$

$$\bigvee_{i \in 1..N} \bar{a}_i \tag{8}$$

$$\bigvee_{i \in 1..N} \bar{b}_i \tag{9}$$

The first two types of clauses (6) and (7) represent the completion principle. The last two types of clauses (8) and (9) disables setting all a_i and b_1 to true, respectively. Hence, the whole formula $\mathtt{CR}_n$ is false because once z is set according to the strategy outlined above, the variables $A \cup B$ must be set such that variables from one of the sets A and B will be all true. Consequently, one of the clauses (8) and (9) must be falsified.

First we show that $\mathtt{CR}_n$ has short refutation in $\forall$Exp+Res.

Proposition 4. *$\mathtt{CR}_n$ has a $\forall$Exp+Res refutation polynomial in size in n*

Proof (sketch). Expand z in both values. The expansion yields the clauses $(x_{(i,j)} \vee a_i^{z \to 0}) \wedge (\bar{x}_{(i,j)} \vee b_i^{z \to 1})$, from which we derive the clauses $a_i^{z \to 0} \vee b_j^{z \to 1}$ for $i,j \in 1..n$. (n^2 resolution steps). From clauses (8) and (9) we obtain $C_a = \bigvee_{i \in 1..n} \bar{a}_i^{z \to 0}$ and $C_b = \bigvee_{i \in 1..n} \bar{b}_i^{z \to 1}$.

For all $j \in 1..n$, using the clauses $a_i^{z \to 0} \vee b_j^{z \to 1}$ for $i \in 1..n$ and the clause C_a derive the unit clause $b_j^{z \to 1}$. (n^2 resolution steps) Together with the unit clauses b_i, the clause C_b leads to the empty clause. (n resolution steps) $\square$

We will show that the formula $\mathtt{CR}_n$ does *not* have short refutations in level-ordered Q-resolution.

Lemma 9. *Let π be a level-ordered Q-resolution refutation of $\mathtt{CR}_n$. Let C be a clause in π such that it does not contain any a_i or b_i variables and let π_C be a DAG rooted in C, i.e. π_C is a proof of C. The proof π_C contains one of the clauses (8), (9).*

PROOF. Construct a path P that starts at C and goes to some leaf of π_C. The path P is constructed so that $\bar{a}_i$ (resp. $\bar{b}_i$) is followed whenever a_i (resp. b_i) is resolved on. Since all axiom clauses (6) and (7) contain a_i or b_i positively, one of (8), (9) must be at the end of P. □

Lemma 10. *Let P be a path from the root of π that contains clauses with x_{ij} variables only and is maximal in that respect, i.e. P cannot be extended by a clause that would have only x_{ij} variables. Let C be the last clause on the path P. The clause C contains a variable x_{ij} for each $i \in 1..N$.*

PROOF. Note that P must exists because at least $\bot$ satisfies the condition that it contains only x_{ij} variables. Let D_1 and D_2 be the antecedents of C and let π_C be the derivation of C. (Note that the antecedents must exists because all the axiom clauses contain a_i,b_i variables.) From the construction of P, the clauses D_1 and D_2 contain some variable y with $y = a_i$ or $y = b_i$ for some $i \in 1..N$ and C does not contain this variable. Therefore, the resolution step of D_1 and D_2 is over the variable y. Due to the order condition on the resolution steps in π, the derivation π_C does not contain any resolution steps over the x_{ij} variables.

From Lemma 9 there is one of (8), (9) in π_C. W.L.O.G. let (8) be in π_C. The clause (8) introduces $\bar{a}_i$ for each $i \in 1..N$. Since C does not contain any $\bar{a}_i$, all these must have been resolved away. As the only clauses containing the positive literals a_i are clauses of type (6), a variable x_{ij} must be introduced for each $i \in 1..N$. Since no x_{ij} variables could have been resolved away in π_C, all the introduced x_{ij} literals must appear also in C. □

Proposition 5. *The refutation π is exponential in N.*

PROOF. Pick an assignment τ to all variables x_{ij}. Construct a path P as follows. Start from $P = \bot$. If the end of P is derived by a resolution step over some variable x_{ij}, extend P with the antecedent that contains the literal l s.t. $\tau(l) = 0$ and $\mathsf{var}(l) = x_{ij}$. If the end of P is derived by a resolution over some a_i, b_i variable, stop. If there is a $\forall$-reduction step, simply extend P.

Due to Lemma 10, the path P ends with a clause C_τ that contains N different variables x_{ij} and $\tau(C_\tau) = 0$.

There are 2^{N*N} different assignments τ to the x_{ij} variables and each clause C_τ covers at most 2^{N*N-N} assignments. Hence there must be at least $2^{N*N}/2^{N*N-N} = 2^N$ clauses C_τ altogether. □

Corollary 1. *Level-ordered Q-resolution does not p-simulate ∀Exp+Res even with bounded number of quantification levels.*

Remark 6. *We should note that real conflict-driven DPLL solvers indeed produce exponential refutations of* $\mathtt{CR}_n$*. In contrast, expansion-based solver refute the formula quickly.*[4]

8. Conclusions and Future Work

This article introduces and studies a proof system ∀Exp+Res aimed at refuting false QCNFs based on a combination of two techniques: *expansion* of universal variables and *propositional resolution*. In order to mitigate the exponential blowup of expansion, we introduce *partial* expansions, which permit considering only certain polarities of the variable being expanded.

The application of expansions for producing a propositional formula is an attractive means of solving QBF as it enables the use of modern SAT technology. Indeed, a number of QBF solvers tackle a given formula by the combination of expansion and SAT solving. The solvers QUBOS [14], Nenofex [16], Quantor [15] expand universal variables from inner- to outermost levels. However, these expansions are possibly interleaved with operations for removal of existential quantifiers. (Nevertheless, Quantor has an option that forces it to expand universal variables only.) It is the subject of future work to develop proof systems that would characterize these solvers.

The solver sKizzo [33] expands all universal quantifiers as is done in ∀Exp+Res (even though the process is called Skolemization) but it does not apply partial expansions. In contrast, the solver RAReQS [28, 17] constructs a partial expansion, which is gradually being augmented (refined) until it becomes sufficient. Hence, out of the mentioned solvers, the workings of RAReQS are closest to ∀Exp+Res as it is the only one considering

[4] See http://sat.inesc-id.pt/~mikolas/expansion13/ for a formula generator and also proofs generated from a DPLL QBF solver.

partial expansions. It should be noted that there is a price for the use of partial expansions as the solver "wastes" effort on partial expansions that are insufficient to show falsity.[5]

The formalization of ∀Exp+Res enables us to further our understanding of the difference between expansion-based QBF solvers and DPLL solvers, whose responses can be certified by Q-resolution. On the positive side, the article shows that any *tree* Q-resolution refutation is p-simulated by ∀Exp+Res (Section 4). This result is shown to be "tight", i.e. *non-tree* Q-resolution cannot be p-simulated by ∀Exp+Res (Section 6). In the opposite direction, the article shows that Q-resolution can p-simulate ∀Exp+Res if the underlying propositional resolution refutation follows a certain *order of variables* (Section 5). It is the subject of future work to show whether or not this result is tight.

To show an advantage of expansion-based solving over DPLL solving, we consider a restriction of Q-resolution, which requires variables to be resolved inside out with the respect to the prefix (Section 7). We show that once this restriction is imposed, Q-resolution does not p-simulate ∀Exp+Res. What makes our result particularly interesting is that we show that such separation already happens for formulas with only 3 quantification levels. Effectively this means that the existential variables of the formula are split into two groups (the 1^{st} and the 3^{rd} quantification level) and it is required that variables from the second group are resolved before the variables from the first group. Such restriction already gives an exponential advantage to ∀Exp+Res over Q-resolution. Hence, this result is different from the standard separation result for *DPLL propositional resolution*, where a total order on the variables is required [32], and the separation result for *regular propositional resolution*, where variables may not repeat [34]. To show the last result, we introduced a principle which we call "completion principle", which takes advantage of the fact that QBF can be seen as a game between ∀-player and ∃-player. To our best knowledge, no such principles were introduced in the context of QBF.

This article opens a number of avenues for future research. It is open whether or not ∀Exp+Res can be p-simulated by Q-resolution. On the other hand, as it was shown that ∀Exp+Res does not p-simulate non-tree Q-resolution, a natural question to ask is how can this gap be bridged? For

[5]The solver is also constructing a partial expansion of the negation of the formula for the case when the formula is true.

further understanding of the workings of QBF solvers, modifications of Q-resolution and ∀Exp+Res should be considered (we have already touched upon long-distance Q-resolution in Remark 5). Last but not least, the introduction of the completion principle motivates investigation of QBF classes of formulas with underlying principles that lead to interesting behavior of the proof systems in question.

Acknowledgments.

We would like to thank Uwe Egly and Will Klieber for various helpful conversations on QBFs. We would also like to thank the anonymous reviewers for their stimulating feedback. This work is partially supported by SFI PI grant BEACON (09/IN.1/I2618), FCT grants ATTEST (CMU-PT/ELE/0009/2009) and POLARIS (PTDC/EIA-CCO/123051/2010), and multiannual PIDDAC program funds (PEst-OE/EEI/LA0021/2011).

[1] S. Oliva, On the complexity of resolution-based proof systems, Ph.D. thesis, Departament de Llenguatges i Sistemes Informàtics Universitat Politècnica de Catalunya (Mar. 2013).

[2] M. Alekhnovich, A. A. Razborov, Satisfiability, branch-width and Tseitin tautologies, Computational Complexity 20 (4) (2011) 649–678.

[3] S. A. Cook, R. A. Reckhow, The relative efficiency of propositional proof systems, J. Symb. Log. 44 (1) (1979) 36–50.

[4] J. Krajíček, P. Pudlák, Quantified propositional calculi and fragments of bounded arithmetic, Mathematical Logic Quarterly 36 (1) (1990) 29–46.

[5] H. K. Büning, M. Karpinski, A. Flögel, Resolution for quantified Boolean formulas, Inf. Comput. 117 (1) (1995) 12–18.

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

[7] U. Egly, On sequent systems and resolution for QBFs, in: Cimatti and Sebastiani [35], pp. 100–113.

[8] A. Van Gelder, Contributions to the theory of practical quantified Boolean formula solving, in: M. Milano (Ed.), CP, Vol. 7514, Springer, 2012, pp. 647–663.

[9] J. Rintanen, Improvements to the evaluation of quantified Boolean formulae, in: T. Dean (Ed.), IJCAI, Morgan Kaufmann, 1999, pp. 1192–1197.

[10] M. Cadoli, M. Schaerf, A. Giovanardi, M. Giovanardi, An algorithm to evaluate quantified Boolean formulae and its experimental evaluation, J. Autom. Reasoning 28 (2) (2002) 101–142.

[11] L. Zhang, S. Malik, Conflict driven learning in a quantified Boolean satisfiability solver, in: ICCAD, 2002, pp. 442–449.

[12] F. Lonsing, A. Biere, DepQBF: A dependency-aware QBF solver, JSAT 7 (2-3) (2010) 71–76.

[13] E. Giunchiglia, P. Marin, M. Narizzano, QuBE 7.0 system description, Journal on Satisfiability, Boolean Modeling and Computation 7 (2010) 83–88.

[14] A. Ayari, D. A. Basin, QUBOS: Deciding quantified Boolean logic using propositional satisfiability solvers, in: M. Aagaard, J. W. O'Leary (Eds.), FMCAD, Vol. 2517, Springer, 2002, pp. 187–201.

[15] A. Biere, Resolve and expand, in: SAT, 2004, pp. 238–246.

[16] F. Lonsing, A. Biere, Nenofex: Expanding NNF for QBF solving, in: H. K. Büning, X. Zhao (Eds.), SAT, Vol. 4996, Springer, 2008, pp. 196–210.

[17] M. Janota, W. Klieber, J. Marques-Silva, E. M. Clarke, Solving QBF with counterexample guided refinement, in: Cimatti and Sebastiani [35], pp. 114–128.

[18] U. Bubeck, H. K. Büning, Bounded universal expansion for preprocessing QBF, in: J. Marques-Silva, K. A. Sakallah (Eds.), SAT, Vol. 4501, Springer, 2007, pp. 244–257.

[19] U. Bubeck, Model-based transformations for quantified Boolean formulas, Ph.D. thesis, University of Paderborn (2010).

[20] M. Janota, J. Marques-Silva, On propositional QBF expansions and Q-resolution, in: M. Järvisalo, A. Van Gelder (Eds.), SAT, Vol. 7962, Springer, 2013, pp. 67–82.

[21] M. Janota, J. Marques-Silva, ∀Exp+Res does not p-simulate Q-resolution, International Workshop on Quantified Boolean Formulas (2013).

[22] H. K. Büning, U. Bubeck, Theory of quantified boolean formulas, in: A. Biere, M. Heule, H. van Maaren, T. Walsh (Eds.), Handbook of Satisfiability, Vol. 185 of Frontiers in Artificial Intelligence and Applications, IOS Press, 2009, pp. 735–760.

[23] M. L. Bonet, J. L. Esteban, N. Galesi, J. Johannsen, Exponential separations between restricted resolution and cutting planes proof systems, in: 39th Annual Symposium on Foundations of Computer Science, FOCS, IEEE Computer Society, 1998, pp. 638–647.

[24] G. S. Tseitin, On the complexity of derivations in the propositional calculus, Studies in Constructive Mathematics and Mathematical Logic Part II, ed. A.O. Slisenko.

[25] A. Urquhart, The complexity of propositional proofs, Bulletin of the EATCS 64.

[26] O. Beyersdorff, Disjoint NP-pairs and propositional proof systems, Ph.D. thesis, Humboldt University, Berlin (2006).

[27] M. Cadoli, M. Schaerf, A. Giovanardi, M. Giovanardi, An algorithm to evaluate quantified Boolean formulae and its experimental evaluation, J. Autom. Reasoning 28 (2).

[28] M. Janota, J. Marques-Silva, Abstraction-based algorithm for 2QBF, in: K. A. Sakallah, L. Simon (Eds.), SAT, Springer, 2011, pp. 230–244.

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

[30] W. Craig, Linear reasoning. a new form of the Herbrand-Gentzen theorem, J. Symb. Log. 22 (3) (1957) 250–268.

[31] QBF gallery, http://www.kr.tuwien.ac.at/events/qbfgallery2013/ (2013).

[32] A. Goerdt, Davis-Putnam resolution versus unrestricted resolution, Ann. Math. Artif. Intell. 6 (1-3) (1992) 169–184.

[33] M. Benedetti, Evaluating QBFs via symbolic Skolemization, in: F. Baader, A. Voronkov (Eds.), LPAR, Vol. 3452, Springer, 2004, pp. 285–300.

[34] M. Alekhnovich, J. Johannsen, T. Pitassi, A. Urquhart, An exponential separation between regular and general resolution, Theory of Computing 3 (1) (2007) 81–102.

[35] A. Cimatti, R. Sebastiani (Eds.), Theory and Applications of Satisfiability Testing - SAT 2012 - 15th International Conference, Trento, Italy, June 17-20, 2012. Proceedings, Vol. 7317, Springer, 2012.