# Definition-Based Dependency Schemes

**David Kattermann**
Institute for Symbolic AI, Johannes Kepler Universität Linz, Austria

**Clemens Hofstadler**
Institute for Symbolic AI, Johannes Kepler Universität Linz, Austria

**Martina Seidl**
Institute for Symbolic AI, Johannes Kepler Universität Linz, Austria

## Abstract

A variable in a quantified Boolean formula (QBFs) is *defined*, if its value is uniquely determined by some other variables. Such definitions are widely exploited in various techniques for QBF solving. In this work, we formalize the concept of using definitions for reducing variable dependencies by introducing a novel dependency scheme and investigate its proof-theoretic impact.

Our analysis shows that a definition-based dependency scheme is able to detect independencies other established dependency schemes cannot and that this can lead to exponentially shorter refutations. We further demonstrate that our scheme can be combined with any other scheme and that such a combined use can exponentially outperform using either scheme alone. Moreover, we study the dynamic application of our definition-based dependency scheme, which leads to another exponential speedup compared to the static application. Finally, we analyze the computational complexity of our dependency scheme and introduce a family of tractable variants.

**2012 ACM Subject Classification** Theory of computation → Proof complexity

**Keywords and phrases** Quantified Boolean formulas, Dependency schemes, Proof calculi

**Digital Object Identifier** 10.4230/LIPIcs.SAT.2026.22

**Funding** This work is supported by the LIT AI Lab funded by the State of Upper Austria and by the Austrian Science Fund (FWF) [10.55776/COE12].

## 1 Introduction

Quantified Boolean formulas (QBFs) are a natural extension of propositional formulas by existential and universal quantification of variables. The satisfiability problem for propositional formulas (SAT) is the prototypical NP-complete problem. As such it has experienced an enormous amount of research over the past decades, leading to the development highly efficient SAT solvers [10]. Similarly, deciding the truth of QBFs is a canonical PSPACE-complete problem, capturing many real-world applications [29].

A major challenge in QBF solving is handling the variable dependencies introduced by the quantifiers. These dependencies are, however, often overly restrictive, as a variable may actually not depend on all its preceding variables. This insight led to the development of *dependency schemes* [28], a class of methods to detect spurious dependencies in a QBF. As observed in [4], dependency schemes implicitly transform a QBF into an equivalent dependency QBF (DQBF) with dependency sets indicated by the given scheme. Numerous dependency schemes have been proposed in the literature [31, 6, 13] and their properties have been extensively studied over the past years [20, 2, 25, 3, 14]. Solvers incorporating dependency schemes, like DepQBF [21] or Qute [23, 24], have proven highly competitive.

This work presents a new dependency scheme that considerably differs from previous approaches. It is based on the following observation: QBFs arising from real-world applications are typically translated into *prenex conjunctive normal form* (PCNF), in which all quantifiers are placed in a *prefix* followed by a propositional formula in conjunctive normal form (CNF),


called *matrix*. This transformation is usually performed using Tseitin transformations [33], where a subformula $\phi$ is replaced by some freshly introduced *Tseitin variable* $t$ and a CNF encoding of the equivalence $t \leftrightarrow \phi$, known as *syntactic* or *gate definition* [15, 16, 35], is added to the formula. A widespread, though suboptimal, practice is to collect all Tseitin variables into a single existential quantifier block at the end of the prefix, thereby introducing lots of artificial dependencies [26]. Ideally, each Tseitin variable should depend only on the variables in its corresponding subformula. Generalizing Tseitin variables, we consider semantically *defined* variables [30]. A variable $x$ is said to be defined in a formula $\psi$ by a set $X \subseteq \mathrm{var}(\psi)$ when, in every satisfying assignment of $\psi$, the value of $x$ is uniquely determined by $X$. Restricting the defined variable $x$ to depend only on variables in $X$ can drastically reduce dependencies compared to the prefix order and even other dependency schemes.

Although there are several well-known preprocessing techniques in QBF solving using some notion of defined variables [35, 30, 26], there is comparatively little theoretical research on this. In this paper, we provide a theoretical foundation for definition-based dependency reductions by capturing such reductions as a dependency scheme $\mathcal{D}^{\mathsf{def}}$ and explore its proof theoretic impact. Our contributions are as follows.

### 1. Introduction of novel dependency schemes based on defined variables

In Section 3, we revisit the notion of defined variables from [30]. We explain how definitions can be used for reducing dependencies in QBFs, culminating in the definition of our novel dependency scheme $\mathcal{D}^{\mathsf{def}}$. Unlike other schemes, our construction does not rely on the matrix being in CNF. Moreover, our formulation allows us to combine $\mathcal{D}^{\mathsf{def}}$ with any other dependency scheme $\mathcal{D}^*$ to create a common generalization $\mathcal{D}^{*+\mathsf{def}}$. Due to the NP-completeness of computing $\mathcal{D}^{\mathsf{def}}$ (Theorem 29), we introduce a family of less general, but tractable dependency schemes $\mathcal{D}^{\mathsf{def}(\lambda)}$ in Section 6, using a syntactic notion of defined variables with gate definitions of size at most $\lambda$.

### 2. Proof theoretic comparison with other dependency schemes

In Section 4, we study the effect of adding $\mathcal{D}^{\mathsf{def}}$ and its variants to the proof systems $\mathsf{Q}-\mathsf{Res}$ [18] and $\mathsf{QU}-\mathsf{Res}$ [34]. We focus on the comparison with one of the most prominent dependency schemes $\mathcal{D}^{\mathsf{rrs}}$ [31] and show that the systems $\mathsf{Q}(\mathcal{D}^{\mathsf{def}})-\mathsf{Res}$ and $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs}})-\mathsf{Res}$ are incomparable by giving QBF families that admit polynomial-size proofs in one system, but require exponential-size proofs in the other (Propositions 14 and 15). We further demonstrate that the combined scheme $\mathcal{D}^{\mathsf{rrs}+\mathsf{def}}$ can exponentially outperform both schemes simultaneously. Our results also apply to the well-known scheme $\mathcal{D}^{\mathsf{std}}$ [28] and more recent generalizations of $\mathcal{D}^{\mathsf{rrs}}$ [6, 13]. Notably, all separations even work with $\mathcal{D}^{\mathsf{def}(\lambda)}$ for $\lambda \geq 4$ (Theorem 27).

### 3. New complexity bounds regarding the dynamic use of dependency schemes

Using the framework from [11, 3], we show in Section 5 that, by using $\mathcal{D}^{\mathsf{def}}$ in a dynamic way, one can achieve exponentially shorter proofs compared to a static use of even the combined scheme $\mathcal{D}^{\mathsf{rrs}+\mathsf{def}}$. In what seems to be the first result of this kind, we also prove that there are formulas that still require exponential proofs in $\mathsf{dynQU}(\mathcal{D}^{\mathsf{rrs}})-\mathsf{Res}$ (Proposition 21). Notably, the corresponding formula family admits constant size proofs in the static system $\mathsf{Q}(\mathcal{D}^{\mathsf{def}})-\mathsf{Res}$ (Proposition 14). As a side result, we obtain an exponential separation between $\mathsf{dynQ}(\mathcal{D}^{\mathsf{rrs}})-\mathsf{Res}$ and $\mathsf{dynQ}(\mathcal{D}^{\mathsf{std}})-\mathsf{Res}$ (Theorem 22), thereby generalizing the static version of this result from [11].

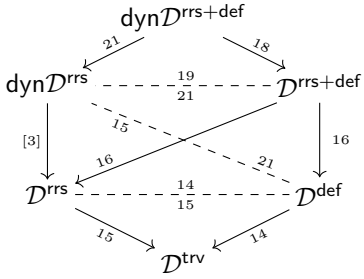


**Figure 1** Separations between static and dynamic proof systems for various schemes $\mathcal{D}$.

Figure 1 provides an overview of our proof-theoretic separations of various dependency schemes $\mathcal{D}$. An arrow from $\mathcal{D}$ to $\mathcal{D}'$ means that the proof system $\mathsf{Q}(\mathcal{D})-\mathsf{Res}$ is stronger than $\mathsf{Q}(\mathcal{D}')-\mathsf{Res}$. The notation $\mathsf{dyn}\mathcal{D}$ means that we refer to the system $\mathsf{dynQ}(\mathcal{D})-\mathsf{Res}$. Dashed lines indicate incomparability of the corresponding proof systems and numbers refer to the theorem numbering in this paper. All separations hold analogously for $\mathsf{QU}(\mathcal{D})-\mathsf{Res}$.

### Related Works

Exploiting definitions is a well-known technique in practical SAT and (D)QBF solving. For SAT, several efficient, incomplete techniques to detect gate definitions in CNF formulas have been proposed, see e.g. [15, 16]. The (D)QBF preprocessor HQSpre [35] has a subroutine to detect gate definitions and either eliminate defined variables or reduce their dependency sets. In [26], several existing tools are used to detect and move syntactically defined variables right after their last defining variable in the prefix. This approach already achieved significant solving speedups on the QBFEVAL'20 benchmarks for multiple QBF solvers, despite still potentially exhibiting spurious dependencies compared to reducing dependency sets to just defining variables. The works [30, 27] consider semantic definitions to obtain unique strategy functions, both in preprocessing and as part of a DQBF solving paradigm.

Complementary to these practical works, our focus lies on the complexity theoretic effect of such techniques. The works by Beyersdorff and Blinkhorn [2, 3, 12] extensively discuss the effects of adding dependency schemes to resolution- and expansion-based QBF proof calculi, usually with a focus on the schemes $\mathcal{D}^{\mathsf{std}}$ or $\mathcal{D}^{\mathsf{rrs}}$. Recently, two more general dependency schemes based on resolution paths have been proposed [6, 13].

## 2 Preliminaries

### Quantified Boolean Formulas

We study *quantified Boolean formulas* (QBFs) in *prenex normal form* (PNF), which can be defined as follows: A *literal* $\ell$ is a Boolean variable $x$ or its negation $\bar{x}$. We write $\mathrm{var}(\ell) = x$ if $\ell \in \{x, \bar{x}\}$. A clause is a disjunction of literals. A propositional formula $\psi$ in *conjunctive normal form* (CNF) is a conjunction of clauses. When convenient, we identify a clause with the set of its literals and a CNF formula with the set of its clauses. For a clause $C$ we write $\mathrm{var}(C) := \bigcup_{\ell \in C} \mathrm{var}(\ell)$, and for a CNF formula $\psi$ write $\mathrm{var}(\psi) := \bigcup_{C \in \psi} \mathrm{var}(C)$. A clause $C$ is a *tautology* if there is some variable $x$ with $x, \bar{x} \in C$. Tautologies can be removed from a CNF formula without affecting its truth value. Throughout this paper, we assume that

all tautologies have been removed from the considered formulas. A QBF in PNF has the form $\Pi.\psi$, where $\Pi = Q_1 x_1 \cdots Q_n x_n$ is the *prefix* with quantifiers $Q_i \in \{\forall, \exists\}$ over Boolean variables $x_i$ and $\psi$ is a propositional formula with $\mathrm{var}(\psi) = \{x_1, \ldots, x_n\} =: \mathrm{var}(\Psi)$ called the *matrix*. If $\psi$ is in CNF, then the QBF is said to be in *prenex conjunctive normal form* (PCNF). Note that we only consider closed formulas. A variable $x_i$ is *universal* if $Q_i = \forall$ and *existential* if $Q_i = \exists$. We write $U_\Psi$ and $E_\Psi$ for the set of universal and existential variables of $\Psi$, respectively. The prefix $\Pi$ induces a linear order on the variables with $x_i <_\Pi x_j$ if and only if $i < j$. For any variable $x \in E_\Psi$ the set $D_x \coloneqq \{u \in U_\Psi \colon u <_\Pi x\}$ is called the *dependency set* of $x$. Throughout this work, we assume QBFs to be in PNF, usually even PCNF, although examples may be presented in non-CNF form for better readability.

An assignment to a set of variables $X \subseteq \mathrm{var}(\psi)$ of a CNF $\psi$ is a function $\sigma \colon X \to \{\bot, \top\}$ from $X$ to the truth values $\bot$ (false) and $\top$ (true). We may interpret $\sigma$ as the set containing $x$ if $\sigma(x) = \top$ and $\bar{x}$ if $\sigma(x) = \bot$. The *restriction* of $\psi$ by $\sigma$, denoted $\psi[\sigma]$, is obtained by replacing in $\psi$ every $x \in X$ by $\sigma(x)$ and simplifying according to the semantics of propositional logic. The formula $\psi$ is *satisfied* by an assignment $\sigma$ to all variables $\mathrm{var}(\psi)$ if $\psi[\sigma] = \top$. In this case, $\sigma$ is a *satisfying assignment* for $\psi$. The restriction of a QBF $\Psi = \Pi.\psi$ by some assignment $\sigma$ to variables $X \subseteq \mathrm{var}(\Psi)$, denoted $\Psi[\sigma]$, is obtained by removing every $x \in X$ from the prefix along with its quantifier and replacing the matrix by $\psi[\sigma]$.

A *Skolem set* for a QBF $\Psi = \Pi.\psi$ is a set $\{f_x\}_{x \in E_\Psi}$ of *Skolem functions* $f_x$. Each $f_x$ maps assignments of the variables of dependency set $D_x$ to $\{\bot, \top\}$ such that for every assignment $\sigma$ to $U_\Psi$ the extension $\sigma'$, defined by $\sigma'(u) = \sigma(u)$ for $u \in U_\Psi$ and $\sigma'(z) \coloneqq f_z(\sigma|_{D_z})$ for $z \in E_\Psi$, is a satisfying assignment for $\psi$. A QBF $\Psi$ is *true* if it admits a Skolem set and *false* otherwise. We may identify Skolem functions $f_x$ as propositional formulas in the variables of $D_x$ such that substituting every existential variable $x$ in $\psi$ by $f_x$ yields a tautology. A Skolem function $f_x$ *depends on* a universal variable $u$, if there is some assignment to all universal variables for which changing the value of $u$ changes the value of $f_x$.

▶ **Example 1.** The QBF $\forall a, b \exists x. (\bar{a} \vee b \vee x) \wedge (a \vee \bar{b} \vee \bar{x})$ is true and a possible Skolem set is $\{f_x\}$, with $f_x = a$. The Skolem function $f_x$ does not depend on $b$. Note that the Skolem function is not unique, other possible choices are $f_x = \bar{b}$ or $f_x = a \wedge \bar{b}$.

QBFs can be naturally generalized to *dependency quantified Boolean formulas* (DQBF), where the total order of the prefix is turned into a partial one by making dependencies explicit. More precisely, a DQBF in PNF is of the form $\forall u_1, \ldots, u_m \exists x_1(D_{x_1}), \ldots, x_n(D_{x_n}).\psi$, where $D_{x_i} \subseteq \{u_1, \ldots, u_m\}$ is the *dependency set* of $x_i$. Note that a QBF $\Psi = \Pi.\psi$ can naturally be interpreted as a DQBF using $D_x = \{u \in U_\Psi \colon u <_\Pi x\}$. The notions of truth and falsity of QBFs lift to DQBFs. If there is some $u \in D_x$ such that $f_x$ does not depend on $u$, then $u$ can be omitted from $D_x$ without changing the truth of the DQBF.

## Dependency Schemes

An existential variable $x$ in a QBF or DQBF is said to *depend* on a universal $u$, if $u$ appears in the dependency set $D_x$ of $x$. As seen in Example 1, however, this does not yet imply that any Skolem function for $x$ also depends on $u$. Dependency schemes [28] are a way to filter out such spurious dependencies in a formula, usually by looking at the structure of the matrix. We only consider dependence of existential from universal variables.

▶ **Definition 2.** *A* dependency scheme *is a function $\mathcal{D}$ that maps any (D)QBF $\Psi$ to some set $\mathcal{D}_\Psi \subseteq \{(u, x) \in U_\Psi \times E_\Psi \colon u \in D_x\}$. For any $x \in E_\Psi$, write $\mathcal{D}_\Psi(x) \coloneqq \{u \in U_\Psi \colon (u, x) \in \mathcal{D}_\Psi\}$ and for $u \in U_\Psi$ write $\mathcal{D}_\Psi(u) \coloneqq \{u\}$. A dependency scheme $\mathcal{D}$ is* more general *than another dependency scheme $\mathcal{D}'$, denoted $\mathcal{D} \geq \mathcal{D}'$, if $\mathcal{D}_\Psi \subseteq \mathcal{D}'_\Psi$ for every $\Psi$. If neither $\mathcal{D} \geq \mathcal{D}'$ nor $\mathcal{D} \leq \mathcal{D}'$, then $\mathcal{D}$ and $\mathcal{D}'$ are* incomparable.

A dependency scheme $\mathcal{D}$ transforms a (D)QBF $\Psi$ into a DQBF $\mathcal{D}(\Psi)$ by replacing each dependency set $D_x$ by its subset $\mathcal{D}_\Psi(x)$. The following property ensures that this transformation preserves truth, i.e., that $\Psi$ and $\mathcal{D}(\Psi)$ are equivalent [4]. It suffices to formulate this for true formulas, as reducing dependency sets can never make a false formula true.

▶ **Definition 3** ([2]). *A dependency scheme $\mathcal{D}$ is* fully exhibited *if every true (D)QBF $\Psi$ admits a Skolem set $\{f_x\}_{x\in E_\Psi}$ in which each $f_x$ only depends on variables $u$ with $(u,x)\in\mathcal{D}_\Psi$.*

If $\mathcal{D}$ is a fully exhibited dependency scheme, then any scheme $\mathcal{D}' \leq \mathcal{D}$ is also fully exhibited [6, Proposition 3.6]. We collect some examples of well-known dependency schemes, all of which are fully exhibited [2, Theorem 18].

▶ **Example 4.** Fix some DQBF $\Psi = \Pi.\psi$. The most basic dependency scheme is the *trivial dependency scheme* $\mathcal{D}^{\mathsf{trv}}$, which gives exactly the dependencies indicated by the prefix, i.e., $\mathcal{D}^{\mathsf{trv}}_\Psi = \{(u,x)\in U_\Psi\times E_\Psi : u\in D_x\}$. By definition, every dependency scheme is more general than $\mathcal{D}^{\mathsf{trv}}$. The *standard dependency scheme* $\mathcal{D}^{\mathsf{std}}$ [28] is defined by $(u,x)\in\mathcal{D}^{\mathsf{std}}_\Psi$ if and only if $u\in D_x$ and there is a sequence of clauses $C_1,\dots,C_k\in\psi$ such that $u\in\mathrm{var}(C_1)$, $x\in\mathrm{var}(C_k)$ and for each $i=1,\dots,k-1$ there is some existential variable $y$ with $u\in D_y$ and $y\in\mathrm{var}(C_i)\cap\mathrm{var}(C_{i+1})$. Its generalization, the *reflexive resolution path dependency scheme* $\mathcal{D}^{\mathsf{rrs}}$ [31, 32] is defined by $(u,x)\in\mathcal{D}^{\mathsf{rrs}}_\Psi$ if and only if there exists a *resolution path* connecting $u$ and $x$. A resolution path is a sequence of clauses $C_1,\dots,C_k\in\psi$ with $u\in C_1, \bar{u}\in C_k$, and for which there are existential literals $\ell_1,\dots,\ell_{k-1}$ such that: (1) $u\in D_{\mathrm{var}(\ell_i)}$ and $\ell_i\in C_i, \bar{\ell}_i\in C_{i+1}$ for $i=1,\dots,k-1$, (2) $\mathrm{var}(\ell_i)\neq\mathrm{var}(\ell_{i+1})$ for $i=1,\dots,k-2$, and (3) $x=\mathrm{var}(\ell_i)$ for some $i=1,\dots,k-1$.

## Proof Calculi

Informally, a (D)QBF *proof calculus* $\mathsf{P}$ is a system of inference rules for deriving new clauses from a (D)QBF $\Psi$ in PCNF. We focus on proofs for false formulas. A $\mathsf{P}$-*refutation* of $\Psi$ is a derivation of the empty clause from $\Psi$ using the rules of $\mathsf{P}$. The calculus $\mathsf{P}$ is *sound* if it only refutes false formulas and it is (refutationally) *complete* if every false formula admits a $\mathsf{P}$-refutation. The *size* of a $\mathsf{P}$-refutation $\pi$, denoted by $|\pi|$, is the number of rule applications. A proof calculus $\mathsf{P}$ *simulates* another calculus $\mathsf{Q}$, if there exists a polynomial $p$ such that for every $\mathsf{Q}$-refutation $\pi$ of any false (D)QBF $\Psi$, there is a $\mathsf{P}$-refutation of $\Psi$ with size at most $p(|\pi|)$. If additionally $\mathsf{Q}$ does not simulate $\mathsf{P}$, then $\mathsf{P}$ is *stronger* than $\mathsf{Q}$. If neither of them simulates the other, then $\mathsf{P}$ and $\mathsf{Q}$ are *incomparable*.

There exists a whole plethora of (D)QBF proof calculi in the literature, see [1] for a survey. We focus on generalizations of one of the most studied ones, $\mathsf{Q}{-}\mathsf{Res}$ [18] (short for *Q–Resolution*). Given an arbitrary dependency scheme $\mathcal{D}$, we consider the parametrized proof calculus $\mathsf{Q}(\mathcal{D}){-}\mathsf{Res}$ [32], defined by the rules in Figure 2. Its extension $\mathsf{QU}(\mathcal{D}){-}\mathsf{Res}$ is defined analogously, but additionally allows resolution over universal variables. The calculus $\mathsf{Q}{-}\mathsf{Res}$ corresponds to $\mathsf{Q}(\mathcal{D}^{\mathsf{trv}}){-}\mathsf{Res}$ and analogously $\mathsf{QU}{-}\mathsf{Res}$ corresponds to $\mathsf{QU}(\mathcal{D}^{\mathsf{trv}}){-}\mathsf{Res}$. Note that $\mathsf{QU}(\mathcal{D}){-}\mathsf{Res}$ simulates $\mathsf{Q}(\mathcal{D}){-}\mathsf{Res}$ for any $\mathcal{D}$. Moreover, $\mathsf{Q}(\mathcal{D}){-}\mathsf{Res}$ simulates $\mathsf{Q}(\mathcal{D}'){-}\mathsf{Res}$ for all $\mathcal{D}'\geq\mathcal{D}$ [11, Section 3.2], and the same holds for $\mathsf{QU}{-}\mathsf{Res}$. If $\mathcal{D}$ is fully exhibited, then $\mathsf{Q}(\mathcal{D}){-}\mathsf{Res}$ and $\mathsf{QU}(\mathcal{D}){-}\mathsf{Res}$ are sound [2, 4] for QBF and DQBF. Both calculi are also refutationally complete for QBF, but incomplete for DQBF as observed in [9].

| | | |
|---|---|---|
| (Axiom) | $\frac{}{C}$ | where $C$ is a clause of $\Psi$ |
| (Resolution) | $\frac{C \vee x \quad D \vee \bar{x}}{C \vee D}$ | where $x$ is an existential variable and $C \vee D$ is not a tautology |
| (Reduction) | $\frac{C \vee \ell}{C}$ | where $u = \mathrm{var}(\ell)$ is universal with $(u, x) \notin \mathcal{D}_\Psi$ for all $x \in \mathrm{var}(C)$ |

**Figure 2** Rules of the $\mathsf{Q}(\mathcal{D})-\mathsf{Res}$ calculus.

## 3 Dependency Reductions for Defined Variables

In this section, we investigate the notion of defined variables from [30, 19] and demonstrate how it can be used to reduce dependencies in a (D)QBF. Intuitively, an existential variable $x$ is *defined* in a formula if its value is uniquely determined by the values of some other variables. This occurs, for example, when Tseitin transformation [33] is used to transform the matrix into CNF. The following notion generalizes this in an semantic way.

▶ **Definition 5.** *Let $\psi$ be a propositional formula, $x \in \mathrm{var}(\psi)$, and $X \subseteq \mathrm{var}(\psi)$ with $x \notin X$. The variable $x$ is* (semantically) defined *by $X$ in $\psi$, if for all satisfying assignments $\sigma, \tau$ of $\psi$ with $\sigma|_X = \tau|_X$, we have $\sigma(x) = \tau(x)$. In this case, $X$ is a* defining set *for $x$ in $\psi$. We define* $\mathrm{def}(\psi, x) \coloneqq \{X \subseteq \mathrm{var}(\psi) \setminus \{x\} \colon x \text{ is defined by } X \text{ in } \psi\}$.

The concept of defining sets allows us to reduce the dependencies of defined variables in a sound manner, i.e., while preserving the truth of the formula. Recall that a QBF $\Psi = \Pi.\psi$ can naturally be viewed as a DQBF with dependency sets $D_x = \{u \in U_\Psi \colon u <_\Pi x\}$ for existential variables $x$. For convenience, we also write $D_u = \{u\}$ for universals $u \in U_\Psi$.

Suppose an existential variable $x$ in a (D)QBF $\Psi = \Pi.\psi$ is defined by a set $X \subseteq \mathrm{var}(\psi)$ in $\psi$. Then the variable $x$ should only depend on variables appearing in $X$. So we aim to reduce the dependency set $D_x$ using $X$. Note that we cannot simply take $D_x \cap X$ as new dependency set, since $X$ may contain existential variables and dependency sets must consist only of universals. Instead, we replace each existential $z \in X$ by its dependency set $D_z$ and reduce $D_x$ to $D'_x = D_x \cap \bigcup_{z \in X} D_z$, which is a subset of $U_\Psi$. Let us consider an example.

▶ **Example 6.** Consider the QBF

$$\Psi = \forall a \exists x \forall b, c \exists y. \, (a \vee c \vee \bar{x}) \wedge (\bar{a} \vee \bar{c} \vee x) \wedge \underbrace{(\bar{y} \vee b) \wedge (\bar{y} \vee x) \wedge (y \vee \bar{b} \vee \bar{x})}_{\delta}.$$

The prefix induces the dependency sets $D_x = \{a\}$ and $D_y = \{a, b, c\}$. The variable $y$ is defined by $X = \{b, x\}$, as the clauses $\delta$ encode $y \leftrightarrow (b \wedge x)$. Using the defining set $X$, we can reduce the dependencies of $y$ to $D'_y = D_y \cap (\{b\} \cup \{a\}) = \{a, b\}$ without changing the truth of the formula. Indeed, $\Psi$ admits a Skolem set $\{f_x, f_y\}$ with $f_y = a \wedge b$ and $f_x = a$.

A formula might contain multiple defining sets for the same variable $x$. In particular, if $x$ is defined by some set $X$, then it is also defined by every superset $X' \supseteq X$. To reduce the dependencies for $x$, it is therefore natural to consider the intersection over all defining sets. Extending the construction discussed above, we reduce $D_x$ to

$$D'_x = D_x \cap \bigcap_{X \in \mathrm{def}(\psi, x)} \bigcup_{z \in X} D_z. \tag{1}$$

In other words, we remove those variables $u$ from $D_x$ for which there is some defining set $X$ for $x$ in which no variable depends on $u$. Note that this reduction is trivially sound for false formulas, as decreasing dependencies cannot turn a false formula into a true one. In particular, if the matrix $\psi$ is unsatisfiable then the empty set is defining for any variable, so we always obtain $D'_x = \varnothing$. We will show that the reduction is also sound for true formulas in Theorem 10 below. In Theorem 29 we show that finding the set $D'_x$ is in general NP-complete. Before formalizing our observations into a new dependency scheme, we illustrate the idea by another example.

▶ **Example 7.** Consider the DQBF

$$\Psi = \forall u, v, w \exists x(\{u, v, w\}), y(\{v\}), z(\{w\}).\ (x \leftrightarrow (u \vee v \vee y)) \wedge (x \leftrightarrow (u \vee w \vee z))\,.$$

The variable $x$ is defined by $\{u, v, y\}$ and $\{u, w, z\}$, as well as by all their supersets. The dependency reduction according to (1) gives $D'_x = \{u\}$. Note that this set is itself not defining for $x$. Nevertheless, reducing $D_x$ to $D'_x$ does not change the truth of the formula. Indeed, $\Psi$ admits a Skolem set $\{f_x, f_y, f_z\}$ with $f_x = f_y = f_z = \top$. This also illustrates that the reduced dependency set $D'_x$ is not necessarily minimal.

The above observations can be generalized by applying the reduction (1) not to the original dependency sets, but to the ones obtained from the application of another dependency scheme $\mathcal{D}^*$. Based on this idea, we now introduce a family of dependency schemes.

▶ **Definition 8.** *For any $\mathcal{D}^*$ dependency scheme its* definition-based generalization $\mathcal{D}^{*+\mathsf{def}}$ *is defined as follows: For any (D)QBF $\Psi = \Pi.\psi$, we have $(u, x) \in \mathcal{D}^{*+\mathsf{def}}_\Psi$ if and only if $u \in \mathcal{D}^*_\Psi(x)$ and for each $X \in \mathrm{def}(\psi, x)$ there is some $z \in X$ with $u \in \mathcal{D}^*_\Psi(z)$. The* definition-based dependency scheme *is $\mathcal{D}^{\mathsf{def}} := \mathcal{D}^{\mathsf{trv}+\mathsf{def}}$, i.e., the special case where $\mathcal{D}^* = \mathcal{D}^{\mathsf{trv}}$.*

Note that, unlike for most other dependency schemes, the scheme $\mathcal{D}^{\mathsf{def}}$ does not rely on the matrix being in CNF due to our general notion of defined variables. Our construction of $\mathcal{D}^{*+\mathsf{def}}$ corresponds to the application of $\mathcal{D}^{\mathsf{def}}$, respectively the dependency reduction from (1), after the application of $\mathcal{D}^*$. In particular it holds $\mathcal{D}^{*+\mathsf{def}}_\Psi \subseteq \mathcal{D}^*_\Psi \cap \mathcal{D}^{\mathsf{def}}_\Psi$ for any $\Psi$ and thus $\mathcal{D}^{*+\mathsf{def}}$ is always more general than both $\mathcal{D}^*$ and $\mathcal{D}^{\mathsf{def}}$. The dependencies of non-defined variables remain unchanged in $\mathcal{D}^{*+\mathsf{def}}$ compared to $\mathcal{D}^*$. Interpreting dependency schemes as maps from (D)QBF to DQBF [4], our above definition gives $\mathcal{D}^{*+\mathsf{def}} = \mathcal{D}^{\mathsf{def}} \circ \mathcal{D}^*$. We provide a small example to illustrate that the order in which the schemes are applied can make a difference. It is a priori not clear which application order is best. A solution to this is to alternate the applications several times until no more changes occur.

▶ **Example 9.** Consider the QBF $\Psi = \forall u, v \exists x, y.\ (x \leftrightarrow (u \vee y)) \wedge (x \vee v)$ and some $\mathcal{D}^*$ with $\mathcal{D}^*_\Psi(x) = \{u, v\}$ and $\mathcal{D}^*_\Psi(y) = \varnothing$. Then we get $\mathcal{D}^{*+\mathsf{def}}_\Psi(x) = \{u\}$. In contrast, $\mathcal{D}^{\mathsf{def}}$ alone would not change neither dependency set, so applying $\mathcal{D}^*$ after $\mathcal{D}^{\mathsf{def}}$ does not remove $v$ from $D_x$.

On the other hand, consider $\mathcal{D}^* = \mathcal{D}^{\mathsf{rrs}}$ and the QBF $\Psi$ with prefix $\forall u, v, w \exists x, y$ and matrix $(u \vee \bar{y}) \wedge (y \vee \bar{u}) \wedge (v \vee y \vee w) \wedge (\bar{y} \vee x \vee w) \wedge (\bar{x} \vee \bar{v} \vee w)$. One can easily check $(v, x), (v, y) \in \mathcal{D}^{\mathsf{rrs}}_\Psi$ and $\mathcal{D}^{\mathsf{def}}_\Psi(y) = \{u\}$, but $x$ is not defined in the matrix. Hence, $v$ only gets removed from $D_x$ if $\mathcal{D}^{\mathsf{rrs}}$ is applied after $\mathcal{D}^{\mathsf{def}}$.

Let us now prove that $\mathcal{D}^{*+\mathsf{def}}$ preserves the truth of formulas, whenever $\mathcal{D}^*$ does so.

▶ **Theorem 10.** *For any fully exhibited dependency scheme $\mathcal{D}^*$, the dependency scheme $\mathcal{D}^{*+\mathsf{def}}$ is fully exhibited.*

**Proof.** Removing dependencies can never turn a false formula into a true one, thus it is enough to show that true formulas remain true. To this end, fix a true DQBF $\Psi = \Pi.\psi$. Since $\mathcal{D}^*$ is assumed to be fully exhibited, there exists a Skolem set $\{f_z\}_{z\in E_\Psi}$ for $\Psi$ in which every $f_z$ only depends on variables $u \in \mathcal{D}^*_\Psi(z)$. Now let $x \in E_\Psi$ be a defined variable and fix some $u \in \mathcal{D}^*_\Psi(x) \setminus \mathcal{D}^{*+\mathsf{def}}_\Psi(x)$. By the construction of $\mathcal{D}^{*+\mathsf{def}}$ this means that there is some defining set $X$ for $x$ with $u \notin \bigcup_{z\in X} \mathcal{D}^*_\Psi(z)$. We have to show that the Skolem function $f_x$ does not depend on $u$. Fix some assignment $\sigma$ to all universal variables. Since $\{f_z\}_z$ is a Skolem set, the matrix $\psi$ must be satisfied under the extended assignment given by $\sigma'(z) = f_z(\sigma|_{D_z})$ for any existential $z$. Let $\tau$ be the assignment to all universals that agrees with $\sigma$, but flips the value of $u$, and let $\tau'$ be its extension to all variables. Due to $u \notin \bigcup_{z\in X} \mathcal{D}^*_\Psi(z)$, we have $u \notin X$. Moreover, since each $f_z$ only depends on variables in $\mathcal{D}^*_\Psi(z)$ it holds $\sigma'(z) = f_z(\sigma|_{D_z}) = f_z(\tau|_{D_z}) = \tau'(z)$ for every existential $z \in X$. Together this shows $\sigma'|_X = \tau'|_X$. Since $x$ is defined by $X$, it follows that $f_x(\sigma|_{D_x}) = \sigma'(x) = \tau'(x) = f_x(\tau|_{D_x})$. So $f_x$ does not depend on $u$. ◀

Full exhibition of $\mathcal{D}$ is sufficient for soundness of the (D)QBF calculi $\mathsf{Q}(\mathcal{D})-\mathsf{Res}$ and $\mathsf{QU}(\mathcal{D})-\mathsf{Res}$. Note that their completeness is not given for DQBF [4, 9].

▶ **Corollary 11.** *For any fully exhibited scheme $\mathcal{D}^*$, the (D)QBF proof calculi $\mathsf{Q}(\mathcal{D}^{*+\mathsf{def}})-\mathsf{Res}$ and $\mathsf{QU}(\mathcal{D}^{*+\mathsf{def}})-\mathsf{Res}$ are sound. For QBF they are also complete.*

## 4 Exponential Separations

In this section, we compare $\mathcal{D}^{\mathsf{def}}$ with other dependency schemes in terms of proof complexity. We provide exponential separations, showing that using one scheme can yield exponentially shorter proofs than another in $\mathsf{Q}-\mathsf{Res}$ or $\mathsf{QU}-\mathsf{Res}$. We always state the strongest possible separation by using the weakest proof system admitting polynomial refutations and the strongest system requiring exponential refutations. For notational convenience, we stick to a QBF presentation in this section, but all results naturally carry over to the DQBF setting.

We often use the following two observations.

▶ **Lemma 12** (cf. [14, Proposition 3.3])**.** *Let $\mathcal{D}, \mathcal{D}'$ be dependency schemes and $\Psi$ a QBF with $\mathcal{D}_\Psi = \mathcal{D}'_\Psi$. Then any $\mathsf{Q}(\mathcal{D})-\mathsf{Res}$ refutation of $\Psi$ is also a $\mathsf{QU}(\mathcal{D}')-\mathsf{Res}$ refutation.*

Analogous to the restriction of a QBF by an assignment $\sigma$ to some existential variables, the restriction of a refutation $\pi$, denoted $\pi[\sigma]$, is obtained by substituting the assigned values in each clause, simplifying, and discarding true clauses [5].

▶ **Lemma 13** ([5, Proposition 3.2])**.** *Let $\pi$ be a $\mathsf{Q}-\mathsf{Res}$ (resp. $\mathsf{QU}-\mathsf{Res}$) refutation of a QBF $\Psi$ and let $\sigma$ be a partial assignment to some existential variables of $\Psi$. Then $\pi[\sigma]$ is a $\mathsf{Q}-\mathsf{Res}$ (resp. $\mathsf{QU}-\mathsf{Res}$) refutation of $\Psi[\sigma]$.*

For our first exponential separation, we consider the formula family $\mathsf{QParity}$ [8]. Using the notation $\mathrm{xor}(a,b,c) \coloneqq (\bar{a} \vee \bar{b} \vee \bar{c}) \wedge (\bar{a} \vee b \vee c) \wedge (a \vee \bar{b} \vee c) \wedge (a \vee b \vee \bar{c})$ to encode a CNF of $a \leftrightarrow (b \oplus c)$, we define for $n \geq 2$

$$\mathsf{QP}_n \coloneqq \exists x_1, \ldots, x_n \forall u \exists t_2, \ldots, t_n. \mathrm{xor}(t_2, x_1, x_2) \wedge \bigwedge_{i=3}^{n} \mathrm{xor}(t_i, x_i, t_{i-1}) \wedge (u \vee t_n) \wedge (\bar{u} \vee \bar{t}_n).$$

▶ **Proposition 14.** *The QBF family $\{\mathsf{QP}_n\}_{n\geq 2}$ admits constant size $\mathsf{Q}(\mathcal{D}^{\mathsf{def}})-\mathsf{Res}$ refutations, but requires $\mathsf{QU}(\mathcal{D}^{\mathsf{rrs}})-\mathsf{Res}$ refutations of size exponential in $n$.*

**Proof.** Note that each $t_i$ for $i \geq 2$ is defined by $\{x_1, \dots, x_i\}$, as the structure of the matrix encodes $t_i \leftrightarrow \bigoplus_{j=1}^{i} x_j$. Since the variables $x_j$ do not have any dependencies, taking the intersection over all defining sets for $t_i$ always yields the empty set, and so $\mathcal{D}^{\mathsf{def}}_{\mathsf{QP}_n} = \varnothing$. In particular, we have $(u, t_n) \notin \mathcal{D}^{\mathsf{def}}_{\mathsf{QP}_n}$. Thus, in $\mathsf{Q}(\mathcal{D}^{\mathsf{def}})\text{−}\mathsf{Res}$ we are allowed to reduce $u$ from the last two clauses and derive the empty clause via resolution over $t_n$. This is a refutation in only three steps, independent of $n$. On the other hand, we have $\mathcal{D}^{\mathsf{rrs}}_{\mathsf{QP}_n} = \mathcal{D}^{\mathsf{trv}}_{\mathsf{QP}_n}$, see [14, Section 4.1]. Hence, any $\mathsf{QU}(\mathcal{D}^{\mathsf{rrs}})\text{−}\mathsf{Res}$ refutation of $\mathsf{QP}_n$ is actually a $\mathsf{QU}\text{−}\mathsf{Res}$ refutation by Lemma 12, which is known to require exponential size [8, Theorem 14]. ◀

Another interesting example is KBKF [18]. This family requires exponential $\mathsf{QU}(\mathcal{D}^{\mathsf{std}})\text{−}\mathsf{Res}$ refutations [11], but a similar proof as above shows that it admits polynomial $\mathsf{Q}(\mathcal{D}^{\mathsf{def}})\text{−}\mathsf{Res}$ refutations. KBKF also admits polynomial $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs}})\text{−}\mathsf{Res}$ refutations [3].

Next, we show that $\mathcal{D}^{\mathsf{rrs}}$ can also outperform $\mathcal{D}^{\mathsf{def}}$ and even its combination with $\mathcal{D}^{\mathsf{std}}$. For this we consider the formula family Equality [5] given by

$$\mathsf{EQ}_n \coloneqq \exists y_1, \dots, y_n \forall u_1, \dots, u_n \exists s_1, \dots, s_n.\ (s_1 \vee \dots \vee s_n) \wedge \bigwedge_{i=1}^{n} (\bar{s}_i \vee y_i \vee u_i) \wedge (\bar{s}_i \vee \bar{y}_i \vee \bar{u}_i).$$

▶ **Proposition 15.** *The QBF family* $\{\mathsf{EQ}_n\}_{n \geq 1}$ *admits* $\mathcal{O}(n)$ *sized* $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs}})\text{−}\mathsf{Res}$ *refutations, but requires* $\mathsf{QU}(\mathcal{D}^{\mathsf{std+def}})\text{−}\mathsf{Res}$ *refutations of size exponential in* $n$.

**Proof.** It can be verified that $\mathcal{D}^{\mathsf{rrs}}_{\mathsf{EQ}_n} = \varnothing$, but $\mathcal{D}^{\mathsf{std}}_{\mathsf{EQ}_n} = \mathcal{D}^{\mathsf{trv}}_{\mathsf{EQ}_n}$, see [11]. Thus, in $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs}})\text{−}\mathsf{Res}$ one can reduce all $u_i$'s from the formula and resolve over each $y_i$ and $s_i$ to get a refutation of size $\mathcal{O}(n)$. On the other hand, we still have $\mathcal{D}^{\mathsf{std+def}}_{\mathsf{EQ}_n} = \mathcal{D}^{\mathsf{trv}}_{\mathsf{EQ}_n}$, as none of the variables $s_i$ is defined. So the claim follows from Lemma 12, since $\mathsf{EQ}_n$ requires exponential $\mathsf{QU}\text{−}\mathsf{Res}$ refutations by [5, Theorem 3.5]. ◀

By combining the two families QParity and Equality, one can show that the combination of the two dependency schemes $\mathcal{D}^{\mathsf{rrs}}$ and $\mathcal{D}^{\mathsf{def}}$ can outperform using only one of them.

▶ **Corollary 16.** *There is a QBF family that admits linear size* $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs+def}})\text{−}\mathsf{Res}$ *refutations, but requires both exponential* $\mathsf{QU}(\mathcal{D}^{\mathsf{rrs}})\text{−}\mathsf{Res}$ *and* $\mathsf{QU}(\mathcal{D}^{\mathsf{def}})\text{−}\mathsf{Res}$ *refutations.*

**Proof.** For $n \geq 2$, denote $P_n \coloneqq \mathrm{xor}(t_2, x_1, x_2) \wedge \bigwedge_{i=3}^{n} \mathrm{xor}(t_i, x_i, t_{i-1})$ and let $E_n$ be the matrix of $\mathsf{EQ}_n$. Consider the QBF family $\{\Psi_n\}_{n \geq 2}$ with

$$\Psi_n \coloneqq \exists a, x_1, \dots, x_n, y_1, \dots, y_n \forall u \exists t_2, \dots, t_n \forall u_1, \dots, u_n \exists s_1, \dots, s_n.$$
$$P_n \wedge (u \vee t_n \vee a) \wedge (\bar{u} \vee \bar{t}_n \vee a) \wedge (E_n \vee \bar{a}).$$

Using a similar argumentation as in the proofs of Proposition 14 and Proposition 15, one can verify that $\mathcal{D}^{\mathsf{rrs}}_{\Psi_n} = \{(u, t_i) \colon i = 2, \dots, n\}$ and $\mathcal{D}^{\mathsf{def}}_{\Psi_n} = \{(u_i, s_j) \colon i, j = 1, \dots, n\}$. Since the intersection of the schemes is empty, it follows that $\mathcal{D}^{\mathsf{rrs+def}}_{\Psi_n} = \varnothing$. The latter implies that $\Psi_n$ admits linear $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs+def}})\text{−}\mathsf{Res}$ refutations, because we can reduce $u$, resolve over $t_n$ and $a$, and finally refute $E_n$ like in the proof of Proposition 15. However, neither $\mathcal{D}^{\mathsf{rrs}}$ nor $\mathcal{D}^{\mathsf{def}}$ alone are enough to ensure this: By Lemma 13, any formula restricted by a (partial) assignment to some of its existential variables admits a refutation of at most the same size as the original formula. Restricting $\Psi_n$ by the assignment $a = \bot$ leads to the formula $\mathsf{QP}_n$, which requires exponential $\mathsf{QU}(\mathcal{D}^{\mathsf{rrs}})\text{−}\mathsf{Res}$ refutations by Proposition 14. The assignment $a = \top$ gives the true formula $\exists x_1, \dots, x_n, t_2, \dots, t_n . P_n$ conjunctively combined with $\mathsf{EQ}_n$. Since these two parts do not share any variables and true formulas cannot be refuted in a sound calculus, this formula still requires exponential $\mathsf{QU}(\mathcal{D}^{\mathsf{def}})\text{−}\mathsf{Res}$ refutations by Proposition 15. ◀

Our results even transfer to some strong generalizations of $\mathcal{D}^{\mathsf{rrs}}$. Recall that a dependency $(u, x)$ in $\mathcal{D}^{\mathsf{rrs}}$ is defined through the existence of a resolution path connecting $u$ and $x$. The idea behind the generalizations of $\mathcal{D}^{\mathsf{rrs}}$ is that a resolution path does not exhibit an actual dependency between $u$ and $x$, if some of its intermediate clauses are always satisfied.

In [6], a whole family of generalizations of $\mathcal{D}^{\mathsf{rrs}}$ based on this idea is introduced, the so-called *implication–free dependency schemes*. By [6, Lemma 5.9], the most general example is $\mathcal{D}^{\mathsf{F}}(R_{\models}, \infty)$. In this scheme, resolution paths in the matrix $\psi$ are not considered, if the involved clauses contain some literals $\ell_1, \ldots, \ell_k$ that are *free*, in the sense that $D_{\mathrm{var}(\ell_i)} = \varnothing$, and for which the implication $\psi \to \bigvee_{i=1}^{k} \ell_i$ is valid. Another generalization of $\mathcal{D}^{\mathsf{rrs}}$ is $\mathcal{D}^{\forall\mathsf{pure}}$ [13]. Here, a resolution path connecting $u$ and $x$ is discarded, if $u$ or $\bar{u}$ appears in any intermediate clause of the path.

By careful inspection, one can check that the formulas of the QParity family, used in Proposition 14, do not exhibit any non-trivial independencies in $\mathcal{D}^{\mathsf{F}}(R_{\models}, \infty)$ or $\mathcal{D}^{\forall\mathsf{pure}}$. This is because we can always find a resolution path from the only universal variable $u$ to any of the existential variables $t_i$ in which the free variables $x_j$ appear only with positive polarity and $u$ does not appear in any intermediate clauses. Similar considerations show that the more general schemes do also not alter the dependencies in Proposition 15 and Corollary 16 compared to $\mathcal{D}^{\mathsf{rrs}}$. Thus, we obtain the following result.

▶ **Theorem 17.** *Let $\mathcal{D}^*$ be any of the implication-free dependency schemes or $\mathcal{D}^* = \mathcal{D}^{\forall\mathsf{pure}}$. Then the proof systems $\mathsf{QU}(\mathcal{D}^*)-\mathsf{Res}$ and $\mathsf{QU}(\mathcal{D}^{\mathsf{def}})-\mathsf{Res}$ are incomparable and the system $\mathsf{QU}(\mathcal{D}^{*+\mathsf{def}})-\mathsf{Res}$ is stronger than both of them.*

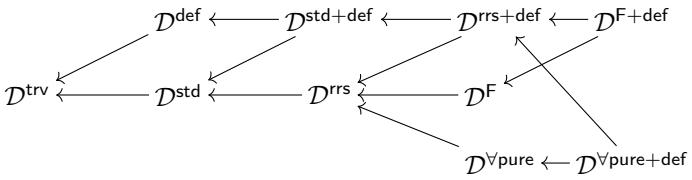


**Figure 3** Relationships between various dependency schemes with arrows from more to less general schemes. $\mathcal{D}^{\mathsf{F}}$ stands for any of the implication-free dependency schemes ($\neq \mathcal{D}^{\mathsf{rrs}}$).

Figure 3 summarizes the relationships between the discussed dependency schemes $\mathcal{D}$ and the separations between of the corresponding proof systems $\mathsf{QU}(\mathcal{D})-\mathsf{Res}$. Two schemes, respectively the corresponding proof systems, are incomparable, if there is no directed path between them in the figure, for example, $\mathcal{D}^{\mathsf{std+def}}$ and $\mathcal{D}^{\forall\mathsf{pure}}$ are incomparable.

## 5 Dynamic Dependency Schemes

A dynamic use of dependency schemes was introduced in [11] to model the recomputation of dependencies during the solving procedure of a QCDCL-based solver. In this section, we demonstrate that using $\mathcal{D}^{\mathsf{def}}$ dynamically can yield exponentially shorter proofs compared to using it in the classical static way. We further generalize the results from the previous section by showing that the static use of $\mathcal{D}^{\mathsf{def}}$ can even outperform the dynamic use of $\mathcal{D}^{\mathsf{rrs}}$.

In our proof-theoretic framework, the dynamic use of a dependency scheme $\mathcal{D}$ is realized as the QBF proof system $\mathsf{dynQ}(\mathcal{D})-\mathsf{Res}$, an extension of $\mathsf{Q}(\mathcal{D})-\mathsf{Res}$ by the so-called *reference rule* defined in Figure 4 below. This rule allows to introduce clauses that are learned

from *subproofs*, which refute restrictions of the formula under an assignment to some of its variables. The size of a $\mathsf{dynQU}(\mathcal{D})\text{–}\mathsf{Res}$ refutation is defined recursively by adding the sizes of the subproofs to the overall proof size. If the scheme $\mathcal{D}$ is fully exhibited, then the QBF calculus $\mathsf{dynQ}(\mathcal{D})\text{–}\mathsf{Res}$ is sound and refutationally complete [3, Theorem 5.6]. Analogous definitions and results apply for the proof system $\mathsf{dynQU}(\mathcal{D})\text{–}\mathsf{Res}$. More details can be found in [11, 3, 12].

The reference rule uses the following terminology: Given a dependency scheme $\mathcal{D}$, an assignment $\sigma$ to some variables $X$ of a PCNF $\Psi$ is called a *$\mathcal{D}$–assignment* if $u \in X$ whenever there is some $x \in X$ with $(u, x) \in \mathcal{D}_\Psi$.

$$\frac{}{R}$$

- $\sigma = \{\ell_1, \dots, \ell_k\}$ is a $\mathcal{D}$–assignment for $\Psi$.
- $\pi$ is a $\mathsf{dynQ}(\mathcal{D})\text{–}\mathsf{Res}$ refutation of $\Psi[\sigma]$.
- $R$ is the clause $\bar{\ell}_1 \vee \dots \vee \bar{\ell}_k$.

**Figure 4** Reference rule $\mathrm{ref}(\sigma, \pi)$ for $\mathsf{Q}(\mathcal{D})\text{–}\mathsf{Res}$.

We first provide exponential separations showing that the dynamic uses of $\mathcal{D}^{\mathsf{def}}$, respectively $\mathcal{D}^{\mathsf{rrs}}$, can outperform the static use of $\mathcal{D}^{\mathsf{rrs+def}}$.

▶ **Proposition 18.** *There is a QBF family that admits constant* $\mathsf{dynQ}(\mathcal{D}^{\mathsf{def}})\text{–}\mathsf{Res}$ *refutations, but requires exponential* $\mathsf{QU}(\mathcal{D}^{\mathsf{rrs+def}})\text{–}\mathsf{Res}$ *refutations.*

**Proof.** Consider again $\{\mathsf{QP}_n\}_{n\geq 2}$ from Proposition 14 and define $\Psi_n := \forall v.(\mathsf{QP}_n \vee v)$ (transformed into PCNF). This formula is still false, as $\mathsf{QP}_n$ is false and $v$ is universal. Since the variable $v$ appears only in positive polarity, it holds $(v, x) \notin \mathcal{D}^{\mathsf{rrs}}_{\Psi_n}$ for any $x$. Moreover, the formula $\Psi_n$ does not contain any defined variables, as every clause can be satisfied by setting $v = \top$. Therefore $\mathcal{D}^{\mathsf{rrs+def}}_{\Psi_n} = \mathcal{D}^{\mathsf{rrs}}_{\Psi_n} = \{(u, t_i) \colon i = 2, \dots, n\}$. Now, if $\Psi_n$ admitted a polynomial $\mathsf{QU}(\mathcal{D}^{\mathsf{rrs+def}})\text{–}\mathsf{Res}$ refutation, then we could get a polynomial $\mathsf{QU}(\mathcal{D}^{\mathsf{rrs}})\text{–}\mathsf{Res}$ refutation for $\mathsf{QP}_n$ by first reducing $v$ from every clause in linear time. But such short refutations are not possible as seen in Proposition 14. In the dynamic setting, however, we get $\Psi_n[\sigma] = \mathsf{QP}_n$ under the assignment $\sigma = \{\bar{v}\}$. So, by Proposition 14, the restriction $\Psi_n[\sigma]$ admits a constant size $\mathsf{Q}(\mathcal{D}^{\mathsf{def}})\text{–}\mathsf{Res}$ refutation $\pi$. Now the reference rule $\mathrm{ref}(\sigma, \pi)$ allows us to introduce the unit clause $v$, which we immediately reduce to the empty clause. The overall size remains constant. ◀

▶ **Proposition 19.** *There is a QBF family that admits polynomial* $\mathsf{dynQ}(\mathcal{D}^{\mathsf{rrs}})\text{–}\mathsf{Res}$ *refutations, but requires exponential* $\mathsf{QU}(\mathcal{D}^{\mathsf{rrs+def}})\text{–}\mathsf{Res}$ *refutations.*

**Proof.** Consider again $\mathsf{EQ}_n$ from Proposition 15. In [3, Theorem 5.11], the formula

$$\mathrm{lock}(\mathsf{EQ}_n) := \exists a.\left((\mathsf{EQ}_n \vee \bar{a}) \wedge a \wedge \bigwedge_{i,j=1}^{n} (u_i \vee s_j \vee a) \wedge (\bar{u}_i \vee \bar{s}_j \vee a)\right),$$

again transformed into PCNF, is shown to admit polynomial $\mathsf{dynQ}(\mathcal{D}^{\mathsf{rrs}})\text{–}\mathsf{Res}$ refutations, but to require $\mathsf{QU}(\mathcal{D}^{\mathsf{rrs}})\text{–}\mathsf{Res}$ refutations exponential in $n$. Due to the big conjunction, it holds $(u_i, s_j) \in \mathcal{D}^{\mathsf{rrs}}_{\mathrm{lock}(\mathsf{EQ}_n)}$ for all $i, j = 1, \dots, n$. And since the formula does not contain any defined variables, except for $a$, it does not admit any non–trivial independencies in $\mathcal{D}^{\mathsf{rrs+def}}$. By Lemma 12, the exponential lower bound also applies to $\mathsf{QU}(\mathcal{D}^{\mathsf{rrs+def}})\text{–}\mathsf{Res}$. ◀

Next, we show that the exponential lower bound for the QParity family in $\mathsf{QU}(\mathcal{D}^{\mathsf{rrs}})-\mathsf{Res}$ from Proposition 14 still applies to the dynamic version of the calculus. For this purpose, we generalize the ideas from [11, Theorem 17] and [3, Theorem 5.7] to prove that if a formula never exhibits any new independencies under any assignment, then its dynamic and static refutations are essentially the same. As usual, the statement also applies to $\mathsf{Q}-\mathsf{Res}$.

▶ **Lemma 20.** *Let $\Psi = \Pi.\psi$ be a false QBF and $\mathcal{D}$ a dependency scheme such that for every $\mathcal{D}$–assignment $\sigma$ to $V := \mathrm{var}(\sigma)$ it holds that $\mathcal{D}_{\Psi[\sigma]} = \mathcal{D}_\Psi \setminus \{(u,x) \colon u \in V \text{ or } x \in V\}$. Then any $\mathsf{dynQU}(\mathcal{D})-\mathsf{Res}$ refutation of $\Psi$ can be transformed into a $\mathsf{QU}(\mathcal{D})-\mathsf{Res}$ refutation with no increase in size.*

**Proof Sketch.** The proof is by induction on the *reference degree* $d$ of a $\mathsf{dynQU}(\mathcal{D})-\mathsf{Res}$ refutation $\pi$, which is the maximal recursive depth of subproofs of $\pi$. The case $d = 0$ is trivial, as it corresponds to a $\mathsf{QU}(\mathcal{D})-\mathsf{Res}$ refutation. So let $\pi$ be a $\mathsf{dynQU}(\mathcal{D})-\mathsf{Res}$ refutation of $\Psi$ of reference degree $d \geq 1$. Let $R$ be the first reference clause derived in $\pi$ by the application of $\mathrm{ref}(\sigma, \pi')$, where $\sigma$ is some $\mathcal{D}$–assignment and $\pi'$ is a $\mathsf{dynQU}(\mathcal{D})-\mathsf{Res}$ refutation of $\Psi[\sigma]$. Note that $\Psi[\sigma]$ still satisfies the dependency assumption from the claim, since for any assignment $\sigma'$ to variables $V' \subseteq \mathrm{var}(\psi) \setminus V$ we have

$$\mathcal{D}_{\Psi[\sigma][\sigma']} = \mathcal{D}_{\Psi[\sigma\cup\sigma']} = \mathcal{D}_\Psi \setminus \{(u,x) \colon u \in V \cup V' \text{or } x \in V \cup V'\} = \mathcal{D}_{\Psi[\sigma]} \setminus \{(u,x) \colon u \in V' \text{or } x \in V'\}.$$

Hence, we can apply the induction hypothesis to $\pi'$, which has reference degree $< d$, to transform it into a $\mathsf{QU}(\mathcal{D})-\mathsf{Res}$ refutation $\rho$ of $\Psi[\sigma]$ with size $|\pi'|$. Next, we want to show that appending the clause $R = \bigvee_{\ell \in \sigma} \bar{\ell}$ to each clause of $\rho$ gives a valid $\mathsf{QU}(\mathcal{D})-\mathsf{Res}$ derivation of $R$. Appending $R$ does not create any tautologies in $\rho$, since the literals from $R$ do not appear in $\rho$ and are only added with a single polarity. Appending $R$ to $\rho$ does also not invalidate any reductions of universals $u \notin V$ in $\rho$ with respect to $\mathcal{D}_\Psi$: Indeed, for any existential $x \notin V$ with $(u,x) \notin \mathcal{D}_{\Psi[\sigma]}$ our dependency assumption ensures $(u,x) \notin \mathcal{D}_\Psi$, and if $x \in V$, then we also have $(u,x) \notin \mathcal{D}_\Psi$, as $\sigma$ is a $\mathcal{D}$–assignment. So there is a valid $\mathsf{QU}(\mathcal{D})-\mathsf{Res}$ derivation $\rho'$ of $R$ with size $|\rho|$. From here on the argument is analogous to [11, Theorem 17] (respectively the journal version [3, Theorem 5.7], which uses a more technical, equivalent definition of $\mathsf{dynQU}(\mathcal{D})-\mathsf{Res}$). There it is argued that $\rho'$ can be transformed into a $\mathsf{QU}(\mathcal{D})-\mathsf{Res}$ derivation of $R$ from $\Psi$ with no increase in size using subsumption of clauses. Repeating this for every reference clause yields a $\mathsf{QU}(\mathcal{D})-\mathsf{Res}$ refutation of $\Psi$ of same size. ◀

▶ **Proposition 21.** *The family $\{\mathsf{QP}_n\}_{n\geq 2}$ requires exponential $\mathsf{dynQU}(\mathcal{D}^{\mathsf{rrs}})-\mathsf{Res}$ refutations.*

**Proof.** By Proposition 14, it suffices to verify the dependency assumptions of Lemma 20 for $\mathsf{QP}_n$. Let $\sigma$ be any $\mathcal{D}^{\mathsf{rrs}}$–assignment to some variables $V$ of $\mathsf{QP}_n$. Inspecting the matrix, we see $(u, t_i) \in \mathcal{D}^{\mathsf{rrs}}_{\mathsf{QP}_n}$ for every $i = 2, \ldots, n$, since there is a resolution path from $u$ to $t_i$ via the literals $t_n, t_{n-1}, \ldots, t_i$ and then back to $u$ in reverse order and polarity. If $u \in V$, then $\mathcal{D}^{\mathsf{rrs}}_{\mathsf{QP}_n[\sigma]} = \varnothing = \mathcal{D}^{\mathsf{rrs}}_{\mathsf{QP}_n} \setminus (V \times E_{\mathsf{QP}_n})$, as $u$ is the only universal variable in $\mathsf{QP}_n$, so we are done. If $u \notin V$, then we cannot have $t_i \in V$ for any $i \geq 2$, since $\sigma$ is a $\mathcal{D}^{\mathsf{rrs}}$–assignment. Furthermore, even if $x_i \in V$ for some $i > 2$, then two of the clauses $(\bar{t}_i \vee \bar{x}_i \vee \bar{t}_{i-1})$, $(\bar{t}_i \vee x_i \vee t_{i-1})$, $(t_i \vee \bar{x}_i \vee t_{i-1})$, and $(t_i \vee x_i \vee \bar{t}_{i-1})$ get deleted in $\mathsf{QP}_n[\sigma]$. But then there is still a resolution path connecting $u$ and $t_{i-1}$. So, in this case $\mathcal{D}^{\mathsf{rrs}}_{\mathsf{QP}_n[\sigma]} = \mathcal{D}^{\mathsf{rrs}}_{\mathsf{QP}_n}$. ◀

The above separation seems to be the first example of a formula family requiring exponential $\mathsf{dynQU}(\mathcal{D}^{\mathsf{rrs}})-\mathsf{Res}$ refutations in the literature. Our inspection of the resolution paths at the end of Section 4 shows the exponential bound generalizes to the implication-free schemes and $\mathcal{D}^{\forall\mathsf{pure}}$. Hence, the static use of the scheme $\mathcal{D}^{\mathsf{def}}$ can outperform even the dynamic use of $\mathcal{D}^{\mathsf{rrs}}$ and its generalizations. This is summarized in Figure 1.

An analogous argument as above using Lemma 20 shows that the family Equality from Proposition 15 requires exponential dynQU($\mathcal{D}^{\mathsf{std}}$)−Res refutations. This yields the following dynamic analog of [3, Theorem 4.12].

▶ **Theorem 22.** *The proof system* dynQU($\mathcal{D}^{\mathsf{rrs}}$)−Res *is stronger than* dynQU($\mathcal{D}^{\mathsf{std}}$)−Res.

## 6 Syntactic Definitions

The notion of defined variables from Section 3 that we used to define $\mathcal{D}^{\mathsf{def}}$ is a very general semantic one. In this section, we explore a less general syntactic notion that requires the existence of an explicit CNF encoding of a gate definition $x \leftrightarrow \phi$ within the matrix. The advantage of such syntactic definitions is that they are easier to detect, for example, by pattern matching or unsatisfiable core extraction [15, 16]. In [26], this is used to shift Tseitin variables in the prefix, which is essentially a weaker version of the dependency scheme we introduce below. By bounding the size (i.e., the number of clauses) of definitions, we can ensure polynomial complexity. Throughout this section we assume (D)QBFs to be in PCNF.

▶ **Definition 23.** *Let $\psi$ be a CNF formula. A variable $x \in \mathrm{var}(\psi)$ is said to be* syntactically defined *in $\psi$ if there exists a set of clauses $\delta \subseteq \psi$, each containing $x$ or $\bar{x}$, such that every assignment $\sigma$ to $\mathrm{var}(\delta)\setminus\{x\}$ there is exactly one value for $x$ to satisfy $\delta[\sigma]$. In this case, $\delta$ is called a* syntactic *or* gate definition *for $x$ in $\psi$. Given a parameter $\lambda \in \mathbb{N} \cup \{\infty\}$, we denote* $\mathrm{def}(\psi, x, \lambda) := \{\delta \colon \delta$ *is a syntactic definition for $x$ in $\psi$ with $|\delta| \leq \lambda\}$.*

Some examples of gate definitions and their corresponding syntactic definitions are given in Table 1. Note that every syntactically defined variable $x$ with syntactic definition $\delta \subseteq \psi$ is also semantically defined in $\psi$ by $\mathrm{var}(\delta)\setminus\{x\}$. However, not every semantically defined variable is also syntactically defined. For instance, $(x \vee \bar{y}) \wedge (y \vee \bar{z}) \wedge (z \vee \bar{x})$ is not a syntactic definition for $x$ (as not every clause contains $x$), but both $y$ and $z$ semantically define $x$.

**Table 1** Examples of gate definitions with CNF encodings of different sizes $\lambda$.

| $\lambda$ | Gate definition | Corresponding syntactic definition $\delta$ for $x$ |
|---|---|---|
| 1 | $x \leftrightarrow \top$ | $x$ |
| 2 | $x \leftrightarrow y$ | $(x \vee \bar{y}) \wedge (\bar{x} \vee y)$ |
| 3 | $x \leftrightarrow y \wedge z$ | $(\bar{x} \vee y) \wedge (\bar{x} \vee z) \wedge (x \vee \bar{y} \vee \bar{z})$ |
| | $x \leftrightarrow y \vee z$ | $(x \vee \bar{y}) \wedge (x \vee \bar{z}) \wedge (\bar{x} \vee y \vee z)$ |
| 4 | $x \leftrightarrow y \oplus z$ | $(\bar{x} \vee \bar{y} \vee \bar{z}) \wedge (\bar{x} \vee y \vee z) \wedge (x \vee \bar{y} \vee z) \wedge (x \vee y \vee \bar{z})$ |

We use syntactic definitions to construct a dependency scheme $\mathcal{D}^{\mathsf{def}(\lambda)}$ analogous to the construction of $\mathcal{D}^{\mathsf{def}}$: Given a syntactic definition $\delta$ of size at most $\lambda$ for an existential variable $x$, we can again remove any variable from $D_x$ that does not appear in $\bigcup_{z \in \mathrm{var}(\delta)\setminus\{x\}} D_z$. One thing that changes is that it might be necessary to reduce dependency sets multiple times, as each reduction can potentially enable further reductions. As an example, consider the QBF $\forall u \exists t_1, \ldots t_n . t_1 \wedge \bigwedge_{i=1}^{n} (t_i \leftrightarrow t_{i+1})$. Here, every variable $t_i$ is semantically defined by the empty set and does not depend on $u$. But since the only syntactic definition for $t_{i+1}$ is $t_i \leftrightarrow t_{i+1}$, the variable $u$ can only be removed from $D_{t_{i+1}}$ *after* removing it from $D_{t_i}$.

We model these iterated reductions via a chain of dependency sets $(D_x^{i,\lambda})_{i \geq 0}$: Fix some dependency scheme $\mathcal{D}^*$, a (D)QBF in PCNF $\Psi = \Pi.\psi$ and some $\lambda \in \mathbb{N} \cup \{\infty\}$. If $x \in U_\Psi$, we set $D_x^{i,\lambda} := \{x\}$ for all $i$. For existential $x$ we inductively define $D_x^{0,\lambda} := \mathcal{D}^*_\Psi(x)$ and for $i \geq 1$

$$D_x^{i,\lambda} := D_x^{i-1,\lambda} \cap \bigcap_{\delta \in \mathrm{def}(\psi, x, \lambda)} \bigcup_{z \in \mathrm{var}(\delta)\setminus\{x\}} D_z^{i-1,\lambda}.$$

For every variable $x$, we get a descending chain $D_x^{0,\lambda} \supseteq D_x^{1,\lambda} \supseteq \cdots$. Since the first set is finite, the chain stabilizes after finitely many steps, i.e., there exists some $N$ such that for all $i \geq N$ and all $x$ it holds that $D_x^{i,\lambda} = D_x^{N,\lambda}$. Denote the minimal such $N$ by $N_\Psi$. Note that we can ensure $N_\Psi \leq |U_\Psi| \cdot |E_\Psi|$, as this is the maximal number of dependencies in $\Psi$.

▶ **Definition 24.** *For any dependency scheme $\mathcal{D}^*$ and $\lambda \in \mathbb{N} \cup \{\infty\}$, the dependency scheme $\mathcal{D}^{*+\mathsf{def}(\lambda)}$ is defined by $\mathcal{D}_\Psi^{*+\mathsf{def}(\lambda)} := \{(u,x) \colon x \in E_\Psi, u \in D_x^{N_\Psi,\lambda}\}$ for any (D)QBF $\Psi$. We write $\mathcal{D}^{\mathsf{def}(\lambda)}$ as a shorthand for $\mathcal{D}^{\mathsf{trv}+\mathsf{def}(\lambda)}$.*

The fact that every syntactically defined variable is also semantically defined indicates that $\mathcal{D}^{\mathsf{def}(\lambda)}$ is a less general scheme than $\mathcal{D}^{\mathsf{def}}$. To prove this, we observe that the above chain stabilizes after one step if semantic definitions are used. In other words, applying $\mathcal{D}^{\mathsf{def}}$ twice does not change anything, i.e., $\mathcal{D}^{\mathsf{def}+\mathsf{def}} = \mathcal{D}^{\mathsf{def}}$.

▶ **Lemma 25.** *For any dependency scheme $\mathcal{D}^*$, any (D)QBF $\Psi = \Pi.\psi$, and any variable $x \in E_\Psi$, we have $\mathcal{D}_\Psi^{*+\mathsf{def}}(x) = \mathcal{D}_\Psi^*(x) \cap \bigcap_{X \in \mathrm{def}(\psi,x)} \bigcup_{z \in X} \mathcal{D}_\Psi^{*+\mathsf{def}}(z)$.*

**Proof.** The inclusion "$\supseteq$" holds, as, by definition, $\mathcal{D}_\Psi^{*+\mathsf{def}}(z) \subseteq \mathcal{D}_\Psi^*(z)$ for any $z$. For the other inclusion, consider some universal $u \in \mathcal{D}_\Psi^*(x)$ that is not in the right-hand side. Then there is some $X \in \mathrm{def}(\psi, x)$ such that $u \notin \mathcal{D}^{*+\mathsf{def}}(z)$ for all $z \in X$. Assume there is some $z \in X$ with $u \in \mathcal{D}_\Psi^*(z)$. Then $u$ must have been removed from this set, which means that there is some set $X_z \in \mathrm{def}(\psi, z)$ such that $u \notin \mathcal{D}_\Psi^*(y)$ for all $y \in X_z$. Thus, $x \notin X_z$ and $(X \setminus \{z\}) \cup X_z$ is a defining set for $x$. Repeating this argument for every $z \in X$ with $u \in \mathcal{D}_\Psi^*(z)$ yields a defining set $X'$ for $x$ with $u \notin \mathcal{D}_\Psi^*(y)$ for every $y \in X'$. This shows $u \notin \mathcal{D}_\Psi^{*+\mathsf{def}}(x)$. ◀

▶ **Proposition 26.** *For any scheme $\mathcal{D}^*$ and $\mu \leq \lambda$, we have $\mathcal{D}^{*+\mathsf{def}(\mu)} \leq \mathcal{D}^{*+\mathsf{def}(\lambda)} \leq \mathcal{D}^{*+\mathsf{def}}$. In particular, the scheme $\mathcal{D}^{*+\mathsf{def}(\lambda)}$ is fully exhibited, whenever $\mathcal{D}^*$ is fully exhibited.*

**Proof.** We have to show $\mathcal{D}_\Psi^{*+\mathsf{def}} \subseteq \mathcal{D}_\Psi^{*+\mathsf{def}(\lambda)} \subseteq \mathcal{D}_\Psi^{*+\mathsf{def}(\mu)}$ for every (D)QBF $\Psi = \Pi.\psi$. The full exhibition then follows from Theorem 10. The second inclusion is clear. For the first inclusion, it suffices to show $\mathcal{D}^{*+\mathsf{def}}(x) \subseteq D_x^{i,\lambda}$ for $i \geq 0$ and any existential $x$. For $i = 0$ this is clear, as $\mathcal{D}_\Psi^{*+\mathsf{def}}(x) \subseteq \mathcal{D}_\Psi^*(x)$. Now suppose the claim holds for any variable at step $i-1$. For any $u \in D_x^{i-1,\lambda} \setminus D_x^{i,\lambda}$ there must be some syntactic definition $\delta \in \mathrm{def}(\psi, x, \lambda)$ such that $u \notin D_z^{i-1,\lambda}$ for all $z \in \mathrm{var}(\delta) \setminus \{x\}$. By the inductive hypothesis, we have $u \notin \mathcal{D}_\Psi^{*+\mathsf{def}}(z)$. Now as $x$ is semantically defined by $\mathrm{var}(\delta) \setminus \{x\}$, Lemma 25 implies $u \notin \mathcal{D}_\Psi^{*+\mathsf{def}}(x)$. ◀

The above result shows that we have the following hierarchy of dependency schemes:

$$\mathcal{D}^{\mathsf{def}(1)} \leq \mathcal{D}^{\mathsf{def}(2)} \leq \cdots \leq \mathcal{D}^{\mathsf{def}(\infty)} \leq \mathcal{D}^{\mathsf{def}}.$$

With increasing $\lambda$, the scheme $\mathcal{D}^{\mathsf{def}(\lambda)}$ is able to capture increasingly complex gate definitions. From Table 1, we see that the scheme $\mathcal{D}^{\mathsf{def}(4)}$ can reduce dependencies of variables defined by xor gates. Since these are exactly the definitions used in QParity, we can generalize our results from Section 4. Again, the same result applies for the strong generalizations of $\mathcal{D}^{\mathsf{rrs}}$.

▶ **Theorem 27.** *For any $\lambda \geq 4$, the proof systems $\mathsf{Q}(\mathcal{D}^{\mathsf{def}(\lambda)})\text{–}\mathsf{Res}$ and $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs}})\text{–}\mathsf{Res}$ are incomparable, while $\mathsf{Q}(\mathcal{D}^{\mathsf{rrs}+\mathsf{def}(\lambda)})\text{–}\mathsf{Res}$ is stronger than both of them.*

## 7 Computational Complexity

We now turn to the computational complexity of the definition-based dependency schemes. Generally, deciding dependence between variables is PSPACE-complete for QBF [28, Proposition 2] and even NEXPTIME-complete for DQBFs [35, Theorem 11]. This causes a tradeoff

between generality and computational complexity of dependency schemes. While $\mathcal{D}^{\mathsf{rrs}}$ can be computed in linear time in the size of the formula [31], some of its generalizations are known to be computationally harder. For example, the computation of the implication-free $\mathcal{D}^{\mathsf{F}}(R_{\models}, \infty)$ is NP-complete [6], while some of its less general variants $\mathcal{D}^{\mathsf{F}}(R, \lambda)$ can be computed in polynomial time with exponent $\lambda < \infty$. A similar picture emerges for $\mathcal{D}^{\mathsf{def}(\lambda)}$.

Our analysis for $\mathcal{D}^{\mathsf{def}}$ is based on the following characterization of defining sets.

▶ **Proposition 28** (Padoa's Theorem [30, 22])**.** *Let $\psi$ be a propositional formula, $x \in \mathrm{var}(\psi)$, and $X \subseteq \mathrm{var}(\psi)$ with $x \notin X$. Let $\psi'$ be the formula obtained by replacing every $y \in \mathrm{var}(\psi) \setminus X$ by a fresh variable $y'$. Then $x$ is defined by $X$ in $\psi$ if and only if $\psi \wedge x \wedge \psi' \wedge \bar{x}'$ is unsatisfiable.*

▶ **Theorem 29.** *Given a (D)QBF $\Psi$ in PNF, deciding $(u, x) \in \mathcal{D}^{\mathsf{def}}_{\Psi}$ is NP-complete.*

**Proof.** We first show that for any (D)QBF $\Psi = \Pi.\psi$ and any $u, x \in \mathrm{var}(\Psi)$ with $u \in D_x$, the problem of deciding $u \in \mathcal{D}^{\mathsf{def}}_{\Psi}(x)$ lies in NP. We have $u \notin \mathcal{D}^{\mathsf{def}}_{\Psi}(x)$ if and only if there is some defining set $X$ for $x$ such that $u \notin D_z$ for all $z \in X$. Since supersets of defining sets are again defining, this is true if and only if $X_0 = \{z \in \mathrm{var}(\Psi) \colon z \neq x, u \notin D_z\}$ is a defining set for $x$. By Proposition 28, this is equivalent to the unsatisfiability of the formula $\psi_0 \coloneqq \psi \wedge x \wedge \psi' \wedge \bar{x}'$ with all variables $y \in \mathrm{var}(\Psi) \setminus X_0$ replaced by new ones in $\psi'$. By negating the previous conditions, we get $(u, x) \in \mathcal{D}^{\mathsf{def}}_{\Psi}$ if and only if $\psi_0$ is satisfiable. So we have a polynomial reduction to a problem in NP.

For the NP-hardness, fix an arbitrary propositional formula $\psi$ with variables $y_1, \ldots, y_n$. Consider the QBF $\Psi = \forall y_1, \ldots, y_n \exists x.(x \vee \psi)$. Note that $x$ has a defining set in $x \vee \psi$ if and only if $\psi$ is unsatisfiable, and in this case, the empty set is defining for $x$. Hence, $(y_1, x) \in \mathcal{D}^{\mathsf{def}}_{\Psi}$ if and only if $\psi$ is satisfiable. So we have polynomially reduced the NP-complete problem SAT to the computation of $\mathcal{D}^{\mathsf{def}}$. ◀

A similar argument shows that the computation of $\mathcal{D}^{\mathsf{rrs+def}}$ is NP-complete. For the weaker schemes $\mathcal{D}^{\mathsf{def}(\lambda)}$ we can achieve polynomial runtime with exponent linear in $\lambda$. We prove this using the following characterization of syntactic definitions, adapted from [26]. We say that a CNF $\phi$ is *propositionally blocked* on a variable $x$ if for every $C \in \phi$ with $x \in C$ and every $D \in \phi$ with $\bar{x} \in D$, there is some literal $\ell$ with $\mathrm{var}(\ell) \neq x$ and $\ell \in C$ and $\bar{\ell} \in D$.

▶ **Proposition 30** (cf. [26])**.** *Let $x$ be a variable in a CNF formula $\psi$ and let $\delta \subseteq \psi$ be a set of clauses, all containing $x$ or $\bar{x}$. Then $\delta$ is a syntactic definition for $x$ if and only if $\delta$ is propositionally blocked on $x$ and $\{C \setminus \{x, \bar{x}\} \colon C \in \delta\}$ is unsatisfiable.*

The blockedness condition says that there is always at least one value for $x$ to satisfy $\delta$, while the unsatifsiability condition ensures that there is at most one. Using these two conditions, we can detect all definitions of bounded size $\lambda$ in a formula in polynomial time with exponent linear in $\lambda$. In the terminology of parameterized complexity [10, Chapter 17], deciding membership in $\mathcal{D}^{\mathsf{def}(\lambda)}$ is XP-tractable parameterized by $\lambda$.

▶ **Proposition 31.** *For any $1 \leq \lambda < \infty$, given a (D)QBF $\Psi$ in PCNF of size $n$ and some variables $u, x$, deciding $(u, x) \in \mathcal{D}^{\mathsf{def}(\lambda)}_{\Psi}$ can be done in $n^{\mathcal{O}(\lambda)}$ many steps.*

**Proof.** Fix a formula of size $n$ with matrix $\psi$. Since every variable needs to appear in $\psi$ there are at most $n$ variables and clauses. So, for fixed $x$, we can take each of the $\mathcal{O}(n^{\lambda})$ many subsets $\delta \subseteq \psi$ of clauses with $|\delta| \leq \lambda$ and use the characterization from Proposition 30 to check if $\delta$ is a definition for $x$. The blockedness condition can be checked in quadratic time by iterating over $\delta$. The formula $\delta' = \{C \setminus \{x, \bar{x}\} \colon C \in \delta\}$ is satisfiable if and only if we can pick one literal $\ell_i$ from each clause of $\delta'$ in such a way that $\ell_i \neq \bar{\ell}_j$ for all $i, j$. Since $\delta'$

has at most $\lambda$ many clauses with $n$ variables, there are $\mathcal{O}(n^\lambda)$ many possible choices and we can perform the unsatisfiability check by testing every choice. Overall, we see that we can determine all definitions for a fixed variable $x$ in $\mathcal{O}(n^{2\lambda})$ steps. We repeat this procedure for each variable to compute all definitions within the formula. Given those definitions, we can compute $\mathcal{D}^{\mathsf{def}(\lambda)}$ by reducing the involved dependency sets $D_x^{i,\lambda}$. Since there are at most $n^2$ dependencies within the formula, we need at most $n^2$ steps for $i$. ◀

A similar proof as in Theorem 29 shows that the computation of $\mathcal{D}^{\mathsf{def}(\infty)}$ is NP-hard. Using Proposition 30 and the iterative construction of the scheme, one can see that its computation is possible using polynomially many NP-oracle calls.

## 8 Conclusion and Outlook

We have introduced a novel dependency scheme $\mathcal{D}^{\mathsf{def}}$ for (D)QBFs that models reducing dependencies of defined variables. The scheme $\mathcal{D}^{\mathsf{def}}$ is sound for Q–Resolution and can be combined with other dependency schemes. Since $\mathcal{D}^{\mathsf{def}}$ only reduces dependencies of defined variables, it should not be seen as a competing approach, but rather as an addition to improve other dependency schemes. Our proof-theoretic considerations have shown that using $\mathcal{D}^{\mathsf{def}}$ can lead to exponentially shorter refutations in resolution-based proof systems compared to virtually any other established dependency scheme.

We expect our proof-theoretic results to also apply to other proof calculi in which dependency schemes can be incorporated, like ∀Exp+Res [17], LDQ–Res [8], or QCDCL [7]. One major drawback of the dependency scheme $\mathcal{D}^{\mathsf{def}}$ is its high computational complexity. However, due to its motivation from Tseitin transformations, we still expect its (partial) computation to be very feasible in many real-world formulas. This belief is supported by empirical findings [26, 30], but more experiments using our framework are required. In particular, a sophisticated study of the polynomial schemes $\mathcal{D}^{\mathsf{def}(\lambda)}$ for different values of $\lambda$ would be interesting, both in terms of theoretical and practical aspects.

## References


1 Olaf Beyersdorff. Proof complexity of quantified Boolean logic–a survey. *Mathematics for Computation (M4C)*, pages 353–391, 2022. doi:10.1142/9789811245220_0015.

2 Olaf Beyersdorff and Joshua Blinkhorn. Dependency Schemes in QBF Calculi: Semantics and Soundness. In *Principles and Practice of Constraint Programming*, volume 9892, pages 96–112, 2016. doi:10.1007/978-3-319-44953-1_7.

3 Olaf Beyersdorff and Joshua Blinkhorn. Dynamic QBF Dependencies in Reduction and Expansion. *ACM Transactions on Computational Logic*, 21(2):1–27, 2020. doi:10.1145/3355995.

4 Olaf Beyersdorff, Joshua Blinkhorn, Leroy Chew, Renate Schmidt, and Martin Suda. Reinterpreting Dependency Schemes: Soundness Meets Incompleteness in DQBF. *Journal of Automated Reasoning*, 63(3):597–623, 2019. doi:10.1007/s10817-018-9482-4.

5 Olaf Beyersdorff, Joshua Blinkhorn, and Luke Hinde. Size, Cost, and Capacity: A Semantic Technique for Hard Random QBFs. *Logical Methods in Computer Science*, Volume 15, Issue 1:4141, 2019. doi:10.23638/LMCS-15(1:13)2019.

6 Olaf Beyersdorff, Joshua Blinkhorn, and Tomáš Peitl. Strong (D)QBF Dependency Schemes via Implication-free Resolution Paths. *ACM Transactions on Computation Theory*, 16(4):1–25, 2024. doi:10.1145/3689345.

7 Olaf Beyersdorff and Benjamin Böhm. Understanding the relative strength of QBF CDCL solvers and QBF resolution. *Logical Methods in Computer Science*, 19(2), 2023. doi:10.46298/LMCS-19(2:2)2023.

8 Olaf Beyersdorff, Leroy Chew, and Mikolás Janota. Proof Complexity of Resolution-based QBF Calculi. *LIPIcs STACS 2015*, 30:76–89, 2015. doi:10.4230/LIPIcs.STACS.2015.76.

9 Olaf Beyersdorff, Leroy Chew, Renate Schmidt, and Martin Suda. Lifting QBF resolution calculi to DQBF. In *International Conference on Theory and Applications of Satisfiability Testing*, pages 490–499. Springer, 2016. doi:10.1007/978-3-319-40970-2_30.

10 Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors. *Handbook of Satisfiability - Second Edition*, volume 336 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2021. doi:10.3233/FAIA336.

11 Joshua Blinkhorn and Olaf Beyersdorff. Shortening QBF proofs with dependency schemes. In *International Conference on Theory and Applications of Satisfiability Testing*, pages 263–280. Springer, 2017. doi:10.1007/978-3-319-66263-3_17.

12 Joshua Lewis Blinkhorn. *Quantified Boolean formulas: Proof complexity and models of solving*. PhD thesis, University of Leeds, 2019. URL: https://etheses.whiterose.ac.uk/id/eprint/26508/.

13 Leroy Chew and Tomás Peitl. Strong (D)QBF Dependency Schemes via Pure Universal Resolution Paths. *Electronic Colloquium on Computational Complexity*, TR25-149, 2025. URL: https://eccc.weizmann.ac.il/report/2025/149.

14 Abhimanyu Choudhury and Meena Mahajan. Dependency Schemes in CDCL-based QBF Solving: A Proof-Theoretic Study. *Journal of Automated Reasoning*, 68(3):16, 2024. doi:10.1007/s10817-024-09707-4.

15 Zhaohui Fu and Sharad Malik. Extracting Logic Circuit Structure from Conjunctive Normal Form Descriptions. In *20th International Conference on VLSI Design held jointly with 6th International Conference on Embedded Systems (VLSID'07)*, pages 37–42, 2007. doi:10.1109/VLSID.2007.81.

16 Markus Iser. *Recognition and Exploitation of Gate Structure in SAT Solving*. PhD thesis, Karlsruher Institut für Technologie (KIT), 2020. doi:10.5445/IR/1000118523.

17 Mikoláš Janota and Joao Marques-Silva. Expansion-based QBF solving versus Q-resolution. *Theoretical Computer Science*, 577:25–42, 2015. doi:10.1016/j.tcs.2015.01.048.

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

19 Jean-Marie Lagniez, Emmanuel Lonca, and Pierre Marquis. Improving Model Counting by Leveraging Definability. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, pages 751–757. IJCAI/AAAI Press, 2016. URL: http://www.ijcai.org/Abstract/16/112.

20 Florian Lonsing. *Dependency Schemes and Search-Based QBF Solving: Theory and Practice*. PhD thesis, Johannes Kepler University Linz, 2012.

21 Florian Lonsing and Armin Biere. DepQBF: A Dependency-Aware QBF Solver. *Journal of Satisfiability, Boolean Modeling and Computation*, 7(2-3):71–76, 2010. doi:10.3233/SAT190077.

22 Alessandro Padoa. Essai d'une théorie algébrique des nombres entiers, précédé d'une introduction logique à une theorie déductive quelconque. In *Bibliothèque du Congrès international de philosophie*, volume 3, pages 309–365, 1901.

23 Tomáš Peitl, Friedrich Slivovsky, and Stefan Szeider. Combining resolution-path dependencies with dependency learning. In *International Conference on Theory and Applications of Satisfiability Testing*, pages 306–318. Springer, 2019. doi:10.1007/978-3-030-24258-9_22.

24 Tomáš Peitl, Friedrich Slivovsky, and Stefan Szeider. Dependency learning for QBF. *Journal of Artificial Intelligence Research*, 65:181–208, 2019. doi:10.1613/JAIR.1.11529.

25 Tomáš Peitl, Friedrich Slivovsky, and Stefan Szeider. Long-Distance Q-Resolution with Dependency Schemes. *Journal of Automated Reasoning*, 63(1):127–155, 2019. doi:10.1007/s10817-018-9467-3.

26 Joseph Reeves, Marijn Heule, and Randal Bryant. Moving definition variables in quantified Boolean formulas. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 462–479. Springer, 2022. doi:10.1007/978-3-030-99524-9_26.

27 Franz-Xaver Reichl, Friedrich Slivovsky, and Stefan Szeider. Certified DQBF solving by definition extraction. In *International Conference on Theory and Applications of Satisfiability Testing*, pages 499–517. Springer, 2021. doi:10.1007/978-3-030-80223-3_34.

28 Marko Samer and Stefan Szeider. Backdoor sets of quantified Boolean formulas. *Journal of Automated Reasoning*, 42(1):77–97, 2009. doi:10.1007/s10817-008-9114-5.

29 Ankit Shukla, Armin Biere, Luca Pulina, and Martina Seidl. A survey on applications of quantified Boolean formulas. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 78–84. IEEE, 2019. doi:10.1109/ICTAI.2019.00020.

30 Friedrich Slivovsky. Interpolation-based semantic gate extraction and its applications to qbf preprocessing. In *International Conference on Computer Aided Verification*, pages 508–528. Springer, 2020. doi:10.1007/978-3-030-53288-8_24.

31 Friedrich Slivovsky and Stefan Szeider. Computing resolution-path dependencies in linear time. In *International Conference on Theory and Applications of Satisfiability Testing*, pages 58–71. Springer, 2012. doi:10.1007/978-3-642-31612-8_6.

32 Friedrich Slivovsky and Stefan Szeider. Soundness of Q-resolution with dependency schemes. *Theoretical Computer Science*, 612:83–101, 2016. doi:10.1016/J.TCS.2015.10.020.

33 Grigori S Tseitin. On the complexity of derivation in propositional calculus. In *Automation of reasoning 2: Classical papers on computational logic 1967–1970*, pages 466–483. Springer, 1983. doi:10.1007/978-3-642-81955-1_28.

34 Allen Van Gelder. Contributions to the theory of practical quantified boolean formula solving. In *International Conference on Principles and Practice of Constraint Programming*, pages 647–663. Springer, 2012. doi:10.1007/978-3-642-33558-7_47.

35 Ralf Wimmer, Christoph Scholl, and Bernd Becker. The (D)QBF Preprocessor HQSpre – Underlying Theory and Its Implementation. *Journal on Satisfiability, Boolean Modeling and Computation*, 11(1):3–52, 2019. doi:10.3233/SAT190115.