Proceedings Paper:
Blinkhorn, J and Beyersdorff, O (2017) Shortening QBF Proofs with Dependency Schemes. In: Theory and Applications of Satisfiability Testing – SAT 2017 (Lecture Notes in Computer Science). International Conference on Theory and Applications of Satisfiability Testing, 28 Aug - 01 Sep 2017, Melbourne, Australia. Springer Nature, pp. 263-280. ISBN: 978-3-319-66262-6. ISSN: 0302-9743.

https://doi.org/10.1007/978-3-319-66263-3_17

Shortening QBF Proofs with Dependency Schemes

Joshua Blinkhorn and Olaf Beyersdorff

School of Computing, University of Leeds, UK

Abstract. We provide the first proof complexity results for QBF dependency calculi. By showing that the reflexive resolution path dependency scheme admits exponentially shorter Q-resolution proofs on a known family of instances, we answer a question first posed by Slivovsky and Szeider in 2014 [30]. Further, we conceive a method of QBF solving in which dependency recomputation is utilised as a form of inprocessing. Formalising this notion, we introduce a new calculus in which a dependency scheme is applied dynamically. We demonstrate the further potential of this approach beyond that of the existing static system with an exponential separation.

1 Introduction

Proof complexity is the study of proof size in systems of formal logic. Since its beginnings the field has enjoyed strong connections to computational complexity [8, 10] and bounded arithmetic [9, 17], and has emerged in the past two decades as the primary means for the comparison of algorithms in automated reasoning.

Recent successes in that area, epitomised by progress in SAT solving, have motivated broader research into the efficient solution of computationally hard problems. Amongst them, the logic of quantified Boolean formulas (QBF) is an established field with a substantial volume of literature. QBF extends propositional logic with the addition of existential and universal quantification, and naturally accommodates more succinct encodings of problem instances. This gives rise to diverse applications in areas including conformant planning [11, 24], verification [1], and ontologies [16].

It is fair to say that much of the early research into QBF solving [13, 26, 33], and later the proof complexity of associated theoretical models [4–6], was built upon existing techniques for SAT. For example, QCDCL [12] is a major paradigm in QBF solving based on conflict-driven clause learning (CDCL [21]), *the* dominant paradigm for SAT. By analogy, the fundamental theoretical model of QCDCL, the calculus Q-resolution (Q-Res [15]), is an extension of propositional resolution, the calculus that underpins CDCL. Given, however, that the decision problem for QBF is **PSPACE**-complete, it is perhaps unsurprising that the implementation of QCDCL presents novel obstacles for the practitioner, beyond those encountered at the level of propositional logic.

Arguably, the biggest challenge concerns the allowable order of variable assignments. In traditional QCDCL, the freedom to assign variables is limited

according to a linear order imposed by the quantifier prefix. Whereas decision variables must be chosen carefully to ensure sound results, coercing the order of assignment to respect the prefix is frequently needlessly restrictive [19]. Moreover, limiting the choice adversely affects the impact of decision heuristics. In contrast, such heuristics play a major role in SAT solving [18, 22, 27, 28], where variables may be assigned in an arbitrary order.

Dependency awareness, as implemented in the solver DepQBF [7], is a QBF-specific paradigm that attempts to maximise the impact of decision heuristics. By computing a *dependency scheme* before the search process begins, the linear order of the prefix is effectively supplanted by a partial order that better approximates the variable dependencies of the instance, granting the solver greater freedom regarding variable assignments. Use of the scheme is static; dependencies are computed only once and do not change during the search. Despite the additional computational cost incurred, empirical results demonstrate improved solving on many benchmark instances [19].

Dependency schemes themselves are tractable algorithms that identify dependency information by appeal to the syntactic form of an instance. From the plethora of schemes that have been proposed in the literature, two have emerged as principal objects of study. The *standard dependency scheme* ($\mathcal{D}^{\text{std}}$ [25]), a variant of which is used by DepQBF, was originally proposed in the context of backdoor sets. This scheme uses sequences of clauses connected by common existential variables to determine a dependency relation between the variables of an instance. The *reflexive resolution path dependency scheme* ($\mathcal{D}^{\text{rrs}}$ [31]) utilises the notion of a *resolution path*, a more refined type of connection introduced in [32].

A solid theoretical model for dependency awareness was only recently proposed in the shape of the calculus $\mathsf{Q}(\mathcal{D})\text{-}\mathsf{Res}$ [31], a parametrisation of Q-resolution by the dependency scheme $\mathcal{D}$. Whereas the body of work on $\mathsf{Q}(\mathcal{D})\text{-}\mathsf{Res}$ and related systems has focused on soundness [2, 23, 31], authors of all three papers have cited open problems in proof complexity. Indeed, prior to this paper there were no proof-theoretic results to support any claims concerning the potential of dependency schemes in the practice of QBF solving.

In this work, not only do we provide the first such results, we also demonstrate the potential of dependency schemes to further reduce the size of proofs if they are applied *dynamically*. We summarise our contributions below.

1. The first separations for QBF dependency calculi. We use the well-known formulas of Kleine Büning et al. [15] to prove the first exponential separation for $\mathsf{Q}(\mathcal{D})\text{-}\mathsf{Res}$. We show that $\mathcal{D}^{\text{rrs}}$ can identify crucial independencies in these formulas, leading to short proofs in the system $\mathsf{Q}(\mathcal{D}^{\text{rrs}})\text{-}\mathsf{Res}$. In contrast, we show that $\mathcal{D}^{\text{std}}$ cannot identify any non-trivial independencies, allowing us to lift the exponential lower bound for $\mathsf{Q}\text{-}\mathsf{Res}$ [3, 15] to $\mathsf{Q}(\mathcal{D}^{\text{std}})\text{-}\mathsf{Res}$. Our result proves conclusively for the first time that the use of a dependency scheme can significantly (i.e. exponentially) reduce the running time of a QBF solver.

2. A model of dynamic dependency analysis. We propose the new calculus dyn-Q($\mathcal{D}$)-Res that models the dynamic application of a dependency scheme in Q-resolution. The system employs a so-called 'reference rule' that allows new axioms, called reference clauses, to be introduced into the proof. The key insight is that the application of an assignment to an instance formula may allow the dependency scheme to unlock new independencies. As such, the reference rule alludes to an explicit refutation of the formula under an appropriate restriction, and is analogous to the recomputation of dependencies at an arbitrary point of the QCDCL procedure. We prove that dyn-Q($\mathcal{D}$)-Res is sound whenever the dependency scheme $\mathcal{D}$ is fully exhibited.

3. Exponential separation of static and dynamic systems. Our final contribution demonstrates that the dynamic application of dependency schemes can shorten Q-resolution proofs even further, yielding an exponential improvement even over the static approach. Using a modification of the aforementioned formulas from [15], we prove that dyn-Q($\mathcal{D}^{\mathsf{rrs}}$)-Res is exponentially stronger than Q($\mathcal{D}^{\mathsf{rrs}}$)-Res.

2 Preliminaries

Quantified Boolean formulas. In this paper, we consider *quantified Boolean formulas* (QBFs) in *prenex conjunctive normal form* (PCNF), typically denoted $\Phi = \mathcal{Q}\,.\,\phi$. A PCNF over Boolean variables $z_1, \ldots, z_n$ consists of a *quantifier prefix* $\mathcal{Q} = \mathcal{Q}_1 z_1 \cdots \mathcal{Q}_n z_n$, $\mathcal{Q}_i \in \{\exists, \forall\}$ for $i \in [n]$, in which all variables are quantified either existentially or universally, and a propositional conjunctive normal form (CNF) formula ϕ called the *matrix*. The prefix $\mathcal{Q}$ imposes a linear ordering $<_\Phi$ on the variables of Φ, such that $z_i <_\Phi z_j$ holds whenever $i < j$, in which case we say that z_j *is right of* z_i.

A literal is a variable or its negation, a clause is a disjunction of literals, and a CNF is a conjunction of clauses. Throughout, we refer to a clause as a set of literals and to a CNF as a set of clauses. We typically write x for existential variables, u for universals, and z for either. For a literal l, we write $\mathrm{var}(l) = z$ iff $l = z$ or $l = \neg z$, for a clause C we write $\mathrm{vars}(C) = \{\mathrm{var}(l) \mid l \in C\}$, and for a PCNF Φ we write $\mathrm{vars}(\Phi)$ for the variables in the prefix of Φ.

A (partial) assignment δ to the variables of Φ is represented as a set of literals, typically denoted $\{l_1, \ldots, l_k\}$, where literal z (resp. $\neg z$) represents the assignment $z \mapsto 1$ (resp. $z \mapsto 0$). The *restriction of* Φ *by* δ, denoted $\Phi[\delta]$, is obtained by removing from ϕ any clause containing a literal in δ, and removing the negated literals $\neg l_1, \ldots, \neg l_k$ from the remaining clauses, while the variables of δ and their associated quantifiers are removed from the prefix $\mathcal{Q}$. For assignments to single variables we may omit the braces; for example, we write $\Phi[l]$ for $\Phi[\{l\}]$.

QBF resolution. *Resolution* is a well-studied refutational proof system for propositional CNF formulas with a single inference rule: the *resolvent* $C_1 \cup C_2$

may be derived from clauses $C_1 \cup \{x\}$ and $C_2 \cup \{\neg x\}$ (variable x is the *pivot*). Resolution is *refutationally* sound and complete: that is, the empty clause can be derived from a CNF iff it is unsatisfiable.

There exist a host of resolution-based QBF proof systems – see [3] for a detailed account. *Q-resolution* (Q-Res) introduced in [15] is the standard refutational calculus for PCNF. In addition to resolution over existential pivots, the calculus has a *universal reduction rule* which allows a clause C to be derived from $C \cup \{l\}$, provided $\mathrm{var}(l)$ is a universal variable right of all existentials in C. Tautologies are explicitly forbidden; one may not derive a clause containing both z and $\neg z$.

For a QBF resolution system P, a P *derivation* of a clause C from a PCNF Φ is a sequence $C_1, \ldots, C_m$ of clauses in which $C = C_m$, and each clause is either an axiom or is derived from previous clauses in the sequence using an inference rule. A *refutation* of Φ is a derivation of the empty clause from Φ.

A proof system P *p-simulates* a system Q (denoted $\mathsf{Q} \leq_p \mathsf{P}$) if each Q-proof can be transformed in polynomial time into a P-proof of the same formula [10]. The systems P and Q are *p-equivalent* (denoted $\mathsf{P} \equiv_p \mathsf{Q}$) if $\mathsf{P} \leq_p \mathsf{Q}$ and $\mathsf{Q} \leq_p \mathsf{P}$.

QBF models. Let $\Phi = \mathcal{Q}_1 z_1 \cdots \mathcal{Q}_n z_n \,.\, \phi$ be a PCNF over existential variables $V_\exists$ and universal variables $V_\forall$. A *model* f for Φ is a mapping from total assignments to $V_\forall$ to total assignments to $V_\exists$ that satisfies two conditions: (a) whenever α and α' agree on all universals left of a variable z_i, then $f(\alpha)$ and $f(\alpha')$ agree on all existential variables left of (and including) z_i; (b) for each α in the domain of f, $\alpha \cup f(\alpha)$ satisfies every clause $C \in \phi$ (that is, $C \cap (\alpha \cup f(\alpha)) \neq \emptyset$). A PCNF is true iff it has a model, otherwise it is false.

Following [26], a model can be depicted naturally as a tree, as shown in Figure 1. For each α in the domain of f, the literals of the set $\alpha \cup f(\alpha)$ are written in prefix order on a unique path from the root of the tree to some leaf. As such, a model can be uniquely identified with a set of $2^{|V_\forall|}$ *paths*, each of which is one of the sets $\alpha \cup f(\alpha)$. This is a convenient interpretation (cf. [2]), and we adopt this approach for all technicalities concerning QBF models.

3 Static dependency awareness in Q-resolution

In this section, we provide the necessary background for dependency schemes and their incorporation into Q-resolution. We recall the definitions of the standard [25] and reflexive resolution path [31] dependency schemes, and the definition of the dependency calculus $\mathsf{Q}(\mathcal{D})$-Res.

3.1 Overview of dependency schemes

For the duration of this work, we deal only with the (in)dependence of existential variables on universal variables[1]. This is a convenience afforded by the fact that

[1] In practice, the dual notion of (in)dependence of universals on existentials is equally important.

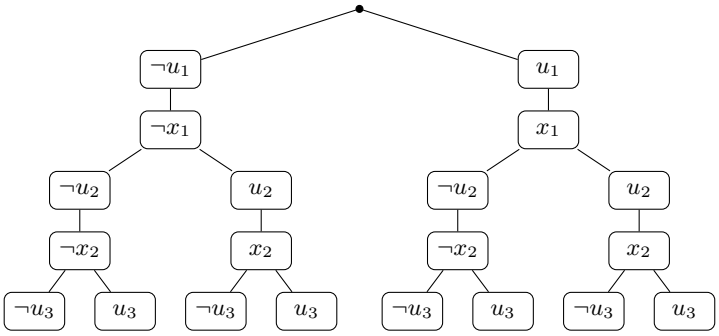

Fig. 1. Tree depiction of a model for the PCNF with prefix $\forall u_1 \exists x_1 \forall u_2 \exists x_2 \forall u_3$ and clauses $\{u_1, \neg x_1\}, \{\neg u_1, x_1\}$ and $\{\neg u_1, \neg u_2, x_2, \neg u_3\}$.

we deal with *refutational* calculi, in which the (in)dependence of universals on existentials does not feature. We therefore take the opportunity to work with tighter (and in some cases considerably simpler) definitions than those referenced in the literature.

A dependency scheme is presented as a function mapping PCNFs to binary relations. The binary relations represent variable dependencies. For an arbitrary PCNF Φ, the *trivial dependency relation* captures the linear order of the quantifier prefix of Φ, and is given by $\mathcal{D}^{\mathrm{trv}}(\Phi) = \{(u, x) \in \mathrm{vars}_\forall(\Phi) \times \mathrm{vars}_\exists(\Phi) \mid u <_\Phi x\}$. Formally, a *dependency scheme* $\mathcal{D}$ is a mapping from the set of all PCNFs that satisfies $\mathcal{D}(\Phi) \subseteq \mathcal{D}^{\mathrm{trv}}(\Phi)$ for each PCNF Φ. The existence of a pair $(u, x) \in \mathcal{D}(\Phi)$ should be interpreted as 'existential x depends on universal u in Φ according to dependency scheme $\mathcal{D}$'. We say that $\mathcal{D}'$ is *at least as general* as $\mathcal{D}$ iff $\mathcal{D}'(\Phi) \subseteq \mathcal{D}(\Phi)$ for each PCNF Φ, and is *strictly more general* if the inclusion is strict for some PCNF.

All non-trivial dependency schemes that have appeared in the literature to date are based in some way or another on connections between clauses in the matrix. In the standard dependency scheme $\mathcal{D}^{\mathrm{std}}$, an existential x depends on a universal u whenever a clause containing variable x is connected to a clause containing variable u, whereby clauses are connected iff they share a common existential variable that is right of u. The absence of such a connection ensures that x is independent of u according to $\mathcal{D}^{\mathrm{std}}$.

Definition 1 (standard dependency scheme [25]). *Let $\Phi = \mathcal{Q} \,.\, \phi$ be a PCNF. The pair $(u, x) \in \mathcal{D}^{\mathrm{trv}}(\Phi)$ is in $\mathcal{D}^{\mathrm{std}}(\Phi)$ iff there exists a sequence of clauses $C_1, \ldots, C_n \in \phi$ with $u \in \mathrm{vars}(C_1)$, $x \in \mathrm{vars}(C_n)$, such that, for each $i \in [n-1]$, $\mathrm{vars}(C_i) \cap \mathrm{vars}(C_{i+1})$ contains an existential variable right of u.*

Whereas connections in $\mathcal{D}^{\mathrm{std}}$ are based on common variables, the reflexive resolution path dependency scheme $\mathcal{D}^{\mathrm{rrs}}$ improves upon $\mathcal{D}^{\mathrm{std}}$ by taking polarity into account. The connecting existential variable must appear in opposite polarities in the connected clauses, yielding a strictly more general scheme. As explained above, we present a simplified formulation of $\mathcal{D}^{\mathrm{rrs}}$ tailored to the current work.

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

$$\frac{}{C}$$

C is a clause in the matrix ϕ.

Reduction rule: $\mathsf{red}(C, l)$

$$\frac{C}{C \setminus \{l\}}$$

- literal l is universal.
- $(\mathrm{var}(l), x) \notin \mathcal{D}(\Phi)$ holds for each existential variable x in $\mathrm{vars}(C)$.

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

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

- variable x is existential.
- $x \in C_1$ and $\neg x \in C_2$.
- the resolvent is non-tautological.

Fig. 2. The rules of $\mathsf{Q}(\mathcal{D})$-Res [31]. $\mathcal{D}$ is a dependency scheme and $\Phi = \mathcal{Q} \,.\, \phi$ is a PCNF.

Definition 2 (reflexive resolution path dependency scheme [31]). *Let $\Phi = \mathcal{Q} \,.\, \phi$ be a PCNF, and let $(u, x) \in \mathcal{D}^{\mathrm{trv}}(\Phi)$. Then $(u, x) \in \mathcal{D}^{\mathrm{rrs}}(\Phi)$ iff there is a sequence of clauses $C_1, \ldots, C_n \in \phi$ and a sequence of existential literals $l_1, \ldots, l_{n-1}$ for which the following four conditions hold:*

(a) $u \in C_1$ *and* $\neg u \in C_n$,
(b) $x = \mathrm{var}(l_i)$, *for some* $i \in [n-1]$,
(c) $u <_\Phi \mathrm{var}(l_i)$, $l_i \in C_i$ *and* $\neg l_i \in C_{i+1}$, *for each* $i \in [n-1]$,
(d) $\mathrm{var}(l_i) \neq \mathrm{var}(l_{i+1})$ *for each* $i \in [n-2]$.

3.2 Dependency schemes in Q-resolution

The theoretical model for the use of dependency schemes in dependency-aware solving is captured by the calculus $\mathsf{Q}(\mathcal{D})$-Res, introduced in [31]. The main idea is to generalise Q-Res by replacing the implicit reference to the trivial dependency scheme with an explicit reference to a strictly more general scheme. Note that Q-Res allows a universal variable u to be reduced only if it is right of all existentials in the clause, or, equivalently, whenever all existentials in the clause are trivially independent of u. By contrast, in $\mathsf{Q}(\mathcal{D})$-Res u can be reduced whenever all existentials in the clause are $\mathcal{D}$-independent of u. We recall the rules of $\mathsf{Q}(\mathcal{D})$-Res in Fig. 2.

Soundness of the calculus $\mathsf{Q}(\mathcal{D})$-Res is not guaranteed, and hinges on the choice of the dependency scheme $\mathcal{D}$. Previous work has shown that the concept of *full exhibition*[2], which imposes a natural condition on $\mathcal{D}$, is sufficient to prove

[2] The term 'full exhibition' was coined in [2]. The concept itself and the term '$\mathcal{D}$-model' originate from [29].

soundness in $\mathsf{Q}(\mathcal{D})$-Res [29], and indeed in stronger dependency calculi for QBF [2]. Following [2], we say that a model f exhibits the independence of x on u iff, for each α in the domain of f, the assignment to x in $f(\alpha)$ remains unchanged when the assignment to u in α is flipped.

Definition 3 (full exhibition [2,29]). *A model f for a PCNF Φ is a $\mathcal{D}$-model iff, for each $(u, x) \in \mathcal{D}^{\mathrm{trv}}(\Phi) \setminus \mathcal{D}(\Phi)$, f exhibits the independence of x on u. A dependency scheme $\mathcal{D}$ is fully exhibited iff each true PCNF has a $\mathcal{D}$-model.*

Informally, full exhibition ensures that a true PCNF has a particular model in which existentials do not depend on the universals from which they are independent according to the dependency scheme. As in [29], we refer to such a model as a $\mathcal{D}$-model. In Section 5, we show that full exhibition remains sufficient for soundness when a dependency scheme is applied dynamically, as opposed to the static application offered in $\mathsf{Q}(\mathcal{D})$-Res.

It should be clear that $\mathsf{Q}(\mathcal{D})$-Res is simulated by $\mathsf{Q}(\mathcal{D}')$-Res whenever $\mathcal{D}'$ is at least as general as $\mathcal{D}$. We conclude this section by noting the following trivial simulations for $\mathsf{Q}(\mathcal{D})$-Res.

Proposition 4. $\mathsf{Q}\text{-Res} \equiv_p \mathsf{Q}(\mathcal{D}^{\mathsf{trv}})\text{-Res} \leq_p \mathsf{Q}(\mathcal{D}^{\mathsf{std}})\text{-Res} \leq_p \mathsf{Q}(\mathcal{D}^{\mathsf{rrs}})\text{-Res}$.

4 Exponential separation of $\mathsf{Q}(\mathcal{D}^{\mathsf{std}})$-Res and $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs}})$-Res

In this section, we prove that $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs}})$-Res is exponentially stronger than $\mathsf{Q}(\mathcal{D}^{\mathsf{std}})$-Res. Given that $\mathsf{Q}(\mathcal{D}^{\mathsf{std}})$-Res p-simulates Q-Res (Prop. 4), we thereby separate $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs}})$-Res and Q-Res, thus answering the question initially posed by Slivovsky and Szeider in [30].

The separating formulas are a well-studied family of PCNFs, originally introduced in [15]. We recall the definition of this formula family, which is referred to as $\Psi(n)$ throughout this paper.

Definition 5 (formulas of Kleine Büning et al. [15]). *The formula family $\Psi(n) := \mathcal{Q}(n) \,.\, \psi(n)$ has prefixes $\mathcal{Q}(n) := \exists x_1 \exists y_1 \forall u_1 \cdots \exists x_n \exists y_n \forall u_n \exists t_1 \cdots \exists t_n$ and matrices $\psi(n)$ consisting of the clauses*

$$
\begin{array}{lll}
A = \{\neg x_1, \neg y_1\}, & & \\
B_i = \{x_i, u_i, \neg x_{i+1}, \neg y_{i+1}\} & B'_i = \{y_i, \neg u_i, \neg x_{i+1}, \neg y_{i+1}\} & i \in [n-1], \\
B_n = \{x_n, u_n, \neg t_1, \ldots, \neg t_n\} & B'_n = \{y_n, \neg u_n, \neg t_1, \ldots, \neg t_n\}, & \\
C_i = \{u_i, t_i\} & C'_i = \{\neg u_i, t_i\} & i \in [n].
\end{array}
$$

We first show that the standard dependency scheme cannot identify any non-trivial independencies for $\Psi(n)$.

Proposition 6. *For each $n \in \mathbb{N}$, $\mathcal{D}^{\mathrm{std}}(\Psi(n)) = \mathcal{D}^{\mathrm{trv}}(\Psi(n))$.*

Proof. Let $n \in \mathbb{N}$ and let $i, j \in [n]$.

For $(u_i, t_j) \in \mathcal{D}^{\mathrm{trv}}(\Psi(n))$, consider the sequence of clauses $B_i, \ldots, B_n$, and observe that $u_i \in \mathrm{vars}(B_i)$ and $t_j \in \mathrm{vars}(B_n)$. For each $k \in [i, n-1]$, the

existential variable x_{k+1}, which is right of u_i, is in the set $\mathrm{vars}(B_k) \cap \mathrm{vars}(B_{k+1})$. Therefore $(u_i, t_j) \in \mathcal{D}^{\mathrm{std}}(\Psi(n))$.

For each $(u_i, x_j) \in \mathcal{D}^{\mathrm{trv}}(\Psi(n))$ with $i < j$, the fact that $(u_i, x_j) \in \mathcal{D}^{\mathrm{std}}(\Psi(n))$ is shown similarly, using the sequence of clauses $B_i, \ldots, B_j$. For the final case $(u_i, y_j) \in \mathcal{D}^{\mathrm{trv}}(\Psi(n))$ take the sequence $B'_i, \ldots, B'_j$. □

The salient consequence of Prop. 6 is that every application of ∀-reduction in a $\mathsf{Q}(\mathcal{D}^{\mathsf{std}})$-Res derivation from $\Psi(n)$ is also available in Q-Res. As a result, the Q-Res lower bound for $\Psi(n)$ lifts directly to $\mathsf{Q}(\mathcal{D}^{\mathsf{std}})$-Res.

Theorem 7. *The QBFs $\Psi(n)$ require exponential-size $\mathsf{Q}(\mathcal{D}^{\mathsf{std}})$-Res refutations.*

Proof. It is known that $\Psi(n)$ require exponential-size Q-Res refutations [3,15]. By Prop. 6, any $\mathsf{Q}(\mathcal{D}^{\mathsf{std}})$-Res refutation of $\Psi(n)$ is a $\mathsf{Q}(\mathcal{D}^{\mathsf{trv}})$-Res refutation of $\Psi(n)$. The result follows since Q-Res and $\mathsf{Q}(\mathcal{D}^{\mathsf{trv}})$-Res are p-equivalent, by Prop. 4. □

In contrast, the more general dependency scheme $\mathcal{D}^{\mathrm{rrs}}$ can identify some crucial non-trivial independencies in $\Psi(n)$.

Proposition 8. *For each $n \in \mathbb{N}$ and for each $i, j \in [n]$, if $i \neq j$ then $(u_i, t_j) \notin \mathcal{D}^{\mathrm{rrs}}(\Psi(n))$.*

Proof. Let $n \in \mathbb{N}$ and let $i, j \in [n]$ with $i \neq j$. Suppose that $D_1, \ldots, D_k \in \psi(n)$ and $l_1, \ldots, l_{k-1}$ are sequences of clauses and literals respectively, satisfying the four conditions of Definition 2 with respect to the pair $(u_i, t_j) \in \mathcal{D}^{\mathrm{trv}}(\Psi(n))$. By condition (b), the literal sequence contains a literal in the variable t_j. Observe that, in the matrix $\psi(n)$, the positive literal t_j occurs only in the clauses $C_j = \{u_j, t_j\}$ and $C'_j = \{\neg u_j, t_j\}$. Hence, by condition (c), there is some clause D in the clause sequence such that $D = C_j$ or $D = C'_j$. Since t_j is the only existential literal in D, the clause must be an endpoint of the sequence by condition (d), and hence we must have $D = D_1$ or $D = D_k$. However, since $i \neq j$, this implies that either $u_i \notin D_1$ or $u_i \notin D_k$, contradicting condition (a). □

According to Prop. 8, a $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs}})$-Res refutation of $\Psi(n)$ may contain ∀-reduction steps that are disallowed in Q-Res. For example, under $\mathcal{D}^{\mathrm{rrs}}$ it is possible to remove literal u_n from the clause $\{x_n, u_n, \neg t_1, \ldots, \neg t_{n-1}\}$. As we demonstrate in the proof of the following theorem, it is precisely this step (which is unavailable in Q-Res due to the presence of existentials right of u) that permits the construction of $O(n)$-size $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs}})$-Res refutations.

Theorem 9. *The formulas $\Psi(n)$ have linear-size $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs}})$-Res refutations.*

Proof. A portion of a linear-size $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs}})$-Res refutation of $\Psi(n)$ is shown in Fig. 3. The clauses $\{x_{n-1}, u_{n-1}, \neg t_1, \ldots, \neg t_{n-1}\}$ and $\{y_{n-1}, \neg u_{n-1}, \neg t_1, \ldots, \neg t_{n-1}\}$ are derived in a constant number of steps, and the task is reduced to the refutation of $\Psi(n-1)$. The complete refutation is therefore linear in size.

According to Prop. 8, in a $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs}})$-Res derivation from $\Psi(n)$ the variable u_i may be removed from a clause D provided that the existential variables in D that are right of u_i are contained in the set $\{t_1, \ldots, t_n\} \setminus \{t_i\}$. Such ∀-reduction steps, which would be disallowed in Q-Res, are marked with an asterisk $(*)$ in Fig. 3. □

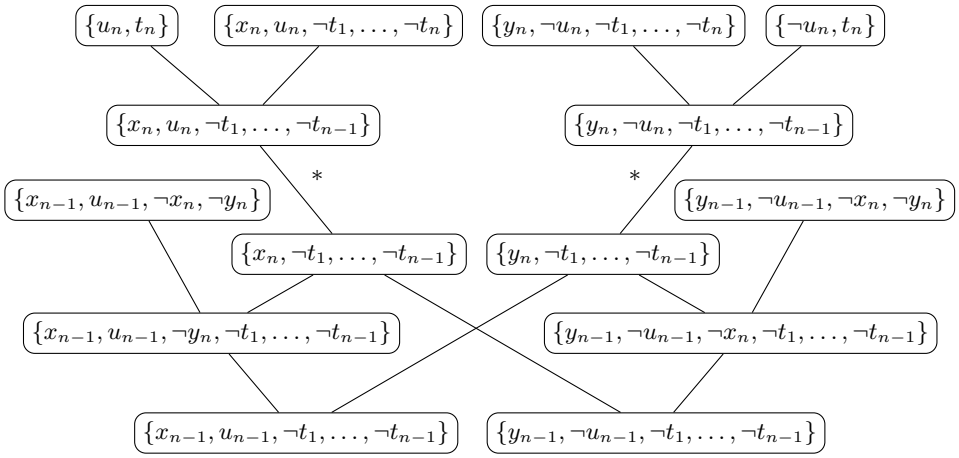

Fig. 3. Portion of a linear size $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs}})$-Res refutation of $\Psi(n)$. The $\forall$-reduction steps marked with $*$ are forbidden in Q-Res, but are allowed in $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs}})$-Res due to Prop. 8.

The following result is an immediate consequence of Theorems 7 and 9.

Theorem 10. $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs}})$-Res *is exponentially stronger than* $\mathsf{Q}(\mathcal{D}^{\mathsf{std}})$-Res.

5 Modelling dynamic dependency awareness

In this section, we introduce the dynamic dependency calculus dyn-$\mathsf{Q}(\mathcal{D})$-Res and prove that it is sound for a fully exhibited scheme $\mathcal{D}$.

5.1 Dynamic dependencies in Q-resolution

We first define a particular kind of assignment to the variables of a PNCF that, in a clear sense, 'respects' the dependency scheme $\mathcal{D}$.

Definition 11 ($\mathcal{D}$-assignment). *Let $\mathcal{D}$ be a dependency scheme and let δ be a partial assignment to the variables of a PCNF Φ. Then δ is a $\mathcal{D}$-assignment for Φ iff, whenever δ assigns an existential literal l, then δ assigns all universal variables in the set $\{u \mid (u, \mathrm{var}(l)) \in \mathcal{D}(\Phi)\}$.*

We also define the *largest falsified clause* of an assignment.

Definition 12 (largest falsified clause). *Let $\delta = l_1, \ldots, l_k$ be an assignment. The largest falsified clause of δ is $\{\neg l_1, \ldots, \neg l_k\}$.*

Definition of the calculus. We define dyn-$\mathsf{Q}(\mathcal{D})$-Res as the proof system that has the rules of $\mathsf{Q}(\mathcal{D})$-Res in addition to the *reference rule* shown in Fig. 4. On an intuitive level, the reference rule is based on the following fact: Given a PCNF Φ and a fully exhibited dependency scheme $\mathcal{D}$, if Φ is false under restriction by

Reference rule: $\mathsf{ref}(\delta, \pi)$

$$\frac{}{C}$$

- δ is a $\mathcal{D}$-assignment for Φ.
- π is a dyn-Q($\mathcal{D}$)-Res refutation of $\Phi[\delta]$
- C is the largest falsified clause of δ.

Fig. 4. The reference rule of dyn-Q($\mathcal{D}$)-Res. $\mathcal{D}$ is a dependency scheme and $\Phi = \mathcal{Q} \,.\, \phi$ is a PCNF.

a $\mathcal{D}$-assignment δ, then adding the largest falsified clause of δ to the matrix of Φ preserves satisfiability[3] (note that this does not hold for an arbitrary assignment δ). Therefore, if the calculus is capable of refuting $\Phi[\delta]$, it should be able to introduce the largest falsified clause of δ.

We refer to a clause derived by application of the reference rule as a *reference clause*. As stated in the rule itself, a reference clause may only be introduced if an explicit refutation π of $\Phi[\delta]$ can be given. This feature allows the size of a dyn-Q($\mathcal{D}$)-Res derivation to be suitably defined. We refer to π as a *referenced refutation*.

The power of the reference rule lies in the fact that the dependency scheme $\mathcal{D}$ may identify (or *unlock*) new non-trivial independencies in the restricted formula, meaning that it may be easier to refute the restricted formula $\Phi[\delta]$ than to derive the reference clause from Φ directly. We note that the referenced refutation π, being a derivation from $\Phi[\delta]$, can make use of these newly unlocked independencies. In this way, the calculus models the recomputation of dependencies during the QCDCL search procedure. We elaborate on this point in Subsection 5.3.

Reference degree. The reference degree of a dyn-Q($\mathcal{D}$)-Res derivation is 0 iff it does not contain any reference clauses (i.e. it is a Q($\mathcal{D}$)-Res derivation). For all other derivations π, the reference degree is $d+1$, where d is the largest reference degree of a refutation referenced from π.

Proof size. The size of a dyn-Q($\mathcal{D}$)-Res derivation π of reference degree 0 is the number of clauses in the proof. The size of a derivation π with non-zero reference degree is $a + b$, where a is the number of clauses in π and b is the sum of the sizes of refutations referenced from π.

5.2 Soundness of dyn-Q($\mathcal{D}$)-Res

The task of proving that dyn-Q($\mathcal{D}$)-Res is sound for a fully exhibited dependency scheme $\mathcal{D}$ (Theorem 15) can essentially be reduced to proving that the reference clauses derived from a true PCNF are satisfied by a $\mathcal{D}$-model (Lemma 14).

[3] We prove this statement formally in Subsection 5.2 (Lemma 14).

In what follows, we find it convenient to introduce a notion of restriction for models. Let f be a model for a PCNF Φ, and let l be a literal with $\mathrm{var}(l) \in \mathrm{vars}(\Phi)$. If l is universal, then $f[l]$ is obtained from f by removing all paths containing $\neg l$ and removing l from all remaining paths. If l is existential, then $f[l]$ is obtained from f by removing all occurrences of l and $\neg l$ from the paths of f.

It should be clear that $f[l]$ is a model for $\Phi[l]$ if l is universal. The same is also true for an existential literal l provided that it is unopposed in f, by which we mean that its negation $\neg l$ does not appear in any path in f. These facts are useful enough in the sequel to be the subject of the following proposition.

Proposition 13. *Let Φ be a PCNF, let f be a model for Φ and let l be a literal with $\mathrm{var}(l) \in \mathrm{vars}(\Phi)$. Then $f[l]$ is a model for $\Phi[l]$ if either (a) l is universal, or (b) l is existential and unopposed in f.*

We extend the restriction of a model to an arbitrary assignment $\delta = \{l_1, \ldots, l_k\}$ in the natural way; that is, $f[\delta]$ is the result of the successive restriction of f by the literals in δ. It should be clear that the order of successive restrictions does not matter.

We proceed to prove that a $\mathcal{D}$-model of a PCNF Φ satisfies any reference clause derivable from it in dyn-Q($\mathcal{D}$)-Res.

Lemma 14. *Let $\mathcal{D}$ be a dependency scheme, let f be a $\mathcal{D}$-model for a PCNF Φ, and let δ be a $\mathcal{D}$-assignment for Φ. If $\Phi[\delta]$ is false, then f satisfies the largest falsified clause of δ.*

Proof. Let $\Phi = \mathcal{Q}\,.\,\phi$, and let C be the largest falsified clause of δ. We prove the contrapositive statement: if f does not satisfy C, then $\Phi[\delta]$ is true.

The idea of the proof is to restrict f by δ, obtaining a model for $\Phi[\delta]$. The simplest way to do this is to restrict first by the universal subassignment of δ, and then by the existential subassignment. To that end, let $\delta_\forall := \{l \in \delta \mid \mathrm{var}(l) \text{ is universal}\}$ and define $\delta_\exists$ similarly.

By successive application of Proposition 13 (a), it follows that $f[\delta_\forall]$ is a model for $\Phi[\delta_\forall]$. We claim that every literal in $\delta_\exists$ is unopposed in $f[\delta_\forall]$. We will therefore prove the result since, by successive application of Proposition 13 (b), it follows that $(f[\delta_\forall])[\delta_\exists] = f[\delta]$ is a model for $(\Phi[\delta_\forall])[\delta_\exists] = \Phi[\delta]$.

It remains to prove that the literals in $\delta_\exists$ are indeed unopposed in $f[\delta_\forall]$. Suppose that f falsifies C. Then there is some path P in f that contains none of the literals in C. Since P contains a literal for every variable, it must therefore contain the negation of every literal in C. It follows, by definition of largest falsified clause (Def. 12), that $\delta \subseteq P$. Then, by definition of model restriction, there is some path $P' = P \setminus \delta_\forall$ in $f[\delta_\forall]$ with $\delta_\exists \subseteq P'$. The result follows since each existential variable in $\mathrm{vars}(\delta_\exists)$ appears in $f[\delta_\forall]$ in a single polarity. To see this, let $x \in \mathrm{vars}(\delta_\exists)$, and note that $\{u \mid (u, x) \in \mathcal{D}(\Phi)\} \subseteq \mathrm{vars}(\delta_\forall)$, since δ is a $\mathcal{D}$-assignment. Hence, $f[\delta_\forall]$ exhibits the independence of x on all remaining universals, and x therefore occurs in a single polarity. □

To prove that dyn-Q($\mathcal{D}$)-Res is sound, we must prove that one cannot derive the empty clause from any true PCNF. The proof is obtained by the addition of Lemma 14 to the literature's existing proof of soundness for Q($\mathcal{D}$)-Res.

Theorem 15. *The dynamic dependency calculus* dyn-Q($\mathcal{D}$)-Res *is sound if* $\mathcal{D}$ *is fully exhibited.*

Proof. In [29] it is shown that Q($\mathcal{D}$)-Res is sound if $\mathcal{D}$ is fully exhibited. The result may be proved by induction on derivation depth in the following way (for a detailed proof cf. [2]): Let $\mathcal{D}$ be a fully exhibited dependency scheme, let $\Phi := \mathcal{Q} \,.\, \phi$ be a true PCNF and assume π is a Q($\mathcal{D}$)-Res refutation of Φ. Since $\mathcal{D}$ is fully exhibited, there exists a fully exhibiting model f for Φ (with respect to $\mathcal{D}$) that satisfies every matrix clause. Moreover, if f satisfies the antecedent clauses of any application of resolution or reduction, then f satisfies the consequent clause. We therefore reach a contradiction, since f satisfies the conclusion of π, the empty clause.

Now, if we instead let π be a dyn-Q($\mathcal{D}$)-Res refutation, the above method can be lifted provided that the fully exhibiting model f satisfies every clause introduced by application of the reference rule. In this way, we prove soundness by induction on the reference degree d of π.

The base case $d = 0$ is already established [29], since any dyn-Q($\mathcal{D}$)-Res refutation of degree 0 is a Q($\mathcal{D}$)-Res refutation. For the inductive step, let $d \geq 1$, and suppose that all dyn-Q($\mathcal{D}$)-Res refutations of reference degree less than d are sound. Further, let C be the first reference clause of π, introduced by application of $\mathsf{ref}(\delta, \pi')$. Since π' is a dyn-Q($\mathcal{D}$)-Res refutation of $\Phi[\delta]$ of reference degree at most $d-1$, $\Phi[\delta]$ is false by the inductive hypothesis. Since C is the largest falsified clause of δ, it is therefore satisfied by f, by Lemma 14. Successive application of the argument demonstrates that f satisfies every reference clause in π. □

As it is known that $\mathcal{D}^{\mathrm{rrs}}$ is fully exhibited [2], the fact that dyn-Q($\mathcal{D}^{\mathsf{rrs}}$)-Res is sound is a corollary to Theorem 15. Since $\mathcal{D}^{\mathrm{rrs}}$ is strictly more general than $\mathcal{D}^{\mathrm{std}}$, every dyn-Q($\mathcal{D}^{\mathsf{std}}$)-Res refutation is a dyn-Q($\mathcal{D}^{\mathsf{rrs}}$)-Res refutation, hence dyn-Q($\mathcal{D}^{\mathsf{std}}$)-Res is also sound.

Corollary 16. *The calculi* dyn-Q($\mathcal{D}^{\mathsf{std}}$)-Res *and* dyn-Q($\mathcal{D}^{\mathsf{rrs}}$)-Res *are both sound.*

5.3 Motivations for dyn-Q($\mathcal{D}$)-Res

We chose to define a $\mathcal{D}$-assignment in order to replicate the kind of assignment that is maintained by a QCDCL solver using a dependency scheme, whereby decision variables are assigned only after all others on which they depend. In line with our discussion in Section 3, we can relax the theoretical model so that only the (in)dependence of existentials on universals is considered, and hence universals may be assigned arbitrarily in a $\mathcal{D}$-assignment.

The motivation for dyn-Q($\mathcal{D}$)-Res is this observation: If, by recomputing dependencies, the solver is able to refute the formula under its current assignment δ, it should be able to learn the largest falsified clause of δ. In this way, the

system shares similarities with 'Q-resolution with generalised axioms' [20]. Regarding proof complexity, a drawback of that calculus is that every false formula may be refuted in a single step. Our system resolves this difficulty, using the notion of referencing to accommodate a suitable definition of proof size.

In line with [20], we could have allowed assignments due to unit propagation and pure literal elimination in dyn-Q($\mathcal{D}$)-Res. This would allow additional existential literals to be included in a $\mathcal{D}$-assignment provided that they are valid assignments under Boolean constraint propagation. Doing so would result in a stronger version of dyn-Q($\mathcal{D}$)-Res, since such a modification extends the set of $\mathcal{D}$-assignments for any instance. However, we prefer to the present simpler system, since propagation is not necessary for the separation in the following section.

Soundness of the system with propagation can be proved by an extension of our argument in Lemma 14. This is because existential literals that become unit under restriction are always unopposed in the restricted model, and hence Proposition 13 still applies. Existential literals that become pure can be assigned unopposed throughout the restricted model without falsifying any clauses.

6 Static vs dynamic dependency awareness in Q($\mathcal{D}$)-Res

In this section, we investigate the relative proof complexities of Q($\mathcal{D}$)-Res and dyn-Q($\mathcal{D}$)-Res. We prove an exponential separation when $\mathcal{D}$ is the reflexive resolution path dependency scheme. In contrast, the two systems are p-equivalent when $\mathcal{D}$ is the trivial dependency scheme.

The latter result, while a perfectly natural conjecture, requires a non-trivial proof.

Theorem 17. Q-Res *and* dyn-Q($\mathcal{D}^{\mathsf{trv}}$)-Res *are p-equivalent proof systems.*

Proof (sketch). Since dyn-Q($\mathcal{D}^{\mathsf{trv}}$)-Res trivially p-simulates Q-Res (Prop. 4), we need only prove the reverse simulation. We prove by induction on reference degree that any dyn-Q($\mathcal{D}^{\mathsf{trv}}$)-Res derivation can be transformed into a Q-Res derivation of the same size, in time linear in the size of the original derivation. To that end, let π be a dyn-Q($\mathcal{D}^{\mathsf{trv}}$)-Res derivation of a clause C from a PCNF Φ of reference degree d.

If $d = 0$, then π is a Q-Res derivation, so the base case is established trivially. For the inductive step, let $d \geq 1$, and let R be a reference clause in π derived by application of rule $\mathsf{ref}(\delta, \pi')$. Note that the reference degree of π' is less than d, and hence, by the inductive hypothesis, π' can be transformed in linear time into a Q-Res refutation ρ of $\Phi[\delta]$ with $|\rho| = |\pi'|$. A $\mathcal{D}^{\mathrm{trv}}$-assignment assigns variables strictly in block order and assigns no variable before the preceding block is fully assigned. As a result, adding the literals in R to each clause of ρ cannot invalidate any $\forall$-reduction step, nor introduce a universal tautology. Moreover, doing so transforms ρ in linear time into a Q-Res derivation ρ' of R, with $|\rho'| = |\rho|$. Since every axiom clause in ρ' is subsumed by some clause in the matrix of Φ, ρ' can be transformed into a derivation of R from Φ simply by omitting any steps that are rendered unnecessary by the absence of a literal.

This last transformation can clearly be carried out in linear time and does not increase the size of the derivation.

Successive application of this method to all the reference clauses in π yields a Q-Res derivation of C of size at most $|\pi|$. The complete transformation can be carried out in time linear in $|\pi|$. □

The remainder of this section is devoted to the separation of dyn-Q($\mathcal{D}^{\text{rrs}}$)-Res from Q($\mathcal{D}^{\text{rrs}}$)-Res. The separating formulas are a modification of $\Psi(n)$, for which we make use of the following operation.

Definition 18 (clause-matrix product). *Let C be a clause and let ϕ be a CNF matrix. The clause-matrix product $C \otimes \phi$ is the CNF matrix with clauses $\{C \cup C' \mid C' \in \phi\}$.*

We modify $\Psi(n)$ by adding two fresh existential variables a and b, quantified at the very beginning and very end of the prefix, respectively. Taking two copies of the matrix $\psi(n)$, to each clause of the first copy we add literals a and b, and to each clause of the second we add literals $\neg a$ and $\neg b$. Finally, we add the clauses $\{a, \neg b\}$ and $\{\neg a, b\}$ so that the modified formulas are false.

Definition 19 (modification of the formulas of Kleine Büning et al.). *Let $\Psi(n) := \mathcal{Q}(n)\,.\,\psi(n)$ be the formulas of Kleine Büning et al. (as in Def. 5). We define the formula family*

$$\Xi(n) := \exists a \mathcal{Q}(n) \exists b\,.\,(\{a, b\} \otimes \psi(n)) \cup (\{\neg a, \neg b\} \otimes \psi(n)) \cup \{\{a, \neg b\}, \{\neg a, b\}\}.$$

The purpose of variable b is to introduce sufficiently many $\mathcal{D}^{\text{rrs}}$ connections between clauses, such that $\mathcal{D}^{\text{rrs}}$ can no longer identify any non-trivial independencies. This means that static application of $\mathcal{D}^{\text{rrs}}$ cannot improve upon Q-Res. However, under either assignment to variable a, one copy of the matrix $\psi(n)$ vanishes, and the connections due to b disappear with it. As a result, the restricted formulas $\Xi(n)[a]$ and $\Xi(n)[\neg a]$ are sufficiently similar to $\Psi(n)$ to admit short Q($\mathcal{D}^{\text{rrs}}$)-Res refutations. Hence, dynamic application of $\mathcal{D}^{\text{rrs}}$ yields shorter proofs.

To prove the lower bound for the static calculus, we first show that $\mathcal{D}^{\text{rrs}}(\Xi(n)) = \mathcal{D}^{\text{trv}}(\Xi(n))$, from which it follows that any Q($\mathcal{D}^{\text{rrs}}$)-Res refutation of $\Xi(n)$ is also a Q-Res refutation. We then show that any Q-Res refutation of $\Xi(n)$ contains an embedded refutation of $\Psi(n)$, which has size at least 2^n [3, 15].

Theorem 20. *The QBFs $\Xi(n)$ require exponential-size* Q($\mathcal{D}^{\text{rrs}}$)-Res *refutations.*

Proof. To see that $\mathcal{D}^{\text{rrs}}$ does not identify any spurious existential dependencies for $\Xi(n)$ – or, equivalently, that $\mathcal{D}^{\text{rrs}}(\Xi(n)) = \mathcal{D}^{\text{trv}}(\Xi(n))$ – we must show that, for each pair $(v, z) \in \mathcal{D}^{\text{trv}}(\Xi(n))$, there exists a sequence of k clauses and a sequence of $k - 1$ literals satisfying the four conditions of Def. 2.

Let $i, j \in [n]$. For $(u_i, b) \in \mathcal{D}^{\text{trv}}(\Xi(n))$, the clauses $\{a, b\} \cup B_i, \{\neg a, \neg b\} \cup B'_i$ and the single literal b form suitable sequences. For $(u_i, t_j) \in \mathcal{D}^{\text{trv}}(\Xi(n))$, the clauses $\{a, b\} \cup B_i, \{\neg a, \neg b\} \cup C_j, \{a, b\} \cup B_n, \{\neg a, \neg b\} \cup B'_i$ and the literals b, t_j, b are suitable. For $(u_i, x_j) \in \mathcal{D}^{\text{trv}}(\Xi(n))$ (with $i < j$), the clauses

$\{a, b\} \cup B_i, \{\neg a, \neg b\} \cup B_j, \{a, b\} \cup \{B_{j-1}\}, \{\neg a, \neg b\} \cup B'_i$ and the literals b, x_j, b are suitable, and the case for $(u_i, y_j) \in \mathcal{D}^{\text{trv}}(\Xi(n))$ is similar.

Now, let π be a $\mathsf{Q}(\mathcal{D}^{\text{rrs}})$-$\mathsf{Res}$ refutation of $\Xi(n)$, and let α be the assignment $\{\neg a, \neg b\}$. Since α assigns only existential variables, $\pi[\alpha]$ is a refutation of $\Xi(n)[\alpha]$ that is no larger than π. Observe that $\Xi(n)[\alpha] = \Psi(n)$, hence the size of $\pi[\alpha]$ is at least 2^n [3, 15], and we must have $|\pi| \geq 2^n$. □

The upper bound argument makes use of the construction of short refutations from the proof of Theorem 9. By referencing those refutations, dyn-$\mathsf{Q}(\mathcal{D}^{\text{rrs}})$-$\mathsf{Res}$ admits simple $O(n)$-size refutations of $\Xi(n)$.

Theorem 21. *The formulas $\Xi(n)$ have linear-size* dyn-$\mathsf{Q}(\mathcal{D}^{\text{rrs}})$-$\mathsf{Res}$ *refutations.*

Proof. We construct linear-size $\mathsf{Q}(\mathcal{D}^{\text{rrs}})$-$\mathsf{Res}$ refutations of $\Xi(n)[\neg a]$ and $\Xi(n)[a]$. Since a and $\neg a$ are $\mathcal{D}^{\text{rrs}}$-assignments for $\Xi(n)$, in dyn-$\mathsf{Q}(\mathcal{D}^{\text{rrs}})$-$\mathsf{Res}$ one can introduce the unit clauses $\{a\}$ and $\{\neg a\}$ by application of the reference rule, from which the empty clause is derived by a single resolution step. As the two referenced refutations are of linear size, so is the complete refutation.

It remains to construct the referenced refutations of $\Xi(n)[\neg a]$ and $\Xi(n)[a]$. We describe the case for $\Xi(n)[\neg a]$ – the other case is similar.

Note that the formula $\Xi(n)[\neg a]$ may be obtained from $\Psi(n)$ by adding the literal b to every clause, and then adding the unit clause $\{\neg b\}$ to the matrix. We make two observations. First, since the negative literal $\neg b$ occurs only in a unit clause, such a modification of $\Psi(n)$ cannot introduce any new existential $\mathcal{D}^{\text{rrs}}$ dependencies; no $\mathcal{D}^{\text{rrs}}$ path can go through variable b. As a result, Prop. 8 lifts from $\Psi(n)$ to $\Xi(n)[\neg a]$; that is, $(u_i, t_j) \notin \mathcal{D}^{\text{rrs}}(\Xi(n)[\neg a])$ for each $i, j \in [n]$ with $i \neq j$. Second, $\Psi(n)$ can be derived in $O(n)$ resolution steps from $\Xi(n)[\neg a]$ simply by resolving the unit clause $\{\neg b\}$ with every other clause (there are $O(n)$ clauses in $\Xi(n)[\neg a]$). It follows that $\mathsf{Q}(\mathcal{D}^{\text{rrs}})$-$\mathsf{Res}$ can refute $\Xi(n)[\neg a]$ in $O(n)$ steps by first deriving the clauses of $\Psi(n)$ and then replicating the refutation given in the proof of Theorem 9. □

Our final result is immediate from Theorems 20 and 21.

Theorem 22. dyn-$\mathsf{Q}(\mathcal{D}^{\text{rrs}})$-$\mathsf{Res}$ *is exponentially stronger than* $\mathsf{Q}(\mathcal{D}^{\text{rrs}})$-$\mathsf{Res}$.

7 Conclusions

We demonstrated that the use of dependency schemes in Q-resolution can yield exponentially shorter proofs. In line with experimental results, we thereby provided strong theoretical evidence supporting the notion that dependency schemes can be utilised for improved QBF solving. With further proof-theoretical results, we also demonstrated that the dynamic use of schemes has further potential for improved solving, beyond that of the static approach in existing implementations.

Finally, we suggest strongly that the results in this paper will lift to further QBF calculi, and most notably to expansion-based systems. We therefore highlight the potential for dependency schemes in expansion solving, and endorse the move in this direction mooted at the end of [14].

References

1. Benedetti, M., Mangassarian, H.: QBF-based formal verification: Experience and perspectives. Journal on Satisfiability, Boolean Modeling and Computation (JSAT) 5(1-4), 133–191 (2008)
2. Beyersdorff, O., Blinkhorn, J.: Dependency schemes in QBF calculi: Semantics and soundness. In: Principles and Practice of Constraint Programming (CP). pp. 96–112 (2016)
3. Beyersdorff, O., Chew, L., Janota, M.: Proof complexity of resolution-based QBF calculi. In: International Symposium on Theoretical Aspects of Computer Science (STACS). Leibniz International Proceedings in Informatics (LIPIcs), vol. 30, pp. 76–89 (2015)
4. Beyersdorff, O., Chew, L., Mahajan, M., Shukla, A.: Feasible interpolation for QBF resolution calculi. In: International Colloquium on Automata, Languages, and Programming (ICALP). pp. 180–192 (2015)
5. Beyersdorff, O., Chew, L., Mahajan, M., Shukla, A.: Are short proofs narrow? QBF resolution is not simple. In: Symposium on Theoretical Aspects of Computer Science (STACS). pp. 15:1–15:14 (2016)
6. Beyersdorff, O., Chew, L., Sreenivasaiah, K.: A game characterisation of tree-like Q-resolution size. In: LATA. pp. 486–498. Springer (2015)
7. Biere, A., Lonsing, F.: Integrating dependency schemes in search-based QBF solvers. In: International Conference on Theory and Applications of Satisfiability Testing (SAT). pp. 158–171. Springer (2010)
8. Buss, S.R.: Towards NP-P via proof complexity and search. Ann. Pure Appl. Logic 163(7), 906–917 (2012)
9. Cook, S.A., Nguyen, P.: Logical Foundations of Proof Complexity. Cambridge University Press (2010)
10. Cook, S.A., Reckhow, R.A.: The relative efficiency of propositional proof systems. Journal of Symbolic Logic 44(1), 36–50 (1979)
11. Egly, U., Kronegger, M., Lonsing, F., Pfandler, A.: Conformant planning as a case study of incremental QBF solving. In: Artificial Intelligence and Symbolic Computation (AISC'14). pp. 120–131 (2014)
12. Giunchiglia, E., Marin, P., Narizzano, M.: Reasoning with quantified boolean formulas. In: Handbook of Satisfiability, pp. 761–780. IOS Press (2009)
13. Giunchiglia, E., Narizzano, M., Tacchella, A.: Clause/term resolution and learning in the evaluation of quantified boolean formulas. Journal of Artificial Intelligence Research (JAIR) 26, 371–416 (2006)
14. Janota, M., Klieber, W., Marques-Silva, J., Clarke, E.M.: Solving QBF with counterexample guided refinement. Journal of Artificial Intelligence 234, 1–25 (2016)
15. Kleine Büning, H., Karpinski, M., Flögel, A.: Resolution for quantified boolean formulas. Information and Computation 117(1), 12–18 (1995)
16. Kontchakov, R., Pulina, L., Sattler, U., Schneider, T., Selmer, P., Wolter, F., Zakharyaschev, M.: Minimal module extraction from DL-lite ontologies using QBF solvers. In: International Joint Conference on Artificial Intelligence (IJCAI). pp. 836–841. AAAI Press (2009)
17. Krajíček, J.: Bounded Arithmetic, Propositional Logic, and Complexity Theory, Encyclopedia of Mathematics and Its Applications, vol. 60. Cambridge University Press, Cambridge (1995)
18. Liang, J.H., Ganesh, V., Zulkoski, E., Zaman, A., Czarnecki, K.: Understanding VSIDS branching heuristics in conflict-driven clause-learning SAT solvers. In: Haifa Verification Conference (HVC). pp. 225–241 (2015)

19. Lonsing, F.: Dependency Schemes and Search-Based QBF Solving: Theory and Practice. Ph.D. thesis, Johannes Kepler University (2012)
20. Lonsing, F., Egly, U., Seidl, M.: Q-resolution with generalized axioms. In: International Conference on Theory and Applications of Satisfiability Testing (SAT). pp. 435–452 (2016)
21. Moskewicz, M.W., Madigan, C.F., Zhao, Y., Zhang, L., Malik, S.: Chaff: Engineering an efficient SAT solver. In: Design Automation Conference (DAC). pp. 530–535 (2001)
22. Moskewicz, M.W., Madigan, C.F., Zhao, Y., Zhang, L., Malik, S.: Chaff: Engineering an efficient SAT solver. In: Design Automation Conference (DAC). pp. 530–535 (2001)
23. Peitl, T., Slivovsky, F., Szeider, S.: Long distance Q-resolution with dependency schemes. In: International Conference on Theory and Applications of Satisfiability Testing (SAT). pp. 500–518 (2016)
24. Rintanen, J.: Asymptotically optimal encodings of conformant planning in QBF. In: National Conference on Artificial Intelligence (AAAI). pp. 1045–1050. AAAI Press (2007)
25. Samer, M., Szeider, S.: Backdoor sets of quantified boolean formulas. Journal of Automated Reasoning 42(1), 77–97 (2009)
26. Samulowitz, H., Bacchus, F.: Using SAT in QBF. In: International Conference on Principles and Practice of Constraint Programming (CP). pp. 578–592 (2005)
27. Shacham, O., Zarpas, E.: Tuning the VSIDS decision heuristic for bounded model checking. In: International Workshop on Microprocessor Test and Verification (MTV). p. 75 (2003)
28. Silva, J.P.M.: The impact of branching heuristics in propositional satisfiability algorithms. In: Portugese Conference on Progress in Artificial Intelligence (EPIA). pp. 62–74 (1999)
29. Slivovsky, F.: Structure in #SAT and QBF. Ph.D. thesis, Vienna University of Technology (2015)
30. Slivovsky, F., Szeider, S.: Variable dependencies and Q-resolution. In: International Conference on Theory and Applications of Satisfiability Testing (SAT). pp. 269–284 (2014)
31. Slivovsky, F., Szeider, S.: Soundness of Q-resolution with dependency schemes. TCS 612, 83–101 (2016)
32. Van Gelder, A.: Variable independence and resolution paths for quantified boolean formulas. In: International Conference on Principles and Practice of Constraint Programming (CP). pp. 789–803. Springer (2011)
33. Zhang, L., Malik, S.: Conflict driven learning in a quantified boolean satisfiability solver. In: International Conference on Computer-aided Design (ICCAD). pp. 442–449 (2002)