Quantified Boolean Formulas: Proof Complexity and Models of Solving

UNIVERSITY OF LEEDS

Joshua Lewis Blinkhorn

School of Computing

University of Leeds

A thesis submitted for the degree of

Doctor of Philosophy

27th September 2019

Declaration

The candidate confirms that the work submitted is their own, except where work which has formed part of jointly authored publications has been included. The contribution of the candidate and the other authors to this work has been explicitly indicated below. The candidate confirms that appropriate credit has been given within the thesis where reference has been made to the work of others.

Results contained in this thesis appeared in the following jointly authored publications. The candidate is the main author of all sections from jointly authored publications from which work appears in this thesis.

[A] Olaf Beyersdorff and Joshua Blinkhorn: *Dependency Schemes in QBF Calculi: Semantics and Soundness*. International Conference on Principles and Practice of Constraint Programming (CP), pages 96–112. Springer, 2016.

[B] Joshua Blinkhorn and Olaf Beyersdorff: *Shortening QBF Proofs with Dependency Schemes*. International Conference on Theory and Practice of Satisfiability Testing (SAT), pages 263–280. Springer, 2017. **Best paper award**.

[C] Joshua Blinkhorn and Olaf Beyersdorff: *Dynamic Dependency Awareness for QBF*. International Joint Conference on Artificial Intelligence (IJCAI), pages 5224–5228. ijcai.org, 2018. (Invited contribution, "sister conferences best paper award track", based on [B])

[D] Olaf Beyersdorff, Joshua Blinkhorn and Luke Hinde: *Size, Cost and Capacity: A Semantic Technique for Hard Random QBFs*. ACM Conference on Innovations in Theoretical Computer Science (ITCS), pages 9:1–9:18. Schloss Dagstuhl, 2018.

[E] Olaf Beyersdorff and Joshua Blinkhorn: *Genuine Lower Bounds for QBF Expansion*. International Symposium on Theoretical Aspects of Computer Science (STACS), pages 12:1–12:15. Schloss Dagstuhl, 2018.

[F] Olaf Beyersdorff, Joshua Blinkhorn and Meena Mahajan: *Building Strategies into QBF Proofs*. International Symposium on Theoretical Aspects of Computer Science (STACS), pages 14:1–14:18. Schloss Dagstuhl, 2019.

[G] Olaf Beyersdorff, Joshua Blinkhorn, Leroy Chew, Renate A. Schmidt and Martin Suda: *Reinterpreting Dependency Schemes: Soundness Meets Incompleteness in DQBF*. Journal of Automated Reasoning, volume 63(3), pages 597–623. 2019.

[H] 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(1). 2019. (Journal version of [D])

[I] Olaf Beyersdorff and Joshua Blinkhorn: *Lower-bound Techniques for QBF Expansion.* Theory of Computing Systems, in press. (Journal version of [E])

[J] Olaf Beyersdorff and Joshua Blinkhorn: *Dynamic QBF Dependencies in Reduction and Expansion.* ACM Transactions on Computational Logic, in press. (Journal version of [B])

The following conference paper was published during the course of the PhD, but does not feature in this thesis.

[K] Olaf Beyersdorff and Joshua Blinkhorn: *Proof Complexity of QBF Symmetry Recomputation.* International Conference on Theory and Practice of Satisfiability Testing (SAT), pages 36–52. Springer, 2019.

Work from jointly authored publications appear in this thesis as follows. Where sections contain material published in one or more of the jointly authored publications above, the venue is indicated. Sections recalling definitions and results from other publications in the literature, or generally describing related work from the literature, are labelled 'review of related work'. Sections containing only original unpublished material are labelled 'unpublished'.

Chapter 4

4.1.1	review of related work
4.2	review of related work
4.3.1	unpublished
4.3.2	published in [E] and [I]
4.3.3	published in [E] and [I]

Chapter 5

5.1	review of related work
5.2	review of related work
5.3	published in [D] and [H]

Chapter 6	6.1 review of related work 6.2 review of related work 6.3 published in [E] and [I] 6.4 published in [E] and [I]
Chapter 7	7.1 review of related work 7.2 published in [F] 7.3 published in [F]
Chapter 8	8.1 review of related work 8.2 review of related work 8.3 unpublished
Chapter 9	9.1 review of related work 9.2 published in [G] 9.3 review of related work
Chapter 10	10.1 unpublished 10.2 published in [J] 10.3 unpublished
Chapter 11	11.1.1 review of related work 11.1.2 review of related work 11.1.3 published in [B] and [J] 11.2 published in [B] and [J]
Chapter 12	12.1 published in [F] 12.2 published in [F] 12.3 published in [F]

Acknowledgements

First of all I would like to express my gratitude to my supervisor Olaf Beyersdorff. When I first met Olaf in Leeds in the summer of 2015, I had no idea about the journey I was embarking on – I had no idea that I was becoming a computer scientist. Looking back, I feel privileged, having had the freedom to cultivate my ideas, and someone like Olaf who takes the time to understand them. I have come to see mathematics as an art, and it has been a most enjoyable experience. His supervision has been nothing short of excellent.

Second, I would like to thank Meena Mahajan for hosting me on three very rewarding visits to IMSc Chennai. I owe much to the immersive environment I was able to find in Chennai. Meena's clarity of thought and ability to use the language of mathematics has been an inspiration, and I'm certain I returned each time as a better researcher.

I also had the opportunity to make a very pleasant and productive visit to the Algorithms and Complexity group at TU Wien, and I would like to thank Friedrich Slivovsky for the invitation.

I am grateful to Barnaby Martin and Brandon Bennett for agreeing to review my thesis. The version you are reading now is certainly improved thanks to their thorough treatment.

Many other researchers and staff members have helped me out on this journey, scientifically and otherwise. In particular, I want to thank Leroy Chew, Luke Hinde, Judith Clymo, Sarah Sigley, Anil Shukla, Valerio Boncompagni, Li Lei, Pedro Copardo-Mendez, Tomáš Peitl, Martin Suda, David Sherratt, Judi Drew, Annemarie Kunze and Silvia Blaser. And a special acknowledgement to Gaynor Butterwick for some timely moral support.

Of course, some friends outside of science need a mention. Respect going out to Alan Smith for rekindling my interest in science, Joe Tordoff for being himself, Oliver Garthwaite for putting me up, and Diāna Spurīte for putting up with me.

Finally, I would like to thank my parents Marjorie and Geoff and my brothers Lee and Stephen. My parents have always supported me on whichever path I have chosen, and I can see both of them in this thesis. As I child, I inherited from my elder brothers a fondness for the Commodore 64 and the BASIC programming language. These are some of my earliest vivid memories, and without them, I wouldn't have done this.

Abstract

Quantified Boolean formulas (QBF), which form the canonical PSPACE-complete decision problem, are a decidable fragment of first-order logic. Any problem that can be solved within a polynomial-size space can be encoded succinctly as a QBF, including many concrete problems in computer science from domains such as verification, synthesis and planning. Automated solvers for QBF are now reaching the point of industrial applicability.

In this thesis, we focus on dependency awareness, a dedicated solving paradigm for QBF. We show that dependency schemes can be envisaged in terms of dependency quantified Boolean formulas (DQBF), exposing strong connections between these two previously disparate entities. By introducing new lower-bound techniques for QBF proof systems, we study the relative strengths of models of dependency-aware solving, including the proposal of new, stronger models.

Proof Complexity. Using the strategy extraction paradigm, we introduce new lower-bound techniques that apply to resolution-based QBF proof systems. In particular, we use the technique to prove exponential lower bounds for a new family of QBFs called the equality formulas. Our technique also affords considerably simpler, more intuitive proofs of some existing QBF proof-size lower bounds.

Models of solving. We apply our lower bound techniques to show new separations for QBF proof systems parametrised by dependency schemes. We also propose new models of dynamic dependency-aware solving and prove that they are exponentially stronger than the existing static models. Finally, we introduce Merge Resolution, a proof system modelling CDCL-style solving for DQBF, which is the first of its kind.

Contents

List of Figures

Part I

Foundations

Chapter 1

Introduction

In this thesis, we contribute to the theory of QBF solving by studying the relative strengths of associated proof systems. We introduce new lower-bound techniques for existing systems, and propose some new systems that model generalised and novel solving techniques.

1.1 Proof complexity and solving

None of this would be possible were it not for the strong connection between logic and computation. The strong connection allows us to build a bridge between theory and practice: on the theory side we have *proof complexity*, which studies the lengths of proofs in formal systems of logic; on the practical side we have the automation of decision procedures for formal languages, an endeavour referred to as *solving*. In the last two decades, solving has seen some quite remarkable advances [70], with proof complexity employed as its analytical toolkit [44].

Proof complexity

A working mathematician tasked with proving a conjecture has a number of worries. First and foremost, is the conjecture correct? Even if it is correct, is it provable? And even if it is provable, is there a proof which is sufficiently short that I might reasonably be expected to find it?

The last question is not so easy to answer, especially where 'hard problems' are concerned. Working mathematicians are well aware of one truth: the length of proof varies depending on the choice of theory. As undergraduates, for example, we encountered real integrals that were much easier to evaluate using techniques from complex analysis. And so, when it comes to hard problems, the working mathematician has a choice: to try and construct a long and complex proof within an existing theory, or to develop a new theory, aiming to construct a short and simple one.

Proof complexity is a branch of mathematics which deals formally with these questions of proof size. The central problem is to determine the length of the shortest proof of a given theorem, in a given theory of formal logic. The main idea is to compare the strengths of theories, whereby stronger theories have shorter proofs.

A seminal work in proof complexity is the 1979 paper from Cook and Reckhow, which introduced the abstract definition of a *proof system*:

Definition ([21])**.** *A* proof system *for a language $\mathcal{L}$ over an alphabet Σ is a poly-time computable function from Σ^* onto $\mathcal{L}$.*

Cook and Reckhow had discovered that the proof complexity of propositional logic, one of the most basic logics consisting only of Boolean variables and connectives, was related to an open problem in computational complexity:

Theorem ([21])**.** *There exists a polynomially bounded proof system for the language of unsatisfiable propositional formulas if, and only if,* $\mathsf{NP} = \mathsf{coNP}$*.*

A proof system is said to be *polynomially bounded* when every theorem has a short proof, where a proof is 'short' when its length is only polynomially larger than the theorem statement, for some fixed polynomial that does not depend on the theorem.

There is no Fundamental Theorem of Proof Complexity, but this result is a good candidate. It tells us that superpolynomial proof-size lower bounds are a valid approach towards the separation of complexity classes. This is quite remarkable; there is no obvious connection between lengths of proofs and consumption of computational resources, but the theorem of Cook and Reckhow establishes a firm correspondence in the spirit of the strong connection between logic and computation.

The following forty-five years saw a great deal of interest surrounding superpolynomial proof-size lower bounds in propositional logic, a prolonged effort which eventually culminated in the development of a range of lower-bound techniques [56, 18]. As yet, proof complexity theorists have not been able to separate NP from coNP. However, their lower-bound techniques became very relevant to the emerging advances in solving.

Solving

A solver is a piece of computer software that automatically solves a decidable problem. The user encodes the problem in the solver's input format, invokes the solver on the instance, and waits for a solution. Depending on the difficulty of the problem, the solver may provide a solution immediately, require five minutes or a couple of days, or work away in the background for as long as the user is willing to wait before terminating the process. Provided the solver is correct and capable of solving every problem instance, the process will eventually terminate and provide a solution.

For example, the *satisfiability problem* (SAT) is to determine whether a formula from propositional logic has a satisfying assignment. Many problems from theoretical and applied computer science, as well as pure mathematics, can be encoded in propositional logic, whereby solving the problem is reduced to solving a SAT instance. So a scientist, faced with solving a problem by hand, can instead encode it in propositional logic, and hope to solve it quickly using a SAT solver.

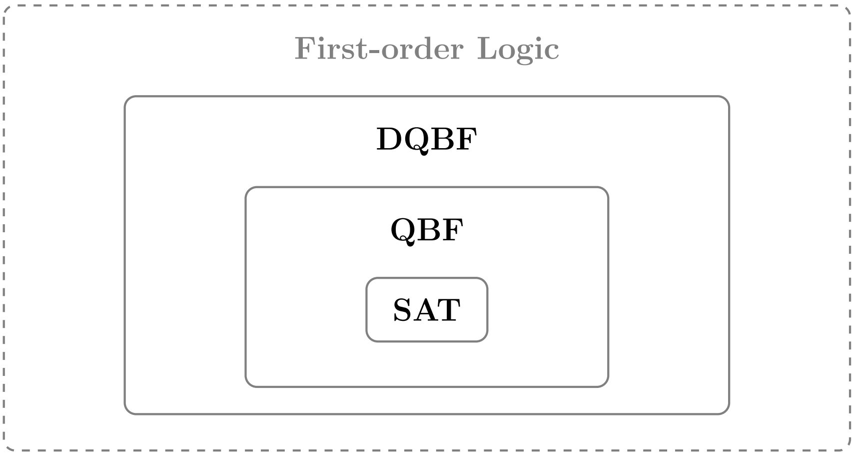
Nowadays, SAT solving serves a host of concrete applications in computer science. To cite only a few examples, it has been employed to formally verify software [31], synthesise circuit designs from specification [17], and to help solve long-standing open problems in number theory [30]. The rise of SAT implementations as universal, efficient, automatic problem solvers is arguably one of the most notable developments in the history of computer science. Authors heralding the 'SAT revolution' have done so in earnest [70].

Why are efficient SAT solvers considered such an important breakthrough? In part, this is due to the Cook-Levin Theorem.

Theorem ([20, 37])**.** *The SAT problem is* NP*-complete.*

There are two messages to be drawn here. First, every decision problem solvable in nondeterministic polynomial time can be encoded succinctly as a SAT instance, meaning that SAT is a universal language for a wide variety of problems. Second, SAT is the classical, prototypically hard problem, and if $\mathsf{P} \neq \mathsf{NP}$, as the average computational complexity theorist appears to believe, then it is not efficiently soluble. As a result, the practical progress in computationally hard problems, fostered by solving, appears to defy our theoretical intution about the difficulty of the problems we are trying to solve.

Another reason for the excitement surrounding SAT is what lies beyond. Progress towards even harder decision problems is already underway. Figure 1.1 depicts two notable cases: QBF, a fragment of first-order logic which is PSPACE-complete, and



SAT	propositional logic	NP	established efficient technology
QBF	propositional logic with totally ordered quantification	PSPACE	reaching industrial applicability
DQBF	propositional logic with partially ordered quantification	NEXP	in its infancy

Figure 1.1: Some logical decision problems in increasing order of complexity.

DQBF, an even larger fragment which is NEXP-complete. Decision procedures for these languages, whose formidable complexities were seemingly unapproachable before SAT, are being seriously developed for industrial applications. Perhaps this does indeed signify the dawn of a new age in the mechanical solution of hard problems.

A bridge between theory and practice

The relationship between proof complexity and solving boils down to the following notion.

Notion. *Every decision procedure implicitly defines a proof system for the language.*

Imagine a decision procedure defined in some computational model, say as a Turing machine [68]. The *trace* of the procedure on some input details the complete configuration of the machine at each time step, until termination. Given that the transition function of the machine is finite, and the basic operations simple, the fact that the trace encodes a correctly executed computation can be verified efficiently, with respect to the size of the trace. Hence the function that maps valid traces to the input string, and invalid traces to some fixed member of the language, is a proof system, as defined by Cook and Reckhow.

The notion itself is quite useful; it allows us to envisage the computational framework in terms of logic and proof complexity. We have already seen (by the Cook-Levin Theorem) that problem instances can be encoded as equivalent logical formulas. Moreover, as depicted in Figure 1.2, a solver can be identified with a proof system and its computation with a proof.

This is where complexity enters: a lower bound on proof size guarantees a lower bound on solver running time. More precisely, a superpolynomial proof size lower bound for a family of logical formulas guarantees that the instances cannot be solved efficiently; it specifies a hard problem for the solver.

If this were all that could be said about proof complexity and solving, it wouldn't be much. The 'proof system' implicitly defined by a solver is nothing other than its source code, and the size of the shortest proof is a redefinition of its running time. The real value of proof complexity is due to the fact that state-of-the-art solvers implicitly define proof systems which are *fragments* of well-known, well-analysed proof systems.

Whereas the value of upper-bounds breaks down when the implicit proof system is weaker (the existence of short proofs in a stronger proof system does not guarantee fast termination), the value of lower-bounds remains concrete. Lower-bound techniques developed by proof complexity theorists identify hard problems for state-of-the-art solvers.

Case study: Resolution and SAT

The most productive period for SAT solving began in 1996, with the advent of a solving technique called conflict-driven clause learning (CDCL) [60]. CDCL, which is based on the DPLL algorithm [22] (named after Davis, Putnam, Logemann and Loveland), implemented a number of innovations which significantly reduced solving times.

Before CDCL, SAT solvers were optimised implementations of a 1960s algorithm. After CDCL, they were efficient technologies, orders of magnitude quicker than their predecessors and capable of solving problems that were previously considered intractable. As it happens, proof complexity provides a compelling explanation for the

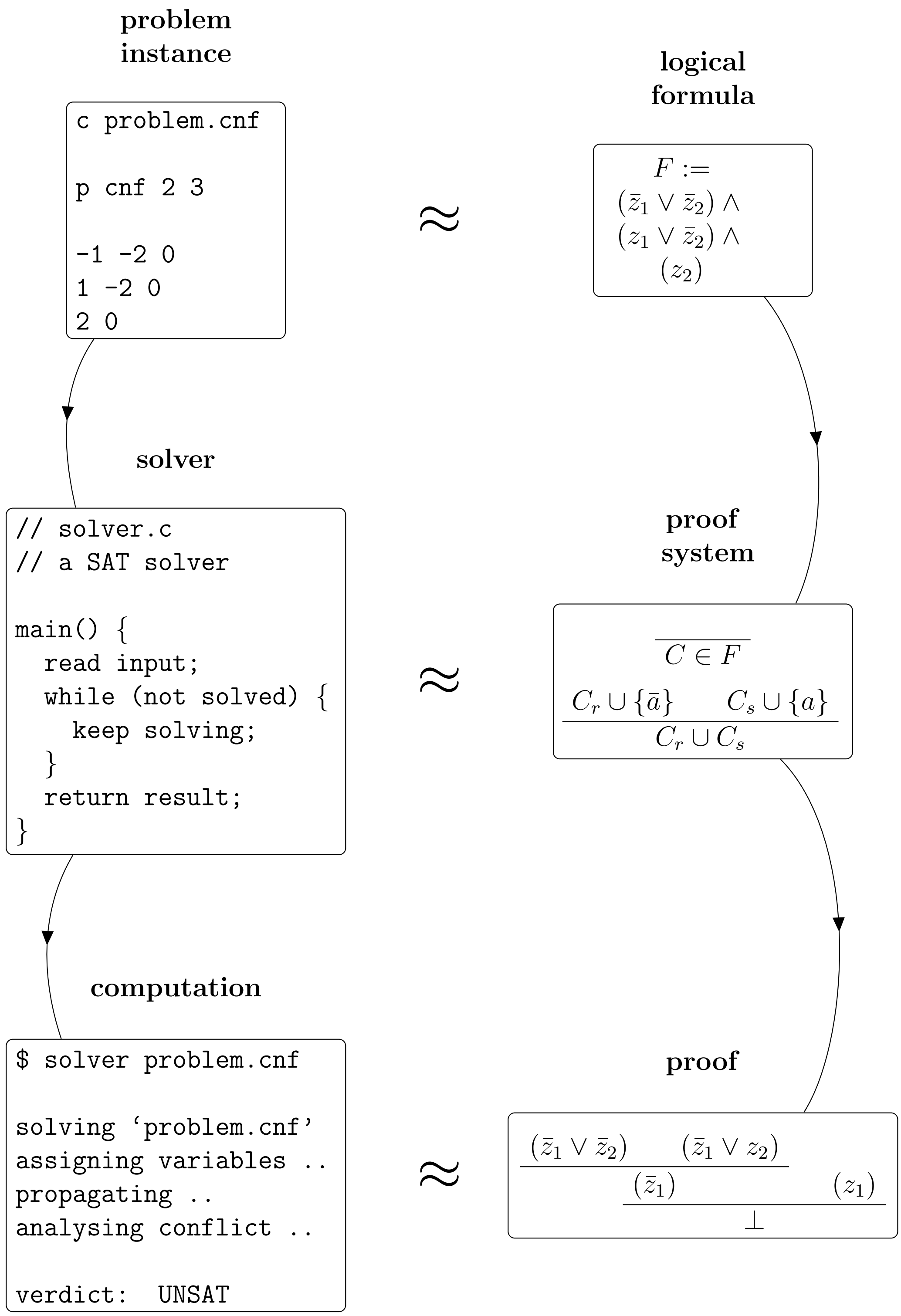


Figure 1.2: A bridge between theory and practice.

experimental gains. This is due to the fact that DPLL and CDCL, when run on unsatisfiable instances, essentially conduct a proof search.

The proof system in question is Resolution [52], a calculus that proves the unsatisfiability of propositional formulas. Superpolynomial lower bounds for Resolution were among the first major results in proof complexity [67], and it has received a great deal of attention ever since. Several lower-bound techniques have been developed, and the relative proof complexities of many fragments of Resolution are now well understood [56, 18, 44].

The most important fragments are Resolution itself, also referred to as General Resolution, and Tree-Like Resolution, the fragment whose derivations have the structure of trees. It is now known that DPLL conducts proof search in Tree-like Resolution [4], whereas CDCL conducts proof search in General Resolution [59]. Furthermore, General Resolution is *exponentially stronger*: there are unsatisfiable formulas (so-called 'pebbling formulas') that require exponential-size tree-like proofs, but admit polynomial-size proofs in general [4].

This is a point on which it is important to be precise. The non-existence of short tree-like proofs means that DPLL provably requires exponential time to solve the pebbling formulas. The existence of short general proofs does not mean that CDCL will solve the pebbling formulas efficiently, since the short proofs may be difficult to find. Nonetheless, it has the potential to do so, whereas DPLL does not.

Let us look at this the other way round for a moment. Suppose we are proof complexity theorists who have studied Resolution, but now we want to get our hands dirty: we want to implement proof search. First we devise an algorithm that searches the simpler tree-like proofs, which happens to be DPLL. Thereafter we modify the algorithm to search the shorter general proofs, and we come up with CDCL. Although we had no theoretical proof that our modified algorithm would efficiently find shorter proofs, we see a significant improvement.

This is the main message, the essence of the relationship between proof complexity and solving, and the theme of this thesis: Solving is proof search, proof systems model solving, and proof complexity calibrates the strength of models. We can look for better models by identifying and overcoming the underlying reasons for proof-size lower bounds.

1.2 Quantified Boolean formulas

In this thesis, we are interested in the proof complexity of models of QBF solving. We will introduce new lower-bound techniques to calibrate the strengths of existing models. Moreover, by identifying and overcoming the underlying reasons for hardness, we propose new models of solving that can potentially foster improved proof search.

A quantified Boolean formula (QBF) is a sentence from propositional logic in which all variables are quantified in a total order preceding the proposition. A typical, simple example would be the formula

$$\exists x \forall u \exists z \cdot (\neg x \vee \neg u \vee z) \wedge (x \vee u \vee z) \wedge (\neg z)\,.$$

In fact, we'll meet with this formula quite frequently throughout the thesis. It is the first instance of a family of hard QBFs called the *equality* family.

This particular QBF is false. Reading from left to right, it is typically interpreted as 'there exists a 0/1 value for x such that, for each 0/1 value for u there exists a 0/1 value for z such that the formula

$$(\neg x \vee \neg u \vee z) \wedge (x \vee u \vee z) \vee (\neg z)$$

evaluates to 1,' where the logical connectives '$\neg$', '$\wedge$' and '$\vee$' take their usual definitions. Taking the time to consider all 0/1 values, we would determine that this is not the case, and that there exists no such value for x. So the sentence, interpreted in the natural way, is not true.

The problem of determining whether or not a quantified Boolean formula is true is complete for the complexity class PSPACE. Thus, any problem that can be decided in polynomial space can be encoded as a polynomial-size sequence of QBFs. Since $\mathsf{NP} \subseteq \mathsf{PSPACE}$ (and the average complexity theorist believes the inclusion is strict), QBFs potentially provide more succinct and natural encodings of problems compared to propositional logic.

Indeed, like SAT, QBF reductions have been employed for formal verification [5] and synthesis [39], but also to further application domains such as automated planning [51], ontological reasoning [36], and fault correction [64]. In fact, the authors of [24] found that the QBF workflow actually outperformed SAT on a particular class of synthesis problems. It appears fair to say that QBF solving has reached the point of industrial applicability.

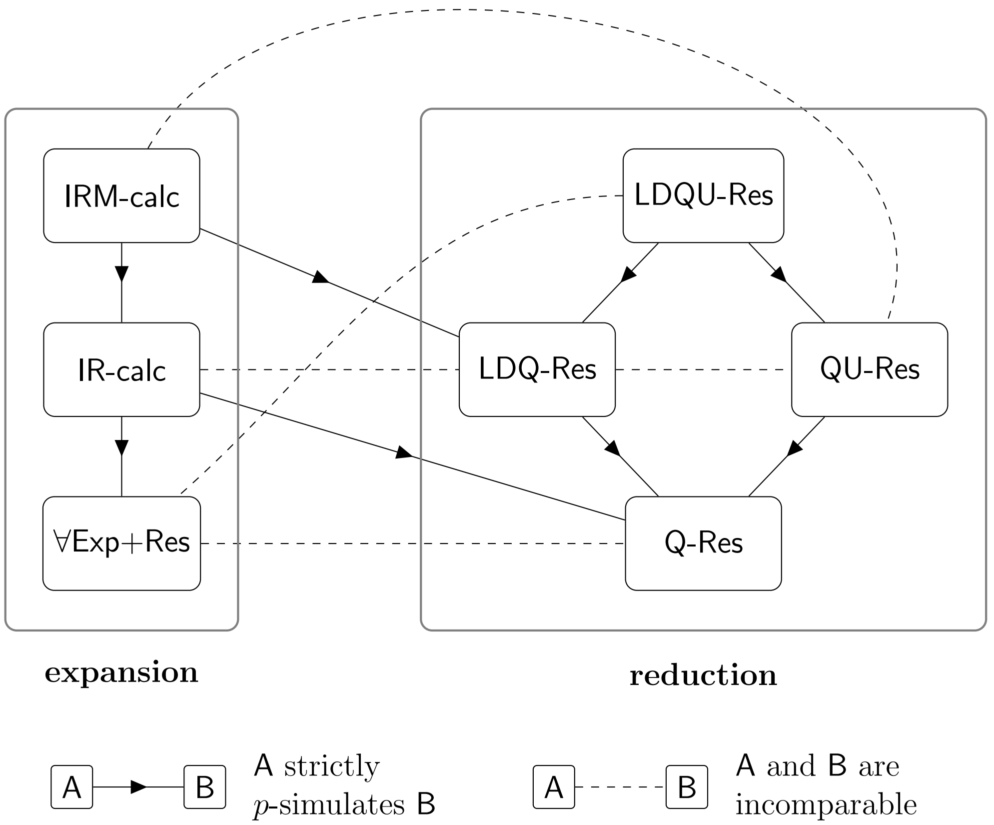


Figure 1.3: Simluation order of seven major QBF proof systems.

QBF proof systems

Research into QBF proof complexity began in 1995 [35]. Since then numerous proof systems have been proposed, many of them based on existing propositional proof systems. Given that state-of-the-art SAT solvers correspond to fragments of Resolution, it is not surprising that almost all of the proof systems pertintent to QBF solving are based on Resolution.

Figure 1.3 (reproduced from [15]) depicts seven resolution-based QBF proof systems along with their *simulation order*. A formal definition of simulation is given in Chapter 2. For now, one proof system *p-simulates* another if proofs in the latter can be translated efficiently into proofs in the former; two proof systems are incomparable if neither p-simlutates the other.

What is really unique to each system is the handling of universal quantification, which falls into one of two brackets: universal expansion and universal reduction. The division of QBF proof systems into these two brackets is analogous to the situation in solving, where universal expansion and universal reduction are two major paradigms. Interestingly, the performance of expansion and reduction-based solvers on benchmark sets are often markedly different [42].

Universal expansion

The expansion-based QBF proof systems $\forall\mathsf{Exp}+\mathsf{Res}$, $\mathsf{IR\text{-}calc}$ and $\mathsf{IRM\text{-}calc}$ are shown on the left-hand side of Figure 1.3.

The idea behind the universal expansion paradigm is quite simple. The universal quantification is handled by a translation to propositional logic, whereby universal variables are 'expanded out'. The resulting formula is a fully existentially quantified QBF, which is true if, and only if, the propositional part is satisfiable.

An expansion-based QBF solver, such as RAReQs [33], typically tries to expand out the universal variables in the most economical way, and then calls a SAT solver. The corresponding proof system is $\forall\mathsf{Exp}+\mathsf{Res}$. The stronger system $\mathsf{IR\text{-}calc}$ is related to the solver iDQ [25]. The strongest of the expansion systems, $\mathsf{IRM\text{-}calc}$, is not known to correspond to a QBF solver.

Universal reduction

The universal reduction systems are shown on the right-hand side of Figure 1.3.

Universal reduction is a fundamentally different approach to solving QBFs. CDCL is augmented with new propagation rules to handle universal quantification during the search process. The resulting decision procedure is called Quantified Conflict-Driven Constraint Learning (QCDCL). Solvers based on QCDCL include DepQBF [41] and Qute [46].

Universal reduction in QCDCL is underpinned by the proof system $\mathsf{Q\text{-}Res}$ [35]. A more sophisticated form of reduction [72] is underpinned by the stronger system $\mathsf{LDQ\text{-}Res}$ [2]. The other two systems $\mathsf{QU\text{-}Res}$ and $\mathsf{LDQU\text{-}Res}$ are not known to correspond to particular solvers.

Dependency schemes

One of the first challenges identified for QBF solving concerns the allowable order of variable assignments [40]. In standard QCDCL, the freedom to assign variables is limited according to the total order imposed by the quantifier prefix.

The prenexed quantifier prefix of a QBF introduces dependencies between variables. For example, consider an existential variable x quantified after a universal variable u. Reading the prefix left to right, the value of the variable x witnessing the existential quantification is allowed to depend on the value of u.

Standard QCDCL solvers must respect variable dependencies during search, insofar as a variable cannot be assigned before any of the variables on which it depends. This carries a drawback, namely, reduced impact of so-called *decision heuristics*, routines which determine the order of variable assignments in a solver. Given that decision heuristics play a major role in the success of CDCL [58, 57, 38, 43], the drawback for QBF is significant.

At the same time, coercing the order of assignment to respect the prefix is frequently needlessly restrictive [40]. Solvers can often turn a blind eye to many of the dependencies implied by the quantifier prefix [46], boosting the utility of the decision heuristic.

Dependency awareness, first implemented in the solver DepQBF [41], 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 total order of the prefix is supplanted by a partial order that better approximates the variable dependencies of the instance, granting the solver greater freedom regarding variable assignments. Despite the additional computational cost incurred, empirical results demonstrate improved solving on many benchmark instances [40].

Dependency schemes themselves are tractable algorithms that identify dependency information. From the plethora of schemes that have been proposed in the literature, two have emerged as the principal ones: the *standard dependency scheme* ($\mathcal{D}^{\mathrm{std}}$ [54]) and the *reflexive resolution path dependency scheme* ($\mathcal{D}^{\mathrm{rrs}}$ [63]). A solid theoretical model for dependency awareness was proposed in the form of the calculus $\mathsf{Q}(\mathcal{D})\text{-}\mathsf{Res}$ [63], a parametrisation of $\mathsf{Q}\text{-}\mathsf{Res}$ by the dependency scheme $\mathcal{D}$.

Dependency schemes are closely related to DQBF, although this was not recognised at the outset (see [54]). In fact, the whole paradigm of dependency-aware solving can be recast in terms of DQBF, taking the form of a detour into a larger fragment of first-order logic. The formal recognition, and the clarification of the relationship between dependency schemes and dependency quantified Boolean formulas, is one of the contributions of the thesis.

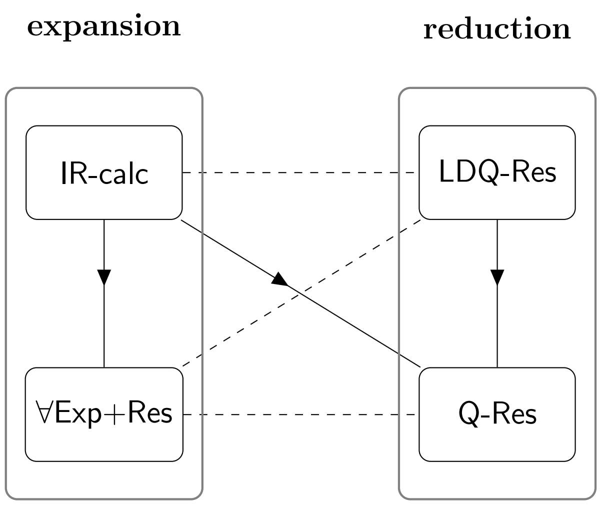


Figure 1.4: Simulation order of our four main QBF proof systems.

1.3 Contributions

In this thesis, we focus on the four resolution-based QBF proof systems shown in Figure 1.4. From the point of view of QBF solving, these are the most important ones of the seven from Figure 1.3.

Technical contributions of the thesis appear in Parts II and III. Part II focuses on lower-bound techniques, whereas Part III focuses on their application to new and existing models of QBF solving.

Convention for results in this thesis. Results taken from the literature are always referenced, the citation appearing in parenthesis immediately before the statement of the result. Folklore results are indicated similarly. All results appearing in this thesis without a citation, and not designated as folklore, are original contributions of the thesis.

Part II: Lower-bound Techniques

In Part II we introduce new semantic techniques and prove some new lower bounds for QBF proof systems, while providing shorter, intuitive proofs of known lower bounds.

- **Chapter 4**: We completely characterise lower bounds in $\forall$Exp+Res (Theorem 4.13) by means of a semantically-grounded lower-bound technique. We prove new exponential lower bounds for the equality family (Corollary 4.15), and the interleaved equality family (Corollary 4.17).

- **Chapter 5**: Exploiting strategy extraction, we lift the lower-bound technique to Q-Res for formulas of bounded quantifier alternation (Corollary 5.17). As an application, we prove a new exponential lower bound for the equality formulas (Theorem 5.18).

- **Chapter 6**: Appealing further to strategy extraction, we lift the lower-bound technique to unbounded quantifier alternation in the stronger system IR-calc (Theorem 6.18). We provide an alternative, semantically-grounded proof of the lower bound for the Kleine Büning et al. family (Theorem 6.20) that first appeared in [14].

- **Chapter 7**: We introduce the squared equality family (Definition 7.7) and prove that they require exponential-size refutations in a particular fragment of LDQ-Res called *deferred* LDQ-Res (Theorem 7.11).

Part III: Models of Solving

Part III explores the wider context of dependency schemes and DQBF. We use the lower-bound techniques from Part II to separate existing proof systems parametrised by dependency schemes. We also propose some new models of solving and separate them from the existing ones.

- **Chapter 8**: We propose the expansion-reduction hypothesis (Idea 8.5) as a credible explanation for the issues associated with lifting QBF proof systems to DQBF.

- **Chapter 9**: Utilising DQBF, we propose a new interpretation for dependency schemes, and show that it accomodates (and simplifies) the existing theory.

- **Chapter 10**: Parametrising the expansion systems by dependency schemes, we introduce two new models of dependency-aware QBF solving (Definitions 10.6 and 10.16). We show that they are exponentially stronger than the base systems when the dependency scheme is $\mathcal{D}^{\mathrm{rrs}}$ (Theorems 10.13 and 10.20).

- **Chapter 11**: We show that the parametrisation of Q-Res by $\mathcal{D}^{\mathrm{rrs}}$ also gives exponentially shorter proofs (Theorem 11.7). We go on to introduce a new model of dynamic dependency-aware QBF solving (Definition 11.10), and show that this promotes a further exponential speedup over the existing static system (Theorem 11.17).

- **Chapter 12**: Following the expansion-reduction hypothesis, we investigate reduction systems in the context of DQBF. We propose a new solving model called Merge Resolution (Definition 12.10), a natural extension of LDQ-Res, which has strategy extraction built in. We show that it p-simulates deferred LDQ-Res on QBFs (Theorem 12.19), and is even exponentially stronger thanks to short proofs of the squared equality family (Theorem 12.21).

Chapter 2

Propositional Logic

In this chapter, we cover the background material on propositional logic, which forms the technical foundation for the rest of the thesis. Our preferred syntax, conjunctive normal form, is covered in Section 2.1, the corresponding semantics follow in Section 2.2. Section 2.3 focuses on Resolution, arguably the most famous propositional proof system. The abstract theory of proof systems due to Cook and Reckhow is covered in Section 2.4.

2.1 Syntax

The two most fundamental objects in the object language for propositional logic are the *domain of discourse* and the *universe*.

Definition 2.1 (domain of discourse)**.** *The* domain of discourse *is the set of symbols* $\{0, 1\}$.

The domain of discourse is denoted '$\mathbb{D}$'. The particular symbols chosen as the elements of $\mathbb{D}$ are unimportant. One can think of $\mathbb{D}$ as the set $\{0, 1\}$ or $\{f, t\}$ or $\{\diamondsuit, \heartsuit\}$ as one pleases; they are merely symbols in an object language, and no relations or operations are defined on them.

Definition 2.2 (universe)**.** *The* universe *is a countably infinite set of symbols.*

The universe is denoted '$\mathbb{U}$'. The elements of the universe are called *variables*. One could think of the universe concretely as the set of natural numbers. In practice, it is better to use lower-case roman letters as our variable symbols, with subscripts and superscripts where convenient. By a *set of variables* we mean a subset of $\mathbb{U}$, and usually a finite subset. The only infinite set of variables we will encounter is $\mathbb{U}$ itself.

Conjunctive normal form

A *conjunctive normal form formula* is traditionally defined as a conjunction of disjunctions of literals, where a literal is a variable or its negation. The following would be a typical example:

$$(z_1 \vee \neg z_2) \wedge (\neg z_1) \wedge (z_1 \vee z_2 \vee \neg z_3)$$

Nowadays it has become commonplace for authors to use a different notation for CNFs: each disjunct is written as a set of literals, the conjunction as a set of sets, and negation denoted with an overline:

$$\{\{z_1, \bar{z}_2\}, \{\bar{z}_1\}, \{z_1, z_2, \bar{z}_3\}\}$$

We use the more convenient set-theoretic notation throughout.

Notice that the explicit binary connectives '$\wedge$' and '$\vee$' disappear, and are now represented implicitly as a set hierarchy. It also makes sense to do away with '$\neg$' as a unary connective, and instead to define a set of literals, which is essentially two distinct copies of $\mathbb{U}$.

For each variable $z \in \mathbb{U}$ we introduce two symbols:

- The negative literal symbol '$\bar{z}$';
- The positive literal symbol 'z', which is identical the the variable symbol itself.

We say that $\bar{z}$ has *negative polarity* and z has *positive polarity*. These symbols form the countable set

$$\mathbb{L} := \{\bar{z} : z \in \mathbb{U}\} \cup \{z : z \in \mathbb{U}\}\,.$$

Definition 2.3 (literal, clause, CNF)**.** *A* literal *is an element of the set* $\mathbb{L}$*, a* clause *is a finite set of literals, and a* CNF *is a finite set of clauses.*

The *complement* of a literal a in $\mathbb{L}$ is

$$\tilde{a} := \begin{cases} z & \text{if } a = \bar{z}\,, \\ \bar{z} & \text{if } a = z\,, \end{cases}$$

and we say that a and $\tilde{a}$ are *complementary.*

The set of all clauses is denoted '$\mathbb{C}$'. A clause is called *tautological* if it includes a pair $\{\bar{z}, z\}$ of complementary literals. The *empty clause*, i.e. the empty set, is denoted '$\emptyset$'. We say that a clause C *subsumes* another clause D when $C \subseteq D$.

The set of all CNFs is denoted '$\mathbb{F}$'. Like the empty clause, the empty CNF is just the empty set, denoted '$\emptyset$'. There is never any ambiguity here, and it is always clear from the context which is meant. The size of a CNF, denoted '$|F|$', is its cardinality, i.e. the number of clauses it contains.

It will be frequently useful to identify the variable corresponding to a literal, the variables appearing in a clause, or those appearing in a formula. For each variable z, and each literal a in $\{\bar{z}, z\}$, the variable of a is

$$\mathrm{var}(a) := z\,.$$

The variables of a clause are the elements of the set

$$\mathrm{vars}(C) := \{\mathrm{var}(a) : a \in C\}\,,$$

and the variables of a formula are the elements of the set

$$\mathrm{vars}(F) := \textstyle\bigcup_{C \in F} \mathrm{vars}(C)\,.$$

Substitution

Substitution is merely the process of replacing one symbol for another. More precisely, a variable symbol is replaced either by another variable symbol, or a symbol from the domain of discourse. We define a substitution as a mapping that details what replaces what.

Definition 2.4. *A* substitution *is a function from a variable set Z into $\mathbb{U} \cup \mathbb{D}$.*

When one applies a substitution to a formula in the traditional sense, it is normal to remove the symbol 0 from disjunctions and the symbol 1 from conjunctions, as this preserves the truth values defined by those connectives. Therefore substitution is not exactly a straight swap – there is a little bit of tidying up to be done.

The same thing happens when we apply substitutions to CNFs in set notation, we perform a swap followed by some tidying. We describe the effect of an arbitrary substitution s on literals, then on clauses, and finally on CNFs.

Given a variable z in the domain of s, the application of s to the negative literal $\bar{z}$ is

$$\bar{z}[s] := \begin{cases} \bar{x} & \text{if } s(z) = x \in \mathbb{U}\,, \\ 1 & \text{if } s(z) = 0\,, \\ 0 & \text{if } s(z) = 1\,, \end{cases}$$

and the application to the positive literal z is

$$z[s] := s(z)\,.$$

For any variable z that is not in the domain of s, application of the substitution has no effect on either literal:

$$\bar{z}[s] := \bar{z} \quad \text{and} \quad z[s] := z\,.$$

Applying s to a clause C is defined as follows:

$$C[s] := \begin{cases} \mathbb{L} & \text{if } a[s] = 1 \text{ for some literal } a \text{ in } C\,, \\ \{a[s] : a \in C\} \setminus \{0\} & \text{otherwise}\,. \end{cases}$$

The choice of the symbol $\mathbb{L}$ will be discussed in the next subsection.

The application of s to a CNF F is

$$F[s] := \{C[s] : C \in F\} \setminus \{\mathbb{L}\}\,.$$

2.2 Semantics

Assignments and satisfiability

Assignments are special kinds of substitutions, namely those which map into the domain of discourse. Hence, they replace variable symbols with constant symbols.

Definition 2.5. *An* assignment *is a function from a variable set Z into $\mathbb{D}$.*

By an *assignment to Z* we mean an assignment whose domain is Z. The set of all assignments to Z is denoted '$\langle Z\rangle$'. An assignment to a subset of Z is called a *partial assignment to Z*. The set of all partial assignments to Z is denoted '$\langle\langle Z\rangle\rangle$'. The domain of an assignment σ is denoted '$\mathrm{vars}(\sigma)$'.

An assignment can be specified explicity as a function, for example

$$\begin{array}{rcrcl} \alpha & : & \{z_1, z_2, z_3\} & \to & \mathbb{D} \\ & & z_1 & \mapsto & 0 \\ & & z_2 & \mapsto & 1 \\ & & z_3 & \mapsto & 0\,. \end{array}$$

However, it is also conventional to represent assignments as sets of literals, where the negative literal $\bar{z}$ represents the assignment $z \mapsto 0$ and the positive literal z represents the assignment $z \mapsto 1$. Hence we could also specify the assignment α by writing

$$\alpha := \{\bar{z}_1, z_2, \bar{z}_3\}\,.$$

We will use both forms throughout the thesis. It is always clear from the context whether a set of literals represents a clause or an assignment.

The *negation* of an assignment is the clause consisting of the complementary literals. For example, the negation of α is the clause $\{z_1, \bar{z}_2, z_3\}$.

Given an assignment σ to a set of variables Z, and a variable z in Z, the *complementary assignment* for σ with respect to z, denoted '$\mathrm{comp}(\sigma, z)$' is the assignment obtained from σ by flipping the value of z; that is

$$\begin{array}{rcl} \mathrm{comp}(\sigma, z) \;:\; Z & \to & \mathbb{D} \\ z' & \mapsto & \begin{cases} 0 & \text{if } z' = z \text{ and } \sigma(z) = 1\,, \\ 1 & \text{if } z' = z \text{ and } \sigma(z) = 0\,, \\ \sigma(z') & \text{if } z' \neq z\,. \end{cases} \end{array}$$

Given a subset Z' of Z, the assignment obtained from σ by restricting the domain to Z' is denoted '$\sigma\restriction_{Z'}$'.

Given a second assignment τ, the *completion* of σ by τ is the assignment

$$\sigma \diamond \tau := \sigma \cup \tau\restriction_{\mathrm{vars}(\tau)\setminus\mathrm{vars}(\sigma)}\,,$$

that is, the assignment obtained from σ by adding τ wherever σ is undefined.

Now we introduce some terminology pertaining to the application of the arbitrary assignment σ. First, literals. We say that σ *falsifies* the literal a when $a[\sigma]$ is 0, and *satisfies* it when $a[\sigma]$ is 1.

We say that σ falsifies a clause when it falsifies all of its literals, and satisfies it when it satisfies at least one of its literals. It is easy to see that

$$\sigma \text{ falsifies } C \Leftrightarrow C[\sigma] = \emptyset \quad \text{ and } \quad \sigma \text{ satisfies } C \Leftrightarrow C[\sigma] = \mathbb{L}\,. \tag{2.1}$$

We use the symbol $\mathbb{L}$ to represent a satisfied clause because it is the closest thing we have to the opposite of the falsified clause $\emptyset$, namely its set complement.

An assignment is said to falsify a CNF when it falsifies at least one of its clauses, and to satisfy a CNF when it satisfies all of its clauses. Therefore

$$\sigma \text{ falsifies } F \Leftrightarrow \emptyset \in F[\sigma] \quad \text{ and } \quad \sigma \text{ satisfies } F \Leftrightarrow F[\sigma] = \emptyset\,. \tag{2.2}$$

It is easy to check that a total assignment to a CNF, i.e. an assignment to $\mathrm{vars}(F)$, either satisfies F or falsifies F, and not both. If F has a satisfying assignment we call it *satisfiable*, otherwise we call it *unsatisfiable.*

Now, the satisfaction of a clause represents a kind of disjunction of the satisfaction of its literals. This explains why occurrences of 0, left behind by falsified literals, are discarded. Also, the satisfaction of a CNF represents a kind of conjunction of the satisfaction of its clauses, and this also explains why occurrences of $\mathbb{L}$, left behind by satsified clauses, are discarded. Indeed, in a clear sense, our definition of the application of a substitution, along with (2.1) and (2.2), really define the truth tables of the logical connectives $\wedge$, $\vee$ and $\neg$, which are absent from our object language.

Semantic entailment

We define the entailment relation on $\mathbb{F} \times \mathbb{F}$ as follows.

$$F \vDash G \quad \Leftrightarrow \quad \text{every assignment in } \langle \mathrm{vars}(F \cup G) \rangle \text{ satisfying } F \text{ also satisfies } G\,.$$

When $F \vDash G$ holds, we say that F *entails* G. Note that, if F indeed entails G, then it entails any subset of G. Every CNF entails the empty CNF.

It is also easy to see that that an unsatisfiable CNF entails all other CNFs. Moreover, a CNF is unsatisfiable if, and only if, it entails $\{\emptyset\}$.

Complexity

Under a suitable encoding as binary strings, the set of satisfiable formulas forms the canonical NP-complete language SAT [20, 37]. This is the famous Cook-Levin Theorem [20, 37]. The set of unsatisfiable formulas forms the canonical coNP-complete language UNSAT.

2.3 Resolution

Resolution [52] is a logical system (*proof system*, *calculus*) employed to identify unsatisfiable CNFs. It is often referred to as a 'refutational' system, by which it is meant that the system refutes the satisfiability of the given formula. Intense research into Resolution began in the mid 1990s, fuelled by advances in satisfiability testing. Indeed, the calculus is closely linked with the DPLL and CDCL algorithms – we return to this topic later in the section.

The Resolution proof system

At the centre of the Resolution proof system is a binary inference rule, whose antecedent is called a *resolvent*. Given a literal p and two clauses C_1 and C_2 with $p \in C_1$ and $\tilde{p} \in C_2$, the resolvent of C_1 and C_2 over *pivot literal* p is

$$\mathrm{res}(C_1, C_2, p) := \big(C_1 \setminus \{p\}\big) \cup \big(C_2 \setminus \{\tilde{p}\}\big)\,.$$

Axiom:	$\dfrac{}{C}$	C is a clause in F
Resolution:	$\dfrac{C_r \qquad C_s}{C}$	C is a resolvent of C_r and C_s
Weakening:	$\dfrac{}{\mathbb{L}} \qquad \dfrac{C_1}{C}$	C_1 subsumes C

Figure 2.1: The rules of Resolution as logical inferences. The input CNF is F.

and the *pivot variable* is $\mathrm{var}(p)$. Any clause C that satisfies $C = \mathrm{res}(C_1, C_2, a)$ for some literal a is referred to as 'a resolvent of C_1 and C_2'.

In essence, a Resolution refutation is just a sequence of resolvents, beginning from some CNF, and ending at the empty clause. In practice, and in this thesis in particular, a more general presentation with a weakening rule is preferred.

Definition 2.6 (Resolution derivation)**.** *A Resolution derivation from a CNF F is a sequence $C_1, \ldots, C_k$ in which at least one of the following holds for each $i \in [k]$:*

- **A** Axiom: *C_i is a clause in F;*
- **R** Resolution: *C_i is resolvent of C_r and C_s, for some $r, s < i$;*
- **W** Weakening: *C_i is $\mathbb{L}$, or is subsumed by C_r for some $r < i$.*

The final clause of a derivation is called its *conclusion*. A Resolution derivation from F whose conclusion is empty is called a *refutation* of F. The size of a derivation is the number of clauses.

The three rules of Resolution, viewed as logical inferences with antecedents and consequents, are depicted in Figure 2.1.

Example 2.7. Consider the following Resolution derivation $\pi := C_1, \ldots, C_7$ from the formula $F := \{\{x, y\}, \{x, \bar{y}\}, \{\bar{x}, z\}, \{\bar{x}, \bar{z}\}\}$.

A	$C_1 := \{x, y\}$	axiom
A	$C_2 := \{x, \bar{y}\}$	axiom
A	$C_3 := \{\bar{x}, z\}$	axiom
A	$C_4 := \{\bar{x}, \bar{z}\}$	axiom
R	$C_5 := \{x\}$	resolution, i.e. $C_5 = \mathrm{res}(C_1, C_2, y)$
R	$C_6 := \{\bar{x}\}$	resolution, i.e. $C_6 = \mathrm{res}(C_3, C_4, z)$
R	$C_7 := \emptyset$	resolution, i.e. $C_7 = \mathrm{res}(C_5, C_6, x)$

Clauses $C_1, \ldots, C_4$ belong to F and are introduced as axioms. The remaining clauses are derived by resolution. Since the conclusion C_7 is the empty clause, π is a refutation of F. The size of π is 7. It is readily verified that F is indeed unsatisfiable. ■

Unlike Example 2.7, our definition of Resolution derivation does not state the choice of rules and antecedents used; they are implicit and, as a result, ambiguous. We could of course make this information explicit, writing it on the side of the derivation as in Example 2.7, but this is rather cumbersome. On the other hand, an unambiguous account of the choice of rules and antecedents is often useful, in particular for proof by mathematical induction.

For that reason, we simply assume that an arbitrary but fixed choice of inference rules and antecedents is associated with any given refutation; that is, in any refutation $\pi := C_1, \ldots, C_k$, each C_i is the consequent of the application of exactly one inference rule whose antecedents are well-defined.

Soundness

We proceed to show that Resolution is both *sound* and *complete* for unsatisfiable formulas; that is, a formula has a Resolution refutation if, and only if, it is unsatisfiable.

To say that Resolution is 'sound' is to say that only unsatisfiable formulas have refutations. This is very easy to prove, and comes down to semantic entailment. In fact, in a derivation every clause C_i is an 'implicant' of the input CNF F in the sense that $F \vDash F \cup \{C_i\}$. Since the conclusion C_k is the empty clause, every assignment to $\mathrm{vars}(F)$ falsifies $F \cup \{C_k\}$, and therefore falsifies F as well.

Fact 2.8. *A formula is unsatisfiable if it has a Resolution refutation.*

Proof. Let $\pi := C_1, \ldots, C_k$ be a Resolution derivation of a formula F. By induction on $i \in [k]$, we prove that $F \vDash \{C_i\}$. For the base case $i = 1$, C_1 was introduced by hypothesis and therefore belongs to F, hence $F \vDash \{C_1\}$ holds trivially. For the inductive step, we let $i \geq 2$, and consider two cases:

A If C_i was introduced by hypothesis the inductive step is identical to the base case.

R If C_i was derived by resolution, then C_i is the resolvent of C_r and C_s over pivot literal p for some $r, s < i$. Let σ be an assignment to $\mathrm{vars}(F)$ satisfying F. By the inductive hypothesis, σ satisfies C_r and C_s. Aiming for contradiction, suppose that σ falsifies C_i. If σ falsifies p then it also falsifies C_r (which subsumes $C_i \cup \{p\}$). On the other hand, if σ falsifies $\tilde{p}$ then it also falsifies C_s (which subsumes $C_i \cup \{\tilde{p}\}$). In either case, we reach a contradiction, since σ satisfies both C_r and C_s by the inductive hypothesis.

W If C_i was derived by weakening, there are two subcases. First, if $C_i = \mathbb{L}$, the inductive step follows trivially, since $\mathbb{L}$ is satisfied by every assignment by definition. Second, if C_i is subsumed by C_r with $r < i$, then $\{C_r\} \vDash \{C_i\}$, so $F \vDash \{C_r\} \vDash \{C_i\}$ by the inductive hypothesis. □

Completeness

To say that Resolution is 'complete' is to say that every unsatisfiable formula has a refutation. This is also very easy to prove, by means of a simple construction.

Fact 2.9. *A formula has a Resolution refutation if it is unsatisfiable.*

Proof. Let F be an unsatisfiable CNF over variables $Z := \{z_1, \dots, z_n\}$, and let $\alpha_1, \dots, \alpha_{2^n}$ define the natural lexicographic ordering of the assignments to Z, as in

$$\begin{array}{ccccc}
\sigma_1 & := & z_1 \mapsto 0, \dots, z_{n-2} \mapsto 0, z_{n-1} \mapsto 0, z_n \mapsto 0 & \approx & 0\cdots 000\,, \\
\sigma_2 & := & z_1 \mapsto 0, \dots, z_{n-2} \mapsto 0, z_{n-1} \mapsto 0, z_n \mapsto 1 & \approx & 0\cdots 001\,, \\
\sigma_3 & := & z_1 \mapsto 0, \dots, z_{n-2} \mapsto 0, z_{n-1} \mapsto 1, z_n \mapsto 0 & \approx & 0\cdots 010\,, \\
\sigma_4 & := & z_1 \mapsto 0, \dots, z_{n-2} \mapsto 0, z_{n-1} \mapsto 1, z_n \mapsto 1 & \approx & 0\cdots 011\,, \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
\sigma_{2^n} & := & z_1 \mapsto 1, \dots, z_{n-2} \mapsto 1, z_{n-1} \mapsto 1, z_n \mapsto 1 & \approx & 1\cdots 111\,.
\end{array}$$

Letting '$\circ$' denote concatenation of sequences, let $\pi := \pi_n \circ \cdots \circ \pi_0$ be the sequence in which each

$$\pi_i := C^i_1, \dots, C^i_{2^i}\,,$$

and the clauses C^i_j are defined recursively as follows:

- for j in $[2^n]$, C^n_j is the largest clause falsifed by σ_j;
- for i in $[n]$ and j in $[2^{i-1}]$, $C^{i-1}_j := \mathrm{res}(C^i_{2j-1}, C^i_{2j}, z_i)$.

It is easy to see that each clause in π_n can be derived from a clause in F by weakening, and it is readily verified by downward induction on i in $\{n, \dots, 0\}$ that each clause in π can be derived from preceding clauses by resolution. Moreover, it is easy to see that the conclusion C^0_1 is the empty clause. Thus

$$\mathrm{seq}(F) \circ \pi$$

is a Resolution refutation of F, where $\mathrm{seq}(F)$ denotes the clauses of F written in an arbitrary sequence. □

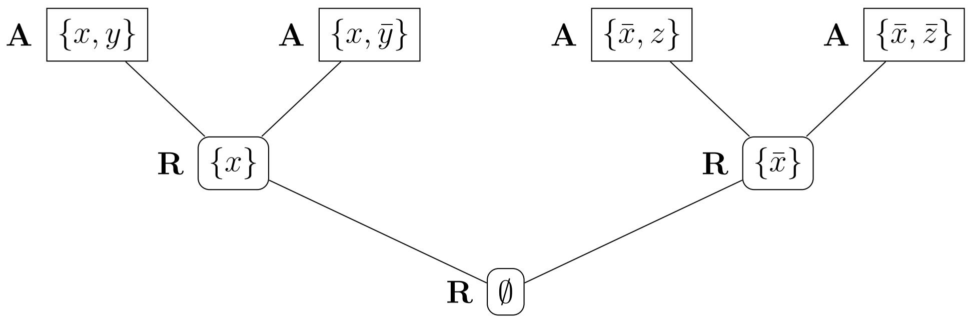


Figure 2.2: Depiction of the tree-like refutation from Example 2.7.

Tree-like and DAG-like derivations

A Resolution derivation can be viewed as a directed acyclic graph (DAG) in which nodes are clauses and edges are inferences. More precisely, given a Resolution derivation $\pi := C_1, \ldots, C_k$, construct a graph $G_\pi := (V, E)$ as follows.

- For each clause C_i add a vertex v_i, labelled with C_i, to V.
- For each clause C_i, and each antecedent C_r of C_i, add an edge (v_r, v_i) to E.

The graph G_π is called the *underlying dag* for π.

Figure 2.2 depicts the underlying DAG for the Resolution refutation in Example 2.7. In this case, the DAG happens to be a tree. Derivations such as this one, for which the underlying DAG is a tree, are called *tree-like* derivations.

Informally, and more intuitively, a derivation is tree-like when each clause is an antecedent of at most one inference. The restriction of Resolution to tree-like derivations is known as *Tree-like Resolution*. When a comparison with the tree-like version is to be emphasised, Resolution proper is often referred to as *General Resolution*.

2.4 Abstract Proof Systems

We will work with concrete proof systems throughout this thesis. Nonetheless, an overview of the abstract theory of proof complexity is essential.

Abstract versus concrete definitions

The following definition is the central one for proof complexity. By a *language*, we mean a subset of $\{0, 1\}^*$.

Definition 2.10 (proof system [21])**.** *A* proof system *for a language $\mathcal{L}$ over an alphabet Σ is a polynomial-time computable function from Σ^* onto $\mathcal{L}$.*

Intuitively, one can interpret the strings over the alphabet Σ as the proofs of the system. Formally, given a proof system P for $\mathcal{L}$ over Σ, a string $\pi \in \Sigma^*$ for which $\mathsf{P}(\pi) = x$ is called a P-proof of $x \in \mathcal{L}$. In this sense, Definition 2.10 captures three important features of a proof system:

- *Soundness*: there exists no P-proof of a string not in $\mathcal{L}$, in other words the codomain of P is $\mathcal{L}$;

- *Completeness*: there exists a P-proof of every string in $\mathcal{L}$, in other words the range of P is $\mathcal{L}$;

- *Polynomial-time checkability*: P-proofs can be checked efficiently, in other words P is polynomial-time computable.

The abstract formalisation of these features, however, is not ideal in the concrete setting. Hence, in practice we recognise that the systems with which we are concerned can be made to fit Definition 2.10, but we refrain from doing so exactly.

Let us illustrate this point with Resolution. To *define* Resolution as a function from the set of Resolution refutations into UNSAT is of course possible. Instead, we prefer to define formally the derivation, and nothing more; the derivation is the central object of study. We show that Resolution refutations indeed give rise to a proof system for the language UNSAT, consistent with the abstract definition, by proving three things:

- *Soundness*: there exists no Resolution refutation of a satisfiable formula;

- *Completeness*: every unsatisfiable formula has a Resolution refutation;

- *Polynomial-time checkability*: Given a derivation π from a CNF F, it can be decided algorithmically in time polynomial in $|F|+|\pi|$ whether π is a Resolution refutation of F.

Of course, these are the same three features that we extracted from the abstract definition, translated into the concrete nomenclature of Resolution. In fact, we have already done most of the work towards proving that all three features are present.

Theorem 2.11. *Resolution is a proof system for* UNSAT.

Proof. Soundness and completeness were established with Facts 2.8 and 2.9. To establish polynomial-time checkability, let $\pi := C_1, \ldots, C_k$ be a Resolution derivation from a CNF F.

Observe that given three clauses C_a, C_b, C_i, one can determine in time

$$O((|C_a| + |C_b|) \cdot |C_i|) = O(|\pi|^2)$$

whether C_i is a resolvent of C_a and C_b, simply by checking whether

$$C_a \setminus C_i = \{p\} \quad \text{and} \quad C_b \setminus C_i = \{\tilde{p}\}$$

for some literal p. For a fixed clause C_i, there are not more than k^2 distinct pairs of earlier clauses, therefore it can be decided in time $O(|\pi|^4)$ whether or not C_i can be derived by resolution.

Whether a clause C_i can be introduced by hypothesis can clearly be determined in time $O(|F|^2)$. Weakening steps can be removed in linear time. Hence, by checking each clause in turn, we can verify that the sequence π is indeed a Resolution refutation in time $O(|F| + |\pi|^4)$. □

Remark 2.12. From a technical standpoint, one might take exception to the fact that an abstract proof system is a total function from Σ^*, whereas Resolution refutations are sequences of clauses. This exception, however, is easily dealt with. One could simply fix an encoding for sequences of clauses on some alphabet; strings that do not encode a sequence of clauses would be considered refutations of a fixed formula. ■

Polynomial bounding

A proof system P is said to be *polynomially bounded* if the following holds for some polynomial p:

$$\text{for each } x \in \mathcal{L}, \quad \text{there exists a } \mathsf{P}\text{-proof } \pi \text{ of } x \text{ with } |\pi| \leq p(|x|).$$

The following theorem, due to [21], proves a fundamental connection between proof complexity and separation of complexity classes.

Theorem 2.13 ([21])**.** *There exists a polynomially bounded proof system for* UNSAT *if, and only if,* $\mathsf{NP} = \mathsf{coNP}$.

Example 2.14. Resolution is not polynomially bounded [67]. To prove this, it suffices to identify a formula family $\{F_n\}_{n\in\mathbb{N}}$ that does not have Resolution refutations of size $O(|F_n|^c)$ for any constant c. In fact, there are several well-known examples of such formula families [29, 69, 19]. ■

Simulation

The collection of all proof systems for a fixed language forms a hierarchy in terms of a natural relation called p-simulation. The relation is very similar to that of reduction between languages in computational complexity. Moreover, it admits an analogous notion of *degree* whereby all proof systems of comparable strength form a single equivalence class. Here, 'comparable' is qualified as 'up to a polynomial proof-size increase'.

Formally, given two proof systems $\mathsf{P}_1, \mathsf{P}_2$ for the same language over alphabets Σ_1, Σ_2, by 'P_1 *p-simulates* P_2' (written $\mathsf{P}_2 \leq_p \mathsf{P}_1$) we mean that there exists a polynomial-time computable function $f : \Sigma_2^* \to \Sigma_1^*$ satisfying $\mathsf{P}_1(h(\pi)) = \mathsf{P}_2(\pi)$ for each $\pi \in \Sigma_2^*$.

If P_1 and P_2 p-simulate one another, then they are said to be *p-equivalent*, written $\mathsf{P}_1 \equiv_p \mathsf{P}_2$. If P_1 p-simulates P_2 but P_2 does not p-simulate P_1, then we say that P_1 is *strictly p-simulates* P_2, written $\mathsf{P}_2 <_p \mathsf{P}_1$. Finally, if neither one of P_1 and P_2 simulates the other, then they are said to be *incomparable*.

Example 2.15. General Resolution trivially p-simulates Tree-like Resolution. On the other hand, Tree-like Resolution does not simulate General Resolution. As we mentioned in Section 2.3, there exist formulas that have linear-size Resolution refutations, but do not have polynomial-size tree-like Resolution refutations.

Therefore we can say that Tree-like Resolution and General Resolution are not p-equivalent, and that the latter is strictly stronger than the former. ■

Chapter 3

Quantified Boolean Formulas

In the final chapter of Part I, we cover the essential background for quantified Boolean formulas, a formalism that generalises propositional logic with existential and universal quantification. We cover syntax in Section 3.1, followed by semantics in Section 3.2. In Section 3.3, we introduce three prominent 'hand-crafted' families of quantified Boolean formulas. Finally in Section 1.2, we take a brief look at the wider landscape of QBF proof complexity, before homing in on the four systems with which we deal in detail.

3.1 Syntax

The syntax of quantified Boolean formulas is an extension of conjunctive normal form. The only additions to the object language are the quantification symbols '$\exists$' and '$\forall$'.

General form

We deal exclusively with quantified Boolean formulas in so-called *prenex conjunctive normal form.* In short, this means that all of the variables appearing in a CNF are first quantified either existentially ($\exists$) or universally ($\forall$).

Definition 3.1 (QBF)**.** *A* quantified Boolean formula (QBF) *is of the form*

$$\forall U_1 \exists X_1 \cdots \forall U_d \exists X_d \cdot F \,,$$

where $U_1, X_1, \dots, U_d, X_d$ are pairwise disjoint sets of Boolean variables, and F is a CNF for which

$$\mathrm{vars}(F) \subseteq \bigcup_{i \in [d]} (U_i \cup X_i) \,.$$

The set of all QBFs is denoted '$\mathbb{Q}$'.

We typically write a QBF as $Q := P \cdot F$, and we call

$$P := \forall U_1 \exists X_1 \cdots \forall U_d \exists X_d$$

the *quantifier prefix* of Q, and F the *matrix* of Q. We refer to the variable sets U_i and X_i as the universal and existential *blocks* of Q respectively. A block may be an empty set. For example, putting U_1 as the empty set mimics the situation in which the first block is existential. The *quantifier depth* of Q is the number of universal blocks.

For ease of reading, we may write the blocks of a prefix as individually quantified variables, rather than as sets. For example, the prefix $\forall U_1 \exists X_1$, with $U_1 := \{u_1, u_2\}$ and $X_1 := \{x_1, x_2\}$, may be written $\forall u_1 \forall u_2 \exists x_1 \exists x_2$. This is merely a typographic convenience; there is no implied order of quantification between the variables in a single block.

In the case where $d = 0$, the matrix contains no variables, and the QBF reduces to either the empty CNF $\emptyset$, or the CNF $\{\emptyset\}$ containing only the empty clause.

When we refer to the universal or existential variables of Q, we mean the elements of the sets

$$\mathrm{vars}_\forall(Q) := \bigcup_{i \in [d]} U_i \quad \text{and} \quad \mathrm{vars}_\exists(Q) := \bigcup_{i \in [d]} X_i \,,$$

or the sets themselves. Unless we specify otherwise, we assume that there are m universal variables u_i indexed from 1 to m, as in

$$U = \{u_1, \dots, u_m\} \,,$$

and n existential variables x_i indexed from 1 to n,

$$X = \{x_1, \dots, x_n\} \,.$$

By a '*total existential assignment*' to Q, we mean a total assignment to $\mathrm{vars}_\exists(Q)$, and by a '*total universal assignment*' we mean a total assignment to $\mathrm{vars}_\forall(Q)$

Applying assignments

The application of an assignment σ to a QBF Q is

$$Q[\sigma] := P[\sigma] \cdot F[\sigma] \,,$$

where the prefix $P[\sigma]$ is obtained from P by deleting the variables in $\mathrm{vars}(\sigma)$, then removing any empty blocks and their associated quantifiers.

3.2 Semantics

A description of QBF semantics must distinguish true formulas from false ones. Some authors choose to define semantics inductively on the syntactic structure (e.g. [32]). We prefer the alternative, equivalent definition based on models and countermodels (e.g. [50]).

Models

For a general QBF Q, the *dependency set* for an existential variable is the set of universal variables that are quantified earlier in the prefix. For example, given an existential x_i that belongs to block X_j, the dependency set for x_i in Q is

$$S_i := \{u \in \mathrm{vars}_\forall(Q) : u \text{ is in } U_k \text{ and } k < j\}\,.$$

For our general QBF, the naming of the existential dependency sets $S_1, \ldots, S_n$ is bound to the indexing of the existential variables $x_1, \ldots, x_n$.

A set of existential dependency functions for Q is a set of mappings $f := \{f_i\}_{i\in[n]}$, where each individual function has the signature

$$f_i \quad : \quad \langle S_i \rangle \quad \to \quad \langle \{x_i\} \rangle\,.$$

When a set of existential dependency functions f satisfies the following property, we call it a *model* for Q.

Definition 3.2 (model)**.** *We call a set of existential dependency functions $\{f_i\}_{i\in[n]}$ a* model *for a QBF Q when, for each μ in $\langle \mathrm{vars}_\forall(Q)\rangle$, the assignment*

$$\mu \cup \{f_i(\mu\restriction_{S_i})\}_{i\in[n]}$$

satisfies the matrix.

Example 3.3. The QBF

$$\forall u_1 \exists x_1 \cdot \{\{\bar{x}_1, \bar{u}_1\}, \{x_1, u_1\}\}$$

has the model $\{f_1\}$, whose dependency function is

$$\begin{array}{cccccc} f_1 & : & \langle\{u_1\}\rangle & \to & \langle\{x_1\}\rangle \\ & & \{\bar{u}_1\} & \mapsto & \{x_1\} \\ & & \{u_1\} & \mapsto & \{\bar{x}_1\}\,. \end{array}$$

We can verify that this is indeed a model, by checking that $\mu \cup f_1(\mu)$ satisfies the matrix for each assignment μ to u_1; that is, by noting that both assignments $\{\bar{u}_1, x_1\}$ and $\{u_1, \bar{x}_1\}$ are satisfying. ■

It is often useful to treat the existential assignment $\{f_i(\mu\restriction_{S_i})\}_{i\in[n]}$ as a whole. For that reason, we commit an abuse of notation: we also treat the symbol f as the function

$$\begin{array}{rcrcl} f & : & \langle \mathrm{vars}_\forall(Q)\rangle & \to & \langle \mathrm{vars}_\exists(Q)\rangle \\ & & \mu & \mapsto & \{f_i(\mu\restriction_{S_i})\}_{i\in[n]}\,, \end{array}$$

whereby $f(\mu)$ becomes an alias for $\{f_i(\mu\restriction_{S_i})\}_{i\in[n]}$. We note two things:

(a) $f(\mu)$ and $f(\nu)$ agree on x_i whenever μ and ν agree on S_i;

(b) f is a model if, and only if, $\mu \cup f(\mu)$ always satisfies the matrix.

Truth, falsity and semantic entailment

Models are the basic objects that witness the truth value of a QBF. We call a QBF *true* when it has a model, and *false* when it does not. We use the phrases '*f is a model for Q*' and '*f models Q*' synonymously.

Models also form the basis of semantic entailment for QBFs, the analogue of semantic entailment for CNFs (Subsection 2.2). We define the entailment relation on $\mathbb{Q}\times\mathbb{Q}$ as follows.

$$Q \vDash R \quad \Leftrightarrow \quad \text{every model for } Q \text{ also models } R.$$

When $F \vDash G$ holds, we say that F *entails* G. We only consider entailment between QBFs which have the same prefix.

Countermodels

A QBF is true when, and only when, it has a model, and hence it is false when no model exists. However, we can also witness falsity by the existence of a *countermodel*, the natural dual.

We consider again our general QBF Q. The dependency set for a universal variable u_i from block U_j is

$$H_i := \{x \in \mathrm{vars}_\exists(Q) : x \text{ is in } X_k \text{ and } k < j\}\,.$$

The naming of the universal dependency sets $H_1,\dots,H_m$ is bound to the indexing of the universal variables $u_1,\dots,u_m$.

A set of universal dependency functions for a QBF is a set of mappings $h := \{h_i\}_{i\in[m]}$, where each individual function has the signature

$$h_i \quad : \quad \langle H_i\rangle \quad \to \quad \langle\{u_i\}\rangle\,.$$

Definition 3.4 (countermodel)**.** *We call a set of universal dependency functions* $\{h_i\}_{i\in[m]}$ *a* countermodel *for a QBF* Q *when, for each* ε *in* $\langle \mathrm{vars}_\exists(Q)\rangle$*, the assignment*

$$\varepsilon \cup \{h_i(\varepsilon\restriction_{H_i})\}_{i\in[m]}$$

falsifies the matrix of Q*.*

Example 3.5. The QBF

$$\exists x_1 \forall u_1 \cdot \{\{\bar{u}_1, \bar{x}_1\}, \{u_1, x_1\}\}\,.$$

has the countermodel $\{h_1\}$, where the dependency function h_1 is defined by

$$\begin{array}{rcccc} h_1 & : & \langle\{x_1\}\rangle & \to & \langle\{u_1\}\rangle \\ & & \{\bar{x}_1\} & \mapsto & \{\bar{u}_1\} \\ & & \{x_1\} & \mapsto & \{u_1\}\,. \end{array}$$

To verify that this is indeed a countermodel, we need to check that $\varepsilon \cup h_1(\varepsilon)$ falsifies the matrix for each assignment ε to x_1. This is clear; $\{\bar{x}_1, \bar{u}_1\}$ falsifies the second clause, and $\{x_1, u_1\}$ falsifies the first. ■

We also commit a dual abuse of notation: we treat the symbol h as the function

$$\begin{array}{rcccc} h & : & \langle \mathrm{vars}_\exists(Q)\rangle & \to & \langle \mathrm{vars}_\forall(Q)\rangle \\ & & \mu & \mapsto & \{h_i(\varepsilon\restriction_{H_i})\}_{i\in[m]}\,, \end{array}$$

whereby $h(\varepsilon)$ becomes an alias for $\{h_i(\varepsilon\restriction_{H_i})\}_{i\in[m]}$. We emphasize

(a) $h(\varepsilon)$ and $h(\delta)$ agree on u_i whenever ε and δ agree on H_i;

(b) h is a countermodel if, and only if, $\varepsilon \cup h(\varepsilon)$ always falsifies the matrix.

Applying assignments to dependency functions

In certain situations, we can preserve the models or countermodels of a QBF under application of assignments. For this, we need to know how to apply assignments to models and countermodels, or more generally, to sets of dependency functions.

Let σ be a partial assignment to a QBF Q, with universal part $\sigma_\forall$ and existential part $\sigma_\exists$. To apply σ to a set of existential dependency functions f, we discard the dependency functions for the variables in $\mathrm{vars}(\sigma_\exists)$, and apply $\sigma_\forall$ to those remaining. Formally,

$$f[\sigma] := \{f_i[\sigma] : x_i \in X \setminus \mathrm{vars}(\sigma_\exists)\}\,,$$

where

$$\begin{array}{rcrcl} f_i[\sigma] & : & \langle S_i \setminus \mathrm{vars}(\sigma_\forall)\rangle & \to & \langle\{u_i\}\rangle \\ & & \mu & \mapsto & f_i(\mu \cup (\sigma_\forall\restriction_{S_i}))\,. \end{array}$$

Likewise, the application of σ to a set of universal dependency functions h is defined as

$$h[\sigma] := \{h_i[\sigma] : u_i \in U \setminus \mathrm{vars}(\sigma_\forall)\}\,,$$

where

$$\begin{array}{rcrcl} h_i[\sigma] & : & \langle H_i \setminus \mathrm{vars}(\sigma_\exists)\rangle & \to & \langle\{x_i\}\rangle \\ & & \varepsilon & \mapsto & f_i(\varepsilon \cup (\sigma_\exists\restriction_{H_i}))\,. \end{array}$$

The following lemma tells us some of the cases where models and countermodels are preserved by assignment.

Lemma 3.6. *Let Q be a QBF, let f and h be sets of existential and universal dependency functions, and let σ be partial assignment to Q.*

(a) *If f_i is identically $\sigma\restriction_{\{x_i\}}$ for each existential x_i in* $\mathrm{vars}(\sigma_\exists)$*, then*

$$f \textit{ models } Q \quad\Rightarrow\quad f[\sigma] \textit{ models } Q[\sigma]\,.$$

(b) *If h_i is identically $\sigma\restriction_{\{u_i\}}$ for each universal u_i in* $\mathrm{vars}(\sigma_\forall)$*, then*

$$h \textit{ countermodels } Q \quad\Rightarrow\quad h[\sigma] \textit{ countermodels } Q[\sigma]\,.$$

Proof. Let $\sigma_\forall$ and $\sigma_\exists$ be the universal and existential subassignments of σ, and let F be the matrix of Q.

(a) Let μ be a total universal assignment to $Q[\sigma]$, and suppose that f models Q. Then,

$$\sigma_\forall \cup \mu \cup f(\sigma_\forall \cup \mu)$$

satisfies F. Since f_i is identically $\sigma\restriction_{\{x_i\}}$ for each x_i in $\mathrm{vars}(\sigma_\exists)$, we have

$$\begin{aligned} f(\sigma_\forall \cup \mu) &= \sigma_\exists \cup \{f_i((\sigma_\forall \cup \mu)\restriction_{S_i}) : x_i \notin \mathrm{vars}(\sigma_\exists)\} \\ &= \sigma_\exists \cup \{f_i[\sigma](\mu\restriction_{S_i}) : x_i \notin \mathrm{vars}(\sigma_\exists)\}\,, \end{aligned}$$

and it follows that $\mu \cup f[\sigma](\mu)$ satisfies $F[\sigma]$. Thus $f[\sigma]$ models $Q[\sigma]$.

(b) The proof is dual to that of (a). Let ε be a total existential assignment to $Q[\sigma]$, and suppose that h countermodels Q. Then,

$$\sigma_\exists \cup \varepsilon \cup h(\sigma_\exists \cup \varepsilon)$$

falsifies F. Since h_i is identically $\sigma\restriction_{\{u_i\}}$ for each u_i in $\mathrm{vars}(\sigma_\forall)$, we have

$$\begin{aligned}
h(\sigma_\exists \cup \varepsilon) &= \sigma_\forall \cup \{h_i((\sigma_\exists \cup \varepsilon)\restriction_{H_i}) : u_i \notin \mathrm{vars}(\sigma_\forall)\} \\
&= \sigma_\forall \cup \{h_i[\sigma](\varepsilon\restriction_{H_i}) : u_i \notin \mathrm{vars}(\sigma_\forall)\}\,,
\end{aligned}$$

and it follows that $\varepsilon \cup h[\sigma](\varepsilon)$ falsifies $F[\sigma]$. Thus $h[\sigma]$ countermodels $Q[\sigma]$. □

Lemma 3.7. *Let Q be a QBF whose first block is Z, let f be a set of existential dependency functions, and let h be a set of universal dependency functions.*

(a) *If Z is universal, then*

$$\begin{array}{lcll}
f \text{ models } Q & \Leftrightarrow & \text{for all } \mu \text{ in } \langle Z \rangle, & f[\mu] \text{ models } Q[\mu]\,, \\
h \text{ countermodels } Q & \Leftrightarrow & \text{for some } \mu \text{ in } \langle Z \rangle, & h[\mu] \text{ countermodels } Q[\mu]\,.
\end{array}$$

(b) *If Z is existential, then*

$$\begin{array}{lcll}
f \text{ models } Q & \Leftrightarrow & \text{for some } \varepsilon \text{ in } \langle Z \rangle, & f[\varepsilon] \text{ models } Q[\varepsilon]\,, \\
h \text{ countermodels } Q & \Leftrightarrow & \text{for all } \varepsilon \text{ in } \langle Z \rangle, & h[\varepsilon] \text{ countermodels } Q[\varepsilon]\,.
\end{array}$$

Proof. The forward directions for all statements follow from Lemma 3.6. The reverse directions follow directly from the definitions of model and countermodel (Definitions 3.2 and 3.4). □

The folklore theorem

It is convenient that we can witness the falsity of a QBF by showing that a countermodel exists, since it usually easier to construct a countermodel than to prove the nonexistence of a model. Moreover, it fosters a pleasant duality between models and countermodels as witnesses of truth and falsity.

That this duality holds is easy to intuit, could probably be taken for granted, and is something of a folklore result [45]. However, the result becomes rather important for us in Part III (in the broader context of DQBF), and for that reason, we formally state and prove it.

Theorem 3.8 (Folklore Theorem)**.** *A QBF is false if, and only if, it has a countermodel.*

Proof. Let Q be a QBF. We prove the fact by induction on the quantifier depth of Q.

First, the base case $d = 0$. The matrix of a QBF with no blocks is either the empty CNF, or the CNF containing only the empty clause. In the former case, the matrix is a tautology, Q is modelled by the empty set of functions and no countermodel exists. In the latter case, the matrix is unsatisfiable, Q is countermodeled by the empty set of functions and no model exists.

Now, for the inductive step, let $d \geq 1$, and let Q be of the form

$$\forall U_1 \exists X_1 \cdots \forall U_d \exists X_d \cdot F\,.$$

For the "only if" direction, suppose that Q has a countermodel h. By Lemma 3.7, there exists some μ_0 in $\langle U_1 \rangle$ such that, for all ε in $\langle X_1 \rangle$, $h[\mu_0 \cup \varepsilon]$ countermodels $Q[\mu_0 \cup \varepsilon]$. Hence $Q[\mu_0 \cup \varepsilon]$, which has quantifier depth $d-1$, does not have a model, by the inductive hypothesis. Now, if we suppose that Q has a model f, we reach a contradiction, since by Lemma 3.7, there exists some ε_0 in $\langle X_1 \rangle$ such that $f[\mu_0 \cup \varepsilon_0]$ models $Q[\mu_0 \cup \varepsilon_0]$.

For the "if" direction, suppose that Q is false, and let f be an arbitrary set of existential dependency functions for Q. By Lemma 3.7, there exists some μ_0 in $\langle U_1 \rangle$ such that, for all ε in $\langle X_1 \rangle$, $f[\mu_0 \cup \varepsilon]$ does not model $Q[\mu_0 \cup \varepsilon]$. Moreover, every set of existential dependency sets for $Q[\mu_0 \cup \varepsilon]$ can be obtained from some set for Q by applying the assignment $\mu_0 \cup \varepsilon$. Hence $Q[\mu_0 \cup \varepsilon]$ does not have a model, so it must have a countermodel, by the inductive hypothesis.

Aiming for contradiction, suppose that Q does not have a countermodel, and let h be an arbitrary set of universal dependency functions for Q. By Lemma 3.7, there exists some ε in $\langle X_1 \rangle$ such that $h[\mu_0 \cup \varepsilon_0]$ does not countermodel $Q[\mu_0 \cup \varepsilon_0]$. Every set of universal dependency sets for $Q[\mu_0 \cup \varepsilon_0]$ can be obtained from some set for Q by applying the assignment $\mu_0 \cup \varepsilon_0$. We reach a contradiction, since this implies that $Q[\mu_0 \cup \varepsilon_0]$ does not have a countermodel. $\square$

The evaluation game

Models and countermodels for QBFs have an equivalent description, given in terms of winning strategies in a two-player *evaluation game*.

The game is contested between two adversaries conventionally named (with some imagination) the existential ($\exists$) and universal ($\forall$) players. At the outset, the board consists of a QBF Q. The players take turns to choose assignments to the blocks of Q, in the order of the prefix, beginning with the leftmost block. Naturally, $\exists$ is responsible for the assignment of existential blocks, and $\forall$ for the universal ones.

The game ends when the assignment to the final block is made. At this point, the players have constructed a total assignment, which either satisfies or falsifies the matrix. In the former case the existential player wins, in the latter, the universal does.

Now, given a model f, the existential player can win the evaluation game by force. She simply plugs the assignments chosen by her opponent into the individual functions f_i to obtain the correct moves. By definition of model, this strategy always yields an assignment that satisfies the matrix, however her opponent plays.

So a model encodes a winning strategy for the existential player in the evaluation game. It is also clear intuitively that a winning strategy for the existential player *defines* a model. Similarly, countermodels are equivalent to winning strategies for the universal player.

Thus, we can use winning strategies to witness the truth values: A QBF is true if, and only if, it has a winning existential strategy, and is false if, and only if, it has a winning universal strategy.

Complexity

Under a suitable encoding as binary strings, the set of true QBFs forms the canonical PSPACE-complete language TQBF [65]. Since $\mathsf{PSPACE} = \mathsf{coPSPACE}$, the set of false QBFs also forms a canonical PSPACE-complete language FQBF.

3.3 Formula Familes

A *QBF family* is an object that associates each natural number with a QBF. One can view this as a countable sequence of QBFs, but formally we choose to define it as a function from $\mathbb{N}$ into $\mathbb{Q}$. The image under a natural number n is referred to as the n^{th} *instance* of the family.

We say that a QBF family is *d-bounded* when every instance has quantifier depth at most d, and has *unbounded quantifier depth* when it is not d-bounded for any d.

Follwing the standard method of proof complexity, we employ QBF families to demonstrate that QBF proof systems are not polynomially bounded; that is, we show that the size of the shortest refutation of the n^{th} instance grows superpolynomially in n. Such a result, which we refer to as a *proof-size lower bound*, is always required whenever one wishes to show constructively that a p-simulation between two systems does not exist. Analogous to the separation of complexity classes (where one must show the absence of reductions), proof-size lower bounds generally comprise the most difficult part of any such argument.

Many of our proof complexity results will be demonstrated by one of three 'hand-crafted' QBF families.

The equality family

The first of our three families is the simplest, and arguably the simplest example of a QBF family which requires large proofs in a non-trivial QBF proof system.

Definition 3.9 (equality family)**.** *The* equality family *is the QBF family $\mathcal{EQ}$ whose n^{th} instance is*

$$\mathrm{EQ}_n := \exists x_1 \cdots x_n \forall u_1 \cdots u_n \exists z_1 \cdots z_n \cdot \mathrm{eq}_n \,,$$

where the CNF eq_n *consists of the clauses*

$$\begin{array}{ll} \{\bar{x}_i, \bar{u}_i, z_i\}\,, & \textit{for } i \textit{ in } [n]\,, \\ \{x_i, u_i, z_i\}\,, & \textit{for } i \textit{ in } [n]\,, \\ \{\bar{z}_1, \ldots, \bar{z}_n\}\,. & \end{array}$$

Note that every instance of $\mathcal{EQ}$ has a single universal block, and quantifier depth 1, so $\mathcal{EQ}$ is 1-bounded.

It is easy to see that the n^{th} equality formula is false, because the universal player has the following winning strategy: set each universal variable u_i to the same value as the existential player sets x_i. Employing this strategy leaves the existential player needing to satisfying all n unit clauses $\{z_i\}$, whereupon the clause containing the full set of negative z_i literals must be falsified.

It is also not so hard to see that the winning strategy for the universal player is unique – any deviation from it allows the existential player to win. Hence, given the equivalence between strategies and countermodels, EQ_n has a unique countermodel.

We will also have cause to use the equality family with a modified prefix.

Definition 3.10 (interleaved equality family)**.** *The* interleaved equality family *is the QBF family $\mathcal{EQ}'$ whose n^{th} instance is*

$$\mathrm{EQ}'_n := \exists x_1 \forall u_1 \exists z_1 \cdots \exists x_n \forall u_n \exists z_n \cdot \mathrm{eq}_n \,.$$

Note that the quantifier depth of the n^{th} instance of the interleaved equality family is n, so $\mathcal{EQ}'$ is not d-bounded for any d; that is, $\mathcal{EQ}'$ has unbounded quantifier depth. The interleaved equality formulas remain false, but the countermodel is no longer unique.

The parity family

Our second QBF family is also 1-bounded with a unique countermodel per instance.

Definition 3.11 (parity family [14])**.** *The* parity family *is the QBF family $\mathcal{PA}$ whose n^{th} instance is*

$$\mathrm{PA}_n := \exists x_1 \cdots x_n \forall u \exists z_1 \cdots z_n \cdot \mathrm{pa}_n \,,$$

where the CNF pa_n *consists of the clauses*

$$\begin{array}{ll}
\{x_1, \bar{z}_1\}\,, & \\
\{\bar{x}_1, z_1\}\,, & \\
\{x_{i+1}, z_i, z_{i+1}\}\,, & \textit{for } i \textit{ in } [n-1]\,, \\
\{\bar{x}_{i+1}, \bar{z}_i, z_{i+1}\}\,, & \textit{for } i \textit{ in } [n-1]\,, \\
\{x_{i+1}, \bar{z}_i, \bar{z}_{i+1}\}\,, & \textit{for } i \textit{ in } [n-1]\,, \\
\{\bar{x}_{i+1}, z_i, \bar{z}_{i+1}\}\,, & \textit{for } i \textit{ in } [n-1]\,, \\
\{u, \bar{z}_n\}\,, & \\
\{\bar{u}, z_n\}\,. &
\end{array}$$

In a nutshell, the parity family encodes the notion that both values of the universal variable u are equal to the parity of the x_i, which is of course false. Accordingly, the unique winning strategy for the universal player is to set u not equal to the parity of the x_i.

The Kleine Büning et al. family

Our final QBF family is arguably the most famous, being the first QBF family for which a non-trivial proof-size lower bound was shown.

Definition 3.12 (Kleine Büning et al. family [35])**.** *The* Kleine Büning et al. family *is the QBF family $\mathcal{KB}$ whose n^{th} instance is*

$$\mathrm{KB}_n := \exists x_1 y_1 \forall u_1 \cdots \exists x_n y_n \forall u_n \exists z_1 \cdots z_n \cdot \mathrm{kb}_n \,,$$

where kb_n *is the CNF consisting of the clauses*

$$\begin{array}{ll}
\{\bar{x}_1, \bar{y}_1\}\,, & \\
\{x_i, \bar{u}_i, \bar{x}_{i+1}, \bar{y}_{i+1}\}\,, & \textit{for } i \textit{ in } [n-1]\,, \\
\{y_i, u_i, \bar{x}_{i+1}, \bar{y}_{i+1}\}\,, & \textit{for } i \textit{ in } [n-1]\,, \\
\{x_n, \bar{u}_n, \bar{z}_1, \ldots, \bar{z}_n\}\,, & \\
\{y_n, u_n, \bar{z}_1, \ldots, \bar{z}_n\}\,, & \\
\{u_i, z_i\}\,, & \textit{for } i \textit{ in } [n]\,, \\
\{\bar{u}_i, z_i\}\,, & \textit{for } i \textit{ in } [n]\,.
\end{array}$$

Note that $\mathcal{KB}$ has unbounded quantifier depth.

Each instance of $\mathcal{KB}$ is a false QBF. One can verify that the universal dependency functions $\{h_i\}_{i\in[n]}$ defined by

$$\begin{aligned} h_i \quad : \quad \{\langle H_i \rangle\} \quad &\to \quad \langle \{u_i\} \rangle \\ \varepsilon \quad &\mapsto \quad \begin{cases} u_i & \text{if } \varepsilon(x_i) = 0 \\ \bar{u}_i & \text{otherwise} \end{cases} \end{aligned}$$

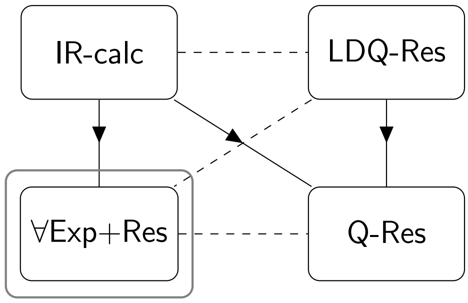
form a countermodel for KB_n.

Part II

Lower-bound Techniques

Chapter 4

Universal Expansion



Universal expansion is one of two major paradigms in QBF solving, the other being universal reduction. In expansion, the aim is to implement a translation from QBF back to propositional logic, in which universal variables are 'expanded out'. The result is a fully existentially quantified QBF, which, as we have seen, is merely a propositional CNF.

Expansion-based solvers such as RAReQs [33] work by implementing this reduction to propositional logic, and passing the result to a SAT solver. In practice, the expansion is carried out piecemeal and multiple SAT calls are made.

A characterisation of expansion lower bounds

In this chapter, we recall ∀Exp+Res [34], the basic theoretical model underpinning expansion-based solving, and we propose a semantic technique for proving proof-size lower bounds.

The technique emerges from a corresponding *semantic* translation from QBF into propositional logic. Studying the semantics of the translation allows us to determine the size increase of the formula due to the expansion. Moreover, the size increase can be phrased in terms of a measure defined on the countermodels of the QBF, that we call *countermodel size*.

The main theorem of this chapter (Theorem 4.13) tells us that lower bounds on countermodel size translate directly into $\forall$Exp+Res proof-size lower bounds. It also tells us that any $\forall$Exp+Res lower bound either comes from a countermodel-size lower bound or a lower bound from propositional Resolution. Thus we completely characterise hardness in $\forall$Exp+Res as either semantic (large countermodels) or propositional (Resolution hardness).

Organisation of the chapter

In Section 4.1, we explain the universal expansion paradigm. In Section 4.2, we describe the proof system $\forall$Exp+Res, followed by a lower-bound technique and a characterisation of hardness in Section 4.3.

4.1 The universal expansion paradigm

Given our goal of proving semantically-grounded lower bounds, we will present universal expansion with an emphasis on semantics. The reader will find the presentation significantly different from the original [34] and subsequent presentations [14]. However, it should be emphasised that this is merely a change of presentation. The proposal of the solving paradigm and the associated proof system originates from [34].

Universal expansion, in the simplest sense, is a method for removing universally quantified variables entirely from QBFs. The process produces a fully existentially quantified QBF, which is usually written simply as a CNF, without the existential quantification. The important point is that the expanded CNF is satisfiable if, and only if, the original QBF is true.

To illustrate, consider a QBF

$$\exists x_0 \forall u_1 \exists x_1 \forall u_2 \exists x_2 \cdot \phi(x_0, u_1, x_1, u_2, x_2)\,.$$

The universal expansion of this QBF is the CNF

$$\phi(x_0, 0, x_1^0, 0, x_2^{00}) \cup \phi(x_0, 0, x_1^0, 1, x_2^{01}) \cup \phi(x_0, 1, x_1^1, 0, x_2^{10}) \cup \phi(x_0, 1, x_1^1, 1, x_2^{11})\,,$$

consisting of four substitution instances of the QBF matrix ϕ. In each one, the universal variables u_1 and u_2 are substituted by constants 0 and 1 (i.e. they are assigned) so that each substitution instance corresponds to some total assignment to the universals.

The existential variables are not assigned, instead they are replaced by new variables. These new variables are named in such a way that the dependencies of the original existentials are recorded in the superscript. For example, the new variables x_1^0 and x_1^1 record the dependency of the existential x_1 on the universal u_1, which is quantified earlier. Notice that x_1 is replaced by x_1^0 in the two subformulas $\phi(x_0, 0, x_1^0, 0, x_2^{00})$ and $\phi(x_0, 0, x_1^0, 1, x_2^{01})$ in which u_1 takes the value 0. In the other two subformulas, in which u_1 is assigned 1, x_1 is replaced by x_1^1.

Similarly, the new variables $x_2^{00}, x_2^{01}, x_2^{10}, x_2^{11}$ record the dependency of x_2 on *both* u_1 and u_2. Each one replaces x_2 in the subformulas corresponding to the appropriate assignment to u_1 and u_2.

With more clarity, and less comfort, one can write the appropriate universal assignments explicitly into the superscripts:

$$\phi(x_0^\emptyset, 0, x_1^{\{\bar{u}_1\}}, 0, x_2^{\{\bar{u}_1,\bar{u}_2\}}) \cup \phi(x_0^\emptyset, 0, x_1^{\{\bar{u}_1\}}, 1, x_2^{\{\bar{u}_1,u_2\}}) \cup$$
$$\phi(x_0^\emptyset, 1, x_1^{\{u_1\}}, 0, x_2^{\{u_1,\bar{u}_2\}}) \cup \phi(x_0^\emptyset, 1, x_1^{\{u_1\}}, 1, x_2^{\{u_1,u_2\}})\,.$$

Superscripts formatted in this way are called *annotations*, and variables that hold them are called *annotated variables*. The annotation is always a total assignment to the dependency set of the original existential variable. An existential in the leftmost block, like x_0, always receives the empty annotation, since its dependency set is empty.

The annotation in the superscript of an annotated literal can get quite large. For this reason, we often write annotated literals x^σ and $\bar{x}^\sigma$ as pairs (x, σ) and $(\bar{x}, \sigma)$.

The explicit use of assignments as annotations yields a neat and tidy form of the universal expansion of a QBF, namely, the union over all universal assignments μ of the application of the substitution

$$\mu \cup \{x_i \mapsto (x_i, \mu\restriction_{S_i}) : i \in [n]\}$$

to the matrix.

Example 4.1. The first instance PA_1 of the parity family is the QBF whose prefix is $\exists x_1 \forall u \exists z_1$ and whose matrix is the CNF

$$\mathrm{pa}_1 := \{\{\bar{x}_1, z_1\}, \{x_1, \bar{z}_1\}, \{\bar{u}, z_1\}, \{u, \bar{z}_1\}\}\,.$$

The expansion of PA_1 is the union of the two CNFs

$$\begin{aligned}
\mathrm{pa}_1[\{u \mapsto 0, x_1 \mapsto x_1^\emptyset, z_1 \mapsto z_1^{\{\bar{u}\}}\}] &= \{\{\bar{x}_1^\emptyset, z_1^{\{\bar{u}\}}\}, \{x_1^\emptyset, \bar{z}_1^{\{\bar{u}\}}\}, \{\bar{z}_1^{\{\bar{u}\}}\}\}\,. \\
\mathrm{pa}_1[\{u \mapsto 1, x_1 \mapsto x_1^\emptyset, z_1 \mapsto z_1^{\{u\}}\}] &= \{\{\bar{x}_1^\emptyset, z_1^{\{u\}}\}, \{x_1^\emptyset, \bar{z}_1^{\{u\}}\}, \{z_1^{\{u\}}\}\}\,.
\end{aligned}$$

PA_1 is false, and it can be verified that its expansion is unsatisfiable, as every assignment to the variables $\{x_1^\emptyset, z_1^{\{\bar{u}\}}, z_1^{\{u\}}\}$ falsifies some clause in the expansion. ■

Partial expansions

In practice, we don't always consider the whole expansion of a QBF. Merely writing out the full expansion can take exponential time, as the number of substitution instances is exponential in the number of universal variables.

However, there are many cases in which the expansion is unsatisfiable, but not minimally unsatisfiable. In these cases, there exists a smaller set of substitution instances whose union is unsatisfiable.

To avoid an inherent, unnecessary size increase, we define the partial expansion of a QBF with respect to an arbitrary set of total universal assignments.

Definition 4.2 (partial expansion)**.** *Let $Q := P \cdot F$ be a QBF with universal variables U and existential dependency sets $S_1, \ldots, S_n$. The* partial expansion *of Q with respect to a set of universal assignments $\Gamma \subseteq \langle U \rangle$ is the CNF*

$$\exp(Q, \Gamma) := \bigcup_{\mu \in \Gamma} F[\mu \cup \{x_i \mapsto x_i^{\mu\restriction S_i} : i \in [n]\}]\,.$$

The total expansion *of Q is*

$$\exp(Q) := \exp(Q, \langle U \rangle)$$

4.1.1 Expansion semantics

We have mentioned that a QBF is true if, and only if, the total expansion is unsatisfiable. Actually, a much stronger statement can be made, and a much tighter semantic relationship exists between the semantics of QBF and propositional logic: There is a natural map between satisfying assignments for the expansion and models of the QBF.

In fact, there exists a one-one correspondence between assignments to the expansion of a QBF and the set consisting of all possible sets of dependency functions, that maps to a model if, and only if, the assignment satisfies the expansion. The correspondence is witnessed by an operation that we call *contraction*.

Definition 4.3 (contraction)**.** *Given a total assignment σ to the expansion of a QBF Q with dependency sets $S_1, \ldots, S_n$, the* contraction *of σ is the set of dependency functions $\{f_i\}_{i \in [n]}$ defined by*

$$\begin{array}{rcrcl} f_i & : & \langle S_i \rangle & \to & \langle x_i \rangle \\ & & \mu & \mapsto & \{x_i \mapsto \sigma(x_i^{\mu})\}\,. \end{array}$$

We first show that contraction maps satisfying assignments to models, and falsifying assignments to dependency functions which are not models.

Lemma 4.4. *A total assignment to the expansion of a QBF Q is satisfying if, and only if, its contraction models Q.*

Proof. Let σ be a total assignment to the expansion of $Q := P \cdot F$, and let $f := \{f_i\}_{i\in[n]}$ be its contraction.

For the "if" direction, suppose that $\{f_i\}_{i\in[n]}$ models Q. Aiming for contradiction, suppose that σ falsifies the expansion of Q. Then there exists a clause C in F and a universal assignment μ such that σ falsifies

$$C[\mu \cup \{x_i \mapsto x_i^{\mu\restriction_{S_i}} : i \in [n]\}] .$$

Since $f_i(\mu\restriction_{S_i})(x_i) = \sigma(x_i^{\mu\restriction_{S_i}})$, the assignment

$$\mu \cup \{f_i(\mu\restriction_{S_i}) : i \in [n]\}$$

falsifies C. But this is a contradiction, since it implies that f does not model Q.

Now for the "only if" direction. Suppose that σ satisfies the expansion of Q. Aiming for contradiction, suppose that f does not model Q. Then there exists a clause C in F and a universal assignment μ such that

$$\mu \cup \{f_i(\mu\restriction_{S_i}) : i \in [n]\}$$

falsifies C. Since $\sigma(x_i^{\mu\restriction_{S_i}}) = f_i(\mu\restriction_{S_i})(x_i)$, σ falsifies the clause

$$C[\mu \cup \{x_i \mapsto x_i^{\mu\restriction_{S_i}} : i \in [n]\}] ,$$

which belongs to the expansion of Q. But this is a contradiction, since it implies that σ does not satisfy the expansion. □

We claimed that contraction is a one-one correspondence, and to prove this we still need to show that every set of dependency functions is the contraction of some assignment to the expansion. This is easy to see: a set of dependency functions $\{f_i\}_{i\in[n]}$ is the contraction of the assignment

$$\bigcup_{\mu\in\langle U\rangle} \{x_i^{\mu\restriction_{S_i}} \mapsto f_i(\mu\restriction_{S_i}) : i \in [n]\} .$$

Fact 4.5. *Each set of dependency functions for a QBF is the contraction of some total assignment to its expansion.*

Lemma 4.4 and Fact 4.5 together imply the following.

Corollary 4.6 ([34])**.** *A QBF is true if, and only if, its total expansion is satisfiable.*

This corollary is the reason why universal expansion is a viable approach to QBF solving. We note that it was not stated or proved in [34] in the same fashion as we have proved it here, rather, it is a direct consequence of [34, Thm. 1, p. 29].

4.2 The proof system ∀Exp+Res

We jump straight in and define the system. Our presentation of the expansion paradigm in the previous section gives rise to a very straightforward definition.

Definition 4.7 (∀Exp+Res [34])**.** *A* ∀Exp+Res refutation *of a QBF Q is a Resolution refutation of the total expansion of Q.*

Remark. The original presentation of ∀Exp+Res was based on expansion trees [34]. Our definition follows more closely the subsequent presentation from [14], but differs in that the initial annotations are defined by the expansion, rather than in the axiom rule [14, Fig. 3, p. 79].

Example 4.8. Figure 4.1 shows a ∀Exp+Res refutation of PA_1, that is, a Resolution refutation of its expansion, namely the CNF

$$\{\{\bar{x}_1^{\emptyset}, z_1^{\{\bar{u}\}}\}, \{x_1^{\emptyset}, \bar{z}_1^{\{\bar{u}\}}\}, \{\bar{z}_1^{\{\bar{u}\}}\}, \{\bar{x}_1^{\emptyset}, z_1^{\{u\}}\}, \{x_1^{\emptyset}, \bar{z}_1^{\{u\}}\}, \{z_1^{\{u\}}\}\}\,.$$

In this case, the expansion is not minimally unsatisfiable, and only four of the six clauses need to be introduced as axioms. The grey clauses sitting above the axioms, connected by dotted lines, show the clauses of PA_1 to which the axioms correspond. ■

As in the foregoing example, a Resolution refutation of a CNF F does not necessarily include every clause in F as an axiom, only some unsatisfiable subset. In the same way, a refutation of a partial expansion of a QBF, which uses only a subset of the total expansion for axioms, is, by definition, still a Resolution refutation of the total expansion. As a result, even if the total expansion of a QBF family grows rapidly with n, ∀Exp+Res may still be able to produce short proofs of small, unsatisfiable partial expansions.

Having proved Corollary 4.6, it is very easy to show that ∀Exp+Res is a refutational QBF proof system.

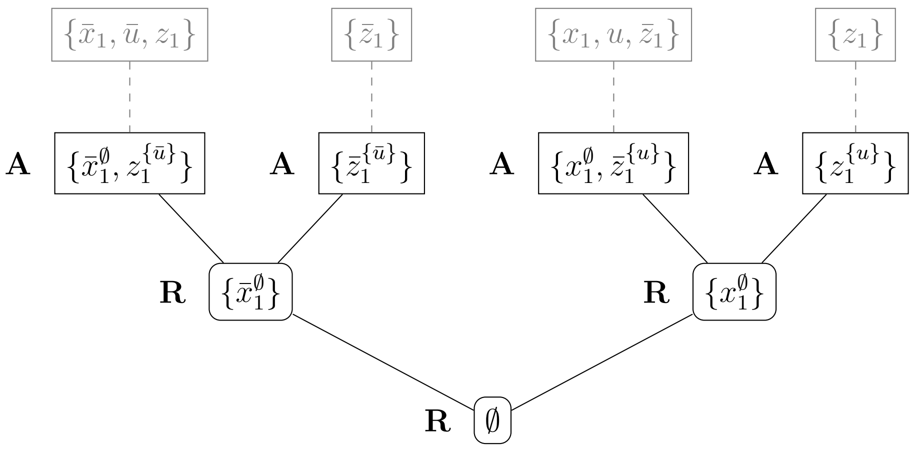


Figure 4.1: A $\forall$Exp+Res refutation of PA_1.

Theorem 4.9 ([34])**.** *$\forall$Exp+Res is a proof system for the language* FQBF*.*

Proof. Soundness. If a QBF Q has an $\forall$Exp+Res refutation, then its total expansion is unsatisfiable, by the soundness of Res. Hence Q is false, by Corollary 4.6. *Completeness.* If Q is false, its total expansion is unsatisfiable by Corollary 4.6, and has a Resolution refutation by completeness of Res. *Checkability.* It can be checked efficiently whether a clause belongs to the complete expansion of a QBF. So checkability of $\forall$Exp+Res follows from that of Res. $\square$

4.3 A lower-bound technique for expansion

Now we turn to the task of demonstrating proof-size lower bounds in $\forall$Exp+Res. We present a technique based on an observation that relates the satisfiability of a partial expansion to the countermodel set of the QBF. We turn to this observation first.

4.3.1 Partial expansions and countermodel range

In $\forall$Exp+Res we refute false QBFs, which have no models and whose expansions are unsatisfiable, and the axioms of a refutation are an unsatisfiable subset of the total expansion. Thus, an obvious factor in the size of $\forall$Exp+Res refutations is the size of the smallest unsatisfiable subset of the expansion. In turn, this is related to the size of the smallest unsatisfiable partial expansion.

Now, in Subsection 4.1.1, we showed that the satisfiability of the total expansion can be determined by finding a model for the QBF. In fact, as we show now, the unsatisfiability of a *partial* expansion can be determined by looking at the countermodels of the QBF. This is the relationship at the centre of our technique.

To make the relationship clear, we need to define the *range* of a countermodel.

Definition 4.10 (countermodel range)**.** *The* range *of a countermodel h for a QBF Q is the set of total universal assignments*

$$\mathrm{rng}(Q) := \{h(\varepsilon) : \varepsilon \in \langle \mathrm{vars}_\exists(Q)\rangle\}\,.$$

The range of a countermodel is exactly its range when viewed as a single function. It can also be understood as the set of total assignments played by the universal player in the corresponding evaluation game strategy.

The next result characterises the satisfiability of a partial expansion based on the ranges of the countermodels.

Lemma 4.11. *Given a QBF Q and a set of universal dependency functions h,*

$$\exp(Q, \mathrm{rng}(h)) \textit{ is unsatisfiable} \quad \Leftrightarrow \quad h \textit{ countermodels } Q\,.$$

Proof. Let Γ be the range of $h := \{h_i\}_{i\in[m]}$, and let $Q := P \cdot F$, where the prefix P is of the form

$$\forall U_1 \exists X_1 \cdots \forall U_d \exists X_d\,.$$

The proof is by induction on the quantifier depth d of Q. The base case $d = 0$ is trivial, since in that case we must have $Q = \{\emptyset\}$.

For the inductive step, let $d \geq 1$. We denote the variables in the first universal block by $U_1 = \{u_1, \ldots, u_j\}$, and those in the first existential block by $X_1 = \{x_1, \ldots, x_k\}$. All the functions $h_1, \ldots, h_j$ are constant, so we define the assignment to U_1 that replicates them, namely

$$\mu_1 := \{h_i(\emptyset) : 1 \leq i \leq j\}\,.$$

Note that the projection of every assignment in Γ to U_1 is μ_1.

Now, the only annotated copies of variables in X_1 that belong to $\mathrm{vars}(\exp(Q, \Gamma))$ are the members of the set

$$X_1^{\mu_1} := \{x_i^{\mu_1} : 1 \leq i \leq k\}\,.$$

For each α in $\langle X_1\rangle$, we define an assignment to $X_1^{\mu_1}$, namely

$$\begin{array}{cccl} \alpha_{\mu_1} & : & X_1^{\mu_1} & \to \mathbb{D} \\ & & x_i^{\mu_1} & \mapsto \alpha(x_i)\,. \end{array}$$

It is easy to see that

$$\exp(Q,\Gamma)[\alpha_{\mu_1}] = \exp(Q[\mu_1 \cup \alpha], \Gamma')\,. \tag{4.1}$$

Further, for each α in $\langle X_1 \rangle$, we define a set of universal dependency functions for $Q[\mu_1 \cup \alpha]$, namely $h_\alpha := \{h_i^\alpha\}_{j<i\leq m}$, where

$$\begin{array}{rcccl} h_i^\alpha & : & \langle H_i \setminus X_1 \rangle & \to & \langle \{u_i\} \rangle \\ & & \varepsilon & \mapsto & h_i(\alpha \cup \varepsilon)\,. \end{array}$$

The range of h_α, denoted Γ_α, is obtained from Γ by deleting μ_1 from every assignment.

For the "$\Rightarrow$" direction, suppose that $\exp(Q,\Gamma)$ is unsatisfiable. By (4.1),

$$\exp(Q[\mu_1 \cup \alpha], \Gamma')$$

is unsatisfiable. Since the quantifier depth of $Q[\mu_1 \cup \alpha]$ is $d-1$, it is countermodelled by h_α, by the inductive hypothesis.

Now we can show that h countermodels Q. Let $\varepsilon := \alpha \cup \beta$ be an arbitrary extension of α to a total existential assignment to Q, where α and β are disjoint. Since h_α countermodels $Q[\mu_1 \cup \alpha]$, the assignment

$$\beta \cup \{h_i^\alpha(\beta\restriction_{H_i \setminus X_1}) : j < i \leq m\}$$

falsifies $F[\mu_1 \cup \alpha]$. Hence the assignment

$$\varepsilon \cup \{h_i(\varepsilon\restriction_{H_i}) : i \in [m]\} = \alpha \cup \beta \cup \mu_1 \cup \{h_i^\alpha(\beta\restriction_{H_i \setminus X_1}) : j < i \leq m\}$$

falsifies F.

For the "$\Leftarrow$" direction, suppose that h countermodels Q. Aiming for contradiction, suppose that $\exp(Q,\Gamma)$ is satisfied by some assignment σ. In contrast to the other direction, we consider the particular α defined by

$$\begin{array}{rcccl} \alpha & : & X_1 & \to & \mathbb{D} \\ & & x_i & \mapsto & \sigma(x_i^{\mu_1})\,. \end{array}$$

It is easy to see that σ extends α_{μ_1}, therefore

$$\exp(Q[\mu_1 \cup \alpha], \Gamma')$$

is satisfiable, by (4.1)

Once more, we take an arbitrary extension $\varepsilon := \alpha \cup \beta$ of α to the existential variables of Q. By definition of countermodel, the assignment

$$\varepsilon \cup \{h_i(\varepsilon\restriction_{H_i}) : i \in [m]\}$$

falsifies F, so the assignment

$$\beta \cup \{h_i^\alpha(\beta\restriction_{H_i \setminus X_1}) : j < i \leq m\}$$

falsifies $F[\mu_1 \cup \alpha]$. Therefore h_α countermodels $Q[\mu_1 \cup \alpha]$. But this leads to a contradition, since it implies that

$$\exp(Q[\mu_1 \cup \alpha], \Gamma')$$

is unsatisfiable, by the inductive hypothesis. □

4.3.2 A tight characterisation of lower bounds

Now we use Lemma 4.11 to characterise the reasons for hardness in $\forall$Exp+Res. Essentially, the characterisation is based on one fairly obvious consequence of the lemma: the minimal number of axioms in a refutation is bounded below by the minimal cardinality of the range of a countermodel.

First some terminology.

Definition 4.12 (countermodel size)**.** *The* size *of a countermodel is the cardinality of its range.*

We say that a QBF family *requires* countermodels of size $t(n)$ when, for each natural number n, the size of every countermodel for the n^{th} instance is at least $t(n)$. A *countermodel family* for a QBF family is a function from the natural numbers that maps each n to a countermodel for the n^{th} instance.

Theorem 4.13 (Characterisation of hardness in $\forall$Exp+Res)**.** *A QBF family $\mathcal{Q}$ requires* $\forall$Exp+Res *refutations of size $t(n)$ if, and only if, either*

(a) *$\mathcal{Q}$ requires countermodels of size $t(n)$, or*

(b) *for each countermodel family $\mathcal{H}$, the CNF family*

$$\begin{array}{rcccl} \mathcal{F} & : & \mathbb{N} & \to & \mathbb{F} \\ & & n & \mapsto & \exp(\mathcal{Q}(n), \mathrm{rng}(\mathcal{H}(n))) \end{array}$$

requires Resolution refutations of size $t(n)$.

Proof. First, the "if" direction. Suppose that condition (a) holds. By Lemma 4.11, the smallest unsatisfiable subsets of total expansions of $\mathcal{Q}$ are of size $t(n)$. Hence $\forall$Exp+Res refutations of $\mathcal{Q}$ contain at least $t(n)$ axioms. On the other hand, suppose that condition (b) holds. Again by Lemma 4.11, all unsatisfiable partial expansions of $\mathcal{Q}$ require Resolution refutations of size $t(n)$, and the same goes for the total expansions.

Now for the "only if" direction. Suppose that $\mathcal{Q}$ requires $\forall\mathsf{Exp}+\mathsf{Res}$ refutations of size $t(n)$. If condition (a) holds, we're done, so we assume it doesn't. For each countermodel family $\mathcal{H}$, the CNF family

$$n \mapsto \exp(\mathcal{Q}(n), \operatorname{rng}(\mathcal{H}(n)))$$

requires Resolution refutations of size $t(n)$, for otherwise, by Lemma 4.11, the total expansions of $\mathcal{Q}$ would not require Resolution refutations of size $t(n)$. Therefore condition (b) holds. $\square$

Theorem 4.13 characterises precisely the reasons for lower bounds in $\forall\mathsf{Exp}+\mathsf{Res}$. It tells us that every lower bound is either due to a countermodel-size lower bound, or due to a Resolution proof-size lower bound, or both.

We are mostly concerned with superpolynomial proof-size lower bounds, i.e. the case where a QBF family requires refutations of size $\Omega(n^c)$ for each constant c. Here, we can apply Theorem 4.13 with $T(n) = n^c$ for each c. This tells us that the lower bound is either

(a) due to the fact that the family does not have polynomial-size countermodels, or

(b) due to the fact that all expansions corresponding to polynomial-size countermodels require superpolynomial-size Resolution refutations.

4.3.3 Concrete exponential lower bounds

Now we apply Theorem 4.13 to prove exponential proof-size lower bounds for three of our handcrafted families.

The equality family

We first prove that the equality family requires countermodels of exponential size. In fact, as we already mentioned, the countermodels for these formulas are unique.

Theorem 4.14. *$\mathcal{EQ}$ requires countermodels of size 2^n.*

Proof. We claim that, for each n, the unique countermodel for $\mathcal{EQ}(n)$ is $\{h_i\}_{i\in[n]}$, where

$$\begin{array}{rccl} h_i & : & \langle\{x_1,\dots,x_n\}\rangle & \to & \langle\{u_i\}\rangle \\ & & \varepsilon & \mapsto & u_i \mapsto \varepsilon(x_i)\,. \end{array}$$

To see that h is indeed a countermodel, let ε be a total existential assignment to $\mathcal{EQ}(n)$, and let δ be its projection to $\{x_1, \ldots, x_n\}$. Noting that $h_i(\delta)$ merely assigns u_i the same value as δ assigns x_i, it is easy to see that applying the assignment

$$\delta \cup \{h_i(\delta) : i \in [n]\}$$

to eq_n gives the CNF

$$\{\{z_i\} : i \in [n]\} \cup \{\{\bar{z}_1, \ldots, \bar{z}_n\}\} ,$$

which is clearly unsatisfiable, so

$$\varepsilon \cup \{h_i(\varepsilon\restriction_{S_i}) : i \in [n]\}$$

falsifies eq_n.

To see that h is unique, let h' be some set of universal dependency functions for which, for some δ in $\langle\{x_1, \ldots, x_n\}\rangle$ and some i in $[n]$,

$$h_i(\delta)(u_i) \neq \delta(x_i) .$$

Assuming without loss of generality that $i = n$, applying the assignment

$$\delta \cup \{h_i(\delta) : i \in [n]\}$$

to eq_n gives

$$\{\{z_i\} : i \in [n-1]\} \cup \{\{\bar{z}_1, \ldots, \bar{z}_n\}\} ,$$

which is satisfied by the assignment δ' to $\{z_1, \ldots, z_n\}$ which maps z_i to 0 when, and only when, $i = n$. Hence, taking ε as $\delta \cup \delta'$, F is satisfied by

$$\delta \cup \{h_i(\delta) : i \in [n]\} ,$$

so h' is not a countermodel.

It is easy to see that the range of h is $\langle\{u_1, \ldots, u_n\}\rangle$, therefore its size is 2^n. $\square$

We now apply the lower-bound characterisation for $\forall$Exp+Res (Theorem 4.13) to obtain an exponential proof-size lower bound.

Corollary 4.15. *$\mathcal{EQ}$ requires $\forall$Exp+Res refutations of size 2^n.*

The interleaved equality family

By a similar method, we can prove an exponential proof-size lower bound for the interleaved equality family. Unlike EQ_n, EQ'_n does not have a unique countermodel, but the range of any countermodel is nonetheless the set of total universal assignments.

Theorem 4.16. *$\mathcal{EQ}'$ requires countermodels of size 2^n.*

Proof. We show that the range of any countermodel for EQ'_n is $\langle\{u_1, \ldots, u_n\}\rangle$, and the theorem follows.

Let h be a countermodel for EQ'_n, and let μ be an arbitrary total assignment to the universal variables. We prove that $\mu = h(\varepsilon)$, where

$$\begin{aligned} \varepsilon(x_i) \quad &:= \quad \begin{cases} 0 & \text{if } \mu(u_i) = 0 \\ 1 & \text{if } \mu(u_i) = 1 \end{cases}, \quad && \text{for } i \in [n]\,, \\ \varepsilon(z_i) \quad &:= \quad 1\,, && \text{for } i \in [n]\,. \end{aligned}$$

Aiming for contradiction, let j be the least integer for which $h(\varepsilon)\restriction_{\{u_j\}}$ does not equal $\mu\restriction_{\{u_j\}}$, and let H_j be the dependency set for u_j. Observe that $\mathrm{EQ}'_n[\varepsilon\restriction_{H_j}]$ is the QBF with prefix

$$\exists z_j \exists x_{j+1} \forall u_{j+1} \exists z_{j+1} \cdots \exists x_n \forall u_n \exists z_n$$

and matrix consisting of the clauses

$$\begin{aligned} &\{a, z_j\}\,, \\ &\{\bar{x}_i, \bar{u}_i, z_i\}\,, \qquad && \text{for } j+1 \leq i \leq n\,, \\ &\{x_i, u_i, \bar{z}_i\}\,, && \text{for } j+1 \leq i \leq n\,, \\ &\{\bar{z}_j, \ldots, \bar{z}_n\}\,, \end{aligned}$$

where a is the literal represented by the assignment $h(\varepsilon)\restriction_{\{u_j\}}$. Note that the matrix is satisfied by the assignment

$$h(\varepsilon)\restriction_{\{u_j\}} \cup \{\bar{z}_j, z_{j+1}, \ldots, z_n\}\,.$$

Now, let δ be any total existential assignment that extends

$$\varepsilon\restriction_{H_j} \cup \{\bar{z}_j, z_{j+1}, \ldots, z_n\}\,.$$

Since ε and δ agree on H_j, the assignments $h(\varepsilon)\restriction_{\{\mu_j\}}$ and $h(\delta)\restriction_{\{\mu_j\}}$ are identical. It follows that the assignment $\delta \cup h(\delta)$ satisfies eq_n, contradicting the fact that h is a countermodel for EQ'_n. $\square$

Once again the proof-size lower bound follows by application of Theorem 4.13.

Corollary 4.17. *$\mathcal{EQ}'$ requires $\forall\mathsf{Exp{+}Res}$ refutations of size 2^n.*

The Kleine Büning et al. family

Finally we apply the technique to prove an exponential proof-size lower bound for the family $\mathcal{KB}$.

Theorem 4.18. *$\mathcal{KB}$ requires countermodels of size 2^n.*

Proof. In this case, the proof follows similar lines to the proof of Theorem 4.16 for the interleaved equality family. To make things more interesting, we give a proof in terms of winning strategies in the evaluation game, an equivalent description of countermodels (see Subsection 3.2). This gives rise to a less formal, yet more intuitive proof.

Aiming for contradiction, suppose that h is a countermodel for KB_n, and that some universal assignment μ does not belong to its range. Hence, in the corresponding evaluation game strategy, the universal player does not play the total assignment μ.

Suppose that the existential player plays the variables $x_1, y_1, \ldots, x_n, y_n$ according to the assignment ε, defined by

$$\begin{aligned} \varepsilon(x_i) \quad &:= \quad \begin{cases} 0 & \text{if } \mu(u_i) = 1 \\ 1 & \text{if } \mu(u_i) = 0 \end{cases}, \quad \text{for } i \in [n]\,, \\ \varepsilon(y_i) \quad &:= \quad \begin{cases} 1 & \text{if } \mu(u_i) = 1 \\ 0 & \text{if } \mu(u_i) = 0 \end{cases}, \quad \text{for } i \in [n]\,. \end{aligned}$$

The final assignment made by the universal player is to the variable u_n, which occurs directly after the existential player assigns y_n. Since μ does not belong to the range of h, there exists a least integer j such that the universal player sets u_j not equal to $\mu(u_j)$.

Now, let δ be the the restriction of ε to $\{x_1, y_1, \ldots, x_j, y_j\}$, let ν be the restriction of μ to $\{u_1, \ldots, u_j\}$. We consider two cases.

(a) Suppose that $j < n$. Observe that the matrix of $\mathrm{KB}_n[\delta \cup \nu]$ consists of the clauses

$$\begin{array}{ll} \{x_i, \bar{u}_i, \bar{x}_{i+1}, \bar{y}_{i+1}\}\,, & \text{for } j+1 \leq i \leq n-1\,, \\ \{y_i, u_i, \bar{x}_{i+1}, \bar{y}_{i+1}\}\,, & \text{for } j+1 \leq i \leq n-1\,, \\ \{x_n, \bar{u}_n, \bar{z}_1, \ldots, \bar{z}_n\}\,, & \\ \{y_n, u_n, \bar{z}_1, \ldots, \bar{z}_n\}\,, & \\ \{z_i\}\,, & \text{for } i \text{ in } [j]\,, \\ \{u_i, z_i\}\,, & \text{for } j+1 \leq i \leq n\,, \\ \{\bar{u}_i, z_i\}\,, & \text{for } j+1 \leq i \leq n\,. \end{array}$$

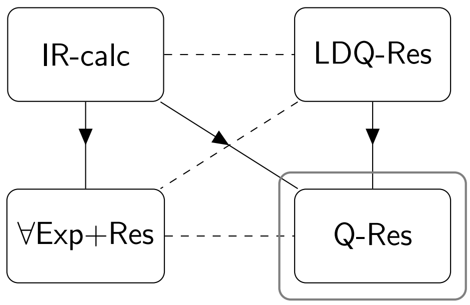
This CNF shows the state of the game according to the winning universal strategy h as the existential player comes to assign the variable x_{j+1}, and it can be satisfied by assigning all existential variables positively. It follows that the existential player can win the game from this point, but this contradicts the fact that h is a winning strategy, which must never give the existential player an opportunity to win.

(b) Suppose on the other hand that $i = n$. Then the matrix of $\mathrm{KB}_n[\delta \cup \nu]$ consists of the n unit clauses $\{z_i\}$. This also contradicts the fact that h is a winning strategy, since the existential player can win the game from here by assigning each z_i positively. $\square$

Corollary 4.19. *$\mathcal{KB}$ requires* $\forall$Exp+Res *refutations of size* 2^n.

Chapter 5

Universal Reduction



Now it's time to take a closer look at the main alternative to universal expansion: universal reduction. Universal reduction was first introduced as a logically correct rule of inference in the QBF proof system Q-Res [35], highlighted in the figure above. Later, it was used as a propagation technique in QBF solvers based on CDCL [27, 72, 40].

Universal reduction constitutes a fundamentally different way to deal with universal quantification. Whereas the expansion paradigm is based on an immediate translation to propositional logic, reduction operates squarely in the realm of quantified Boolean logic, and exploits inferences which are logically correct in terms of QBF models.

Strategy-size lower bounds for reduction

In this chapter, we investigate how our lower-bound technique for expansion carries over to Q-Res. We find that the technique does not lift straight away to Q-Res. There exist formulas which require large (exponential-size) countermodels, yet admit short (linear-size) Q-Res refutations.

In fact, this situation can only occur when the quantifier depth of the family is unbounded. Indeed, we find that strategy size *does* give rise to Q-Res proof-size lower bounds when quantifier depth is bounded above by a constant. In particular, this allows us to prove an exponential lower bound for the equality family.

To prove the main result (Theorem 5.16), we must venture into *strategy extraction*, a well-known notion for practitioners and theoreticians alike. The main message of strategy extraction is that refutations represent countermodels, or equivalently, winning strategies for the universal player in the evaluation game.

Organisation of the chapter

We recall Q-Res in Section 5.1, followed by a description of strategy extraction in Section 5.2. In Section 5.3, we present the lower-bound technique for Q-Res, and obtain a concrete lower bound for the equality family.

5.1 The proof system Q-Res

Q-Resolution is arguably the most natural quantified version of propositional Resolution. It is obtained from Resolution by adding a single inference rule, called universal reduction, which allows universal literals to be deleted under certain conditions.

Universal reduction is based on the notion of a 'trailing' literal. Given a prefix P, we say that a universal literal a belonging to a clause C is *trailing* in C with respect to P when $\mathrm{var}(a)$ does not belong to any of the dependency sets for the existential variables in C. Trailing literals are always universal.

For example, with respect to the prefix $\forall u_1 \exists x_1 \forall u_2 \exists x_2$, we have

(a) $\bar{u}_2$ is trailing in $\{\bar{u}_1, x_1, \bar{u}_2\}$,

(b) both $\bar{u}_1$ and u_2 are trailing in $\{\bar{u}_1, u_2\}$,

(c) no literals are trailing in $\{\bar{u}_1, \bar{u}_2, \bar{x}_2\}$.

We emphasise that all literals are trailing in a clause containing no existential variables, as in (b) above.

Definition 5.1 (Q-Res [35])**.** *A* Q-Res *derivation from a QBF $Q := P \cdot F$ is a sequence $C_1, \dots, C_k$ of non-tautological clauses in which at least one of the following holds for each $i \in [k]$:*

A Axiom: *C_i is a clause in F;*

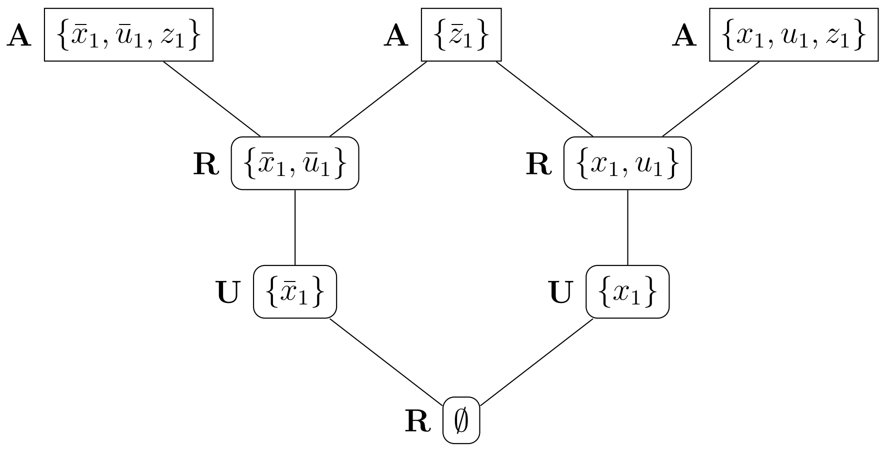


Figure 5.1: A Q-Res refutation of EQ_1.

R Resolution: $C_i = \text{res}(C_r, C_s, p)$, *for some* $r, s < i$ *and existential literal* p*;*

U Reduction: $C_i = C_r \setminus \{a\}$, *for some* $r < i$, *where* a *is universal and trailing in* C_r *with respect to* P*;*

W Weakening: C_i *is* $\mathbb{L}$, *or is subsumed by* C_r *for some* $r < i$.

A Q-Res derivation from Q whose conclusion is empty is called a *refutation* of Q. As usual, the size of a derivation is the number of clauses.

Example 5.2. A Q-Res refutation of the first instance of the equality family is shown in Figure 5.1. The universal reduction steps are marked '**U**'. Note that, with respect to the prefix of EQ_1, namely $\exists x_1 \forall u_1 \exists z_1$, the literal $\bar{u}_1$ is trailing in $\{\bar{x}_1, \bar{u}_1\}$ and the literal u_1 is trailing in $\{x_1, u_1\}$ ■

At first, it may seem strange that tautological clauses, which are harmless in propositional logic, are explicitly forbidden in Q-Res. The next example demonstrates why this is necessary.

Example 5.3. We consider again the true QBF from Example 3.3, namely

$$\forall u_1 \exists x_1 \cdot \{\{\bar{u}_1, \bar{x}_1\}, \{u_1, x_1\}\}\,.$$

If we were to allow tautological clauses in Q-Res, we could resolve the clauses in the matrix to obtain $\{\bar{u}_1, u_1\}$, and then apply universal reduction to obtain the empty clause. Thus, we would have a refutation of a true QBF, and an unsound proof system. ■

Instead of the deletion of trailing literals, universal reduction can be construed as the assignment of trailing literals. Of course, we would always choose to assign a universal literal so as to falsify it, since satisfying it returns $\mathbb{L}$, which is useless in a refutation. Moreover, assigning the variable of a complementary pair of universal literals would also always return $\mathbb{L}$. Hence, this arguably makes for a better definition, since we would no longer need to disallow tautologies.

However, for better or for worse, universal reduction is conventionally the deletion of trailing literals, so we stick to this convention, and disallow tautologies.

Soundness and completeness

Now we turn to the task of proving that Q-Res is indeed a refutational QBF proof system.

We deal with soundness first. Our soundness proof can be thought of as the quantified version of the soundness proof for Resolution (Fact 2.8). This time we prove that the input QBF semantically entails every derived clause, when considered as a QBF under its prefix. We emphasise that we are no longer talking about entailment in propositional logic, rather entailment in quantified Boolean logic, based on QBF models.

Lemma 5.4 ([35]). *A QBF is false if it has a* Q-Res *refutation.*

Proof. Let $\pi := C_1, \ldots, C_k$ be a Q-Res refutation of a QBF $Q := P \cdot F$. For each j in $[k]$, let $F_j = \{C_1, \ldots, C_j\}$.

Aiming for contradiction, suppose that Q has a model $f := \{f_i\}_{i \in [n]}$. We prove by induction on j in $[k]$ that f models $P \cdot F_j$. Hence at step $j = k$, we reach a contradiction, since F_k contains the empty clause C_k.

The base case $j = 1$ is trivial, since C_1 is an axiom, and belongs to F.

For the inductive step, let $2 \leq j \leq k$, suppose that f is a model for $P \cdot F_{j-1}$, and let μ be a total assignment to the universal variables of Q. We consider four cases, depending on how C_j was derived. In each case we show that

$$\sigma := \mu \cup \{f_i(\mu\restriction_{S_i})\}_{i \in [n]}$$

satisfies C_j, and hence f is a model for $P \cdot F_{j-1} \cup \{C_j\} = P \cdot F_j$.

A If C_j is an axiom, the inductive step is identical to the base case.

R Suppose that $C_j = \mathrm{res}(C_r, C_s, p)$ for some $r, s < j$, and some existential literal p. Then, since both C_r and C_s are in F_{j-1}, σ satisfies both of them, by the inductive hypothesis. Hence σ satisfies C_j by the logical correctness of propositional Resolution.

U Suppose that $C_j = C_r \setminus \{a\}$, for some $r < j$, where a is trailing in C_r with respect to P. Aiming for contradiction, suppose that σ falsifies C_i.

We consider the assignment

$$\nu := \mathrm{comp}(\mu, \mathrm{var}(a)) \ .$$

By definition of trailing literal, $\mathrm{var}(a)$ does not belong to the dependency set of any existential variable in C_r. It follows that $\{f_i(\nu\restriction_{S_i})\}_{i\in[n]}$ falsifies all the existential literals in C_r.

As σ satisfies C_r by the inductive hypothesis, μ must satisfy a, therefore ν falsifies a. Then ν, which agrees with μ on all universal variables except $\mathrm{var}(a)$, must falsify the universal literals in C_r. But then the assignment

$$\nu \cup \{f_i(\nu\restriction_{S_i})\}_{i\in[n]}$$

falsifies C_r, contradicting the inductive hypothesis.

W Suppose that $C_j = \mathbb{L}$, or is subsumed by C_r with $r \leq j$. In the former case, σ satisfies C_j trivially. In the latter case, σ satisfies C_r by the inductive hypothesis, and therefore satisfies the larger clause C_j. $\square$

For completeness, we construct a canonical refutation based on a countermodel for the input QBF.

Lemma 5.5 ([35]). *Every false QBF has a* Q-Res *refutation.*

Proof. Let

$$\forall U_1 \exists X_1 \cdots \forall U_d \exists X_d \cdot F$$

be a false QBF with countermodel h. We prove the lemma by induction on the quantifier depth d of Q.

The base case $d = 0$ is trivial since, by definition of countermodel, F contains the empty clause, and the sequence consisting of the empty clause itself is a Q-Res refutation of Q.

For the inductive step, let $d \geq 1$. Consider the assignment to U_1 defined by

$$\mu := \{h_j(\emptyset) : u_j \in U_1\} ,$$

and the assignment set defined by

$$A := \{\mu \cup \varepsilon : \varepsilon \in \langle X_1 \rangle\} .$$

The assignment set A has two important properties.

(a) The negation of every assignment in A can be derived from Q in Q-Res.

(b) In a Q-Res derivation from Q, the empty clause can be derived from the negations of the assignments in A.

We prove the lemma by showing these two properties in turn.

(a) Let ε be an assignment to X_1 and let C be the negation of $\mu \cup \varepsilon$. Now, $Q[\mu \cup \varepsilon]$ has quantifier depth $d-1$, and is false by Lemma 3.6, therefore it has a Q-Res refutation

$$C_1, \dots, C_k$$

by the inductive hypothesis.

Now, if C_i is an axiom, it belongs to $F[\mu \cup \varepsilon]$, and $C_i \cup C$ is subsumed by some clause in F. Moreover, no universal variable in $Q[\mu \cup \varepsilon]$ belongs to the dependency set of any existential in C.

Letting 'seq(F)' denote the clauses of F written as a sequence and '$\circ$' denote concatenation of sequences, it follows that

$$\mathrm{seq}(F) \circ C_1 \cup C, \dots, C_k \cup C ,$$

is a valid Q-Res derivation from F, since the uniform addition of C cannot invalidate any reduction steps. Moreover, $C_k \cup C = C$, since C_k is the empty clause.

(b) It is easy to see that the negations of the assignments in $\langle X_1 \rangle$ form an unsatisfiable set of clauses, which have a Resolution refutation

$$D_1, \dots, D_k$$

by the completeness of Resolution. Hence, putting E as the negation of μ, the sequence

$$D_1 \cup E, \dots, D_k \cup E$$

is a derivation of E from the negations of the assignments in A. Moreover, since E contains no existential literals, every literal in E is trailing, and the empty clause can be derived from it by reduction. □

So, we only need to show checkability, and we have proved that Q-Res is indeed a refutational QBF proof system. Given the simple nature of universal reduction, this is almost trivial.

Theorem 5.6. Q-Res *is a proof system for the language* FQBF.

Proof. Soundness and completeness. Established by Lemmata 5.4 and 5.5. *Checkability.* Universal reduction steps can clearly be checked efficiently, so the checkability of Q-Res follows from that of Resolution. □

5.2 Strategy extraction

Now we turn to strategy extraction, a QBF specific paradigm which underpins our lower-bound technique for Q-Res.

Overview

Strategy extraction originated in QBF solving, where many practical applications require not only the truth value of the instance, but also the certifying model or countermodel. Many QBF solvers (such as DepQBF [41], CAQE [49] and RAReQs [33]) have support for outputting these witnesses. QCDCL solvers, which correspond to reduction-based proof systems, essentially build the witness as the solver works its way through the search space. Hence, on false QBFs, the trace of the solver houses a countermodel in a fairly natural way.

Analogously, strategy extraction in theoretical models of solving refers to the idea that a QBF refutation *represents* a countermodel, much in the same way that a CNF *represents* a Boolean function. To compute a Boolean function from a CNF, we would take an element of the domain, which is a total assignment to the variables, and apply it to the formula. We output 1 if the resulting CNF is the empty set of clauses, and 0 otherwise, in which case it contains the empty clause.

Similarly, in QBF proof systems that have strategy extraction, we can define an algorithm which, given a refutation, computes a countermodel. More precisely, we can define a polynomial-time computable function which takes a refutation and a total existential assignment ε, and returns a total universal assignment μ, for which $\varepsilon \cup \mu$ always falsifies the matrix of the refuted QBF.

Relation to existing literature

The main result of this section is Theorem 5.12 [28], which proves constructively that Q-Res has strategy extraction. The reader may wonder why we need a whole section to reprove an existing result. The answer is that our presentation of the result differs significantly from [28]. Our handling of the details leads to an original lower-bound technique, and this merits (and even requires) a complete, formal treatment of strategy extraction in Q-Res.

A lower-bound technique for Q-Res, also based on strategy extraction, has already been proposed. In [12], Beyersdorff, Bonacina and Chew developed a technique lifting circuit lower bounds to proof-size lower bounds in 'P+∀red' QBF proof systems, of which QU-Res (Figure 1.3) is an example. Our construction and proof differs from [12], which uses decision lists as an intermediate computational model, and ultimately builds countermodels as AC_0 circuits. We do not use any auxiliary computational models; we rather compute the extracted strategy directly from the refutation, as described above. Moreover, our technique establishes lower bounds that cannot be proved using the technique of Beyersdorff et al. ($\mathcal{EQ}$), and vice versa ($\mathcal{PA}$).

The relation between the two techniques, both of which are based on strategy extraction, but which appear to prove entirely different lower bounds, is an interesting moot point to which we return in Chapter 13.

Organisation of the section

Our first goal in this section is to define the 'extracted strategy' (Definition 5.11), that is, we must define an algorithm which maps existential assignments to universal assignments, based on a refutation. Our second goal is to prove that it really works (Theorem 5.12). Inevitably this entails the application of existential assignments to refutations (Subsection 5.2.1), but we must also ensure that we work with weakening-free refutations (Subsection 5.2.2) that are also 'conclusion-unique' (Subsection 5.2.3). After dealing with these three small tasks, we present the extracted strategy in Subsection 5.2.4.

5.2.1 Closure under existential assignments

As the first of three small tasks preceding the definition of the extracted strategy, we show that Q-Res is 'closed under existential assignments'. By this we mean that the application of an existential assignment to a refutation returns a refutation of the input QBF under the same assignment. We first define the application of an existential assignment, and then prove the result.

Definition 5.7. *The application of an existential assignment ε to a* Q-Res *refutation $\pi := C_1, \ldots, C_k$ returns the sequence*

$$\pi[\varepsilon] := C_1[\varepsilon], \ldots, C_k[\varepsilon] .$$

Resolution refutations are 'closed' under existential assignments in the precise sense of the following fact.

Fact 5.8 ([28])**.** *If π is a* Q-Res *refutation of a QBF Q, and ε is a partial assignment to* $\mathrm{vars}_\exists(Q)$*, then $\pi[\varepsilon]$ is a* Q-Res *refutation of $Q[\varepsilon]$.*

Proof. Let $\pi := C_1, \ldots, C_k$. We show by induction on $i \in [k]$ that each clause $C_i[\varepsilon]$ is a valid Q-Res inference in $\pi[\varepsilon]$. Observe that, if ε satisfies C_i, then $C_i[\varepsilon]$ is $\mathbb{L}$ and can be derived by weakening. Hence, we can assume from now on that ε does not satisfy C_i.

For the base case $i = 1$, observe that C_1 is derived by axiom, so C_1 belongs to F. Then $C_1[\varepsilon]$ belongs to $F[\varepsilon]$, so $C_1[\varepsilon]$ can be derived by axiom.

For the inductive step, let $i \geq 2$. We consider four cases.

A If C_i was derived by hypothesis, the inductive step is identical to the base case.

R If C_i was derived by resolution from C_r and C_s over the existential pivot literal p, we consider three further cases.

(i) If ε satisfies the pivot literal p, then $C_i[\varepsilon]$ is subsumed by $C_s[\varepsilon]$, and can therefore be derived by weakening.

(ii) If ε falsifies p, then $C_i[\varepsilon]$ is subsumed by $C_r[\varepsilon]$, and can be derived similarly by weakening.

(iii) If ε neither satisfies nor falsifies p, then, since ε satisfies neither C_r nor C_s, $C_i[\varepsilon]$ can be derived by resolution from $C_r[\varepsilon]$ and $C_s[\varepsilon]$ over pivot literal p.

U If C_i was derived by reduction, say by removing the trailing literal a from C_r, then ε does not satisfy C_r, and $C_i[\varepsilon]$ can be derived by reduction from $C_r[\varepsilon]$ by removing the same trailing literal.

W If C_i was derived by weakening from C_r, then ε does not satisfy C_r, which subsumes C_i. It is easy to see that $C_r[\varepsilon]$ subsumes $C_i[\varepsilon]$, so the latter can be derived by weakening. $\square$

5.2.2 Removing weakening steps

Weakening steps are a hindrance for strategy extraction, but unfortunately the application of an existential assignment may introduce them, even if the original refutation is weakening-free. For our second task, we need to show that weakening steps can be removed algorithmically from $\mathsf{Q\text{-}Res}$ refutation with no size increase.

Fact 5.9 (folklore)**.** *Weakening inferences can be removed algorithmically from* $\mathsf{Q\text{-}Res}$ *refutations with no increase in size, while preserving the refutation.*

Proof. Let $\pi := C_1, \ldots, C_k$ be a $\mathsf{Q\text{-}Res}$ refutation of a QBF $Q := P \cdot F$

Since $\mathbb{L}$ cannot be an antecedent of any inference in π, and the conclusion C_k is not $\mathbb{L}$, deleting instances of $\mathbb{L}$ preserves the refutation. Therefore we can assume without loss of generality that $\mathbb{L}$ does not occur in π.

Now, we transform π into a weakening-free refutation $\pi' := C'_1, \ldots, C'_k$ by processing the clauses C_i in order, as follows:

A if C_i was introduced as an axiom, then define $C'_i := C_i$;

R if C_i was derived by resolution from C_r and C_s over pivot p, then define

$$C'_i := \begin{cases} C'_r & \text{if } p \notin C'_r\,, \\ C'_s & \text{if } p \in C'_r \text{ and } \tilde{p} \notin C'_s\,, \\ \operatorname{res}(C'_r, C'_s, p) & \text{if } p \in C'_r \text{ and } \tilde{p} \in C'_s\,; \end{cases}$$

U If C_i was derived by reduction, say by removing the trailing literal a from C_r, then define $C'_i := C'_r \setminus \{a\}$.

W If C_i was derived by weakening from C_r, then define $C'_i := C'_r$.

To conclude, we show by induction on $i \in [k]$ that C'_i is a subset of C_i, and is the consequent of a valid non-weakening inference in π'. The base case $i = 1$ is established trivially, since $C'_1 = C_1$ is a clause in F. For the inductive step, let $i \geq 2$. We consider four cases.

A If C_i was introduced as an axiom, the inductive step is identical to the base case.

R If C_i was derived by resolution we consider three further cases.

(i) If $p \notin C'_r$, then $C'_i = C'_r$ subsumes C_i, and can be derived by a non-weakening inference by the inductive hypothesis.

(ii) If $p \in C'_r$ and $\tilde{p} \notin C'_s$, then $C'_i = C'_s$ subsumes C_i, and can be derived by a non-weakening inference by the inductive hypothesis.

(iii) If $p \in C'_r$ and $\tilde{p} \in C'_s$, then $C'_i = \text{res}(C'_r, C'_s, p)$. So C'_i is a valid resolution inference in π', and

$$C'_i = (C'_r \setminus \{p\}) \cup (C'_s \setminus \{\tilde{p}\}) \subseteq (C_r \setminus \{p\}) \cup (C_s \setminus \{\tilde{p}\}) = C_i$$

holds by the inductive hypothesis.

U If C_i was derived by reduction, then C'_r is a subset of C_r by the inductive hypothesis. If $a \notin C'_r$, then $C'_i = C'_r$ is a subset of C_i, and can be derived by a non-weakening inference, by the inductive hypothesis. Otherwise, $C'_i = C'_r \setminus \{a\}$ is a subset of C_i by the inductive hypothesis, and can be derived by reduction.

W If C_i was derived by weakening, then C_i is a subsumed by C_r, and

$$C'_i = C'_r \subseteq C_r \subseteq C_i \,,$$

by the inductive hypothesis. Moreover, $C'_i = C'_r$ is a valid non-weakening inference, by the inductive hypothesis. □

5.2.3 Tidy refutations

As our third and final task, we show that weakening-free refutations that have a unique conclusion have a particular property. The property concerns the appearance of universal literals, and is crucial for strategy extraction.

We call a Q-Res derivation *conclusion-unique* when there is exactly one clause in the sequence which is not the antecedent of an inference. For example, the refutation in Figure 5.1 is conclusion-unique, as every clause is the antecedent of a resolution or reduction step, with the exception of the empty clause. If we remove the empty clause, the derivation is no longer conclusion-unique, because now neither clause $\{\bar{x}_1\}$ nor $\{x_1\}$ is the antecedent of an inference.

We call a refutation *tidy* when it is both conclusion-unique and weakening-free. The crucial property of tidy refutations is that complementary literals in first-block universal variables never occur.

Fact 5.10 (folklore)**.** *For each QBF Q whose first block is universal, and each tidy* Q-Res *derivation π from Q, first-block variables appear in at most one polarity in π.*

Proof. Let Q be a QBF whose first block is universal, and let $C_1, \ldots, C_k$ be a tidy derivation from Q. We prove the result by induction on k in $\mathbb{N}$.

The base case $k = 1$ is trivial, since C_1 is a non-tautological clause from the matrix of the input QBF.

For the inductive step, let $k \geq 2$. Since π is tidy, C_k cannot be an axiom. We consider two cases, based on how the conclusion C_k was derived.

R Suppose that C_k was derived by resolution from C_r and C_s over pivot literal p. Aiming for contradiction, suppose that complementary literals in some universal variable u in the first block of Q appear in π.

By the inductive hypothesis, u appears in exactly one polarity in π_r, say negatively, and appears positively in π_s. Since C_k was derived by resolution over an existential pivot, and is not a tautology, variable u is absent from at least one of C_r and C_s, say C_r. This implies that $\bar{u}$ is reduced from some clause C_t in π_r.

Note that C_t contains no existential variables. Since π_r is tidy, and resolution is only allowed over existential pivots, it is easy to see that C_r contains no existential variables. But this contradicts the fact that $C_k = \text{res}(C_r, C_s, p)$.

U Suppose that C_k was derived by reduction from C_r. Now, if $r \neq k - 1$, we reach a contradiction, since then C_{k-1} is not the antecedent of any inference step, and π is not conclusion-unique. Hence $r = k - 1$, and the subderivation of C_r is $C_1, \ldots, C_{k-1}$. By the inductive hypothesis, each first-block variable appears in at most one polarity in $C_1, \ldots, C_{k-1}$, and therefore also in $C_1, \ldots, C_k$. $\square$

5.2.4 The extracted strategy

We are now ready to show how to extract strategies from Q-Res refutations.

Fact 5.9 tells us that we can remove weakening steps algorithmically from Q-Res refutations. Therefore, we can algorithmically transform an arbitrary refutation π into a tidy refutation tidy(π), by taking the subderivation of the first occurrence of the empty clause and removing any weakening steps.

We use $\pi[[\varepsilon]]$ as a shorthand for tidy($\pi[\varepsilon]$). We emphasise that this merely represents the standard substitution operation, followed by some neatening-up which gives us a weakening-free, conclusion-unique refutation.

The following definition, which formally describes the extracted strategy, is a simplified adaptation of the algorithm in [28].

Definition 5.11 (extracted strategy [28]). *Given a* Q-Res *refutation* π *of a QBF* Q*, the extracted strategy for* π *is the set of universal dependency functions* $\{h_j\}_{j\in[m]}$ *defined by*

$$\begin{array}{rcl} h_j & : & \langle H_j\rangle \;\to\; \langle\{u_j\}\rangle \\ \varepsilon & \mapsto & \begin{cases} \{u_j\} & \text{if } \bar{u}_j \text{ appears in } \pi[[\varepsilon\restriction_{\{x_1\}}]]\cdots[[\varepsilon\restriction_{\{x_{n_j}\}}]] \\ \bar{u}_j & \text{otherwise}. \end{cases} \end{array}$$

where $H_j = \{x_1, \dots, x_{n_j}\}$.

Note that Fact 5.10 guarantees that the extracted strategy is well-defined. Now we prove that the extracted strategy is indeed a countermodel.

Theorem 5.12 ([28]). *The extracted strategy for a* Q-Res *refutation of a QBF* Q *is a countermodel for* Q.

Proof. Let $\{h_j\}_{j\in[m]}$ be the extracted strategy for a refutation π of $Q := P\cdot F$, and let the existential variables of Q be indexed $x_1, \dots, x_n$.

Take an arbitrary total existential assignment ε to Q. By the closure of Q-Res under existential assignments (Fact 5.8),

$$\pi[[\varepsilon\restriction_{\{x_1\}}]]\cdots[[\varepsilon\restriction_{\{x_n\}}]] \quad \text{is a refutation of} \quad Q[\varepsilon\restriction_{\{x_1\}}]\cdots[\varepsilon\restriction_{\{x_n\}}] = Q[\varepsilon]\,.$$

Now, $Q[\varepsilon]$ is a fully universally quantified QBF. By Fact 5.10, the definition of extracted strategy (Definition 5.11), and the fact that restriction of a refutation cannot introduce new literals, the first clause of this refutation is falsified by $\{h_j(\varepsilon\restriction_{H_j})\}_{j\in[m]}$. Since the first clause is introduced as an axiom, it belongs to $F[\varepsilon]$. Therefore

$$\varepsilon \cup \{h_j(\varepsilon\restriction_{H_j})\}_{j\in[m]}$$

falsifies some clause in F. □

Two proofs of soundness

Theorem 5.12 actually constitutes an alternative proof of soundness for Q-Res. In our original proof of soundness (Theorem 5.4) we showed that a QBF with a Q-Res refutation has no model, and is therefore false by definition. This time we proved that a QBF with a Q-Res refutation has a countermodel, and is therefore false by the Folklore Theorem (Theorem 3.8).

5.3 Lower bounds in Q-Res

In this section, we show how to use strategy extraction to prove Q-Res lower bounds based on countermodel size. The technique works only for QBFs whose quantifier depth is bounded above by a constant, but this is already good enough to prove an exponential lower bound for the equality formulas.

5.3.1 Unbounded quantifier depth and strategy size

We first show that the lower-bound technique for $\forall$Exp+Res does not lift to Q-Res on unbounded formula families. In particular, the interleaved equality family EQ' requires countermodels of exponential size (Theorem 4.16), but admits Q-Res refutations of linear size.

Lemma 5.13. *The formula family $\mathcal{EQ}'$ admits linear-size* Q-Res *refutations.*

Proof. Let n be a natural number. We claim that, for each i in $[n]$, in a Q-Res derivation from EQ'_n, all of the clauses in the CNF eq_{i-1} can be derived from those of eq_i in a constant number of steps. Moreover, it is easy to see that the QBF

$$\exists x_1 \forall u_1 \exists z_1 \cdots \exists x_n \forall u_n \exists z_n \cdot \mathrm{eq}_1$$

is false, and hence has a constant-size Q-Res refutation. It follows that EQ' admits linear-size Q-Res refutations.

Now for the claim. In fact, the only clause in eq_{i-1} that cannot be introduced as an axiom from EQ'_n is $\{\bar{z}_1, \dots, \bar{z}_{i-1}\}$. A constant-size Q-Res derivation of this clause from eq_i, with the respect to the prefix of EQ'_n, is shown in Figure 5.2. $\square$

This upper bound, along with the $\forall$Exp+Res lower bound (Corollary 4.17), shows that the interleaved equality family exponentially separates Q-Res from $\forall$Exp+Res. Indeed, EQ' can be seen as a simplified version of the original separating formulas [34, (2), p. 38].

5.3.2 A lower-bound technique for bounded depth

We present a lower-bound argument based largely on the following observation.

Fact 5.14. *In a tidy refutation from a QBF whose first block is universal, all first block literals appearing in the derivation appear together in a single clause.*

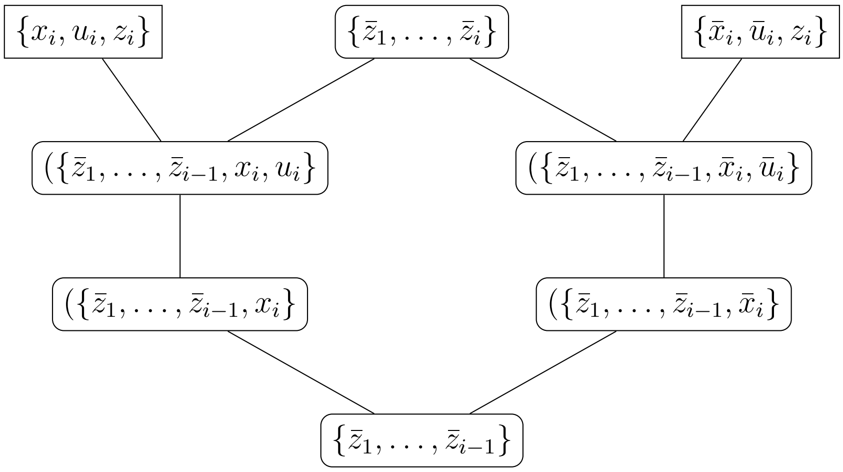


Figure 5.2: Portion of a linear size Q-Res refutation of EQ'_n.

Proof. Let $\pi := C_1, \ldots, C_k$ be a tidy refutation of a QBF whose first block U_1 is universal. Let $C_r := \{a_1, \ldots a_r\}$ be the first clause in π from which a first-block literal is reduced.

Since π is a tidy refutation, every universal literal occurring in it is reduced somewhere. Moreover, the subsequence $C_r, \ldots, C_k$ consists of the r reduction steps that remove the literals of C_r one by one. Hence all the first-block literals appear in C_r. $\square$

We also need to the use the fact that the universal subclauses appearing after the application of an existential assignment were also present beforehand.

Fact 5.15. *Given a* Q-Res *refutation π of a QBF, an assignment ε to a single existential variable, and a clause C in $\pi[[\varepsilon]]$, there exists a clause in π whose universal subclause is the same as that of C.*

Proof. Putting $\varepsilon := \bar{x}$, the application of an existential assignment (Definition 5.7) replaces each clause C in π with either $\mathbb{L}$ or $C \setminus \{x\}$. It is easy to see that the removal of weakening steps (proof of Fact 5.9) removes all occurrences of $\mathbb{L}$ and replaces each remaining clause D with a subset of D, while preserving the universal literals. $\square$

We are now ready to state and prove the central theorem of our lower-bound technique for Q-Res.

Theorem 5.16. *If a QBF with quantifier depth d has a* Q-Res *refutation of size k, then it has a countermodel of size k^d.*

Proof. Let π be a refutation of a QBF

$$Q := \forall U_1 \exists X_1 \cdots \forall U_d \exists X_d \cdot F$$

of size k. We show that the extracted strategy $h := \{h_j\}_{j\in[m]}$ for π has size at most k^d.

In fact, for each i in $[d]$ and each total existential assignment ε to Q, we show that the negation of the assignment

$$\phi(\varepsilon, U_i) := \{h_j(\varepsilon\restriction_{H_j}) : u_j \in U_i\}$$

is equal to

$$\{a \in C_\varepsilon : \mathrm{var}(a) \in U_i\} \diamond \{u_j : u_j \in U_i\} .$$

for some clause C_ε appearing in π. In other words, the projection to U_i of each element in the range of the extracted strategy is the negation of the projection to U_i of some clause in the refutation, padded with positive literals.

It follows that the assignment set

$$\{\phi(\varepsilon, U_i) : \varepsilon \in \langle \mathrm{vars}_\exists(Q)\rangle\}$$

has cardinality at most k. As each assignment in the range of h is equal to

$$\bigcup_{i\in[d]} \phi(\varepsilon, U_i)$$

for some ε, the size of the extracted strategy is at most k^d.

So, let i be an integer in $[d]$ and ε a total existential assignment. For each u_j in U_i, the dependency set for u_j is

$$X_1 \cup \cdots \cup X_{i-1} = \{x_1, \ldots, x_{n_i}\} ,$$

for some integer n_i; and looking at the definition of the extracted strategy, we see that $h_j(\varepsilon\restriction_{H_j}) = \{u_j\}$ if, and only if, $\bar{u}_j$ appears in the sequence

$$\pi_i := \pi[[\varepsilon\restriction_{\{x_1\}}]]\cdots[[\varepsilon\restriction_{\{x_{n_i}\}}]] \ .$$

By Fact 5.8, π_i is a tidy Q-Res refutation of $Q[\delta]$, a QBF whose first block is U_i. Hence, by Fact 5.14, there is some clause C_ε in π_i which satisfies the following for each u_j in U_i:

$$\bar{u}_j \text{ appears in } \pi_i \quad \Rightarrow \quad \bar{u}_j \in C_\varepsilon .$$

Hence, by Fact 5.15, the same condition is satisfied by some clause D_ε appearing in π. It follows that the negation of $\phi(\varepsilon, U_i)$ is the clause

$$\{a \in D_\varepsilon : \mathrm{var}(a) \in U_j\} \diamond \{u_j : u_j \in U_i\} . \qquad \square$$

The contrapositive statement of Theorem 5.16 describes Q-Res proof-size lower bounds in terms of countermodel size, with the strength of the lower bound decreasing with increasing quantifier depth.

Corollary 5.17. *If a d-bounded family of false QBFs requires countermodels of size $t(n)$, then it requires* Q-Res *refutations of size $\sqrt[d]{t(n)}$.*

Recall that a QBF family is d-bounded if every instance has at most d universal blocks.

5.3.3 Application and limitations

We can use our technique to prove that the equality formulas are hard for Q-Res. Since $\mathcal{EQ}$ is a 1-bounded family requiring countermodels of size 2^n (Theorem 4.14), applying Corollary 5.17 gives the following result.

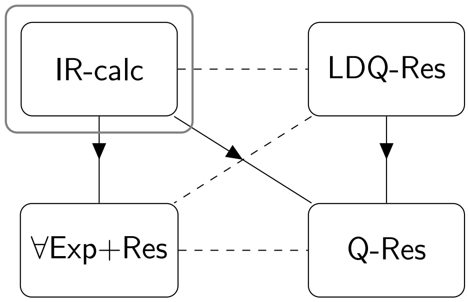
Theorem 5.18. *$\mathcal{EQ}$ requires* Q-Res *refutations of size 2^n.*

The technique also establishes the hardness of a large class of random QBFs in Q-Res [8].

But what about the family $\mathcal{KB}$? Corollary 5.17 doesn't offer much for this family, because their quantifier depth is unbounded. However, these formulas are hard for Q-Res, and our technique is not too far from showing it. In the next chapter, we refine the method to work with families of unbounded quantifier depth, and we even do so in the context of the stronger system IR-calc.

Chapter 6

Universal Instantiation



Quantified Boolean formulas are a decidable fragment of first-order logic. As such, solving techniques and proof systems from first-order logic are always applicable to QBF, often in a simplified form. Arguably the clearest example of a dedicated first-order paradigm that has found its way into QBF solving and proof complexity is *universal instantiation*.

Universal instantiation is a treatment of universal quantification that originates from first-order resolution, which in turn was the inspiration for the QBF proof system IR-calc [13], highlighted in the figure above. Like ∀Exp+Res, IR-calc is an expansion-based system that operates on annotated clauses.

The most prominent feature of IR-calc is that it simulates both ∀Exp+Res and Q-Res, and thus unifies to some extent the two major QBF paradigms, expansion and reduction. This means that lower bounds for IR-calc are already lower bounds for both ∀Exp+Res and Q-Res. However, since neither calculus simulates IR-calc, we should expect instantiation lower bounds to be harder to come by.

A lower bound technique for IR-calc

Our goal in this chapter is to extend our Q-Res technique (Corollary 5.17) to obtain proof-size lower bounds in IR-calc. Moreover, we want to overcome the inherent restriction to bounded quantifier depth.

It turns out that we can reuse the methodology of the previous chapter, provided we introduce a refined notion of countermodel, and a stricter measure called *weight*. Our main result is Theorem 6.19, which states that minimum countermodel weight is an IR-calc proof-size lower bound, regardless of quantifier depth. With these modifications, our technique is applicable to QBF families with unbounded quantifier depth, in particular, the Kleine Büning family. Our method provides a much simpler, more intuitive proof of hardness compared to the original (cf. [14]).

Organisation of the chapter

In Section 6.1, we give a high-level description of the main features of instantiation, followed by the formal presentation of IR-calc in Section 6.2. We revisit strategy extraction in Section 6.3, introduce refined countermodels, and define the weight measure. In Section 6.4, we present the improved lower-bound technique, and apply it to the $\mathcal{KB}$ family.

6.1 Instantiation versus expansion

Universal instantiation differs from expansion in three major ways. First, annotations are partial assignments to the dependency set of the underlying existential variable, as opposed to total assignments in ∀Exp+Res. Second, the allowable axiom clauses do not come from the total expansion, but from a rather different form of expansion of the input QBF, which exploits the use of partial annotations. Thirdly, the instantiation rule allows partial annotations to be enlarged over the course of the refutation.

Partial assignments as annotations

The first major difference between IR-calc and ∀Exp+Res lies in the size of the annotations. In ∀Exp+Res, every annotation is a total assignment to the dependency set of the underlying variable. In IR-calc, the annotations are partial assignments.

This means that set of annotated variables on which IR-calc operates is much larger. More precisely, the variable set available in an IR-calc derivation from a QBF

Q is

$$Z_Q^{\mathsf{IR}} := \{x_i^\mu : x_i \in \mathrm{vars}_\exists(Q), \mu \in \langle\langle S_i \rangle\rangle\}\,,$$

whereas the variables appearing in $\forall$Exp+Res derivations are the variables of the total expansion, all of which belong to the subset

$$\{x_i^\mu \in Z_Q^{\mathsf{IR}} : \mu \in \langle S_i \rangle\}\,.$$

The weak expansion of a QBF

The second major difference is the axiom rule. Axiom clauses in IR-calc are not taken from the total expansion of the QBF. Instead, they belong to a rather different CNF that we call the *weak expansion.* In the weak expansion, only individual variable assignments which actually falsify universal literals make their way into the annotations.

Definition 6.1 (weak expansion)**.** *Let $Q := P \cdot F$ be a QBF with existential dependency sets $S_1, \ldots, S_n$. The* weak expansion *of Q is the CNF*

$$\big\{C[\mu_C \cup \{x_i \mapsto (x_i, \mu_C{\restriction_{S_i}})\}_{i\in[n]}]\big\}_{C\in F}\,,$$

where μ_C is the negation of the universal subclause of C.

Example 6.2. Consider again the first instance of the equality family

$$\exists x_1 \forall u_1 \exists z_1 \cdot \{\{\bar{x}_1, \bar{u}_1, z_1\}, \{x_1, u_1, z_1\}, \{\bar{z}_1\}\}\,.$$

The weak expansion is the CNF

$$\{\{\bar{x}_1^{\emptyset}, z_1^{\{u_1\}}\}, \{x_1^{\emptyset}, z_1^{\{\bar{u}_1\}}\}, \{\bar{z}_1^{\emptyset}\}\}$$

Notice that the literal in the unit clause $\{\bar{z}_1\}$ receives the empty annotation, since the universal subclause is empty. In contrast, in the expansion of EQ_1 (Definition 4.2), it is always annotated with some assignment to u_1, which belongs to the dependency set for z_1.

Even though EQ_1 is false, its weak expansion is satisfiable. It is satisfied, for example, by the partial assignment $\{x_1 \mapsto 0, z_1^{\{\bar{u}_1\}} \mapsto 1, z_1^{\emptyset} \mapsto 0\}$. ■

Instantiation and completion

The third major difference between IR-calc and ∀Exp+Res is instantiation itself.

Instantiation is a means of extending annotations. As we will see, its purpose is to unify the annotations of prospective pivot literals, which, being partial assignments, may be consistent but not equal. In order for a resolution step to be valid, the pivot literals must be complementary, meaning that their annotations must match exactly. It is perhaps worth emphasising that literals whose annotations do not match are literals in distinct variables.

The *instantiation* of an annotated clause C by a universal assignment μ with respect to a prefix P is the clause

$$\operatorname{inst}(C, \mu, P) := C[\{(x_i, \nu) \mapsto (x_i, \nu \diamond \mu\restriction_{S_i}) : (x_i, \nu) \in \operatorname{vars}(C)\}] \ ,$$

where, as usual, S_i is the dependency set for the existential variable x_i.

As illustrated in the following example, due to the definition of completion, instantiation never overwrites the assignments in annotations, it only extends them.

Example 6.3. With respect to the prefix $\forall u_1 \exists x_1 \forall u_2 \exists x_2$, the instantiation of the annotated clause

$$C := \{(x_1, \{\emptyset\}), (x_2, \{u_1\})\}$$

by the universal assignment $\mu := \{\bar{u}_1, \bar{u}_2\}$ is the annotated clause

$$\operatorname{inst}(C, \mu, P) := \{(x_1, \{\bar{u}_1\}), (x_2, \{u_1, \bar{u}_2\})\}$$

Notice that $\{u_1\} \diamond \mu = \{u_1, \bar{u}_2\}$, so the literal $(x_2, \{u_1\})$ in C, which is already annotated with the positive assignment to u_1, does not have that assignment overwritten by the complementary assignment, which appears in μ. The individual assignments in the annotations of the consequent clause are always preserved in this way by an instantiation. ■

6.2 The proof system IR-calc

We are now ready to set out the definition.

Definition 6.4 (IR-calc [13]). *An* IR-calc *derivation from a QBF $Q := P \cdot F$ is a sequence $C_1, \ldots, C_k$ clauses in which at least one of the following holds for each $i \in [k]$:*

A Axiom: *C_i is a clause in the weak expansion of Q;*

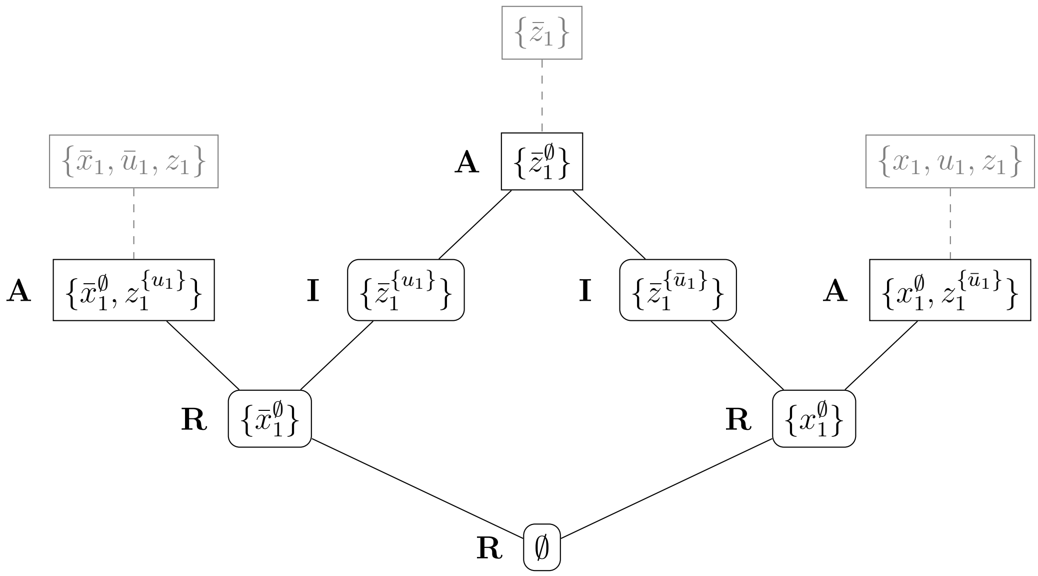


Figure 6.1: An IR-calc refutation of EQ_1.

R Resolution: $C_i = \mathrm{res}(C_r, C_s, p)$, *for some* $r, s < i$ *and existential literal* p*;*

I Instantiation: $C_i = \mathrm{inst}(C_r, \mu, P)$, *for some* $r < i$ *and universal assignment* μ*;*

W Weakening: C_i *is* $\mathbb{L}$*, or is subsumed by* C_r *for some* $r < i$*.*

As usual, the final clause of a derivation is called its conclusion, and a derivation whose conclusion is the empty clause is called a refutation.

The set of *download clauses* of a derivation π is the unique subset G of the matrix for which the axioms of π are the weak expansion of $P \cdot G$. More precisely, the download clauses are the subset of axioms C satisfying

$$C \quad = \quad D[\mu_D \cup \{x_i \mapsto (x_i, \mu_C\restriction_{S_i})\}_{i\in[n]}]\,,$$

for some clause D in F, where μ_D is the negation of the universal subclause of D.

Example 6.5. Figure 6.1 shows an IR-calc refutation of the first instance of the equality family. The unit clause $\{\bar{z}_1^{\emptyset}\}$, which belongs to the weak expansion of EQ_1, is introduced as an axiom and subsequently instantiated in both possible ways, i.e. by the assignments $\{u_1\}$ and $\{\bar{u}_1\}$. The download clauses, which in this case form the complete matrix of the input QBF, are depicted connected to the corresponding axioms with dotted lines. ■

Soundness

We can prove that IR-calc is sound via a translation into $\forall$Exp+Res, which we already know is a sound refutational QBF proof system. Since $\forall$Exp+Res does not simulate IR-calc, any such translation must incur a superpolynomial proof-size inflation. However, when soundness is all we seek, the complexity of the translation is irrelevant.

Lemma 6.6 ([13]). *A QBF is false if it has an* IR-calc *refutation.*

Proof. Let $\pi := C_1, \ldots, C_k$ be an IR-calc refutation of a QBF $Q := P \cdot F$.

By induction on $i \in [k]$, we show that, for each total universal assignment μ in $\langle \text{vars}_\forall(Q) \rangle$, there exists a $\forall$Exp+Res derivation of $\text{inst}(C_i, \mu, P)$ from Q. Since instantiation has no effect on the empty clause, at the final step $i = k$, we show that there exists an $\forall$Exp+Res refutation of Q. So Q is false by the soundness of $\forall$Exp+Res (Theorem 4.9).

Now, let μ be an arbitrary total universal assignment. For the base case $i = 1$, C_1 is a clause in the weak expansion of Q. It is easy to see that $\text{inst}(C_1, \mu, P)$ is a clause in the total expansion of Q, which can be introduced as an axiom in an $\forall$Exp+Res derivation from Q. For the inductive step, let $i \geq 2$. We consider four cases.

A If C_i was introduced as an axiom, the inductive step is identical to the base case.

R Suppose that C_i was derived by resolution from C_r and C_s over the existential pivot literal p^ν. It is easy to see that

$$\text{inst}(C_i, \mu, P) \quad = \quad \text{res}(\text{inst}(C_r, \mu, P), \text{inst}(C_s, \mu, P), (p, \nu \diamond \mu))\,.$$

Since both $\text{inst}(C_r, \mu, P)$ and $\text{inst}(C_s, \mu, P)$ can be derived in $\forall$Exp+Res by the inductive hypothesis, $\text{inst}(C_i, \mu, P)$ can be derived from them by resolution.

I If C_j was derived by instantiation, say by applying the assignment ν to C_r, then $\text{inst}(C_i, \mu, P)$, which is equal to $\text{inst}(C_r, \nu \diamond \mu, P)$, can be derived in $\forall$Exp+Res, by the inductive hypothesis.

W If C_j was derived by weakening from C_r, then $\text{inst}(C_r, \mu, P)$ can be derived in $\forall$Exp+Res by the inductive hypothesis, so

$$\text{inst}(C_i, \mu, P) \quad = \quad \text{inst}(C_r, \mu, P) \cup \text{inst}(C_i \setminus C_r, \mu, P)$$

can be derived from it by weakening. $\square$

Theorem 6.7 ([13])**.** IR-calc *is a proof system for the language* FQBF*.*

Proof. Soundness. Established by Lemma 6.6. *Completeness.* Follows from the completeness of $\forall$Exp+Res which is trivially p-simulated by IR-calc. To see this, it is enough to observe that every clause in the total expansion of a QBF can be derived from some clause in the weak expansion by a single instantiation. *Checkability.* It can be checked efficiently whether a clause belongs to the weak expansion of a QBF. Moreover, instantiation can also be checked efficiently, so checkability of IR-calc follows from that of Res. ☐

6.3 Extracting strategies from IR-calc refutations

Now we turn to the task of extracting strategies from IR-calc refutations, which, as in the previous chapter, forms the basis of our lower-bound technique. Extraction of strategies from IR-calc refutations was already shown in [13], along similar lines as for Q-Res [28]. We also follow the same approach, but once again we handle the details slightly differently, and for that reason we include all the details.

6.3.1 Existential assignments and tidy refutations

Much of the technical details here follow similar lines to those of the previous chapter, but, due to the use of annotated literals, are sufficiently different to merit full proofs. We deal first with the application of existential assignments.

Closure under existential assignments

Due to the nature of annotated clauses, applying existential assignments to IR-calc refutations is not as straightforward as in Q-Res.

What we want to show is the analogue of Fact 5.8, namely that IR-calc is closed under existential assignments. However, when we come to apply the existential assignment, we notice that the variables appearing in the annotated clauses are completely disjoint from those of the input QBF. To deal with this, we must translate the assignment to the QBF into a corresponding assignment to the annotated variables.

Definition 6.8. *Given a QBF Q and a partial existential assignment ε, the application of ε to an* IR-calc *refutation $\pi := C_1, \dots, C_k$ of Q returns the sequence*

$$\pi[\varepsilon] := C_1[\delta], \dots, C_k[\delta] \,,$$

where δ is the assignment to the variable set

$$Z_{\varepsilon}^{\mathsf{IR}} := \{x_i^{\mu} : x_i \in \mathrm{vars}(\varepsilon), \mu \in \langle\langle S_i \rangle\rangle\}$$

defined by

$$\begin{array}{rccc} \delta & : & Z_{\varepsilon}^{\mathsf{IR}} & \to & \mathbb{D} \\ & & x_i^{\mu} & \mapsto & \varepsilon(x_i)\,. \end{array}$$

This assignment translation, which basically just ignores the annotations, indeed gives rise to the closure we seek.

Fact 6.9 ([13])**.** *Given an* IR-calc *refutation of a QBF Q and a partial existential assignment ε, $\pi[\varepsilon]$ is an* IR-calc *refutation of $Q[\varepsilon]$.*

Proof. Let $\pi := C_1, \ldots, C_k$, and let δ be the assignment given in Definition 6.8. We show by induction on $i \in [k]$ that each clause $C_i[\delta]$ is a valid Q-Res inference in $\pi[\varepsilon]$.

Observe that, if δ satisfies C_i, then $C_i[\delta]$ is $\mathbb{L}$ and can be derived by weakening. Hence, we can assume from now on that δ does not satisfy C_i.

For the base case $i = 1$, C_1 is introduced as an axiom and belongs to the weak expansion of F. It is easy to verify that $C_1[\delta]$ belongs to the weak expansion of $F[\varepsilon]$, so $C_1[\delta]$ can be introduced as an axiom.

For the inductive step, let $i \geq 2$. We consider four cases.

A If C_i was introduced as an axiom, the inductive step is identical to the base case.

R If C_i was derived by resolution from C_r and C_s over the existential pivot literal p^{μ}, we consider three further cases.

(i) If δ satisfies the pivot literal p^{μ}, then $C_i[\delta]$ is subsumed by $C_s[\delta]$, and can therefore be derived by weakening.

(ii) If δ falsifies p^{μ}, then $C_i[\varepsilon]$ is subsumed by $C_r[\delta]$, and can be derived similarly by weakening.

(iii) If δ neither satisfies nor falsifies p^{μ}, then, since δ satisfies neither C_r nor C_s, $C_i[\delta]$ can be derived by resolution from $C_r[\delta]$ and $C_s[\delta]$ over pivot literal p^{μ}.

I If C_i was derived by instantiation, say by applying the assignment ν to C_r, then δ does not satisfy C_r, and $C_i[\delta]$ can be derived by instantation, applying the same assignment ν to $C_r[\delta]$.

W If C_i was derived by weakening from C_r, then δ does not satisfy C_r, which subsumes C_i. It is easy to see that $C_r[\delta]$ subsumes $C_i[\delta]$, so the latter can be derived by weakening. $\square$

Removing weakening steps

Removing weakening steps algorithmically from IR-calc refutations is very similar to the corresponding process for Q-Res. In fact, we only need extend the construction to handle instantiation steps, and this is quite straightforward.

Fact 6.10 (folklore)**.** *Weakening inferences can be removed algorithmically from* IR-calc *refutations with no increase in size, while preserving the refutation.*

Proof. Let $\pi := C_1, \ldots, C_k$ be a Q-Res refutation of a QBF $Q := P \cdot F$.

Since $\mathbb{L}$ cannot be an antecedent of any inference in π, and the conclusion C_k is not $\mathbb{L}$, deleting instances of $\mathbb{L}$ preserves the refutation. Therefore we can assume without loss of generality that $\mathbb{L}$ does not occur in π.

Now, we transform π into a weakening-free refutation $\pi' := C'_1, \ldots, C'_k$ by processing the clauses C_i in order, as follows:

A if C_i was introduced as an axiom, then define $C'_i := C_i$;

R if C_i was derived by resolution from C_r and C_s over pivot p^μ, then define

$$C'_i := \begin{cases} C'_r & \text{if } p^\mu \notin C'_r\,, \\ C'_s & \text{if } p^\mu \in C'_r \text{ and } \tilde{p}^\mu \notin C'_s\,, \\ \mathrm{res}(C'_r, C'_s, p^\mu) & \text{if } p^\mu \in C'_r \text{ and } \tilde{p}^\mu \in C'_s\,; \end{cases}$$

I If C_i was derived by instantiation, say by applying the assignment ν to C_r, then define $C'_i := \mathrm{inst}(C'_r, \nu, P)$.

W If C_i was derived by weakening from C_r, then define $C'_i := C'_r$.

It is clear that the size of π' is equal to that of π, and that any annotation appearing in π' also appears in π. Hence, to conclude, we show by induction on i in $[k]$ that C'_i is a subset of C_i, and is the consequent of a valid non-weakening inference in π'. The base case $i = 1$ is established trivially, since $C'_1 = C_1$ is a clause in the weak expansion of Q. For the inductive step, let $i \geq 2$. We consider four cases.

A If C_i was introduced as an axiom, the inductive step is identical to the base case.

R If C_i was derived by resolution we consider three further cases.

(i) If $p^\mu \notin C'_r$, then $C'_i = C'_r$ subsumes C_i, and can be derived by a non-weakening inference by the inductive hypothesis.

(ii) If $p^\mu \in C'_r$ and $\tilde{p}^\mu \notin C'_s$, then $C'_i = C'_s$ subsumes C_i, and can be derived by a non-weakening inference by the inductive hypothesis.

(iii) If $p^\mu \in C'_r$ and $\tilde{p}^\mu \in C'_s$, then $C'_i = \mathrm{res}(C'_r, C'_s, p^\mu)$. So C'_i is a valid resolution inference in π', and

$$C'_i = (C'_r \setminus \{p^\mu\}) \cup (C'_s \setminus \{\tilde{p}^\mu\}) \subseteq (C_r \setminus \{p^\mu\}) \cup (C_s \setminus \{\tilde{p}^\mu\}) = C_i$$

holds by the inductive hypothesis.

I If C_i was derived by instantiation, then C'_r is a subset of C_r by the inductive hypothesis. Moreover, $C'_i = \mathrm{inst}(C'_r, \nu, P)$ is trivially a valid inference in π'.

W If C_i was derived by weakening, then C_i is a subsumed by C_r, and

$$C'_i = C'_r \subseteq C_r \subseteq C_i \,,$$

by the inductive hypothesis. Moreover, $C'_i = C'_r$ is a valid non-weakening inference, by the inductive hypothesis. □

Tidy refutations

Similar to the nomenclature of the previous chapter, we call an IR-calc refutation *conclusion-unique* when there is exactly one clause in the sequence which is not the antecedent of an inference. We call a refutation *tidy* when it is both conclusion-unique and weakening-free.

Further, we call a refutation *non-trivial* when at least one of its clauses is non-empty. It is easy to see that a tidy non-trivial refutation has at least one application of resolution, and therefore has at least three clauses. In a non-trivial IR-calc refutation, we call the annotation of the final pivot variable the *final annotation.*

Example 6.11. The refutation in Figure 6.1 is non-trivial, and its final annotation is the empty assignment. ■

In tidy IR-calc refutations, first block universal variables behave in a particular way, analogous to Q-Res, insofar as they appear in at most one polarity amongst the annotations. In fact, first-block universal literals accumulate as the proof progresses, and in non-trivial tidy refutations, all such literals can be found together in the final annotation.

Later on, when we come to strategy extraction, the important first-block assignments will actually be those that appear in the annotations of axiom clauses (those introduced by instantiation can essentially be ignored) and – for non-trivial refutations – these are exactly the complements of those appearing in the download clauses.

Fact 6.12 ([13]). *Let π be a tidy* IR-calc *refutation of a QBF whose first block U is universal, and let μ be the set of literals in variables from U whose complements appear in the download clauses. If π is non-trivial, then the final annotation includes μ.*

Proof. Let $\pi := C_1, \ldots, C_k$ be the refutation. For each i in $[k]$, we let μ_i be the set of literals in variables from U appearing in the download clauses for the subderivation of C_i.

By induction on i in $[k]$, we show that there exists an assignment ν_i for which

(a) $\mu_i \subseteq \nu_i$, and

(b) for each annotation ζ appearing in C_i, $\zeta\restriction_U = \nu_i$.

Since π is tidy, the final annotation appears in both C_{k-2} and C_{k-1}, and the combined download clauses of their subderivations are the download clauses for π. Hence the final annotation includes μ.

For the base case $i = 1$, C_1 is an axiom. Since every variable in U belongs to the dependency set of every existential variable, the restriction to U of every annotation in C_1 is μ_1. Hence putting $\nu_1 := \mu_1$ satisfies conditions (a) and (b).

For the inductive step, let $i \geq 2$. Since π is tidy, C_i is not the consequent of a weakening step. So, we consider three cases.

A If C_i was introduced as an axiom, the inductive step is identical to the base case.

R Suppose that C_i was derived by resolution from C_r and C_s over pivot literal p^{ζ_0}. By the inductive hypothesis, there exist assignments ν_r and ν_s satisfying conditions (a) and (b) with respect to r and s. Since ζ_0 appears as an annotation in both C_r and C_s, we must have $\nu_r = \nu_s$. Thus, setting $\nu_i := \nu_r$ satisfies both conditions (a) and (b) with respect to i.

I Suppose that C_i was derived by instantiation from C_r and the universal assignment θ. By the inductive hypothesis, there exists an assignment ν_r satisfying conditions (a) and (b) with respect to r. It is easy to verify that the assignment $\nu_i := \nu_r \diamond \theta$ satisfies both conditions with respect to i. $\square$

Fact 6.12 is essentially the analogue of Facts 5.10 and 5.14 for Q-Res refutations. An immediate corollary is that the first-block universal literals from the download clauses contain no complementary pairs, and therefore form a partial assignment to the first block.

Corollary 6.13. *Given a tidy* IR-calc *refutation π of a QBF whose first block U is universal, complementary literals in variables in U do not appear amongst the download clauses of π.*

6.3.2 Refined countermodels

We are working towards extending the lower bound technique for Q-Res (Chapter 5) to IR-calc, in such a way that we can prove lower bounds for formula families of unbounded quantifier depth. For this, it turns out that we need to refine our definition of countermodel.

The refinement is based on two observations. Given a countermodel h for a QBF

$$\forall U_1 \exists X_1 \cdots \forall U_d \exists X_d \cdot F\,,$$

and some total existential assignment ε,

(a) the assignment $h(\varepsilon)$ does not depend on $\varepsilon\restriction_{X_d}$, and

(b) the assignment $\varepsilon \cup h(\varepsilon)$ may falsify the matrix even when $h(\varepsilon)$ is restricted to a partial universal assignment.

Definition 6.14 (refined countermodel)**.** *A* refined countermodel *for a QBF*

$$\forall U_1 \exists X_1 \cdots \forall U_d \exists X_d \cdot F$$

is a function with signature

$$h \quad : \quad \langle \mathrm{vars}_\exists(Q) \setminus X_d \rangle \quad \rightarrow \quad \langle\langle \mathrm{vars}_\forall(Q) \rangle\rangle$$

satisfying the following conditions for each ε, δ in $\langle \mathrm{vars}_\exists(Q) \setminus X_d \rangle$:

(a) $F[\varepsilon \cup h(\varepsilon)]$ *is unsatisfiable;*

(b) *for each j in $[d]$ and each universal variable u in U_j,*

$$\varepsilon \textit{ and } \delta \textit{ agree on } X_1 \cup \cdots \cup X_{j-1} \quad \Rightarrow \quad h(\varepsilon) \textit{ and } h(\delta) \textit{ do not disagree on } u_j\,.$$

Weight of a refined countermodel

A set of assignments is called pairwise-inconsistent when every pair of assignments disagrees on at least one variable. The *weight* of a refined countermodel is the maximal cardinality of a pairwise-inconsistent subset of its range.

We emphasise that, in contrast to countermodels, the range of a refined countermodel is a set of partial universal assignments, so non-identical elements of the range are not necessarily inconsistent.

Example 6.15. The first instance of the interleaved equality family, namely

$$\exists x_1 \forall u_1 \exists x_1 \cdot \{\{\bar{x}_1, \bar{u}_1, z_1\}, \{x_1, u_1, z_1\}, \{\bar{z}_1\}\}$$

has the unique refined countermodel

$$\begin{array}{lll} \langle\{x_1\}\rangle & \to & \langle\langle\{u_1\}\rangle\rangle \\ \{\bar{x}_1\} & \mapsto & \{\bar{u}_1\} \\ \{x_1\} & \mapsto & \{u_1\}\,, \end{array}$$

whose weight is 2, since the domain itself consists of two inconsistent assignments. In this case, each element of the range of the refined countermodel is actually a total universal assignment.

As a further example demonstrating the use of partial assignments, the reader can verify that the second instance, namely

$$\exists x_1 \forall u_1 \exists z_1 \exists x_2 \forall u_2 \exists z_2 \cdot \{\{\bar{x}_1, \bar{u}_1, z_1\}, \{x_1, u_1, z_1\}, \{\bar{x}_2, \bar{u}_2, z_2\}, \{x_2, u_2, z_2\}, \{\bar{z}_1, \bar{z}_1\}\}$$

has the following (non-unique) refined countermodel:

$$\begin{array}{lll} \langle\{x_1, y_1, x_2\}\rangle & \to & \langle\langle\{u_1, u_2\}\rangle\rangle \\ \{\bar{x}_1, \bar{z}_1, \bar{x}_2\} & \mapsto & \{\bar{u}_1\} \\ \{\bar{x}_1, \bar{z}_1, x_2\} & \mapsto & \{\bar{u}_1\} \\ \{\bar{x}_1, z_1, \bar{x}_2\} & \mapsto & \{\bar{u}_2\} \\ \{\bar{x}_1, z_1, x_2\} & \mapsto & \{u_2\} \\ \{x_1, \bar{z}_1, \bar{x}_2\} & \mapsto & \{u_1\} \\ \{x_1, \bar{z}_1, x_2\} & \mapsto & \{u_1\} \\ \{x_1, z_1, \bar{x}_2\} & \mapsto & \{\bar{u}_2\} \\ \{x_1, z_1, x_2\} & \mapsto & \{u_2\}\,. \end{array}$$

The only pairwise-inconsistent susbsets of the range of this countermodel are

$$\{\{\bar{u}_1\}, \{u_1\}\} \quad \text{and} \quad \{\{\bar{u}_2\}, \{u_2\}\}\,,$$

and hence its weight is also 2. Since EQ′ requires exponential-size countermodels (Theorem 4.16), this example demonstrates that a QBF can have a refined countermodel whose weight is less than the minimal countermodel size. ■

Relation to countermodels

Like regular countermodels, refined countermodels also witness the falsity of a QBF. In fact, it is fairly easy to translate between the two.

For example, consider an arbitrary false QBF

$$\forall U_1 \exists X_1 \cdots \forall U_d \exists X_d \cdot F$$

with existential variables $\{x_1, \ldots, x_n\}$. Given a countermodel h for Q, any function defined by

$$\begin{array}{rcl} \langle \text{vars}_\exists(Q) \setminus X_d \rangle & \to & \langle\langle \text{vars}_\forall(Q) \rangle\rangle \\ \varepsilon & \mapsto & h(\varepsilon \cup \alpha)\,, \end{array}$$

where α is an arbitrary total assignment to X_d, is a refined countermodel for Q.

On the other hand, given a refined countermodel h' for Q, it is easy to see that the set of universal dependency functions $\{h_i\}_{i\in[n]}$ defined by

$$\begin{array}{rcccl} h_i & : & \langle H_i \rangle & \to & \langle \{u_i\} \rangle \\ & & \delta & \mapsto & \begin{cases} (h'(\delta \diamond \beta))\restriction_{\{u_i\}} & \text{if } u_i \in \text{vars}(h(\delta \diamond \beta)) \\ \{\bar{u}_i\} & \text{otherwise}\,. \end{cases} \end{array} .$$

where β is the total existential assignment that is identically 0, forms a countermodel for Q.

6.3.3 The extracted strategy

Now we show how to extract a refined countermodel from an IR-calc refutation. We first define the extracted strategy, and then prove that it is indeed a refined countermodel.

Definition 6.16 (extracted strategy)**.** *Given an* IR-calc *refutation* π *of a QBF*

$$\forall U_1 \exists X_1 \cdots \forall U_d \exists X_d \cdot F\,,$$

the extracted strategy for π *is the function*

$$\begin{array}{rcccl} h & : & \langle \text{vars}_\exists(Q) \setminus X_d \rangle & \to & \langle\langle \text{vars}_\forall(Q) \rangle\rangle \\ & & \varepsilon & \mapsto & \mu_\varepsilon \end{array}$$

where μ_ε *is the set of universal literals whose complements appear in the download clauses of*

$$\pi[[\varepsilon\restriction_{X_1}]] \cdots [[\varepsilon\restriction_{X_{d-1}}]]\,.$$

Theorem 6.17. *The extracted strategy for an* IR-calc *refutation is a refined countermodel for the input QBF.*

Proof. Let π be an IR-calc refutation of a QBF Q. We show that the extracted strategy

$$\begin{array}{rcrcl} h & : & \langle \text{vars}_\exists(Q) \setminus X_d \rangle & \to & \langle\langle \text{vars}_\forall(Q) \rangle\rangle \\ & & \varepsilon & \mapsto & \mu_\varepsilon \end{array}$$

satisfies both conditions for a refined countermodel (Definition 6.14).

Let ε and δ be arbitrary assignments in $\langle \text{vars}_\exists(Q) \setminus X_d \rangle$.

(a) By Facts 6.9 and 6.10,

$$\pi_\varepsilon := \pi[[\varepsilon\restriction_{X_1}]] \cdots [[\varepsilon\restriction_{X_{d-1}}]]$$

is an IR-calc refutation of

$$\forall (U_1 \cup \cdots \cup U_d) \exists X_d \cdot F[\varepsilon] \, ,$$

which is a false QBF, by the soundness of IR-calc (Lemma 6.6).

Let G be the download clauses of π_ε. By definition of extracted strategy (Definition 6.16), μ_ε contains the complement of each universal literal occurring in G. It is easy to see, therefore, that the false QBF

$$\forall (U_1 \cup \cdots \cup U_d) \exists X_d \cdot G$$

has a model if $G[\mu_\varepsilon]$ is satisfiable, and hence it is unsatisfiable. It follows immediately that $F[\varepsilon \cup \mu_\varepsilon]$ is unsatisfiable.

(b) Let $j \in [d]$ and $u \in U_j$, and suppose that ε and δ agree on $X_1 \cup \cdots \cup X_{j-1}$. By Facts 6.9 and 6.10,

$$\pi'_\varepsilon := \pi[[\varepsilon\restriction_{X_1}]] \cdots [[\varepsilon\restriction_{X_{j-1}}]]$$

is an IR-calc refutation of

$$Q[[\varepsilon\restriction_{X_1}]] \cdots [[\varepsilon\restriction_{X_{j-1}}]] \, .$$

Now, by Corollary 6.13, u appears in at most one polarity in the download clauses for π'_ε. Note that the download clauses of both π_ε and

$$\pi_\delta := \pi[[\delta\restriction_{X_1}]] \cdots [[\delta\restriction_{X_{d-1}}]]$$

are obtained from the download clauses of π'_ε by the application of existential assignments, so u appears in at most one polarity amongst their combined download clauses. Therefore $h(\varepsilon)$ and $h(\delta)$ do not disagree on u, by definition of the extracted strategy (Definition 6.16). $\square$

6.4 Lower bounds in IR-calc

We are now ready to show that IR-calc proof size is related to weight. More precisely, the minimal refined countermodel weight is an IR-calc proof-size lower bound (Theorem 6.18). Thereafter we reprove the exponential IR-calc lower bound for the Kleine Büning family, by showing that it requires refined countermodels of exponential weight.

6.4.1 Extending the technique to unbounded quantifier depth

Our lower-bound technique for IR-calc rests on the following theorem.

Theorem 6.18. *If a QBF has an* IR-calc *refutation of size k, then it has a refined countermodel of weight k.*

Proof. Let π be an IR-calc refutation of a QBF

$$Q := \forall U_1 \exists X_1 \cdots \forall U_d \exists X_d \cdot F \,,$$

and let k be the size of π. First, we observe that trailing literals can be removed from a QBF while preserving its weak expansion, and therefore its set of IR-calc refutations. Hence, we may assume without loss of generality that the clauses in the matrix of Q contain no trailing literals.

Let s be the size of the extracted strategy h for π. We show that s is at most k. We hence prove the result, since the extracted strategy is a refined countermodel by Theorem 6.17.

We claim that every assignment in the range of h appears as a subset of an annotation in π. As the range of h contains a subset of s pairwise-inconsistent assignments, π contains at least s annotations, and, therefore, at least as many literals. Thus $s \leq k$.

It remains to prove the claim. We take an arbitrary assignment $\mu = h(\varepsilon)$ in the range of h. By definition of the extracted strategy (Definition 6.16), μ is the set of universal literals whose complements appear in the download clauses for

$$\pi_\varepsilon := \pi[[\varepsilon\restriction_{X_1}]] \cdots [[\varepsilon\restriction_{X_{d-1}}]] \,,$$

which is a refutation of the QBF

$$Q_\varepsilon := \forall (U_1 \cup \cdots \cup U_d) \exists X_d \cdot F[\varepsilon] \,.$$

We consider two cases

(a) Suppose that $|\pi_\varepsilon| = 1$. Then π_ε is the sequence consisting of the empty clause only, and has exactly one download clause, which is the negation of μ. It follows that π has a download clause D which, for some restriction δ of ε, is the negation of $\delta \cup \mu$. Since D contains no trailing literals, μ appears as a subset of some annotation appearing in the axiom corresponding to D.

(b) On the other hand, suppose that $|\pi_\varepsilon| > 1$. Since Q_ε has a single universal block, μ appears as a subset of the final annotation in π_ε, by Fact 6.12. Since neither the application of existential assignments nor the removal of weakening steps enlarge the annotation set of a refutation, μ appears as a subset of an annotation in π. $\square$

An immediate corollary of Theorem 6.18 is the following.

Corollary 6.19. *If a family of false QBFs requires refined countermodels of size $t(n)$, then it requires* Q-Res *refutations of size $t(n)$.*

6.4.2 Application to the Kleine Büning family

Finally, we apply the technique to $\mathcal{KB}$. We first show that the family requires refined countermodels of exponential size.

Theorem 6.20. *$\mathcal{KB}$ requires refined countermodels of size 2^n.*

Proof. Let n be a natural number, and Let

$$h \quad : \quad \langle\{x_1, y_1, \dots, x_n, y_n\}\rangle \quad \to \quad \langle\langle\{u_1, \dots, u_n\}\rangle\rangle$$

be a refined countermodel for EQ_n. We show that $\mathrm{rng}(h) = \langle\{u_1, \dots, u_n\}\rangle$, which is itself a set of 2^n pairwise-inconsistent assignments. It follows that the size of h is 2^n.

Let μ be an arbitrary total universal assignment, and let ε be the assignment in the domain of h defined by

$$\begin{aligned} \varepsilon(x_i) \quad &:= \quad \begin{cases} 0 & \text{if } \mu(u_i) = 1 \\ 1 & \text{if } \mu(u_i) = 0 \end{cases}, \quad \text{for } i \in [n]\,, \\ \varepsilon(y_i) \quad &:= \quad \begin{cases} 1 & \text{if } \mu(u_i) = 1 \\ 0 & \text{if } \mu(u_i) = 0 \end{cases}, \quad \text{for } i \in [n]\,. \end{aligned}$$

We complete the proof by showing that $\mu = h(\varepsilon)$.

Now, $\mathrm{kb}_n[\varepsilon]$ is the CNF consisting of the clauses

$$\begin{array}{ll} \{a_n, \bar{z}_1, \dots, \bar{z}_n\}\,, & \\ \{u_i, z_i\}\,, & \text{for } i \text{ in } [n]\,, \\ \{\bar{u}_i, z_i\}\,, & \text{for } i \text{ in } [n]\,, \end{array}$$

where a_n is the unique literal in the variable u_n falsified by μ. It is easy to see that a partial universal assignment leaves $\mathrm{kb}_n[\varepsilon]$ unsatisfiable only if it is a total universal assignment falsifying a_n. Therefore $h(\varepsilon)$ belongs to $\langle\{u_1, \dots, u_n\}\rangle$ and agrees with μ on u_n, by definition of refined countermodel (Definition 6.14), condition (a).

Finally, we show that $h(\varepsilon)(u_i) = \mu(u_i)$ for each i in $[n-1]$. To see this, let $i \in [n-1]$, and consider the assignment in the domain of h specified by

$$\delta_i(x_j) := \begin{cases} \varepsilon(x_j) & \text{if } j < i \\ 1 & \text{otherwise}\,, \end{cases}$$

$$\delta_i(y_j) := \begin{cases} \varepsilon(y_j) & \text{if } j < i \\ 1 & \text{otherwise}\,. \end{cases}$$

Now, $\mathrm{kb}_n[\delta_i]$ is the CNF consisting of the clauses

$$\begin{array}{ll} \{a_i\}\,, & \\ \{u_i, z_i\}\,, & \text{for } i \text{ in } [n]\,, \\ \{\bar{u}_i, z_i\}\,, & \text{for } i \text{ in } [n]\,, \end{array}$$

where a_i is the unique literal in the variable u_i falsified by μ. Similar to the above, a partial universal assignment leaves $\mathrm{kb}_n[\delta_i]$ unsatisfiable only if it falsifies a_i. Hence $h(\delta_i)$ agrees with μ on u_i, by definition of refined countermodel (Definition 6.14), condition (a). Since ε and δ agree on the dependency set for u_i, namely

$$\{x_1, y_1, \dots, x_{i-1}, y_{i-1}\}\,,$$

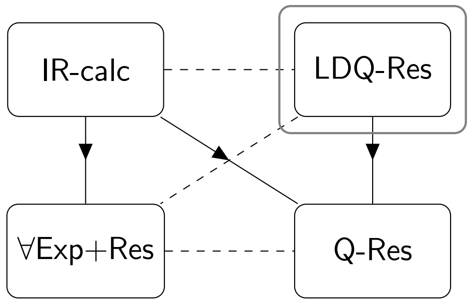
$h(\varepsilon)$ agrees with μ on u_i, by definition of refined countermodel (Definition 6.14), condition (b). $\square$

The exponential proof-size lower bound follows immediately.

Theorem 6.21 ([13]). *$\mathcal{KB}$ requires* IR-calc *refutations of size* 2^n.

Chapter 7

Universal Merging



The concept of *universal merging* originated in the learning mechanism of the QDCDL solver Quaffle [72]. Notwithstanding the difficulties posed by tautological clauses in Q-Res, the authors of Quaffle realised that certain tautological clauses are in fact harmless. They devised a learning scheme based on so-called 'long-distance' resolution, which works with tautologies in a non-trivial way. Zhang and Malik didn't give a semantic account of these tautological clauses, and nor were they expected to – they were mainly interested in efficient solving.

A decade later, a theoretical model for solvers like Quaffle was proposed in the shape of the QBF proof system LDQ-Res [2], highlighted above. The system operates rather like Q-Res, except that tautological clauses may be derived under safe conditions. It was subsequently shown that LDQ-Res is exponentially stronger than Q-Res [23], corroborating the practical appeal of the approach.

Deferred universal reduction

The addition of long-distance resolution gives rise to an interesting normal form for refutations, first introduced by Bjørner, Janota, and Klieber to model their solver GhostQ [16]. In this normal form, all universal reduction steps are carried out at the end of the refutation, after all the resolution steps. Such structured refutations form the fragment that we call *deferred* LDQ-Res (it is elsewhere referred to as 'reductionless LDQ-Res' [11] and 'Q^w' [16]).

In this chapter, we prove that deferred LDQ-Res is complete. Whereas our lower-bound techniques do not seem to be applicable to LDQ-Res, we are able prove an exponential proof-size lower bound in deferred LDQ-Res with a creative modification of the equality formulas.

Organisation of the chapter

We recall LDQ-Res in Section 7.1, and prove that the deferred fragment is complete in Section 7.2. Hardness of the squared equality family is proved in Section 7.3.

7.1 The proof system LDQ-Res

The rules of LDQ-Res are almost identical to those of Q-Res. The only difference is that the ban on tautological clauses is lifted, and a new side condition appears in the resolution rule.

Definition 7.1 (LDQ-Res [2])**.** *A* long-distance Q-Resolution (LDQ-Res) *derivation from a QBF $Q := P \cdot F$ is a sequence $C_1, \dots, C_k$ of clauses in which at least one of the following holds for each $i \in [k]$:*

- **A Axiom**: *C_i is a clause in F;*
- **L Long-distance resolution**: *$C_i = \text{res}(C_r, C_s, p)$, for some $r, s < i$ and existential literal p, $\text{var}(p) = x$ is existential, and, for each universal u in $\text{vars}(C_r \cup C_s)$,*

$$\{u, \bar{u}\} \subseteq C_i \quad \Rightarrow \quad u \textit{ is not in the dependency set for } x$$

- **U Reduction**: *$C_i = C_r \setminus \{a\}$, for some $r < i$, where a is universal and trailing in C_r with respect to P;*
- **W Weakening**: *C_i is $\mathbb{L}$, or is subsumed by C_r for some $r < i$.*

It is conventional to write pairs of complementary universal literals $\bar{u}$ and u as a single merged literal $\overset{*}{u}$. For example, the clause $\{\bar{x}, \bar{u}, u\}$ can be written $\{\bar{x}, \overset{*}{u}\}$.

Example 7.2. Figure 7.1 shows an LDQ-Res refutation of EQ_1. Notice that the merged literal $\overset{*}{u}_1$, which represents the pair of literals $\bar{u}_1$ and u_1, is depicted as being reduced in a single step. Formally this requires two reduction steps, one for each literal. ■

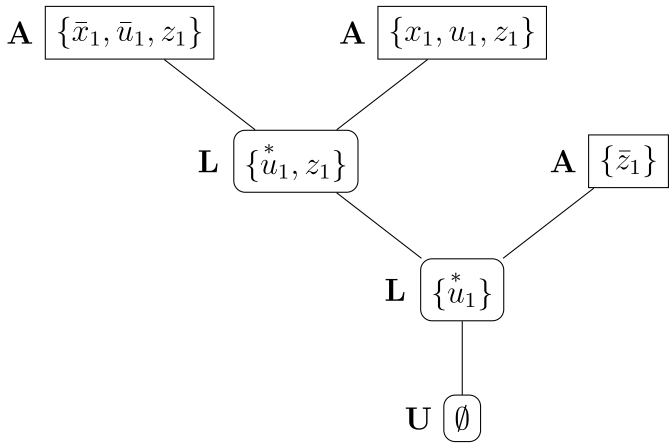


Figure 7.1: An LDQ-Res refutation of EQ_1.

It is easy to see that every Q-Res refutation is an LDQ-Res refutation, and so the completeness of LDQ-Res follows from that of Q-Res. The soundness of LDQ-Res is a different matter, and any proof confirming that a QBF with an LDQ-Res refutation is false must somehow interpret the semantics of merged literals. Reinterpreting tautologies, however, is a non-trivial task, and, it is fair to say, one which appears to be at odds with a conventional interpretation of logic.

For this reason, we do not prove the soundness of LDQ-Res, we only reference the result which can be found in the literature. We come shortly to the semantics of merged literals, and we defer proofs of soundness until we understand properly how to present them consistently with the semantics of conjunctive normal form.

Theorem 7.3 ([2]). LDQ-Res *is a proof system for* FQBF.

Breakdown of lower bound techniques

It is instructive to compare the LDQ-Res refutation of EQ_1 with the Q-Res refutation in Figure 5.1. In the Q-Res refutation, the resolution over z_1 is performed before the resolution over x_1. However, in LDQ-Res, the availability of universal tautologies, in the form of merged literals, allows the resolution over x_1 to take place first. In fact, as the next result shows, merging even allows linear size refutations of $\mathcal{EQ}$.

Theorem 7.4. *$\mathcal{EQ}$ admits* LDQ-Res *refutations of size $O(n)$.*

Proof. For each i in $[n]$, we can resolve the two clauses $\{\bar{x}_i, \bar{u}_i, z_i\}$ and $\{x_i, u_i, z_i\}$ to obtain the clause $\{\overset{*}{u}_i, z_i\}$. Resolving each of these in turn with the long clause $\{\bar{z}_1, \dots, \bar{z}_n\}$, we obtain the fully universal clause $\{\overset{*}{u}_1, \dots, \overset{*}{u}_n\}$. Applying universal reduction to each literal in this clause, we obtain the empty clause, completing the refutation. It is easy to see that the whole refutation is of size linear in n. $\square$

Theorem 7.4 demonstrates that our lower-bound techniques do not lift to long-distance Q-Resolution. More precisely, strategy size is not a lower bound on proof size in LDQ-Res, even for formulas of bounded quantifier depth.

The reason for this, in a nutshell, is that we are now dealing with tautological clauses. The projection of a clause to a set of universal variables does not necessarily represent an assignment, so the method of proof from Theorem 5.16 doesn't work.

The role of the merged literal

The role of the merged literal became something of a talking point amongst the QBF community. The crux of the matter is that the use of tautological clauses is quite difficult to interpret semantically. Certainly we cannot interpret them as we are accustomed to, since a derived tautology, which entails only other tautologies, can never be useful in refuting a formula. However, as we saw in Theorem 7.4, the tautological clauses used in LDQ-Res can actually shorten refutations.

A clear and precise account of the role of merged literals was finally given in [66]: they represent universal dependency functions. (In fact, they represent *partial countermodels*, which we come to in Chapter 12.) This was the result of a great deal of interest in merged literals, particularly in relation to strategy extraction, and the notion is evident to a greater or lesser degree in all of the earlier papers [23, 3, 47, 45, 7].

It is certainly worth taking a moment to elaborate. When a universal variable is merged, it is not in the dependency set of the existential pivot, or equivalently, the pivot is in the dependency set of the merged variable. The merged literal implicitly represents the function that always falsifies the literal in the antecedent clause in which the pivot is falsified. For example, the merged literal $\overset{*}{u}$ in Figure 7.1 represents the function

$$\begin{array}{lcl} \langle\{x_1\}\rangle & \to & \langle\{u_1\}\rangle \\ \bar{x}_1 & \mapsto & \bar{u}_1 \\ x_1 & \mapsto & u_1 \,. \end{array}$$

Further merging of literals with other literals, which may also be merged, produces progressively more complex functions.

What is perhaps unfortunate for LDQ-Res is that these dependency functions are represented *implictly*, and one must traverse the subderivation to determine exactly which function is represented.

7.2 Deferred LDQ-Res

Unlike Q-Resolution, long-distance Q-Resolution actually forms a complete QBF proof system even when all universal reduction steps are performed at the end of the refutation.

A *deferred* LDQ-Res derivation is an LDQ-Res derivation in which each application of reduction follows every application of every other rule [16]. The LDQ-Res refutation in Figure 7.1 is a deferred refutation; it has a single universal reduction which is performed last.

We call the restriction of long-distance Q-Resolution to deferred derivations *deferred* LDQ-Res.

Completeness of deferred refutations

Given a false QBF Q with a countermodel h, we construct a canonical reductionless LDQ-Res refutation based on the 'full binary tree' representation of a countermodel [55].

For each $x \in \langle \mathrm{vars}_\exists(Q) \rangle$, there exists some C_ε in the matrix falsified by $\varepsilon \cup h(\varepsilon)$. The set of all such C_ε may be successively resolved over existential pivots in reverse prefix order, finally producing a clause containing no existentials. Merged literals never block resolution steps in this construction, as they only ever appear to the right of the pivot variable.

Lemma 7.5. *Every false QBF has a deferred* LDQ-Res *refutation.*

Proof. Let $Q := P \cdot F$ be a false QBF with countermodel h. Denote the existential variables of Q by $X := \{x_1, \dots, x_n\}$, such that whenever $i < j$ holds, x_j does not appear in a block quantified before the block in which x_i appears.

Let $\varepsilon_1, \dots, \varepsilon_{2^n}$ define the natural lexicographic ordering of the total assignments to X, as in

$$\begin{array}{lllll}
\varepsilon_1 & := & z_1 \mapsto 0, \dots, z_{n-2} \mapsto 0, z_{n-1} \mapsto 0, z_n \mapsto 0 & \approx & 0 \cdots 000\,, \\
\varepsilon_2 & := & z_1 \mapsto 0, \dots, z_{n-2} \mapsto 0, z_{n-1} \mapsto 0, z_n \mapsto 1 & \approx & 0 \cdots 001\,, \\
\varepsilon_3 & := & z_1 \mapsto 0, \dots, z_{n-2} \mapsto 0, z_{n-1} \mapsto 1, z_n \mapsto 0 & \approx & 0 \cdots 010\,, \\
\varepsilon_4 & := & z_1 \mapsto 0, \dots, z_{n-2} \mapsto 0, z_{n-1} \mapsto 1, z_n \mapsto 1 & \approx & 0 \cdots 011\,, \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
\varepsilon_{2^n} & := & z_1 \mapsto 1, \dots, z_{n-2} \mapsto 1, z_{n-1} \mapsto 1, z_n \mapsto 1 & \approx & 1 \cdots 111\,.
\end{array}$$

We define a sequence $\pi := \pi_n, \dots, \pi_0$ in which each $\pi_i := C^i_1, \dots, C^i_{2^i}$, and the clauses C^i_j are defined recursively as follows. For $j \in [2^n]$, C^n_j is any clause in F

falsified by $\varepsilon_j \cup h(\varepsilon_j)$ (at least one such clause exists by definition of countermodel). For $i \in [n]$ and $j \in [2^{i-1}]$, $C_j^{i-1} := \mathrm{res}(C_{2j-1}^i, C_{2j}^i, x_i)$ if this resolvent exists, otherwise

$$C_j^{i-1} := \begin{cases} C_{2j-1}^i\,, & \text{if } x_i \notin C_{2j-1}^i\,,\\ C_{2j}^i\,, & \text{if } \bar{x}_i \notin C_{2j}^i\,. \end{cases}$$

It is readily verified by downwards induction on $i \in [n]$ that each C_j^i contains no complementary universal literals in variables left of x_i. Moreover, it is easy to see that the conclusion C_1^0 contains no existential literals. So a deferred LDQ-Res refutation of Q is obtained from π by reducing all the literals in the final clause. □

This is enough to show that deferred LDQ-Res is a refutational QBF proof system.

Theorem 7.6. LDQ-Res *is a proof system for the language* FQBF*.*

Proof. Completeness. Established by Lemma 7.5. *Soundness and checkability.* Both follow from the (trivial) fact that LDQ-Res p-simulates deferred LDQ-Res. □

7.3 The squared equality family

We have already seen that Theorem 7.4 marks the breakdown of our lower-bound technique in the long-distance context. In fact, since the short refutations constructed in the proof of Theorem 7.4 are in fact deferred, countermodel size is no proof-size lower bound in deferred LDQ-Res either.

However, we can show a lower bound for deferred LDQ-Res by modifying the equality family. The modification is a kind of squaring.

Definition 7.7 (equality family)**.** *The* squared equality family *is the QBF family whose n^{th} instance is*

$$\mathrm{EQ}_n^2 := \exists x_1 \cdots x_n y_1 \cdots y_n \forall u_1 \cdots u_n v_1 \cdots v_n \exists z_1 \cdots z_n \cdot \mathrm{eq}_n^2\,,$$

where the CNF eq_n^2 *consists of the clauses*

$$\begin{array}{ll} \{\bar{x}_i, \bar{y}_j, \bar{u}_i, \bar{v}_j, z_{i,j}\}\,, & \text{for } i,j \text{ in } [n]\,,\\ \{x_i, \bar{y}_j, u_i, \bar{v}_j, z_{i,j}\}\,, & \text{for } i,j \text{ in } [n]\,,\\ \{\bar{x}_i, y_j, \bar{u}_i, v_j, z_{i,j}\}\,, & \text{for } i,j \text{ in } [n]\,,\\ \{x_i, y_j, u_i, v_j, z_{i,j}\}\,, & \text{for } i,j \text{ in } [n]\,,\\ \{\bar{z}_{i,j} : i,j \in [n]\}\,. & \end{array}$$

We call the final clause in the matrix eq_n^2 the *square clause.*

Proof of hardness in deferred LDQ-Res

The squared equality family actually requires exponential-size deferred refutations. To prove this, we first need a formal definition of a refutation *path.* A path is a sequence of consecutive resolvents beginning with an axiom and ending at the final resolvent.

Definition 7.8 (path)**.** *Let π be a deferred* LDQ-Res *refutation. A* path *from a clause C in π is a subsequence $C_1, \dots, C_k$ of π in which:*

(a) *$C = C_1$ is an axiom of π;*

(b) *C_k contains no existential literals;*

(c) *for each i in $[k-1]$, C_{i+1} is an antecedent of C_i.*

The lower-bound proof is based upon two facts, which we prove as preliminary lemmata.

(1) Every total existential assignment corresponds to a path, all of whose clauses are consistent with the assignment (Lemma 7.9).

(2) Every path from the square clause contains a 'wide' clause containing either all the x_i or all the y_j variables (Lemma 7.10).

It is then possible to deduce the existence of exponentially many wide clauses, by considering the set of assignments ε for which each $\varepsilon(x_i) = \varepsilon(y_i)$ and each $\varepsilon(z_{i,j}) = 0$, all of whose corresponding paths begin at the square clause.

Lemma 7.9. *Let π be a tidy deferred* LDQ-Res *refutation of a QBF Q, and let T be a clause with $\mathrm{vars}(T) = \mathrm{vars}_\exists(Q)$. Then there exists a path in π in which no existential literal outside of T occurs.*

Proof. We describe a procedure that constructs a sequence $P := C_k, \dots, C_1$ of clauses in reverse order as follows. Let the clause C_1 be the antecedent of the final resolution step in π. At each step, let the next clause C_{i+1} be the unique antecedent of C_i which contains the pivot literal in the same polarity as it occurs in T. The procedure terminates as soon an axiom is encountered; that is, C_k is the unique axiom in the sequence.

P is clearly a path in π by construction. By induction we show that the existential subclause of C_i is a subset of T, for each i in $[n]$. The base case $i = 1$ holds trivially since there are no existential literals in the conclusion C_1 of π. The inductive step $i \geq 2$ holds trivially by construction. $\square$

The second lemma is more technical, and its proof more involved. The proof works directly on the definition of path, the rules of LDQ-Res, and the syntax of the squared equality formulas, to show the existence of a wide clause in all paths from the square clause.

Lemma 7.10. *Let $n \geq 2$, and let π be a tidy deferred* LDQ-Res *refutation of* EQ_n^2. *On each path from the square clause, there occurs a clause C for which either $\{x_1, \dots, x_n\} \subseteq \mathrm{vars}(C)$ or $\{y_1, \dots, y_n\} \subseteq \mathrm{vars}(C)$.*

Proof. Put $X := \{x_1, \dots, x_n\}$ and $Y := \{y_1, \dots, y_n\}$. For any variable p, we call a clause in π a *p-resolvent* if it is the consequent of a resolution step over pivot variable p.

Now, we let $P := C_1, \dots, C_k$ be any path from the square clause. For each l in $[k]$ we define an $n \times n$ matrix M_l, where

$$M_l[i,j] := \begin{cases} 1 & \text{if } \bar{z}_{i,j} \in C_l \\ 0 & \text{otherwise} . \end{cases}$$

We choose l_0 as the least integer such that M_{l_0} has either a 0 in each row or a 0 in each column. Note that $l_0 \geq 2$, as M_1 has no zeros.

We consider two exhaustive cases. In the first we show that $X \subseteq \mathrm{vars}(C_{l_0})$, and in the second we show that $Y \subseteq \mathrm{vars}(C_{l_0})$.

(a) Suppose that M_{l_0} has a 0 in each row. We first show that every row in M_{l_0} also has at least one 1.

Aiming for contradiction, suppose that M_{l_0} contains a full 0 row r (this implies that $l_0 \geq 2$, and hence that $M_{l_0 - 1}$ exists). By definition of resolution there can be at most one element that changes from 1 in M_{l_0-1} to 0 in M_{l_0}. Since M_{l_0-1} does not have a 0 in every column, it does not contain a full zero row. Hence it must be the case that the unique element that went from 1 in M_{l_0-1} to 0 in M_{l_0} is in row r. Since $n \geq 2$, we deduce that M_{l_0-1} has a 0 in each row, contradicting the minimality of l_0.

Consider the following three statements, which we claim hold for all $i, j \in [n]$:

(1) for each clause C_l on P, if $\bar{z}_{i,j}$ is in C_l, then $\{\bar{u}_i, u_i\} \not\subseteq C_l$;

(2) each x_i-resolvent in π contains $\{\bar{u}_i, u_i\}$ as a subset;

(3) for each $z_{i,j}$-resolvent R in π, if $x_i \notin \mathrm{vars}(R)$ then $\{\bar{u}_i, u_i\} \subseteq R$.

We prove the claims afterwards.

For now, let i in $[n]$. Since the i^{th} row in M_{l_0} contains a 1, there is some j in $[n]$ for which $\bar{z}_{i,j}$ is in C_{l_0}. From claim (1) it follows that $\{\bar{u}_i, u_i\} \nsubseteq C_{l_0}$. Moreover, as universal literals accumulate along the path, each clause on P up to and including C_{l_0} does not contain $\{\bar{u}_i, u_i\}$ as a subset. Since the i^{th} row in M_{l_0} contains a 0, there exists some j_0 in $[n]$ for which $\bar{z}_{i,j_0}$ is not in C_{l_0}.

As $\bar{z}_{i,j_0}$ is in C_1, there must be a z_{i,j_0}-resolvent preceding C_{l_0} on P, which contains variable x_i by claim (3). Also, each clause up to and including C_{l_0} is not an x_i-resolvent by claim (2). It follows that $x_i \in \text{vars}(C_{l_0})$, and since i was chosen arbitrarily, we have $X \subseteq \text{vars}(C_{l_0})$.

(b) Suppose on the other hand that M_{l_0} does not contain a 0 in each row. Then M_{l_0} contains a 0 in each column, and a symmetrical argument shows that $Y \subseteq \text{vars}(C_{l_0})$.

It remains to prove the three claims.

(1) First, observe that each clause in π containing the positive literal $z_{i,j}$ also contains the variable u_i (this holds for every axiom and universal literals are never removed).

Now, let C_l be a clause on the path P for which $\bar{z}_{i,j}$ is in C_l, and, for the sake of contradiction, suppose that $\{\bar{u}_i, u_i\} \subseteq C_l$. Since u_i belongs to the dependency set for $z_{i,j}$, there cannot be a $z_{i,j}$-resolvent on P following C, as such a resolution step would be forbidden.

This means that $\bar{z}_{i,j}$ occurs in C_k, the final clause of P. This is a contradiction, since C_k is the antecedent of the final resolution step in the tidy refutation π; it is followed only by reduction steps which derive the empty clause, and hence contains no existential literals.

(2) First, observe that each clause in π containing $\bar{x}_i$ also contains $\bar{u}_i$, and each clause containing x_i also contains u_i. Again, this holds for every axiom and universal literals are never removed. It follows immediately that an x_i-resolvent contains $\{\bar{u}_i, u_i\}$ as a subset.

(3) Observe that each axiom in π containing the positive literal $z_{i,j}$ contains variable x_i. Hence, any clause in π that contains literal $z_{i,j}$, but not variable x_i, must

appear after an x_i-resolvent on some path, and therefore contains $\{\bar{u}_i, u_i\}$ by Claim (2).

Now, let R be a $z_{i,j}$-resolvent of R_1 and R_2 in π. Suppose that $x_i \notin \mathrm{vars}(R)$, which implies that $x_i \notin \mathrm{vars}(R_1)$. Since $z_{i,j}$ is in R_1, we have $\{\bar{u}_i, u_i\} \subseteq R_1$, and it follows that $\{\bar{u}_i, u_i\} \subseteq R$. □

Now we prove the lower bound from the preceding lemmata.

Theorem 7.11. *$\mathcal{EQ}^2$ requires deferred* LDQ-Res *refutations of size 2^{n-1}.*

Proof. Let n be a natural number, and let π be a tidy deferred LDQ-Res refutation of EQ^2_n. Once again we put $X := \{x_1, \dots, x_n\}$ and $Y := \{y_1, \dots, y_n\}$.

We show that $|\pi| \geq 2^{n-1}$. The size bound is trivially true for $n = 1$, so we assume $n \geq 2$.

Now, we call a non-tautological clause S *symmetrical* when it satisfies the following three properties:

(a) $\mathrm{vars}(S) = \mathrm{vars}_\exists(\mathrm{EQ}^2_n)$;

(b) for each i in $[n]$, x_i and y_i appear in the same polarity in S;

(c) for each i, j in $[n]$, $z_{i,j}$ appears in negative polarity in S.

It is easy to see that there are 2^n distinct symmetrical clauses.

By Lemma 7.9, for each symmetrical clause S, there exists a path P_S in π in which all appearing existential literals belong to S. Moreover, each P_S begins at the long clause, since every other clause in eq^2_n contains some positive $t_{i,j}$ literal that does not occur in S.

Hence, by Lemma 7.10, on each P_S there exists a clause C_S for which either $X \subseteq \mathrm{vars}(C_S)$ or $Y \subseteq \mathrm{vars}(C_S)$. It follows that we can define a function f that maps each symmetrical assignment S to a clause $f(S)$ in π for which either $\mathrm{proj}(S, X) \subseteq f(S)$ or $\mathrm{proj}(S, Y) \subseteq f(S)$, or both.

Moreover, since distinct symmetrical clauses S_1 and S_2 satisfy

$$\mathrm{proj}(S_1, X) \neq \mathrm{proj}(S_2, X) \quad \text{and} \quad \mathrm{proj}(S_1, Y) \neq \mathrm{proj}(S_2, Y)\,,$$

each $f(S)$ is the image of at most two distinct symmetrical clauses. Hence, π contains at least 2^{n-1} clauses. □

Part III

Models of Solving

Chapter 8

Dependency Quantified Boolean Formulas

Much of our work in this part of the thesis is based on translations from the set of QBFs into a larger set, the set of *dependency quantified Boolean formulas* (DQBF). At the same time, we want to lift our QBF proof systems to DQBF, but we encounter some problems on the reduction side. It turns out that the semantics of DQBF, which is richer than QBF, has a non-trivial impact on our translations.

In this chapter we provide the background on DQBFs that is needed for the remaining chapters in Part III.

Organisation of the chapter

In Section 8.1, we deal with DQBF syntax, and turn to semantics in Section 8.2. In Section 8.3, we discuss the proof complexity landscape for DQBF, and give a possible explanation for the issues with reduction systems.

8.1 S-form and H-form

DQBFs can be written in one of two forms, Skolem form (*S-form*) and Herbrand form (*H-form*). Skolem form DQBFs give the dependencies for the existential variables, whereas Herbrand form gives the dependencies for the universals.

Definition 8.1 (DQBF)**.** *An S-form* dependency quantified Boolean formula (DQBF) *is of the form*

$$\forall u_1 \cdots \forall u_m \exists x_1(S_1) \cdots \exists x_n(S_n) \cdot F$$

and an H-form dependency quantified Boolean formula is of the form

$$\exists x_1 \cdots \exists x_n \forall u_1(H_1) \cdots \forall u_m(H_m) \cdot F$$

where

(a) $X := \{x_1, \ldots, x_n\}$ *and* $U := \{u_1, \ldots, u_m\}$ *are disjoint sets of Boolean variables,*

(b) *for each* i *in* $[n]$, $S_i \subseteq U$, *and for each* $j \in [m]$, $H_j \subseteq X$,

(c) F *is a CNF with* $\mathrm{vars}(F) \subseteq U \cup X$.

We denote the set of S-form DQBFs by '$\mathbb{S}$' and the set of H-form DQBFs by '$\mathbb{H}$'.

A DQBF is rather like a QBF whose dependency sets, either existential or universal, have been given explicitly. In an S-form DQBF, the dependency sets for the existential variables are given: each S_i is the dependency set for the existential x_i. Similarly, in H-form each universal dependency set H_j is given to variable u_j explicitly. This is in contrast to QBF where the dependency sets are defined implicitly in the prefix.

It is easy to see that the set of QBFs $\mathbb{Q}$ is a subset of both of $\mathbb{S}$ and $\mathbb{H}$. For example, given a QBF

$$Q := \forall U_1 \exists X_1 \cdots \forall U_d \exists X_d \cdot F$$

with existential variables $\{x_1, \ldots, x_n\}$ and universal variables $\{u_1, \ldots, u_m\}$, we can write it either as the S-form DQBF or the H-form DQBF in Definition 8.1, where the S_i and H_j are the existential and universal dependency sets of Q.

A QBF prefix prescribes a total order on blocks, meaning that the dependency sets for QBFs always form nested subsets. More precisely, there always exists some enumeration of the existential variables, say $\{x_1, \ldots, x_n\}$, for which

$$S_1 \subseteq S_2 \subseteq \cdots \subseteq S_{n-1} \subseteq S_n \,,$$

and some enumeration of the universal variables, say $\{u_1, \ldots, u_m\}$, for which

$$H_1 \subseteq H_2 \subseteq \cdots \subseteq H_{m-1} \subseteq H_m \,.$$

DQBFs in general do not have this property, since the dependency sets can be arbitrary subsets of the oppositely quantified variables. As a result, the semantics of DQBF is a much richer affair compared to QBF.

8.2 Semantics

Models and countermodels for DQBF are defined exactly as for QBF (Definitions 3.2 and 3.4). Since a DQBF specifies dependency sets for only one quantification type, models only make sense for S-form DQBFs, and countermodels only for H-form.

For example, a model for the S-form DQBF

$$\forall u_1 \cdots \forall u_m \exists x_1(S_1) \cdots \exists x_n(S_n) \cdot F$$

is a set of existential dependency functions $\{f_i\}_{i \in [n]}$ for which, for each total universal assignment μ, the assignment

$$\mu \cup \{f_i(\mu \restriction_{S_i}) : i \in [n]\}$$

satisfies the matrix F. A countermodel for the H-form DQBF

$$\exists x_1 \cdots \exists x_n \forall u_1(H_1) \cdots \forall u_m(H_m) \cdot F$$

is a set of universal dependency functions $\{h_j\}_{j \in [m]}$, where

$$\varepsilon \cup \{h_j(\varepsilon \restriction_{H_j}) : j \in [m]\}$$

falsifies F, for each total existential assignment ε.

We call an S-form DQBF *true* when it has a model, and *false* when it does not. In contrast, we call an H-form DQBF *false* when it has a countermodel, and *true* when it does not.

Complexity

Under a suitable encoding as binary strings, the set of true S-form DQBFs forms the canonical NEXP-complete language TSDQBF [65]. We refer to the language of false S-form DQBFs as FSDQBF, and the language of false H-form DQBFs as FHDQBF.

Complementation

To understand what is unusual about DQBF semantics, we recall a natural bijection between $\mathbb{S}$ and $\mathbb{H}$. The application of this bijection is called *complementation.*

Definition 8.2 (complement [1])**.** *The* complement *of an S-form DQBF*

$$Q := \forall u_1 \cdots \forall u_m \exists x_1(S_1) \cdots \exists x_n(S_n) \cdot F$$

is the H-form DQBF

$$\sim\!Q := \exists x_1 \cdots \exists x_n \forall u_1(H_1) \cdots \forall u_m(H_m) \cdot F$$

where $H_j := \{x_i : u_j \notin S_i\}$, for each j in $[m]$.

(What we refer to here as the complement is actually the negation of the complement in [1].) The interesting thing about complementation is that it preserves truth value on QBFs, but not on DQBFs in general.

Complementation on QBF

Imagine for a moment that Q in Definition 8.2 is a QBF, i.e. it has nested dependency sets, and let us compute H_j, the dependency set for u_j in the complement of Q. This is the set of existential variables x_i for which u_j is not in S_i, that is, for which u_j is quantified after x_i in the total order of blocks. So H_j is exactly the universal dependency set for u_j in Q.

Applying complementation to Q really just swaps the existential dependency sets for the universal ones. Now we have a good opportunity use the QBF folklore theorem (Theorem 3.8), which tells us that Q has either a model or a countermodel, but not both. We deduce two consequences:

(a) if Q has a model, it is a true S-form DQBF, and its H-form complement is also true because it doesn't have a countermodel.

(b) on the other hand, if Q has a countermodel, it is a false H-form DQBF, and its S-form complement, which doesn't have a model, is also false.

These two together show that complementation preserves QBF truth values.

Counterexamples to the folklore theorem for DQBF

The folklore theorem is not true however for DQBFs in general [1]. If it were, it would read something like

'*an S-form DQBF is false if, and only if, its complement has a countermodel.*'

As we see in the next examples, neither implication is true!

Example 8.3 (adapted from [1]). First we show a false S-form DQBF whose complement does not have a countermodel, namely

$$\forall u_1 \forall u_2 \exists x_1(\{u_1\}) \exists x_2(\{u_2\}) \cdot \{\{\bar{u}_1, u_2, x_1\}, \{u_1, \bar{u}_2, x_2\}, \{\bar{u}_1, \bar{u}_2, \bar{x}_1, \bar{x}_2\}\}$$

We can check, for example by an exhaustive search of dependency functions, that this DQBF has no model, so it is false. We can also verify that its complement

$$\exists x_1 \exists x_2 \forall u_1(\{x_2\}) \forall u_2(\{x_1\}) \cdot \{\{\bar{u}_1, u_2, x_1\}, \{u_1, u_2, x_2\}, \{\bar{u}_1, \bar{u}_2, \bar{x}_1, \bar{x}_2\}\}$$

has no countermodel. ■

Example 8.4. This time we show a true S-form DQBF whose complement has a countermodel, namely

$$Q := \forall u_1 \forall u_2 \exists x_1(\{u_1\}) \exists x_2(\{u_2\}) \exists x_3(\{u_1, u_2\}) \cdot F$$

whose matrix is the CNF

$$F := \{\{\bar{u}_1, \bar{x}_2, \bar{x}_3\}, \{u_1, x_2, \bar{x}_3\}, \{\bar{u}_2, \bar{x}_1, x_3\}, \{u_2, x_1, x_3\}\}\,.$$

The following is a model for Q:

$$\begin{array}{rcccc} f_1 & : & \langle\{u_1\}\rangle & \to & \langle\{x_1\}\rangle \\ & & \{\bar{u}_1\} & \mapsto & \{\bar{x}_1\} \\ & & \{u_1\} & \mapsto & \{x_1\} \end{array} \qquad \begin{array}{rcccc} f_2 & : & \langle\{u_2\}\rangle & \to & \langle\{x_2\}\rangle \\ & & \{\bar{u}_2\} & \mapsto & \{x_2\} \\ & & \{u_2\} & \mapsto & \{\bar{x}_2\} \end{array}$$

$$\begin{array}{rcccc} f_3 & : & \langle\{u_1, u_2\}\rangle & \to & \langle\{x_3\}\rangle \\ & & \{\bar{u}_1, \bar{u}_2\} & \mapsto & \{x_3\} \\ & & \{\bar{u}_1, u_2\} & \mapsto & \{\bar{x}_3\} \\ & & \{\bar{u}_1, u_2\} & \mapsto & \{\bar{x}_3\} \\ & & \{u_1, u_2\} & \mapsto & \{x_3\} \end{array}$$

We can verify that this is indeed a model by checking, for each μ in $\langle\{u_1, u_2\}\rangle$, that the assignment

$$\mu \cup f_1(\mu\restriction_{\{u_1\}}) \cup f_2(\mu\restriction_{\{u_2\}}) \cup f_3(\mu)$$

satisfies F.

The complement of Q is the H-form DQBF

$$Q := \exists x_1 \exists x_2 \exists x_3 \forall u_1(\{x_2\}) \forall u_2(\{x_1\}) \cdot F\,.$$

which has the following countermodel:

$$\begin{array}{rcccc} h_1 & : & \langle\{x_2\}\rangle & \to & \langle\{u_1\}\rangle \\ & & \{\bar{x}_2\} & \mapsto & \{\bar{u}_1\} \\ & & \{x_2\} & \mapsto & \{u_1\} \end{array} \qquad \begin{array}{rcccc} h_2 & : & \langle\{x_1\}\rangle & \to & \langle\{u_2\}\rangle \\ & & \{\bar{x}_1\} & \mapsto & \{\bar{u}_2\} \\ & & \{x_1\} & \mapsto & \{u_2\} \end{array}$$

To check that this is indeed a countermodel, we just verify that the assignment

$$\varepsilon \cup h_1(\varepsilon\restriction_{\{u_1\}}) \cup h_2(\varepsilon\restriction_{\{u_2\}})$$

falsifies F, for each ε in $\langle\{x_1, x_2\}\rangle$. ■

This completes the total breakdown of the folklore theorem for DQBF. As a result, we can partition S-form into four distinct non-empty classes, based on the truth values of the DQBF and its complement:

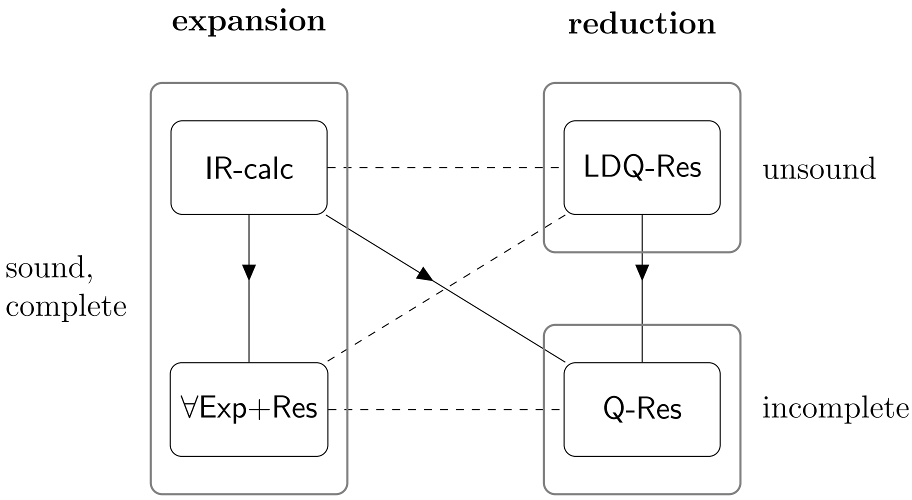


Figure 8.1: Lifting QBF proof systems to S-form DQBF.

- $\mathbb{S}_S^\circ := \{Q \in \mathbb{S} : Q$ has a model, $\sim Q$ has no countermodel$\}$
- $\mathbb{S}_\circ^H := \{Q \in \mathbb{S} : Q$ has no model, $\sim Q$ has a countermodel$\}$
- $\mathbb{S}_\circ^\circ := \{Q \in \mathbb{S} : Q$ has no model, $\sim Q$ has no countermodel$\}$
- $\mathbb{S}_S^H := \{Q \in \mathbb{S} : Q$ has a model, $\sim Q$ has a countermodel$\}$.

All QBFs are either in $\mathbb{S}_S^\circ$ or $\mathbb{S}_\circ^H$. $\mathbb{S}_\circ^\circ$ and $\mathbb{S}_S^H$ are classes of DQBFs whose semantic properties are markedly different from QBF. Note that the QBF from Example 8.3 belongs to the class $\mathbb{S}_\circ^\circ$, whereas the one in Example 8.4 belongs to $\mathbb{S}_S^H$.

8.3 Lifting QBF proof systems

All of our four main QBF proof systems have been trialled on S-form, with varying success. The two papers [1] and [15] together demonstrated what happens when we try to lift them to S-form DQBF in the most obvious way. In each case, the lifted system is something that tries to refute S-form DQBFs.

Figure 8.1 shows some results from the two papers. It turns out that $\forall$Exp+Res and IR-calc are still sound and complete [6], whereas Q-Res is incomplete [1] and LDQ-Res is unsound [15]. The conclusion of Figure 8.1 seems to be that expansion systems lift to S-form DQBF whereas reduction systems do not.

It is not clear a priori why this should be the case.

The expansion-reduction hypothesis

In the coming chapters, we present some evidence for the following idea:

Idea 8.5 (expansion-reduction hypothesis)**.** *Expansion systems prove the non-existence of models, whereas reduction systems prove the existence of countermodels.*

In fact, this one idea can explain the whole picture of Figure 8.1.

If expansion systems indeed prove the non-existence of models, then the use of S-form DQBFs, which are false when they have no models, is perfect for a refutational proof system. In this sense, the fact that $\forall$Exp+Res and IR-calc are both sound and complete for S-form is consistent with our hypothesis.

In contrast, when we lift reduction proof systems to S-form, they are really working to refute the complement H-form DQBF. So we should expect trouble from the classes $\mathbb{S}^{\circ}_{\circ}$ and $\mathbb{S}^{H}_{S}$, where complementation does not preserve the truth value.

Indeed, the class $\mathbb{S}^{\circ}_{\circ}$ is directly responsible for the incompleteness. As we will see in Chapter 11, the DQBF from Example 8.3, which belongs to $\mathbb{S}^{\circ}_{\circ}$, is a counterexample to the completeness of Q-Res; it is a false formula that has no refutation. All of this follows quite normally from the expansion-reduction hypothesis: If reduction is indeed proving the existence of countermodels, it may not be able to refute false formulas in $\mathbb{S}^{\circ}_{\circ}$.

Moreover, the other class $\mathbb{S}^{H}_{S}$ is directly responsible for the unsoundness. We show later in Chapter 12 that the true DQBF from Example 8.4, belonging to $\mathbb{S}^{H}_{S}$, is a counterexample to the soundness of LDQ-Res; it is a true formula with a refutation. Again there is a clear reason: If reduction is indeed proving the existence of countermodels, it is liable to refute true formulas in $\mathbb{S}^{H}_{S}$.

The upshot is that reduction and S-form are not always compatible, and the obvious fix is to use H-form instead. In Chapter 12, we propose a new reduction calculus, Merge Resolution, which is sound and complete for the language of false H-form DQBFs.

It is worth pausing for a moment to see that something like the expansion-reduction hypothesis, which completely explains Figure 8.1, could never be advocated from the QBF perspective, where complementation preserves truth values, and the non-existence of a model is equivalent to the existence of a countermodel. One really needs to know the behaviour of the classes $\mathbb{S}^{\circ}_{\circ}$ and $\mathbb{S}^{H}_{S}$ to understand why these two things are not equivalent for DQBFs in general.

Chapter 9

Dependency Schemes

Solvers for quantified Boolean formulas are restricted in their choice of variable selection in a way that satisfiability solvers are not. For example, when solving a QBF

$$\forall U_1 \exists X_1 \cdots \forall U_d \exists X_d \cdot F$$

under normal circumstances, the solver would be forced to assign all the variables in the first block U_1, before moving on to the second block X_1. Then, all the variables in the second block must be assigned before moving to the third, and so on.

This is a result of the inter-block dependencies that arise due to the order of quantification in the prefix. In propositional logic, we only have a single existential block, so the issues for variable selection do not apply to SAT solvers. The main drawback for QBF is that the scope of decision heuristics (algorithms that decide which variable to assign next) are reduced when the allowable decisions are restricted. Decision heuristics are one of the main drivers in CDCL solving.

Fortunately for QBF, it seems to occur frequently that some dependencies given by the prefix are needlessly restrictive, and can be bypassed. This could be due to prenexing, for example, which forces all quantification to the front of the formula, meanwhile introducing 'spurious' dependencies between blocks that need not be there. The subfield of QBF solving that works with this idea is called *dependency-aware* solving.

Dependency awareness

Dependency awareness, as implemented in the solver DepQBF [41], 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

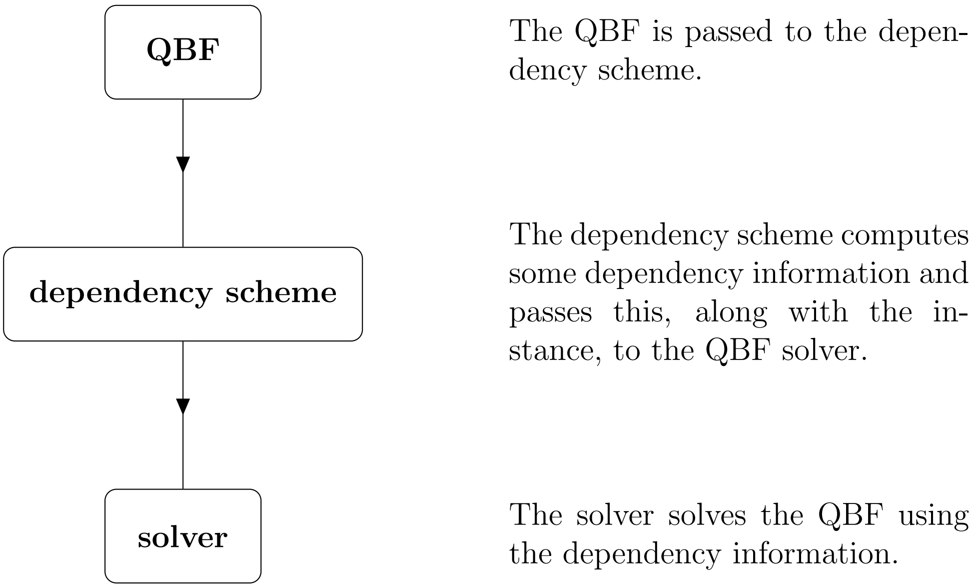


Figure 9.1: Dependency-aware QBF solving.

dependencies of the instance, granting the solver greater freedom regarding variable assignments.

The situtation is depicted in Figure 9.1. 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 [40].

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* ($\mathcal{D}^{\text{std}}$ [54]), 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 variables. The *reflexive resolution path dependency scheme* ($\mathcal{D}^{\text{rrs}}$ [63]) utilises the notion of a *resolution path*, a more refined type of connection introduced in [26].

Organisation of the chapter

We recall the traditional interpretation of dependency schemes in Section 9.1, and move on to the DQBF interpretation in Section 9.2. In Section 9.3, we turn to $\mathcal{D}^{\text{std}}$ and $\mathcal{D}^{\text{rrs}}$, and investigate how they operate on hand-crafted instances.

9.1 The traditional interpretation

The traditional interpretation of dependency schemes originates from [53]. Soon after it was modified to work with binary relations [54], and was subsequently employed by a number of authors, e.g. [40, 62, 7].

The trivial dependency relation for a QBF Q is the binary relation on $\mathrm{vars}(Q) \times \mathrm{vars}(Q)$ consisting of the pairs (z, z') for which

(a) z and z' are oppositely quantified, and

(b) z is left of z'.

The simplest dependency scheme is the trivial dependency scheme $\mathcal{D}^{\mathrm{trv}}$. It is the function that maps each QBF to its trivial dependency relation.

A dependency scheme $\mathcal{D}$ is a function that maps a QBF Q to a subrelation of its trivial dependencies. Equivalently, $\mathcal{D}$ is a mapping from $\mathbb{Q}$ into the set of binary relations on $\mathbb{U} \times \mathbb{U}$, satisfying

$$\mathcal{D}(Q) \subseteq \mathcal{D}^{\mathrm{trv}}(Q), \text{ for each QBF } Q.$$

So a dependency scheme maps QBFs to binary relations on variables. But how should we interpret them?

The binary relation identifies pairs (z, z') for which z' is considered dependent on z. So, the existence of a pair (z, z') in $\mathcal{D}(Q)$ should be interpreted as 'z' depends on z in Q according to the dependency scheme $\mathcal{D}$'. Pairs not included in the binary relation represent independencies. A pair (z, z') absent from $\mathcal{D}(Q)$ should be interpreted as 'z' is independent of z in Q according to $\mathcal{D}$'.

Existential and universal dependencies

The traditional interpretation of a dependency scheme as a binary relation on variables allows both kinds of dependencies to be expressed, namely, dependence of existentials on universals and dependence of universals on existentials. However, when dealing with refutational proof systems, it turns out that we can ignore the latter: we are only concerned with the independence of existentials on universals. So we can view the dependencies of Q according to $\mathcal{D}$ as a binary relation on $\mathrm{vars}_\forall(Q) \times \mathrm{vars}_\exists(Q)$.

This is helpful for several reasons.

9.2 The DQBF interpretation

The reader may already suspect that there is some relationship between dependency schemes and DQBF, and indeed there is. In summary, the binary relation can be written as an S-form quantifier prefix.

When we employ a dependency scheme, we are really replacing the linear order of the QBF prefix with a partial order, hoping for a better approximation to the 'true' dependencies. A DQBF prefix represents exactly this kind of partial order. So, instead of associating a QBF with a binary relation, we can associate it with an S-form DQBF (with the same matrix), whose quantifier prefix expresses the binary relation. S-form specifies dependency sets for the existential variables only, but this is fine for us – we only consider existential dependencies in our theoretical models.

Since dependency schemes are in the business of identifying non-trivial independencies, we inevitably want to map QBFs to DQBFs in which the dependency sets are smaller (whereas the variable sets and matrix should not change). The *strength* relation captures this.

Definition 9.1 (strength)**.** *Given two DQBFs*

$$Q := \forall u_1 \cdots \forall u_m \exists x_1(S_1) \cdots \exists x_n(S_n) \cdot F\,,$$

$$Q' := \forall u_1 \cdots \forall u_m \exists x_1(S_1') \cdots \exists x_n(S_n') \cdot F\,,$$

we say that

(a) *Q' is* stronger *than Q when each S_i' is a subset of S_i,*

(b) *Q' is* strictly stronger *than Q when Q' is stronger than Q and some S_i' is a strict subset of S_i.*

Now we can redefine the dependency scheme using the notion of strength.

Definition 9.2 (dependency scheme)**.** *A* dependency scheme *is a function from $\mathbb{Q}$ into $\mathbb{S}$ that maps each QBF to a stronger DQBF.*

A dependency scheme is only useful when it maps to stronger formulas, so the definition excludes mappings that do not. This is the analogue of the condition, in the traditional interpretation, that the binary relation is a subrelation of the trivial dependencies.

Moreover, we now define the trivial dependency scheme as the identity mapping. That is, $\mathcal{D}^{\mathrm{trv}}$ is the unique function that maps each QBF to itself.

Full exhibition

Full exhibition [61, 7] is a property of dependency schemes that guarantees they are semantically correct. This means that their use in a solver can be verified as correct by proving the soundness of the associated proof system. We look at proof systems with dependency schemes beginning in Chapter 10.

But for now, back to full exhibition. The DQBF interpretation gives us a nice definition.

Definition 9.3 (full exhibition [61, 7])**.** *We call a dependency scheme* fully exhibited *when it maps each true QBF to a true DQBF.*

Note that a false QBF is always mapped to a false DQBF by any dependency scheme, due to the strength condition – a DQBF that is stronger than some false DQBF is itself false.

To check full exhibition of a dependency scheme $\mathcal{D}$, we must search for models for the image of a QBF under $\mathcal{D}$. These models turn out to be quite important, so much so that we give them their own name: $\mathcal{D}$-models.

A $\mathcal{D}$-model for a QBF is a model for the DQBF $\mathcal{D}(Q)$. This is analogous to the $\mathcal{D}$-models of the traditional interpretation [61, p.36]. A dependency scheme is fully exhibited when, and only when, every true QBF has a $\mathcal{D}$-model, and indeed this is how the property was originally defined. The term 'full exhibition' was coined in [7], the property itself originates from [61].

Pairwise comparison

The pairwise comparison of schemes also fits neatly into the DQBF interpretation. Schemes are compared by the notion of generality, whereby one scheme is considered more general than another if it is capable of identifying more independencies.

Definition 9.4 (generality)**.** *Given two dependency schemes $\mathcal{D}$ and $\mathcal{D}'$, we say that*

(a) *$\mathcal{D}'$ is* more general *than $\mathcal{D}$ when, for each Q, $\mathcal{D}'(Q)$ is stronger than $\mathcal{D}(Q)$,*

(b) *$\mathcal{D}'$ is* strictly more general *than $\mathcal{D}$ when $\mathcal{D}'$ is stronger than $\mathcal{D}$ and, for some Q, $\mathcal{D}'(Q)$ is strictly stronger than $\mathcal{D}(Q)$.*

It follows from the definitions of dependency scheme (Definition 9.2), full exhibition (Definition 9.3) and generality (Definition 9.4) that the full exhibition of a dependency scheme implies the full exhibition of any less general scheme.

Fact 9.5. *Given two dependency schemes $\mathcal{D}$ and $\mathcal{D}'$, if $\mathcal{D}$ is fully exhibited and more general than $\mathcal{D}'$, then $\mathcal{D}'$ is also fully exhibited.*

9.3 Concrete dependency schemes

Many dependency schemes have been proposed in the literature (see [61] for a taxonomy). Here we focus on $\mathcal{D}^{\text{std}}$ and $\mathcal{D}^{\text{rrs}}$, the two main schemes that are actually implemented in solvers.

Both of these schemes work on 'connections' between clauses in the matrix. An existential is considered dependent on a universal if there exists a connection to it, i.e. a sequence of clauses in the matrix satisfying some property. The absence of a connection represents independence. Therefore stronger notions of connection give rise to stronger dependency schemes.

9.3.1 The standard dependency scheme

The standard dependency scheme was historically both the first to be proposed [54] and to be implemented in a solver (DepQBF [41]).

In $\mathcal{D}^{\text{std}}$, an existential x depends on a universal u whenever a clause containing variable x is connected to a clause containing variable u. A connection is a sequence of clauses, where successive clauses share a common existential variable right of u.

Definition 9.6 (standard dependency scheme [54])**.** *The standard dependency scheme $\mathcal{D}^{\text{std}}$ is the function that maps a QBF*

$$Q := \forall u_1 \cdots \forall u_m \exists x_1(S_1) \cdots \exists x_n(S_n) \cdot F$$

to the S-form DQBF

$$Q' := \forall u_1 \cdots \forall u_m \exists x_1(S'_1) \cdots \exists x_n(S'_n) \cdot F\,,$$

where S'_i is the set of universal variables u in S_i for which there exists a sequence $C_1, \ldots, C_m$ of clauses in F satisfying the following conditions:

(a) *u in $\text{vars}(C_1)$ and x_i in $\text{vars}(C_m)$;*

(b) *for each j in $[m-1]$, $\text{vars}(C_j) \cap \text{vars}(C_{j+1})$ contains an existential variable x_k for which u is in S_k.*

Example 9.7. We apply $\mathcal{D}^{\text{std}}$ to the first instance of the equality family, namely

$$\exists x_1 \forall u_1 \exists z_1 \cdot \{\{\bar{x}_1, \bar{u}_1, z_1\}, \{x_1, u_1, z_1\}, \{\bar{z}_1\}\}\,.$$

Here, the dependency set for x_1 is empty, and the dependency set for z_1 is $\{u_1\}$. To apply $\mathcal{D}^{\text{std}}$ to Q, we need to find out whether or not u_1 should be removed from the dependency set for z_1.

For this, we look for a sequence satisfying conditions (a) and (b) of Definition 9.6. It is easy to see that the single clause $\{\bar{x}_1, \bar{u}_1, z\}$ fits the bill. Both variables u_1 and z_1 appear in the clause, satisfying (a). Condition (b) is satisfied vacuously since the sequence is of length 1.

Hence u_1 remains in the dependency set for z_1, and $\mathcal{D}^{\text{std}}(\text{EQ}_1)$ is EQ_1 itself. ■

The standard dependency scheme does not perform so well on our hand-crafted QBF families. In fact, it reduces to the weakest possible dependency scheme, i.e. the identity mapping $\mathcal{D}^{\text{trv}}$, on all of them. Unfortunately for $\mathcal{D}^{\text{std}}$, connections based on common variables are too weak to identify independencies in these formulas.

We show this first for the equality family.

Fact 9.8. *For each natural number n, $\mathcal{D}^{\text{std}}(\text{EQ}_n) = \mathcal{D}^{\text{trv}}(\text{EQ}_n)$.*

Proof. In EQ_n, the dependency set for each x_i is empty. Since a dependency scheme maps to stronger DQBFs, the dependency set for x_i in $\mathcal{D}^{\text{std}}(\text{EQ}_n)$ is also empty, for any $\mathcal{D}$. Therefore we need only show that the dependency set for each t_i in EQ_n, namely $\{u_j\}_{j\in[n]}$, is equal to that of $\mathcal{D}^{\text{std}}(\text{EQ}_n)$.

Now, if we put $C_1 := \{x_j, u_j, z_j\}$ and $C_2 := \{\bar{z}_1, \dots, \bar{z}_n\}$, the sequence C_1, C_2 satisfies both

(a) $u_j \in \text{vars}(C_1)$ and $z_i \in \text{vars}(C_2)$,

(b) $z_j \in \text{vars}(C_1) \cap \text{vars}(C_2)$, where u_j is in the dependency set for z_j in EQ_n.

Hence u_j is in the dependency set for z_i in $\mathcal{D}^{\text{std}}(\text{EQ}_n)$. □

Something similar happens for the other families $\mathcal{PA}$ and $\mathcal{KB}$. It is easy to see, by inspecting the formulas, that $\mathcal{D}^{\text{std}}$ connections exist between all relevant variable pairs.

Fact 9.9. *For each natural number n, and each Q in $\{\text{EQ}_n, \text{PA}_n, \text{KB}_n\}$, $\mathcal{D}^{\text{std}}(Q) = Q$.*

Whereas Fact 9.9 does not look so good for $\mathcal{D}^{\text{std}}$, it is useful for proving lower bounds later on, when we incorporate $\mathcal{D}^{\text{std}}$ into some QBF proof systems.

Now we turn to a stronger dependency scheme that can identify independencies in our QBF families.

9.3.2 The reflexive resolution path dependency scheme

Whereas connections in the standard dependency scheme are based on common variables, the reflexive resolution path dependency scheme ($\mathcal{D}^{\text{rrs}}$) also takes polarity into account. The connecting existential variables must appear in opposite polarities in the connected clauses. This turns out to be a much stronger method for dependency awareness.

Definition 9.10 (reflexive resolution path dependency scheme [63]). *The* reflexive resolution path dependency scheme $\mathcal{D}^{\text{rrs}}$ *is the function that maps a QBF*

$$Q := \forall u_1 \cdots \forall u_m \exists x_1(S_1) \cdots \exists x_n(S_n) \cdot F$$

to the S-form DQBF

$$Q' := \forall u_1 \cdots \forall u_m \exists x_1(S_1') \cdots \exists x_n(S_n') \cdot F\,,$$

where S_i' is the set of universal variables u in S_i for which there exists a sequence $C_1, \ldots, C_k$ of clauses in F and a sequence $a_1, \ldots, a_{k-1}$ of existential literals satisfying the following conditions:

(a) *$u \in C_1$ and $\bar{u} \in C_m$,*

(b) *for some j in $[k-1]$, $x_i = \text{var}(a_j)$,*

(c) *for each j in $[k-1]$, $a_j \in C_j$ and $\bar{a}_j \in C_{j+1}$,*

(d) *for each j in $[k-1]$, u is in the dependency set for* $\text{var}(a_j)$ *in Q,*

(e) *for each $j \in [k-2]$, $\text{var}(a_j) \neq \text{var}(a_{j+1})$.*

Example 9.11. Now we consider what happens when we apply $\mathcal{D}^{\text{rrs}}$ to the first instance of the equality family. Once again, we need to find out whether u_1 remains in the dependency set for z_1.

We need to find a sequence of clauses and a sequence of literals that satisfy conditions (a)-(e) in the definition of $\mathcal{D}^{\text{rrs}}$, with respect to u_1 and z_1. It is easy to see that no clause in the sequence can be a unit clause. Hence no such sequence can exist, because the negative literal $\bar{z}_1$ only appears in a unit clause.

Hence u_1 is removed from the dependency set for z_1 and $\mathcal{D}^{\text{rrs}}(Q)$ is the DQBF

$$\forall u_1 \exists x_1(\emptyset) \exists z_1(\emptyset) \cdot \{\{\bar{x}_1, \bar{u}_1, z_1\}, \{x_1, u_1, z_1\}, \{\bar{z}_1\}\}\,,$$

which in this case is also a QBF. ■

We can view $\mathcal{D}^{\text{rrs}}$ as a generalisation of $\mathcal{D}^{\text{std}}$, which uses an improved notion of connection. So it is easy to see that $\mathcal{D}^{\text{rrs}}$ is more general than $\mathcal{D}^{\text{std}}$. In fact, the preceding example already shows it is strictly more general.

Fact 9.12. *$\mathcal{D}^{\text{rrs}}$ is strictly more general than $\mathcal{D}^{\text{std}}$.*

As further examples, we investigate how $\mathcal{D}^{\text{rrs}}$ operates on our hand-crafted formula families. The results are quite different compared to the standard dependency scheme.

On the equality family, $\mathcal{D}^{\text{rrs}}$ works like the strongest possible dependency scheme: all the dependency sets are empty.

Fact 9.13. *For each natural number n, the dependency sets of $\mathcal{D}^{\text{rrs}}(\text{EQ}_n)$ are all empty.*

Proof. Along the same lines of the proof of Fact 9.8, the dependency set for each x_i in EQ_n is empty, so its dependency set in $\mathcal{D}^{\text{rrs}}(\text{EQ}_n)$ is also empty. So we need only show that the dependency set for each z_i in $\mathcal{D}^{\text{rrs}}(\text{EQ}_n)$ is empty.

Aiming for contradiction, suppose that $C_1, \dots, C_m$ is a sequence of clauses in eq_n and $a_1, \dots, a_{m-1}$ a sequence of existential literals satisfying conditions (a)-(e) of Definition 9.10 with respect to the universal variable u_j and the existential variable z_i.

By (a), C_1 is the unique clause in eq_n containing literal u_j, namely $\{u_j, x_j, z_j\}$. Therefore $a_1 = z_j$, by (d). By (c), C_2 is the unique clause in eq_n containing $\bar{z}_j$, namely $\{\bar{z}_1, \dots, \bar{z}_n\}$. Then, by (e), $a_2 = \bar{z}_k$, for some $k \neq j$. Hence, by (c), C_3 contains z_k, and must be either $\{x_k, u_k, z_k\}$ or $\{\bar{x}_k, \bar{u}_k, z_k\}$. In both clauses, z_k is the only existential variable whose dependency set contains u_j. Hence, by (e), C_3 concludes the sequence; that is, $m = 3$. Since $k \neq j$, we have $\bar{u}_j \notin C_m$, contradicting (a). $\square$

In contrast, on the parity family, $\mathcal{D}^{\text{rrs}}$ reduces to the trivial dependency scheme.

Fact 9.14. *For each natural number n, $\mathcal{D}^{\text{rrs}}(\text{PA}_n) = Q$.*

Proof. To show that $\mathcal{D}^{\text{rrs}}$ does not recognise any new independencies, we must show, for each i in $[n]$, that there exists a sequence of clauses and a sequence of literals satisfying the five conditions of Definition 9.10 with respect to u and z_i.

We take the sequence of clauses

$$\{u, \bar{z}_n\}, \{\bar{x}_n, \bar{z}_{n-1}, z_n\} \cdots \{\bar{x}_{i+1}, \bar{z}_i, z_{i+1}\}, \{\bar{x}_{i+1}, z_i, \bar{z}_{i+1}\}, \dots, \{\bar{x}_n, z_{n-1}, \bar{z}_n\}, \{\bar{u}, z_n\}\,,$$

and the sequence of literals

$$\bar{z}_n, \bar{z}_{n-1}, \dots, \bar{z}_{i+1}, \bar{z}_i, \bar{z}_{i+1}, \dots \bar{z}_n\,.$$

It is easy to verify that they satisfy all five conditions. $\square$

On the KBKF family, $\mathcal{D}^{\text{rrs}}$ lies somewhere in the middle of what we see with the equality and parity families. $\mathcal{D}^{\text{rrs}}$ does identify some non-trivial independencies in $\mathcal{KB}$, and later on in Chapter 10, we show how they are crucial for constructing short refutations.

Fact 9.15. *For each natural number n and each $i, j \in [n]$, $i \neq j$, the dependency set for z_i in $\mathcal{D}^{\text{rrs}}(\text{KB}_n)$ does not contain u_j.*

Proof. Let $n \in \mathbb{N}$ and let $i, j \in [n]$ with $i \neq j$. Suppose that $C_1, \dots, C_k \in \text{kb}_n$ and $a_1, \dots, a_{k-1}$ are sequences of clauses and literals respectively, satisfying the five conditions of Definition 9.10 with respect to u_i and z_j.

By condition (b), the literal sequence contains a literal in the variable z_j. Observe that, in the matrix kb_n, the positive literal z_j occurs only in the clauses $\{\bar{u}_j, z_j\}$ and $\{u_j, z_j\}$. Hence, by condition (c), there is some clause C_r in the clause sequence which is one of these clauses. Since z_j is the only existential literal in C_r, the clause must be an endpoint of the sequence by condition (e), and hence we must have $r = 1$ or $r = k$. However, since $i \neq j$, this implies that either $u_i \notin C_1$ or $u_i \notin C_k$, contradicting condition (a). $\square$

9.3.3 Full exhibition of $\mathcal{D}^{\text{std}}$ and $\mathcal{D}^{\text{rrs}}$

It is known that both $\mathcal{D}^{\text{std}}$ and $\mathcal{D}^{\text{rrs}}$ are fully exhibited dependency schemes. The fact that $\mathcal{D}^{\text{rrs}}$ is fully exhibited follows directly from Thereoms 3 and 4 in [71]. The result is also proved (somewhat differently) in [6].

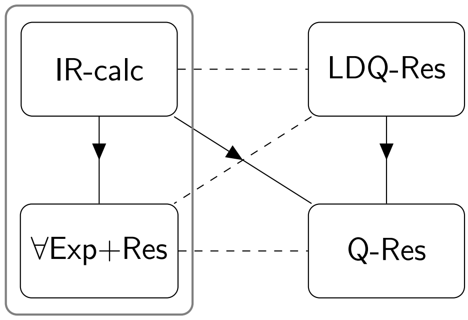
Theorem 9.16 ([71, 6])**.** *The reflexive resolution path dependency scheme is fully exhibited.*

Since $\mathcal{D}^{\text{rrs}}$ is fully exhibited and more general than $\mathcal{D}^{\text{std}}$ (Fact 9.12), $\mathcal{D}^{\text{std}}$ is also fully exhibited, by Fact 9.5.

Corollary 9.17. *The standard dependency scheme is fully exhibited.*

Chapter 10

Dependency Schemes in Expansion



In this chapter, we investigate models of dependency-aware expansion solving, focusing on the two systems highlighted above. We show how to incorporate dependency schemes into these models, and prove some complexity results. It turns out that using the reflexive resolution path dependency scheme can exponentially shorten expansion refutations. Thus, dependency schemes can potentially foster improved expansion-based solving.

Incorporating dependency schemes

Adding a dependency scheme to a QBF expansion system like $\forall$Exp+Res takes the form of a particular fragment of $\forall$Exp+Res. The particular fragment corresponds to the image of the dependency scheme. Since the image of a dependency scheme is a set of S-form DQBFs, we will be using $\forall$Exp+Res extended to S-form.

This is essentially the dependency-aware workflow from Figure 9.1, built into $\forall$Exp+Res. In terms of correctness, two things are crucial for the detour into DQBF. First, the dependency scheme is fully exhibited, guaranteeing that the translation preserves truth values. Second, S-form $\forall$Exp+Res is refutationally sound and complete on the image of the dependency scheme.

Organisation of the chapter

In Section 10.1, we show that much of the universal expansion paradigm lifts straightforwardly to S-form DQBF, including the proof system $\forall$Exp+Res. In Section 10.2, we incorporate dependency schemes into $\forall$Exp+Res. We prove that the equality family has linear-size refutations under $\mathcal{D}^{\mathrm{rrs}}$, but requires exponential-size under $\mathcal{D}^{\mathrm{std}}$. In Section 10.3, we extend the picture to include instantiation. We prove that the Kleine Büning family admits linear-size refutations under $\mathcal{D}^{\mathrm{rrs}}$, but requires exponential-size for $\mathcal{D}^{\mathrm{std}}$.

10.1 Universal expansion revisited

The universal expansion paradigm extends to S-form DQBF in a very natural way. If we were to look back at our introduction to universal expansion for QBF in Sections 4.1 and 4.2, we would find that all of it is applicable to S-form DQBFs in general.

QBFs are the subset of S-form DQBFs for which the dependency sets form a total order with respect to set inclusion. For the most of our work on QBF expansion, we did not use this property, and wherever it is not used, definitions and results lift immediately to DQBF.

Expansion of a DQBF

A good example is the definition of expansion (Definition 4.2), which makes no assumptions about the dependency sets, and lifts straight away to DQBF.

Definition 10.1 (expansion of a DQBF)**.** *The* expansion *of an S-form DQBF*

$$Q := \forall u_1 \cdots \forall u_m \exists x_1(S_1) \cdots \exists x_n(S_n) \cdot F\,,$$

is the CNF

$$\exp(Q) := \bigcup_{\mu \in \langle U \rangle} F[\mu \cup \{x_i \mapsto x_i^{\mu\restriction_{S_i}} : i \in [n]\}]\,.$$

Nor were any restrictions placed on dependency sets throughout Subsection 4.1.1, so the whole semantic connection between models and satisfying assignments for the expansion can be established for S-form. The main point is the analogue of Corollary 4.6.

Corollary 10.2. *An S-form DQBF is true if, and only if, its expansion is satisfiable.*

$\forall$Exp+Res for DQBF

One can also verify that we did not make any assumptions about the dependency sets throughout Section 4.2, so we can define $\forall$Exp+Res in exactly the same way.

Definition 10.3 (S-form $\forall$Exp+Res [15]). *A* $\forall$Exp+Res refutation *of an S-form DQBF Q is a Resolution refutation of the expansion of Q.*

Example 10.4. Consider again the S-form DQBF

$$\forall u_1 \forall u_2 \exists x_1(\{u_1\}) \exists x_2(\{u_2\}) \cdot \{\{\bar{u}_1, u_2, x_1\}, \{u_1, \bar{u}_2, x_2\}, \{\bar{u}_1, \bar{u}_2, \bar{x}_1, \bar{x}_2\}\}$$

from Example 8.3. The total expansion is the CNF

$$\{\{x_1^{\{u_1\}}\}, \{x_2^{\{u_2\}}\}, \{\bar{x}_1^{\{u_1\}}, \bar{x}_2^{\{u_2\}}\}\},$$

which has an obvious Resolution refutation, namely,

$$\{x_1^{\{u_1\}}\}, \{x_2^{\{u_2\}}\}, \{\bar{x}_1^{\{u_1\}}, \bar{x}_2^{\{u_2\}}\}, \{\bar{x}_1^{\{u_1\}}\}, \emptyset$$

This constitutes an S-form $\forall$Exp+Res refutation. ■

The proof that $\forall$Exp+Res is a refutational proof system for S-form DQBF is also identical to that of its QBF analogue, Theorem 4.9, thanks to the absence of any restriction on the dependency sets.

Theorem 10.5 ([15]). $\forall$Exp+Res *is a proof system for the language* FSDQBF.

10.2 Dependency schemes in expansion

Now that we know $\forall$Exp+Res is a refutational proof system for S-form DQBF, it is time to incorporate a fully exhibited dependency scheme.

10.2.1 Parametrisation of $\forall$Exp+Res

First we define what it means to use a dependency scheme in $\forall$Exp+Res. Actually this is quite simple. We first apply the scheme to the QBF, then try to refute the resulting DQBF by finding a Resolution refutation of its expansion.

Definition 10.6 ($\forall$Exp($\mathcal{D}$)+Res). *A* $\forall$Exp($\mathcal{D}$)+Res refutation *of a QBF Q is a Resolution refutation of the expansion of $\mathcal{D}(Q)$.*

Example 10.7. We show a $\forall\mathsf{Exp}(\mathcal{D}^{\mathrm{rrs}})$+$\mathsf{Res}$ refutation of the first instance EQ_1 of the equality family. By Fact 9.13, the image of EQ_1 under $\mathcal{D}^{\mathrm{rrs}}$ is the S-form DQBF

$$\forall u_1 \exists x_1(\emptyset) \exists z_1(\emptyset) \cdot \{\{\bar{x}_1, \bar{u}_1, z_1\}, \{x_1, u_1, z_1\}, \{\bar{z}_1\}\},$$

whose total expansion is the CNF

$$\{\{\bar{x}_1^{\emptyset}, z_1^{\emptyset}\}, \{x_1^{\emptyset}, z_1^{\emptyset}\}, \{\bar{z}_1^{\emptyset}\}\}.$$

Hence the sequence of clauses

$$\{\bar{x}_1^{\emptyset}, z_1^{\emptyset}\}, \{x_1^{\emptyset}, z_1^{\emptyset}\}, \{z_1^{\emptyset}\}, \{\bar{z}_1^{\emptyset}\}, \emptyset$$

forms a $\forall\mathsf{Exp}(\mathcal{D}^{\mathrm{rrs}})$+$\mathsf{Res}$ refutation of EQ_1. ■

Notice how the universal assignments in the annotations have disappeared with the dependency scheme. This demonstrates how dependency schemes help in expansion solving: they reduce the size of the expansion.

As it happens, full exhibition characterises the dependency schemes which can be used correctly in expansion. Dependency schemes that are not fully exhibited give rise to unsound proof systems.

Theorem 10.8. $\forall\mathsf{Exp}(\mathcal{D})$+$\mathsf{Res}$ *is a proof system for the language* FQBF *if, and only if,* $\mathcal{D}$ *is fully exhibited.*

Proof. For the "if" direction, suppose that $\mathcal{D}$ is fully exhibited. We show that $\forall\mathsf{Exp}(\mathcal{D})$+$\mathsf{Res}$ is a proof system for FQBF.

Soundness and completeness. By the full exhibition of $\mathcal{D}$, a QBF Q is false if, and only if, $\mathcal{D}(Q)$ is false. By Corollary 10.2, $\mathcal{D}(Q)$ is false if, and only if, its expansion is unsatisfiable. By the soundness and completeness of Resolution, the expansion of $\mathcal{D}(Q)$ is unsatisfiable if, and only if, it has a Resolution refutation. *Checkability.* Since $\mathcal{D}$ is polynomial-time computable, it can be checked in polynomial time whether a clause belongs to the expansion of $\mathcal{D}(Q)$. Checkability of $\forall\mathsf{Exp}(\mathcal{D})$+$\mathsf{Res}$ then follows from the checkability of Resolution.

Now for the "only if" direction. Suppose that $\mathcal{D}$ is not fully exhibited. Then there exists a true QBF Q for which $\mathcal{D}(Q)$ is false. Since $\forall\mathsf{Exp}$+Res is complete for false S-form DQBFs (Theorem 10.5), there exists a $\forall\mathsf{Exp}$+Res refutation of $\mathcal{D}(Q)$, that is, there exists a $\forall\mathsf{Exp}(\mathcal{D})$+$\mathsf{Res}$ refutation of Q. Since $\forall\mathsf{Exp}(\mathcal{D})$+$\mathsf{Res}$ refutes a true QBF, it is not a proof system for FQBF. □

Simulations by generality

Moving to a more general scheme can indeed shorten proofs – we will we see some examples of this shortly. However, as one might expect, moving to a less general scheme can never shorten proofs.

Fact 10.9. *Given two fully exhibited dependency schemes $\mathcal{D}$ and $\mathcal{D}'$,*

$$\mathcal{D} \text{ is more general than } \mathcal{D}' \quad \Rightarrow \quad \forall\mathsf{Exp}(\mathcal{D}')\text{+}\mathsf{Res} \leq_p \forall\mathsf{Exp}(\mathcal{D})\text{+}\mathsf{Res}$$

Proof. Let $\mathcal{D}$ and $\mathcal{D}'$ be two fully exhibited dependency schemes, and suppose that $\mathcal{D}$ is more general than $\mathcal{D}'$. A $\forall\mathsf{Exp}(\mathcal{D}')$+$\mathsf{Res}$ refutation of a QBF Q can always be turned into an $\forall\mathsf{Exp}(\mathcal{D})$+$\mathsf{Res}$ refutation by shortening the annotations; more precisely, by dropping the assignment to u in the annotation to variable x whenever u belongs to the dependency set for x in $\mathcal{D}'(Q)$, but not in $\mathcal{D}(Q)$. $\square$

Since $\mathcal{D}^{\text{rrs}}$ is fully exhibited (Theorem 9.16) and more general than $\mathcal{D}^{\text{std}}$ (Fact 9.12), which is also fully exhibited (Corollary 9.17), we obtain the following simulations.

Fact 10.10. $\forall\mathsf{Exp}$+$\mathsf{Res} \leq_p \forall\mathsf{Exp}(\mathcal{D}^{\text{std}})$+$\mathsf{Res} \leq_p \forall\mathsf{Exp}(\mathcal{D}^{\text{rrs}})$+$\mathsf{Res}$.

10.2.2 Separation under $\mathcal{D}^{\text{rrs}}$

Now we show that $\forall\mathsf{Exp}(\mathcal{D}^{\text{rrs}})$+$\mathsf{Res}$ is exponentially stronger than $\forall\mathsf{Exp}(\mathcal{D}^{\text{std}})$+$\mathsf{Res}$.

We have already seen that the standard dependency scheme is rendered ineffective on our handcrafted families $\mathcal{EQ}$ and $\mathcal{KB}$, as it is reduced to the identity mapping (Fact 9.9). This means that $\forall\mathsf{Exp}(\mathcal{D}^{\text{std}})$+$\mathsf{Res}$ refutations of any of these formulas are also $\forall\mathsf{Exp}$+Res refutations. As a result, the exponential lower bounds for $\mathcal{EQ}$ (Corollary 4.15) and $\mathcal{KB}$ (Corollary 4.19) still hold under the standard dependency scheme.

Fact 10.11. *The formula families $\mathcal{EQ}$ and $\mathcal{KB}$ both require $\forall\mathsf{Exp}(\mathcal{D}^{\text{std}})$+$\mathsf{Res}$ refutations of size 2^n.*

For the upper bound, we show that equality family admits short proofs under the reflexive resolution path dependency scheme.

Lemma 10.12. *The formula family $\mathcal{EQ}$ admits linear-size $\forall\mathsf{Exp}(\mathcal{D}^{\text{rrs}})$+$\mathsf{Res}$ refutations.*

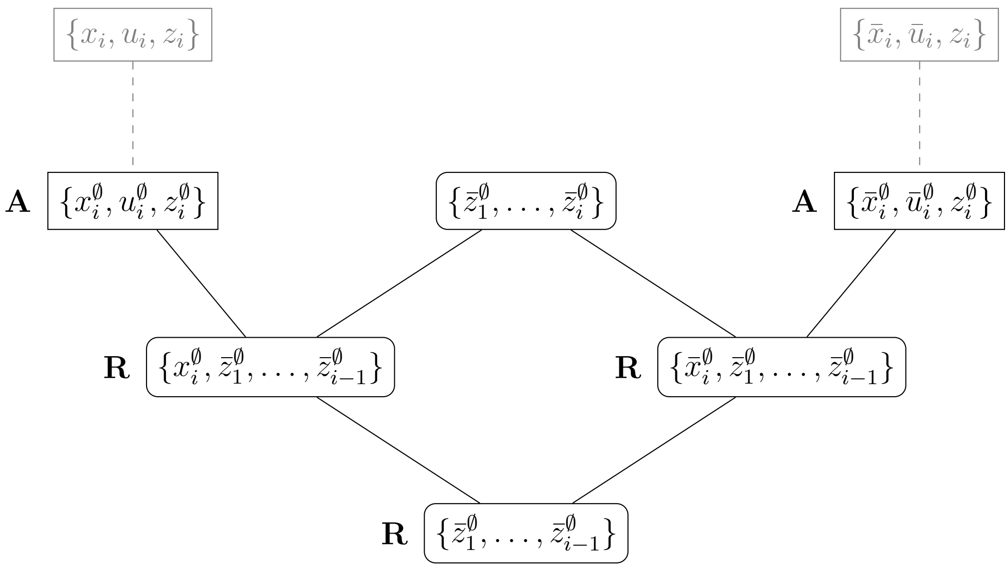


Figure 10.1: Portion of a linear-size $\forall\mathsf{Exp}(\mathcal{D}^{\mathrm{rrs}})+\mathsf{Res}$ refutation of EQ_n.

Proof. For each i in $[n]$, we define the CNF F_i consisting of the clauses

$$\begin{gathered}\{x_i^\emptyset, z_i^\emptyset\}\,,\\ \{\bar{x}_i^\emptyset, z_i^\emptyset\}\,,\\ \{\bar{z}_1^\emptyset, \ldots, \bar{z}_i^\emptyset\}\,.\end{gathered}$$

By Fact 9.13, the existential dependency sets of $\mathcal{D}^{\mathrm{rrs}}(\mathrm{EQ}_n)$ are empty. Hence, each clause in F_n can be introduced as an axiom. Also, F_1 is a constant-size unsatisfiable CNF, which therefore has a constant-size Resolution refutation.

We complete a linear-size refutation by showing that, for each i in $[n-1]$, the clauses in F_i can be derived from F_{i+1}. Now, two of the clauses of F_i, namely

$$\begin{gathered}\{x_i^\emptyset, z_i^\emptyset\}\,,\\ \{\bar{x}_i^\emptyset, z_i^\emptyset\}\,,\end{gathered}$$

already belong to the expansion of $\mathcal{D}^{\mathrm{rrs}}(\mathrm{EQ}_n)$, and can be introduced as axioms. As shown in Figure 10.1, from F_{i+1} we can derive the remaining clause

$$\{\bar{z}_1^\emptyset, \ldots, \bar{z}_i^\emptyset\} \in F_i\,,$$

in a constant number of resolution steps. $\square$

Fact 10.11 and Lemma 10.12 together show that $\forall\mathsf{Exp}(\mathcal{D}^{\mathrm{std}})+\mathsf{Res}$ does not simulate $\forall\mathsf{Exp}(\mathcal{D}^{\mathrm{rrs}})+\mathsf{Res}$.

Theorem 10.13. $\forall\mathsf{Exp}(\mathcal{D}^{\mathrm{std}})+\mathsf{Res} <_p \forall\mathsf{Exp}(\mathcal{D}^{\mathrm{rrs}})+\mathsf{Res}$.

10.3 DQBF Instantiation

The whole framework for incorporating dependency schemes into expansion works perfectly well for instantiation. In this section, we go through the motions again, swapping $\forall$Exp+Res for IR-calc. In Subsection 10.3.2, we show that incorporating $\mathcal{D}^{\text{rrs}}$ into IR-calc admits short refutations of $\mathcal{KB}$.

10.3.1 IR-calc revisited

Earlier in the chapter, we saw that much of our work on the universal expansion paradigm for QBF lifted immediately to S-form DQBF, simply because we made no assumptions about the dependency sets. Likewise, some of our work on QBF instantiation made no assumptions about the dependency sets; in particular, the definition of IR-calc for QBF (Definition 6.4) is also appropriate for S-form DQBFs. For convenience, we recall the definition below.

Definition 10.14 (S-form IR-calc [15]). *An* IR-calc refutation *of an S-form DQBF*

$$Q := \forall u_1 \cdots \forall u_m \exists x_1(S_1) \cdots \exists x_n(S_n) \cdot F\,,$$

is a sequence $C_1, \dots, C_k$ in which C_k is the empty clause and at least one of the following holds for each i in $[k]$:

A Axiom: *C_i is a clause in the weak expansion of Q;*

R Resolution: *$C_i = \text{res}(C_r, C_s, p)$, for some $r, s < i$ and existential literal p;*

I Instantiation: *$C_i = \text{inst}(C_r, \mu, P)$, for some $r < i$ and universal assignment μ;*

W Weakening: *C_i is $\mathbb{L}$, or is subsumed by C_r for some $r < i$.*

Moreover, the proof of soundness of IR-calc for QBF, which consisted of a translation into $\forall$Exp+Res, made no assumptions about dependency sets, and hence exactly the same translation establishes that instantiation is sound for S-form DQBF, based on the soundness of expansion (Theorem 10.5). Similarly the completeness (via p-simulation of $\forall$Exp+Res) and checkability of IR-calc on S-form DQBF are identical to the QBF analogues. So IR-calc indeed forms a refutational proof system for S-form DQBF.

Theorem 10.15 ([15]). IR-calc *is a proof system for the language* FSDQBF*.*

10.3.2 Dependency schemes in instantiation

Incorporating dependency schemes into IR-calc works just like it did for $\forall$Exp+Res. Given Theorem 10.15, it forms a QBF proof system if, and only if, the dependency scheme is fully exhibited.

Definition 10.16 (IR($\mathcal{D}$)-calc)**.** *An* IR($\mathcal{D}$)-calc refutation *of a QBF Q is an* IR-calc *refutation of $\mathcal{D}(Q)$.*

Theorem 10.17. IR($\mathcal{D}$)-calc *is a proof system for the language* FQBF *if, and only if, $\mathcal{D}$ is fully exhibited.*

It is also easy to see that more general schemes always simulate less general ones. The argument is the same as for $\forall$Exp+Res (Fact 10.9); from all annotations in a refutation, we merely delete the assignments that become redundant under the more general scheme.

Fact 10.18. IR-calc $\leq_p$ IR($\mathcal{D}^{\mathrm{std}}$)-calc $\leq_p$ IR($\mathcal{D}^{\mathrm{rrs}}$)-calc.

Short refutations of $\mathcal{KB}$

Now we show that IR($\mathcal{D}^{\mathrm{rrs}}$)-calc admits linear-size refutations of $\mathcal{KB}$.

Lemma 10.19. *The formula family $\mathcal{KB}$ admits linear-size* IR($\mathcal{D}^{\mathrm{rrs}}$)-calc *refutations.*

Proof. We construct linear-size refutations by defining, for each i in $[n-1]$, the CNF F_i consisting of the clauses

$$\begin{aligned}
&\{z_i^{\{\bar{u}_i\}}\}\,,\\
&\{z_i^{\{u_i\}}\}\,,\\
&\{x_{i-1}^{\emptyset}, \bar{x}_i^{\{u_i\}}, \bar{y}_i^{\{u_i\}}\}\,,\\
&\{y_{i-1}^{\emptyset}, \bar{x}_i^{\{\bar{u}_i\}}, \bar{y}_i^{\{\bar{u}_i\}}\}\,,\\
&\{x_i^{\emptyset}, \bar{z}_1^{\emptyset}, \dots, \bar{z}_{i-1}^{\emptyset}, \bar{z}_i^{\{u_i\}}\}\,,\\
&\{y_i^{\emptyset}, \bar{z}_1^{\emptyset}, \dots, \bar{z}_{i-1}^{\emptyset}, \bar{z}_i^{\{\bar{u}_i\}}\}\,.
\end{aligned}$$

Then we show three things:

(a) each clause in F_{n-1} can be introduced as an axiom from KB_n;

(b) for each i in $[n-2]$, F_i can be derived from F_{i+1} in a constant number of steps;

(c) $F_1 \cup \{\{x_1^{\emptyset}, y_1^{\emptyset}\}\}$ can be refuted in a constant number of steps, where $\{x_1^{\emptyset}, y_1^{\emptyset}\}$ can be introduced as an axiom from KB_n

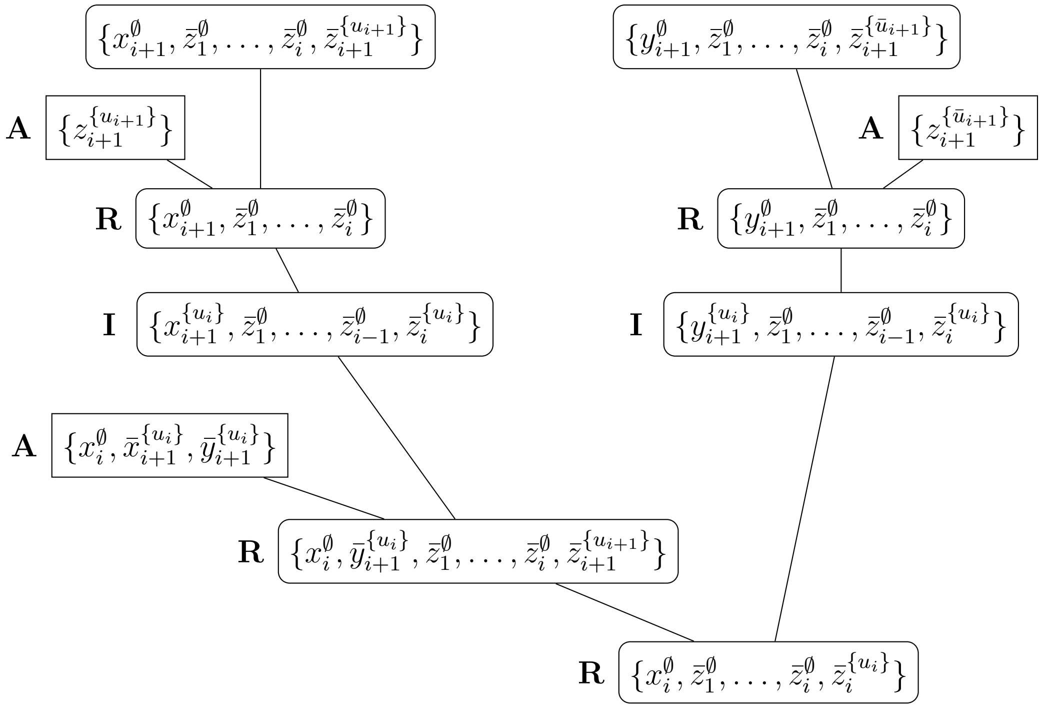


Figure 10.2: Portion of a linear-size $\mathsf{IR}(\mathcal{D}^{\text{rrs}})$-$\mathsf{calc}$ refutation of KB_n.

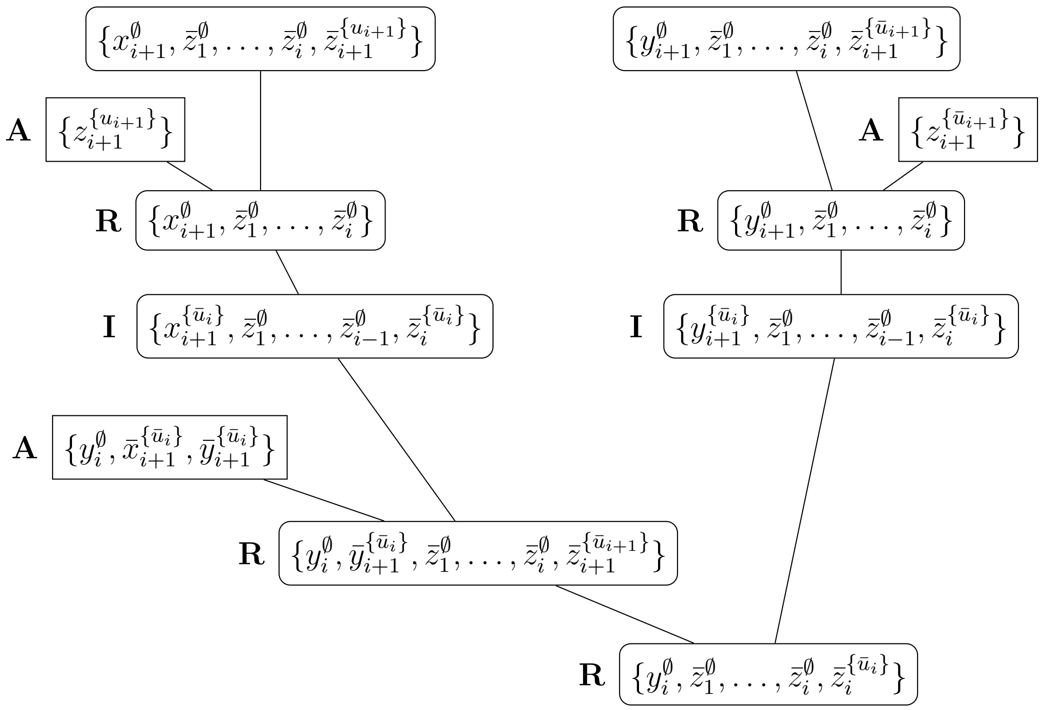


Figure 10.3: Symmetrical portion of the linear-size $\mathsf{IR}(\mathcal{D}^{\text{rrs}})$-$\mathsf{calc}$ refutation of KB_n.

Hence we obtain linear-size $\mathsf{IR}(\mathcal{D}^{\mathrm{rrs}})$-$\mathsf{calc}$ refutations of $\mathcal{EQ}$.

(a) This can be verified easily by inspection.

(b) It is easily verified that the first four clauses of F_i can be introduced as axioms. Constant-size derivations of the remaining two clauses from F_{i+1} are shown in Figures 10.2 and 10.3.

(c) The CNF F_1 contains the clauses

$$\begin{array}{l} \{z_1^{\{\bar{u}_1\}}\}\,, \\ \{z_1^{\{u_1\}}\}\,, \\ \{x_i^{\emptyset}, \bar{z}_i^{\{u_i\}}\}\,, \\ \{y_i^{\emptyset}, \bar{z}_i^{\{\bar{u}_i\}}\}\,. \end{array}$$

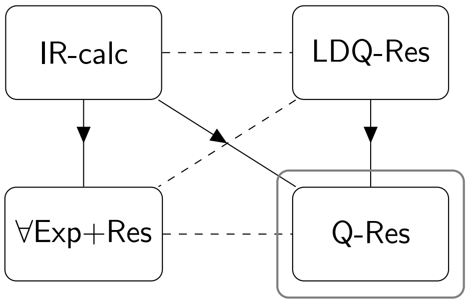
It is easy to see that $F_1 \cup \{\{x_1^{\emptyset}, y_1^{\emptyset}\}\}$ is an unsatisfiable, constant-size CNF. It therefore has a constant-size Resolution refutation. □

On the other hand, since the standard dependency scheme is the identity mapping on $\mathcal{KB}$, the exponential lower bound for $\mathcal{KB}$ in IR-calc (Theorem 6.21) lifts immediately to $\mathsf{IR}(\mathcal{D}^{\mathrm{std}})$-$\mathsf{calc}$. Thus we obtain the following strict simulation.

Theorem 10.20. $\mathsf{IR}(\mathcal{D}^{\mathrm{std}})$-$\mathsf{calc} <_p \mathsf{IR}(\mathcal{D}^{\mathrm{rrs}})$-$\mathsf{calc}$.

Chapter 11

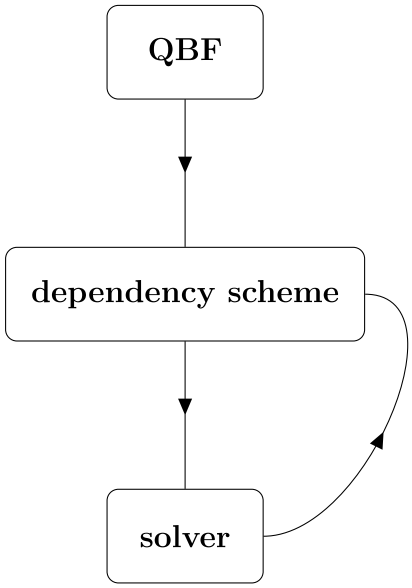
Universal Reduction and Dynamic Dependency Awareness



In the previous chapter we saw how to incorporate dependency schemes into models of expansion-based solving, as fragments of DQBF proof systems. In this chapter we continue to work with models of dependency-aware solving, focusing this time on Q-Resolution. As predicted by the expansion-reduction hypothesis, lifting Q-Res to S-form DQBF doesn't go quite as smoothly: it is incomplete.

But all is not lost. It turns out that incompleteness of S-from Q-Res is not a major obstacle for dependency schemes. The major obstacle is unsoundness, but we won't encounter that until we look at long-distance resolution in the next chapter.

So, we take the same approach, modelling dependency-aware QCDCL QBF solving using fragments of S-form Q-Res. We will see that dependency schemes can indeed shorten Q-Res proofs. In fact, the whole setup for static dependencies is very similar to the previous chapter, and it is merely a formality to verify that the same picture emerges.



The QBF is passed to the dependency scheme.

The dependency scheme computes some dependency information and passes this, along with the instance, to the QBF solver.

The solver solves the QBF using the dependency information, and can also call the dependency scheme for dependency information on the current subformula.

Figure 11.1: Dependency recomputation in QBF solving.

Dynamic dependency awareness

QCDCL solvers work by systematically testing variable assignments until the truth value of the instance can be determined. Variable assignments are constantly being selected and rejected, so that the size of the current assignment, as well as the assignment itself, is always changing. At any search node, the solver is really trying to find the truth value of the QBF *under the application* of the current assignment.

This is very relevant to dependency-aware solving, since application of assignments has a non-trivial interplay with dependency schemes. Applying assignments always preserves independencies, but it may also introduce new ones. We will see that $\mathcal{D}^{\text{rrs}}$, when applied to a QBF under assignment, can determine independencies that cannot be ascertained from the original formula.

This points towards the *dynamic* use of dependency schemes as a new avenue for QBF solving. The new situation, in which the solver can call the dependency scheme during search, is depicted in Figure 11.1. In contrast, the static models that we have seen up to now call the dependency scheme only once at the start.

To model this dynamic setting, we propose a new calculus $\mathsf{dyn\text{-}Q}(\mathcal{D})\mathsf{\text{-}Res}$. We will see that the dynamic use of $\mathcal{D}^{\text{rrs}}$ can have exponentially shorter proofs, compared to the static approach we have seen up to now.

Organisation of the chapter

We begin with a recap of the static approach in Section 11.1. In Section 11.2, we introduce the model for dynamic dependency awareness. We prove that it is sound and complete for fully exhibited dependency schemes, and exponentially separated from the static model in the case of $\mathcal{D}^{\text{rrs}}$.

11.1 Static dependency awareness

A great deal of what was said about dependency schemes in expansion carries over to Q-Res. In the first section of this chapter, we quickly show how.

11.1.1 S-form Q-Res

S-form Q-Res can be defined just as it was for QBF. Even the notion of trailing literal need not change: a universal literal a belonging to a clause C is *trailing* in C with respect to P when $\text{var}(a)$ does not belong to any of the dependency sets for the existential variables in C.

For example, with respect to the S-form prefix $\forall u_1 \forall u_2 \exists x_1(u_1) \exists x_2(u_2)$, we have

(a) $\bar{u}_2$ is trailing in $\{\bar{u}_1, \bar{u}_2, x_1\}$,

(b) $\bar{u}_1$ is trailing in $\{\bar{u}_1, \bar{u}_2, \bar{x}_2\}$,

(c) no literals are trailing in $\{\bar{u}_1, \bar{u}_2, \bar{x}_1, \bar{x}_2\}$,

Definition 11.1 (S-form Q-Res [35])**.** *A* Q-Res *derivation from a QBF $Q := P \cdot F$ is a sequence $C_1, \ldots, C_k$ of non-tautological clauses in which at least one of the following holds for each $i \in [k]$:*

A Axiom: *C_i is a clause in F;*

R Resolution: *$C_i = \text{res}(C_r, C_s, p)$, for some $r, s < i$ and existential literal p;*

U Reduction: *$C_i = C_r \setminus \{a\}$, for some $r < i$, where a is universal and trailing in C_r with respect to P;*

W Weakening: *C_i is $\mathbb{L}$, or is subsumed by C_r for some $r < i$.*

Example 11.2. An S-form Q-Res refutation of

$$\forall u_1 \forall u_2 \exists x_1(\{u_1\}) \exists x_2(\{u_2\}) \cdot \{\{\bar{u}_1, \bar{x}_1, \bar{x}_2\}, \{\bar{u}_1, x_1, \bar{x}_2\}, \{\bar{u}_2, x_2\}\}$$

is shown in Figure 11.2. ■

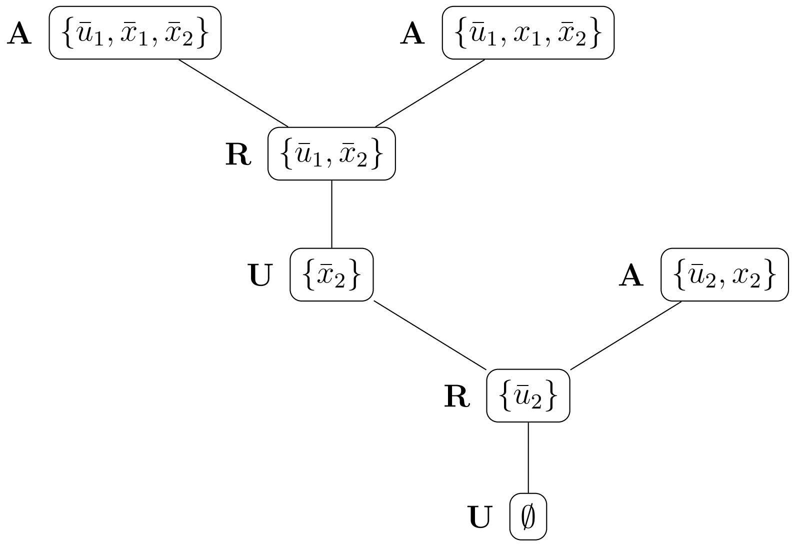


Figure 11.2: An S-form Q-Res refutation.

Soundness

We prove that S-form Q-Res is sound by showing that the rules are logically correct at the DQBF level. This means that every model for the input formula models the whole collection of derived clauses, under the given prefix.

Lemma 11.3 ([1])**.** *If an S-form DQBF has a* Q-Res *refutation, then it is false.*

Proof. Let $\pi := C_1, \ldots, C_k$ be a Q-Res refutation of an S-form DQBF $Q := P \cdot F$. For each i in $[k]$, let $F_i = \{C_1, \ldots, C_i\}$.

Aiming for contradiction, suppose that Q has a model $f := \{f_i\}_{i \in [n]}$. We prove by induction on $i \in [k]$ that f models $P \cdot F_i$. Hence at step $i = k$, we reach a contradiction, since F_k contains the empty clause C_k.

The base case $i = 1$ is trivial, since C_1 is an axiom, and belongs to F.

For the inductive step, let $1 < i \leq k$, suppose that f is a model for $P \cdot F_{i-1}$, and let μ be a total assignment to the universal variables of Q. If C_i is an axiom, the inductive step is identical to the base case, so we consider three further cases. In each case we show that

$$\sigma := \mu \cup \{f_i(\mu\restriction_{S_i})\}_{i \in [n]}$$

satisfies C_i, and hence f is a model for $P \cdot F_{i-1} \cup \{C_i\} = P \cdot F_i$.

R Suppose that $C_i = \text{res}(C_a, C_b, p)$ for some $a, b < i$, and some existential literal p. Then, since both C_a and C_b are in F_{i-1}, σ satisfies both of them, by the inductive hypothesis. Hence σ satisfies C_i by the logical correctness of propositional Resolution.

U Suppose that $C_i = \mathrm{red}(C_a, P)$ for some $a < i$. Let ν be the negation of $C_a \setminus C_i$, i.e. the universal assignment falsiying the reduced literals.

Now, for each x_i in $\mathrm{vars}_\exists(C_i)$, we have

$$f_i((\nu \circ \mu)\restriction_{S_i}) = f_i(\mu\restriction_{S_i})\,.$$

Aiming for contradiction, suppose that σ falsifies C_i. But then

$$\tau := (\nu \circ \mu) \cup \{f_i((\nu \circ \mu)\restriction_{S_i})\}_{i\in[n]}$$

falsifies C_i. Since ν is a subset of τ, τ also falsifies C_a, contradicting the inductive hypothesis.

W Suppose that $C_i = \mathbb{L}$, or is subsumed by C_a with $a \leq i$. In the former case, σ satisfies C_i trivially. In the latter case, σ satisfies C_a by the inductive hypothesis, and therefore satisfies the larger clause C_i. □

Incompleteness

Whereas Q-Res is sound for S-form DQBF, it is not complete. Following the expansion-reduction hypothesis, which tells us that reduction systems prove the existence of countermodels, we should expect to have difficulty refuting false formulas in the set $\mathbb{S}^\circ_\circ$. For example, consider again the S-form DQBF from Example 8.3:

$$\forall u_1 \forall u_2 \exists x_1(\{u_1\}) \exists x_2(\{u_2\}) \cdot \{\{\bar{u}_1, u_2, x_1\}, \{u_1, \bar{u}_2, x_2\}, \{\bar{u}_1, \bar{u}_2, \bar{x}_1, \bar{x}_2\}\}\,.$$

Applying universal reduction to the first two clause, we can derive $\{u_2, x_1\}$ and $\{u_1, x_2\}$. But from there, there are no more reductions, and all possible resolutions produce tautological clauses. So this false DQBF has no refutation.

11.1.2 Dependency schemes in reduction

The fact that Q-Res is incomplete for S-form DQBF is not a major obstacle for incorporating dependency schemes. Unlike unsoundness, incompleteness is harmless.

Incorporating a dependency scheme into Q-Res works much like it did for $\forall$Exp+Res in Chapter 10. The resulting system is the fragment of S-form Q-Res corresponding to the range of the dependency scheme.

Definition 11.4 (Q($\mathcal{D}$)-Res [63])**.** *A* Q($\mathcal{D}$)-Res refutation *of a QBF Q is a* Q-Res *Resolution refutation of $\mathcal{D}(Q)$.*

In contrast to expansion, full exhibition here does not characterise the dependency schemes that give rise to QBF proof systems. In Q-Res, full exhibition is merely a sufficient condition.

Theorem 11.5 ([61]). Q($\mathcal{D}$)-Res *is a proof system for the language* FQBF *if* $\mathcal{D}$ *is fully exhibited.*

Proof. Soundness Let Q be a true QBF. By the full exhibition of $\mathcal{D}$, $\mathcal{D}(Q)$ is a true S-form DQBF, which has no Q-Res refutation by Lemma 11.3. *Completeness* It is easy to see that Q($\mathcal{D}$)-Res always simulates Q-Res, which is complete. *Checkability.* Follows from the checkability of S-form Q-Res. □

11.1.3 Proof complexity of Q($\mathcal{D}$)-Res

In terms of proof complexity, a picture for Q($\mathcal{D}$)-Res emerges which is quite similar to ∀Exp($\mathcal{D}$)+Res.

For example, we have the analogue of Fact 10.10. A more general dependency scheme will remove more universal literals during reduction, so it is easy to see that more a general scheme always simulates a less general one.

Fact 11.6. Q-Res $\leq_p$ Q($\mathcal{D}^{\text{std}}$)-Res $\leq_p$ Q($\mathcal{D}^{\text{rrs}}$)-Res.

Moreover, we can separate $\mathcal{D}^{\text{std}}$ and $\mathcal{D}^{\text{rrs}}$, with essentially the same method as Theorem 10.13.

Theorem 11.7. Q($\mathcal{D}^{\text{std}}$)-Res $<_p$ Q($\mathcal{D}^{\text{rrs}}$)-Res.

The separation can be shown with $\mathcal{EQ}$. As $\mathcal{D}^{\text{std}}$ has no effect on $\mathcal{EQ}$ (Fact 9.8), the exponential lower bound for Q-Res (Theorem 5.18) lifts to Q($\mathcal{D}^{\text{std}}$)-Res. For the upper bound, the construction of linear-size Q($\mathcal{D}^{\text{rrs}}$)-Res refutations is almost identical to those for ∀Exp+Res. The important part of the construction is shown in Figure 11.3. Notice that universal reduction steps marked with $*$ are forbidden in Q-Res, but allowed in Q($\mathcal{D}^{\text{rrs}}$)-Res, since the dependency sets of $\mathcal{D}^{\text{rrs}}(\text{EQ}_n)$ are all empty (Fact 9.13).

11.2 Dynamic dependency awareness

In this section, we make use of a binary operation '⊗', which is a kind of direct product on CNFs. Given two CNFs F and G, we define

$$F \otimes G := \{C \cup D : C \in F, D \in G\}\,.$$

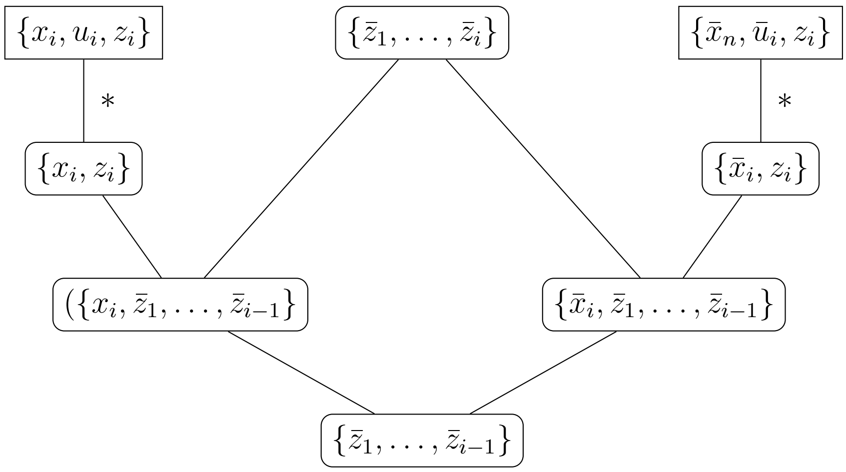


Figure 11.3: Portion of a linear-size $\mathsf{Q}(\mathcal{D}^{\mathrm{rrs}})$-$\mathsf{Res}$ refutation of EQ_n.

11.2.1 Linear assignments

In Chapter 9, we mentioned that QBF solvers are restricted in their choice of variable assignments. Assignments must always respect the prefix dependencies, meaning that a variable cannot be assigned before all the variables from preceding blocks have been assigned.

These are the kind of assignments we would see in use within a QCDCL solver. Moreover, they are the kind of assignments that we want to use in dyn-$\mathsf{Q}(\mathcal{D})$-Res.

Definition 11.8 (linear assignment)**.** *A partial assignment to an S-form DQBF is called* linear *when it assigns the whole dependency set for every assigned existential.*

Linear assignments have a rather special property, namely, if a DQBF is false under a linear assignment, then every model for the DQBF models the negation of the linear assignment.

Lemma 11.9. *Given a linear assignment σ to an S-form DQBF Q, it holds that*

$$Q[\sigma] \text{ is false} \quad \Rightarrow \quad Q \vDash P \cdot \{A\}\,,$$

where P is the prefix of Q and A is the negation of σ.

Proof. If Q is false, the lemma holds vacuouly, so we assume that it is true. Actually, we will prove the contrapositive statement: if Q does not entail $P \cdot \{A\}$, then $Q[\sigma]$ is true.

Let $f := \{f_i\}_{i\in[m]}$ be a model for $\mathcal{D}(Q)$ that does not model $P \cdot \{A\}$. Further, let $\sigma_\forall$ and $\sigma_\exists$ be the universal and existential subassignments of σ.

Now, any universal assignment that does not extend $\sigma_\forall$ will satisfy A. Hence, since f does not model $P \cdot \{A\}$, there exists some total universal assignment μ extending $\sigma_\forall$, for which

$$\mu \cup \{f_i(\mu\restriction_{S_i})\}_{i\in[n]}$$

does not satisfy A.

Now, consider an existential variable x_i assigned in $\sigma_\exists$. Since σ is a linear assignment, $\sigma_\forall$ assigns the whole of S_i, so the function $f_i[\sigma_\forall]$ is constant. Moreover, $f_i[\sigma_\forall]$ does not satisfy A, so it must be identically $\sigma_\exists\restriction_{\{x_i\}}$.

Now we can apply Lemma 3.6. We deduce that $f[\sigma_\forall]$ models $Q[\sigma_\forall]$, and

$$f[\sigma_\forall][\sigma_\exists] \text{ models } Q[\sigma_\forall][\sigma_\exists] = Q[\sigma]\,. \qquad \square$$

Lemma 11.9 does not hold in general for arbitrary assignments. Without going into details, this is the essential reason why the order of variable assignments must be restricted. On the other hand, the propositional version always holds: a CNF entails the negation of any assignment under which it is unsatisfiable.

11.2.2 The proof system dyn-Q($\mathcal{D}$)-Res

Lemma 11.9 tells us that the negation of a falsifying linear assignment is a QBF implicant. Therefore, if a sound system is capable of refuting $Q[\sigma]$, there can be no harm in introducing the negation of σ as if it were an extra axiom. This is the central notion with which we build refutations in dyn-Q($\mathcal{D}$)-Res.

In the following definition, $\circ$ denotes concatenation of sequences.

Definition 11.10 (dyn-Q($\mathcal{D}$)-Res). *Given a dependency scheme $\mathcal{D}$ and a QBF Q, refutations in* dyn-Q($\mathcal{D}$)-Res *are defined recursively by degree:*

- *a degree-0 refutation is a* Q($\mathcal{D}$)-Res *refutation of Q;*
- *for $d \in \mathbb{N}$, a degree-d refutation is a sequence $\pi := \pi_0 \circ \rho_1 \circ \cdots \circ \rho_k$ satisfying*
  - (a) *π_0 is a* Q($\mathcal{D}$)-Res *refutation of Q with extra axioms $A_1, \ldots, A_k$,*
  - (b) *each A_i is the negation of a linear assignment σ_i to $\mathcal{D}(Q)$,*
  - (c) *each ρ_i is a* dyn-Q($\mathcal{D}$)-Res *refutation of $Q[\sigma_i]$,*
  - (d) *the maximum degree of the ρ_i is $d-1$.*

Notice that a $\mathsf{Q}(\mathcal{D})$-Res refutation of a QBF $P \cdot F$ with extra axioms $A_1, \ldots, A_k$ is not the same thing as a $\mathsf{Q}(\mathcal{D})$-Res refutation of $P \cdot F \cup \{A_1, \ldots, A_k\}$, since in the former case the A_i have no effect on the allowable universal reductions, whereas in the latter case they may.

The power of the system lies in the fact that the dependency scheme $\mathcal{D}$ may identify (or *unlock*) new independencies on the restricted formula, meaning that it may be easier to refute the restricted formula $Q[\sigma_i]$ than to derive the extra axiom A_i directly from Q. Of course, the 'referenced' refutation ρ_i, being a refutation of $Q[\sigma_i]$, can make use of these newly unlocked independencies. In this way, the calculus models the recomputation of dependencies during the QCDCL search procedure.

Now we show that full exhibition remains sufficient for soundness even in the dynamic setting.

Lemma 11.11. *Let $\mathcal{D}$ be a fully exhibited dependency scheme. If a QBF has a* dyn-$\mathsf{Q}(\mathcal{D})$-Res *refutation, then it is false.*

Proof. Let π be a dyn-$\mathsf{Q}(\mathcal{D})$-Res refutation of a QBF $Q := P \cdot F$. We prove that Q is false by induction on the degree of π.

For the base case, suppose the degree of π is 0. By definition, a degree 0 refutation is a $\mathsf{Q}(\mathcal{D})$-Res refutation. So Q is false by the soundness of $\mathsf{Q}(\mathcal{D})$-Res (Theorem 11.5).

For the inductive step, suppose the degree of π is $d \geq 1$. Then π is of the form

$$\pi := \pi_0 \circ \rho_1 \circ \cdots \circ \rho_k \,,$$

satisfying conditions (a) to (d) for dyn-$\mathsf{Q}(\mathcal{D})$-Res refutations (Definition 11.10).

By conditions (b), (c) and (d), each ρ_i is refutation of $Q[\sigma_i]$ of degree at most $d-1$, where σ_i is a linear assignment to $\mathcal{D}(Q)$. By the inductive hypothesis, $Q[\sigma_i]$ is false. Applying Lemma 11.9, we see that

$$\mathcal{D}(Q) \vDash P' \cdot \{A_i\} \,,$$

where P' is the prefix of $\mathcal{D}(Q)$, and A is the negated expansion of σ_i. Therefore

$$\mathcal{D}(Q) \vDash P' \cdot F \cup \{A_i : i \in [k]\} \,. \tag{11.1}$$

Now, by condition (a), π_0 is a $\mathsf{Q}(\mathcal{D})$-Res refutation of Q with extra axioms $A_1, \ldots, A_k$. In other words, π_0 is a Q-Res refutation of the S-form DQBF on the right hand side of (11.1). By the soundness of S-form Q-Res (Lemma 11.3), that DQBF is false, therefore so is $\mathcal{D}(Q)$.

Since fully exhibited dependency schemes preserve truth values, Q is also false. ☐

It is now a small step to show that $\mathsf{dyn\text{-}Q}(\mathcal{D})\text{-}\mathsf{Res}$ is a refutational QBF proof system.

Theorem 11.12. $\mathsf{dyn\text{-}Q}(\mathcal{D})\text{-}\mathsf{Res}$ *is a proof system for the language* FQBF *if* $\mathcal{D}$ *is fully exhibited.*

Proof. Soundness. Established by Lemma 11.11. *Completeness.* $\mathsf{dyn\text{-}Q}(\mathcal{D})\text{-}\mathsf{Res}$ trivially p-simulates $\mathsf{Q}(\mathcal{D})\text{-}\mathsf{Res}$, and is therefore complete by Theorem 11.5. *Checkability.* As $\mathcal{D}$ is polynomial-time computable by definition, it can be determined in time polynomial in the size of the prefix whether a partial assignment to $\mathcal{D}(Q)$ is linear. Checkability of $\mathsf{dyn\text{-}Q}(\mathcal{D})\text{-}\mathsf{Res}$ then follows from that of $\mathsf{Q}(\mathcal{D})\text{-}\mathsf{Res}$. $\square$

Simulating dynamic trivial dependencies

As we noted earlier, the appeal of $\mathsf{dyn\text{-}Q}(\mathcal{D})\text{-}\mathsf{Res}$ lies in the ability for the system to go to work on suitable restrictions of the input formula, whereby the system can leverage any independencies that may be 'unlocked' by the restriction. In the case of the trivial dependency scheme, however, there should be no advantage in doing so; the dependencies remain trivial under restriction, so there is nothing to leverage.

If $\mathsf{dyn\text{-}Q}(\mathcal{D})\text{-}\mathsf{Res}$ behaves correctly, then, the static and dynamic systems for the trivial dependency scheme should be equivalent. The next fact establishes that this is indeed the case. We will need it afterwards to prove the separation (Theorem 11.16).

Fact 11.13. $\mathsf{dyn\text{-}Q}(\mathcal{D}^{\mathrm{trv}})\text{-}\mathsf{Res}$ *refutations can be translated to* Q-Res *refutations with no increase in size.*

Proof. We show that a degree-1 dyn-Q(Dtrv)-Res refutation can be transformed into a degree-0 refutation of the same QBF with no increase in size. Hence, given a degree-d refutation π, one can repeatedly search for and transform the first associated degree-1 refutation until no associated refutations are present. This procedure returns a degree-0 refutation, that is, a Q-Res refutation, of size at most $|\pi|$.

To that end, let π be a degree-1 $\mathsf{dyn\text{-}Q}(\mathcal{D}^{\mathrm{trv}})\text{-}\mathsf{Res}$ refutation of a QBF $Q := P \cdot F$. Then π is of the form

$$\pi := \pi_0 \circ \rho_1 \circ \cdots \circ \rho_k$$

as in Definition 11.10. Recall that π_0 is a Q-Res refutation of Q with extra axioms $A_1, \ldots, A_k$, and each ρ_i is a Q-Res refutation of $Q[\sigma_i]$, where σ_i is a linear assignment to Q whose negation is A_i.

Now, consider a particular associated refutation

$$\rho_i := C_1, \ldots, C_{k_i} .$$

We first observe that

$$\rho_i^* := A_i \cup C_1, \ldots, A_i \cup C_{k_i}$$

is a Q-Res derivation from $P \cdot F \otimes \{A_i\}$. To see this, observe that every existential variable in A_i is left of every universal variable in $Q[\sigma_i]$ (since σ_i is a linear assignment to Q), so the addition of A_i to each line cannot block any universal reduction steps.

It follows that

$$\mathrm{seq}(F) \circ C_1 \cup A_i, \ldots, C_{k_i} \cup A_i ,$$

where $\mathrm{seq}(F)$ is the matrix F written as a sequence, is also a Q-Res derivation, since each axiom of ρ_i^* is subsumed by some clause in F. Moreover, by removing weakening steps (Fact 5.9), it is easy to see that we obtain a Q-Res derivation

$$\rho_i' := C_1', \ldots, C_{k_i}'$$

in which $C_j' \subseteq C_j \cup A_i$, for each j in $[k_i]$. In particular, $C_{k_i}' \subseteq A_i$, since C_{k_i} is empty. Therefore

$$\rho_1' \circ \cdots \circ \rho_k' \circ A_1, \ldots, A_k \circ \pi_0$$

is a Q-Res refutation of Q.

Finally, we remove any weakening steps from π_0, while rewriting it as π_0', which renders the subsequence $A_1, \ldots, A_k$ redundant; hence

$$\rho_1' \circ \cdots \circ \rho_k' \circ \pi_0'$$

is a Q-Res refutation of size at most $|\pi|$. □

11.2.3 A further separation under $\mathcal{D}^{\mathrm{rrs}}$

We conclude our investigation into dyn-Q($\mathcal{D}$)-Res by showing an exponential separation over Q($\mathcal{D}$)-Res when $\mathcal{D}$ is $\mathcal{D}^{\mathrm{rrs}}$. This demonstrates that the dynamic application of $\mathcal{D}^{\mathrm{rrs}}$ in principle offers an exponential speedup over the static approach.

Now, formulas that separate dyn-Q($\mathcal{D}^{\mathrm{rrs}}$)-Res from Q($\mathcal{D}^{\mathrm{rrs}}$)-Res can in fact be obtained from those separating Q($\mathcal{D}^{\mathrm{rrs}}$)-Res and Q-Res in a general fashion. Since there are various candidates for the latter separation, we take a general approach to the construction of our separating formulas.

Definition 11.14 (locked QBF)**.** *The* lock *of a QBF* $Q := P \cdot F$ *is the QBF*

$$\mathrm{lock}(Q) := \exists a P \cdot \big(\{\{\bar{a}\}\} \otimes F\big) \cup \big(\{\{a\}\} \otimes \mathrm{trv}(Q)\big) \cup \big(\{\{a\}\}\big)\,,$$

where a is a fresh variable not in $\mathrm{vars}(Q)$*, and*

$$\mathrm{trv}(Q) := \{\{u, x\}, \{\bar{u}, \bar{x}\} : u \textit{ is in the dependency set for } x \textit{ in } Q\}\,.$$

The main idea is that whenever some QBFs $\{Q_n\}_{n\in\mathbb{N}}$ separate $\mathsf{Q}(\mathcal{D}^{\mathrm{rrs}})$-$\mathsf{Res}$ from Q-Res, $\{\mathrm{lock}(Q_n)\}_{n\in\mathbb{N}}$ will separate dyn-$\mathsf{Q}(\mathcal{D}^{\mathrm{rrs}})$-$\mathsf{Res}$ from $\mathsf{Q}(\mathcal{D}^{\mathrm{rrs}})$-$\mathsf{Res}$.

The role of the CNF $\mathrm{trv}(Q)$ is to introduce all the necessary connections so that the reflexive resolution dependencies of $\mathrm{lock}(Q)$ are identical to the trivial dependencies. Of course, all of these connections disappear as soon as the assignment $\{a\}$ is made, which returns the original formula Q. However, until this happens, the reflexive resolution path dependency scheme is useless.

Fact 11.15. *For any QBF Q,*

(a) $\mathrm{lock}(Q)[\{a\}] = Q$*,*

(b) $\mathcal{D}^{\mathrm{rrs}}(\mathrm{lock}(Q)) = \mathrm{lock}(Q)$*.*

Proof. One can verify (a) by inspection. To prove (b), we need show that, for each $x \in \mathrm{vars}_\exists(Q)$ and each universal u in the dependency set for x in Q, there exists a clause sequence and a literal sequence satisfying the five conditions of Definition 9.10, with respect to u and x. Indeed, the clauses

$$\{a, u, x\}, \{a, \bar{u}, \bar{x}\}$$

and the single literal x form suitable sequences. □

We are now prepared to prove the separation formally.

Theorem 11.16. *Given a QBF family $\mathcal{Q}$ which*

(a) *requires* Q-Res *refutations of size* $\Omega(s(n))$*, and*

(b) *admits* $\mathsf{Q}(\mathcal{D}^{\mathrm{rrs}})$-$\mathsf{Res}$ *refutations of size* $O(t(n))$*,*

the QBF family whose n^{th} instance is $\mathrm{lock}(\mathcal{Q}(n))$

(c) *requires* $\mathsf{Q}(\mathcal{D}^{\mathrm{rrs}})$-$\mathsf{Res}$ *refutations of size* $\Omega(s(n))$*, and*

(d) *admits* dyn-$\mathsf{Q}(\mathcal{D}^{\mathrm{rrs}})$-$\mathsf{Res}$ *refutations of size* $O(t(n))$*.*

Proof. Suppose that $\mathcal{Q}$ satisfies conditions (a) and (b), and let $\mathcal{L}$ be the QBF family whose n^{th} instance is lock($\mathcal{Q}(n)$).

First we prove statement (c). Aiming for contradiction, suppose that $\mathcal{Q}$ admits $\textsf{dyn-Q}(\mathcal{D}^{\text{rrs}})\textsf{-Res}$ refutations of size $o(s(n))$. By Fact 11.15(b), $\mathcal{D}^{\text{rrs}}$ is the identity transformation on $\mathcal{L}$, so $\textsf{dyn-Q}(\mathcal{D}^{\text{rrs}})\textsf{-Res}$ refutations of $\mathcal{L}$ are $\textsf{dyn-Q}(\mathcal{D}^{\text{trv}})\textsf{-Res}$ refutations. By Fact 11.13, there exist Q-Res refutations of $\mathcal{L}$ of size $o(s(n))$. Hence, by Fact 11.15(a) and closure of Q-Res under existential assignments (Theorem 5.8), $\mathcal{Q}$ admits Q-Res refutations of size $o(s(n))$, which contradicts (a).

Now for statement (d). Let $\{\rho_1^n\}_{n\in\mathbb{N}}$ be $\textsf{Q}(\mathcal{D}^{\text{rrs}})\textsf{-Res}$ refutations of $\mathcal{Q}$ of size $O(t(n))$. We define the sequences $\{\pi_n\}_{n\in\mathbb{N}}$, where

$$\pi_n := \pi_0^n \circ \rho_1^n, \qquad \pi_0^n := \{\bar{a}\}, \{a\}, \emptyset\,.$$

We show that each π_n is a degree-1 $\textsf{dyn-Q}(\mathcal{D}^{\text{rrs}})\textsf{-Res}$ refutation of $\mathcal{L}(n)$ by verifying conditions (a) to (d) in turn.

(a) Since the unit clause $\{a\}$ belongs to $\mathcal{L}(n)$, π_0^n is a $\textsf{dyn-Q}(\mathcal{D}^{\text{rrs}})\textsf{-Res}$ refutation of Q with an extra axiom $\{\bar{a}\}$.

(b) $\{a\}$ is a linear assignment to $\mathcal{D}^{\text{rrs}}(\mathcal{L})$ and $\{\bar{a}\}$ is its negation.

(c) By Fact 11.15 (a), each ρ_1^n is a $\textsf{Q}(\mathcal{D}^{\text{rrs}})\textsf{-Res}$ refutation of $\mathcal{Q}(n)[\{a\}]$.

(d) The degree of ρ_n^1 is 0.

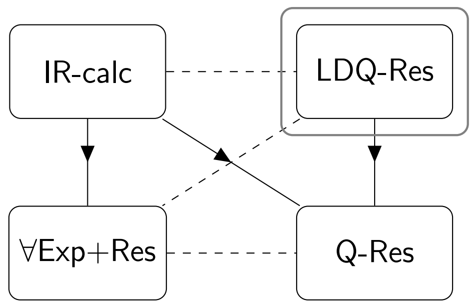
This completes the proof, since it is clear that the size of the π_n is $O(t(n))$. $\square$

The dynamic system $\textsf{dyn-Q}(\mathcal{D}^{\text{rrs}})\textsf{-Res}$ trivially p-simulates $\textsf{Q}(\mathcal{D}^{\text{rrs}})\textsf{-Res}$. Since we already have two formula families separating Q-Res and Q(Drrs)-Res, Theorem 11.16 has the following corollary.

Theorem 11.17. $\textsf{Q}(\mathcal{D}^{\text{rrs}})\textsf{-Res} <_p \textsf{dyn-Q}(\mathcal{D}^{\text{rrs}})\textsf{-Res}$.

Chapter 12

DQBF Merging



It is an unfortunate fact of life that long-distance Q-Resolution is not sound for S-form DQBFs, but what is there to do about it? In this chapter, we use an idea related to the expansion-reduction hypothesis: switch to H-form.

We first take a moment to see the trouble. Suppose we took the definition of deferred LDQ-Res (Definition 7.1) and lifted it straight to S-form DQBF. Unfortunately, this allows refutations of S-form DQBFs in $\mathbb{S}_S^H$, all of which are true. The problem is that their complements have countermodels, and LDQ-Res can find some of them.

For example, the S-form DQBF from Example 8.4 belongs to the class $\mathbb{S}_S^H$. It has a fairly simple deferred LDQ-Res refutation, shown in Figure 12.1. Here, the merged literals $\overset{*}{u}_1$ and $\overset{*}{u}_2$ implicitly represent a countermodel for the complement. One can verify by inspection that they represent exactly the countermodel $\{h_1, h_2\}$ given in Example 8.4.

Building in strategies

We introduce a proof system for false H-form DQBFs called Merge Resolution (M-Res). An M-Res proof looks a little different to the other QBF proofs we have seen. It is a resolution-based system, but rather than a sequence of clauses, a derivation is a

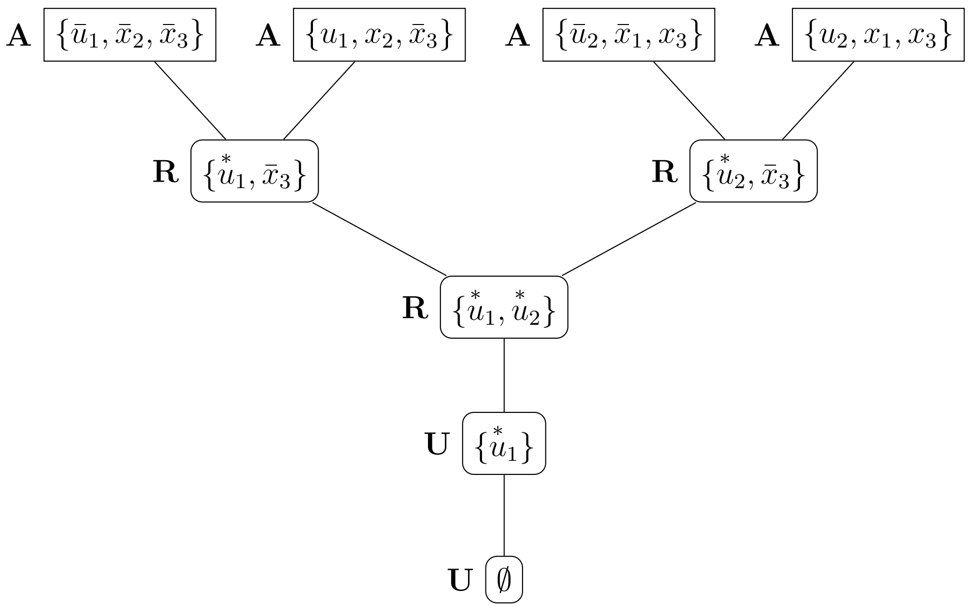


Figure 12.1: A deferred LDQ-Res refutation of a true S-form DQBF.

sequence of lines:

$$
\begin{array}{lcl}
L_1 & := & (C_1, \{M_1^j\}_{j\in[m]})\,, \\
\vdots & \vdots & \qquad\vdots \\
L_k & := & (C_k, \{M_k^j\}_{j\in[m]})\,.
\end{array}
$$

Each line consists of a clause C_i, which contains only existential literals, accompanied by a set $\{M_i^j\}_{j\in[m]}$ of *merge maps*. The merge maps are a set of dependency functions, each of whic is represented as a binary decision diagram (BDD).

The main idea behind M-Res is to build the partial countermodels, which are implicit in long-distance resolution proofs, explicitly into the derivation. The partial countermodels are recorded as merge maps and built up step-by-step as the proof progresses. The existential clauses $C_1, \ldots, C_k$ form a Resolution derivation, and the merge maps determine which resolution steps are allowed.

Organisation of the chapter

In Section 12.1, we introduce a computational model called merge maps, based on binary decision diagrams, that we use to represent partial strategies inside proofs. In Section 12.2, we introduce the system Merge Resolution and prove that it is a proof system for false H-form DQBFs. In Section 12.3, we show that Merge Resolution is exponentially stronger than deferred long-distance Q-Resolution on QBF.

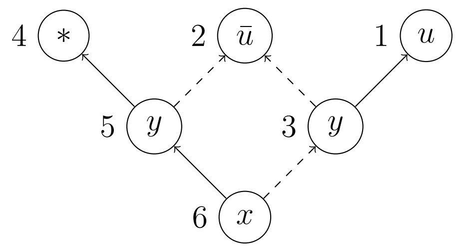


Figure 12.2: Binary decision diagram depiction of a merge map.

12.1 Merge maps

A merge map is a BDD that queries a set of existential variables and outputs either an assignment to some universal variable, or a placeholder that stands for 'no assignment'. For typographical convenience, we will often use the standard convention of representing assignments by literals. We represent the assignment $u \mapsto 1$ by the positive literal u, the assignment $u \mapsto 0$ by the negative literal $\bar{u}$, and 'no assignment' by the symbol $*$.

We write merge maps as a list of instructions that encode the BDD, such as

$$\begin{array}{ccl} 6 & \mapsto & (x, 5, 3) \\ 5 & \mapsto & (y, 4, 2) \\ 4 & \mapsto & * \\ 3 & \mapsto & (y, 1, 2) \\ 2 & \mapsto & \bar{u} \\ 1 & \mapsto & u\,. \end{array}$$

A triple of the form (x, r, s) represents the instruction 'if x is assigned 1 then goto r else goto s', and the literals $\bar{u}$, u, and $*$ represent output values. The highest instruction number is executed first, in this case, instruction 6. You can see this merge map depicted as a binary decision diagram in Figure 12.2.

We opt for a definition of merge map in which the instruction numbers need not be sequential. This comes in useful later, when we want to use the proof line indexing to label instructions.

Definition 12.1 (merge map). *A merge map M for a Boolean variable u over a finite set X of Boolean variables is a function from a finite set N of natural numbers satisfying, for each $i \in N$, either*

(a) $M(i) \in \{\bar{u}, u, *\}$, *or*

(b) $M(i) \in X \times N_{<i} \times N_{<i}$,

where $N_{<i} := \{i' \in N : i' < i\}$.

A merge map for u over X computes a function from $\langle X \rangle$ into $\{\bar{u}, u, *\}$, which is the function computed by the associated BDD. The exact computation is formalised below.

Definition 12.2 (computed function)**.** *Let M be a merge map for u over X with domain N. The* function computed by M *is the function*

$$h : \langle X \rangle \to \{\bar{u}, u, *\}$$

mapping σ to the output of the following algorithm:

1. $i := \max(N)$
2. **while** $M(i) \notin \{\bar{u}, u, *\}$
3. $\quad (x, r, s) := M(i)$
4. $\quad$ **if** $\sigma(x) = 1$ **then** $i := r$ **else** $i := s$
5. **return** $M(i)$

12.1.1 Relations and operations on merge maps

Merge Resolution requires two binary operations on merge maps, which we call *select* and *merge*. Later, *select* and *merge* will appear in the M-Res proof rules, where they help to define the allowable resolution steps. To determine preconditions for the operations, we also need to introduce two relations, *isomorphism* and *consistency*. We turn to these relations first.

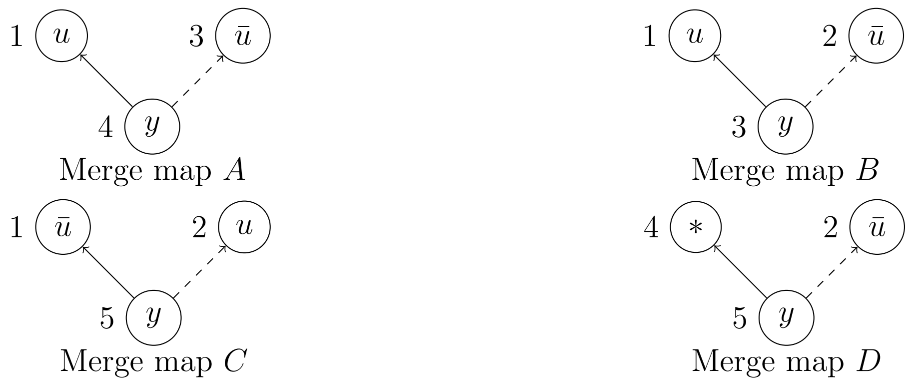
Isomorphism

We call merge maps isomorphic when they are the same, up to some renumbering of instructions.

Definition 12.3 (isomorphism)**.** *We call two merge maps M_1 and M_2 for u over X with domains N_1 and N_2* isomorphic *(written $M_1 \simeq M_2$) when there exists a bijection $f : N_1 \to N_2$ such that the following hold for each $i \in N_1$:*

(a) *if $M_1(i)$ is a literal in $\{\bar{u}, u, *\}$ then $M_2(f(i)) = M_1(i)$;*

(b) *if $M_1(i)$ is the triple (x, a, b) then $M_2(f(i)) = (x, f(a), f(b))$.*



relation	isomorphic	not isomorphic
consistent	$A \bowtie C$; $A \simeq C$	$B \bowtie D$; $B \not\simeq D$
not consistent	$A \not\bowtie B$; $A \simeq B$	$C \not\bowtie D$; $C \not\simeq D$

Figure 12.3: Relations on merge maps.

Isomorphism is useful in M-Res because isomorphic merge maps compute the same function. To see why, let M_1 and M_2 be merge maps computing functions h_1 and h_2, and let f be a bijection satisfying the properties of isomorphism (Definition 12.3). For any ε in $\mathrm{dom}(h_1)$, the computation of $h_2(\varepsilon)$ (as in Definition 12.2) is identical to that of $h_1(\varepsilon)$, except that each natural number i in $\mathrm{dom}(M_1)$ is replaced with $f(i)$.

Fact 12.4. *Any two isomorphic merge maps compute the same function.*

Consistency

Our second relation, consistency, simply identifies whether or not two merge maps agree on the intersection of their domains.

Definition 12.5 (consistency)**.** *Two merge maps M_1 and M_2 for u over X with domains N_1 and N_2 are* consistent *(written $M_1 \bowtie M_2$) iff $M_1(i) = M_2(i)$ for each $i \in N_1 \cap N_2$.*

Isomorphism and consistency, for some example merge maps, are illustrated in Figure 12.3.

The select operation

The select operation identifies equivalent merge maps by means of the isomorphism relation. It also allows a *trivial* merge map to be discarded. We call a merge map trivial when it is isomorphic to $1 \mapsto *$, that is, when it has a single node labelled with

'$*$'. The operation is not defined when the merge maps are neither isomorphic, nor is one of them trivial.

Definition 12.6 (select)**.** *Given two merge maps M_1, M_2 for which $M_1 \simeq M_2$ or one of M_1, M_2 is trivial,*

$$\text{select}(M_1, M_2) := \begin{cases} M_2 & \textit{if } M_1 \textit{ is trivial}\,, \\ M_1 & \textit{otherwise}\,. \end{cases}$$

Example 12.7. The merge maps A, B and C from Figure 12.3 are all isomorphic. Therefore $\text{select}(A, B) = \text{select}(A, C) = A$. ■

The merge operation

The merge operation allows two consistent merge maps to be combined as the children of a fresh instruction. Intuitively, it corresponds to taking two BDDs, and adding a new source query node that goes to the first BDD when satisfied, and to the other when falsified.

Definition 12.8 (merge)**.** *Given consistent merge maps M_1, M_2 for u over X with domains N_1, N_2, a natural number $n > \max(N_1 \cup N_2)$, and a variable x in X,*

$$\text{merge}(M_1, M_2, x, n)(i) := \begin{cases} (x, \max(N_1), \max(N_2)) & \textit{if } i = n\,, \\ M_1(i) & \textit{if } i \in N_1\,, \\ M_2(i) & \textit{if } i \in N_2 \setminus N_1\,. \end{cases}$$

Example 12.9. Looking again at the merge maps B and D in Figure 12.3, it is easy to check that the merge map depicted in Figure 12.2 is $\text{merge}(D, B, x, 6)$. ■

12.2 Merge Resolution

We are now ready to put down the rules of Merge Resolution.

Definition 12.10 (merge resolution)**.** *An* M-Res *derivation from an H-form DQBF*

$$Q := \exists x_1 \cdots \exists x_n \forall u_1(H_1) \cdots \forall u_m(H_m) \cdot F$$

is a sequence of lines

$$\begin{aligned} L_1 \quad &:= \quad (C_1, \{M_1^j : j \in [m]\})\,, \\ \vdots \quad & \quad \vdots \qquad\qquad \vdots \\ L_k \quad &:= \quad (C_k, \{M_k^j : j \in [m]\})\,, \end{aligned}$$

in which at least one of the following holds for each $i \in [k]$:

A Axiom. *There exists a clause C in F for which C_i is the existential subclause of C, and, for each j in $[m]$,*

$$M_i^j \quad = \quad \begin{cases} i \mapsto u_j & \text{if } \bar{u}_j \in C\,, \\ i \mapsto \bar{u}_j & \text{if } u_j \in C\,, \\ i \mapsto * & \text{otherwise}\,; \end{cases}$$

R Resolution. *There exist integers $r, s < i$ and p in $[n]$ for which*

(a) $C_i = \text{res}(C_r, C_s, x_p)$, *and*

(b) *for each j in $[m]$, either*

(i) $M_i^j = \text{select}(M_r^j, M_s^j)$, *or*

(ii) $x_p \in H_j$ *and* $M_i^j = \text{merge}(M_r^j, M_s^j, x_p, i)$;

I Instantiation. *There exists an integer $r < i$ such that C_i is an existential superclause of C_r and, for each j in $[m]$, either*

(i) $M_i^j = M_r^j$, *or*

(ii) M_r^j *is trivial and, for some literal a in $\{\bar{u}, u\}$,* $M_i^j = i \mapsto a$.

A derivation whose final clause C_k is empty is called a refutation. The size of a derivation is the number of lines.

Example 12.11. Figure 12.4 shows an M-Res refutation of the H-form DQBF

$$\exists x \exists y \forall u(\{y\}) \cdot \{\{\bar{x}, \bar{y}, \bar{u}\}, \{x, y, u\}, \{\bar{x}, y\}, \{x, \bar{y}\}\}\,.$$

Lines 1 to 4 are axioms which introduce the matrix clauses in order. Lines 5 and 6 instantiate the trivial merge maps from lines 3 and 4.

Line 7 is the resolution over variable y of lines 1 and 5. Since x is in the dependency set for u, we can apply the merge operation to the merge maps from lines 1 and 5, which are consistent. Notice that this satisfies condition (b)(ii) in the M-Res definition (Definition 12.10).

Line 9 is a resolution, over the other existential variable x, of lines 7 and 8. Note that the merge maps in lines 7 and 8 are isomorphic, so we simply copy the map from line 7, as specified by the select operation. This satisfies condition (b)(i). ■

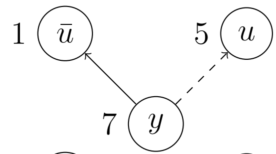
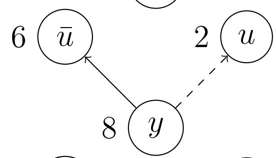
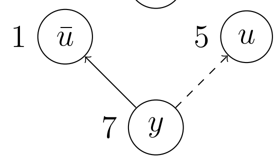
Line	Clause	Merge map	BDD
1 **Axiom**	$\{\bar{x}, \bar{y}\}$	$1 \mapsto \bar{u}$	1 $\bar{u}$
2 **Axiom**	$\{x, y\}$	$2 \mapsto u$	2 u
3 **Axiom**	$\{\bar{x}, y\}$	$3 \mapsto *$	3 $*$
4 **Axiom**	$\{x, \bar{y}\}$	$4 \mapsto *$	4 $*$
5 **Instantiation**	$\{\bar{x}, y\}$	$5 \mapsto u$	5 u
6 **Instantiation**	$\{x, \bar{y}\}$	$6 \mapsto \bar{u}$	6 $\bar{u}$
7 **Resolution**	$\{x\}$	$7 \mapsto (y, 1, 5)$ $5 \mapsto u$ $1 \mapsto \bar{u}$	
8 **Resolution**	$\{\bar{x}\}$	$8 \mapsto (y, 6, 2)$ $6 \mapsto \bar{u}$ $2 \mapsto u$	
9 **Resolution**	$\emptyset$	$7 \mapsto (y, 1, 5)$ $5 \mapsto u$ $1 \mapsto \bar{u}$	

Figure 12.4: An example Merge Resolution refutation.

12.2.1 Soundness

We now turn to an important feature of Merge Resolution, namely, that the merge maps compute *partial* countermodels. At the conclusion of a refutation, this becomes a total countermodel, witnessing that the instance is false. This is as though strategy extraction were built directly into the proof system.

We first define formally what we mean by a partial countermodel.

Definition 12.12. *We call a set of dependency functions* $\{h_j\}_{j \in [m]}$ *a* partial countermodel *for a DQBF* Q against *a clause* C *when, for each* ε *in* $\langle \mathrm{vars}_\exists(Q) \rangle$ *falsifying* C*, the assignment*

$$\varepsilon \cup \{f_i(\varepsilon\restriction_{H_i}) : i \in [m]\}$$

falsifies the matrix of Q.

The rules of Merge Resolution are devised so that the merge maps on any given line compute a partial countermodel against its clause.

Lemma 12.13. *Given an* M-Res *refutation*

$$\begin{array}{rcl} L_1 & := & (C_1, \{M_1^j : j \in [m]\})\,, \\ \vdots & \vdots & \vdots \\ L_k & := & (C_k, \{M_k^j : j \in [m]\})\,, \end{array}$$

of an H-form DQBF Q, for each i in $[k]$, the merge maps $\{M_i^j\}_{j\in[m]}$ compute a partial countermodel for Q against C_i.

Proof. We prove the lemma by induction in $i \in [k]$. We let σ_i be the negation of C_i.

For the base case $i = 1$, L_1 is an axiom, so C_1 belongs to F. The merge maps $\{M_j\}_{j\in[m]}$ are constant, and together compute an assignment that falsifies the universal subclause of C_1, which iteself belongs to $F[\sigma_i]$. Hence the merge maps compute a countermodel for $Q[\sigma_i]$.

For the inductive step, let $i \geq 2$. Let σ be a total existential assignment to $Q[\sigma_i]$. For each $j \in [m]$, and each $r \leq i$ let h_r^j be the function computed by M_r^j.

The case where L_i is introduced as an axiom is identical to the base case, so we consider two cases.

R Suppose that L_i was derived by resolution. Then there exist integers $r, s < i$ and an existential pivot x for which

(a) $C_i = \mathrm{res}(C_r, C_s, x)$, and

(b) for each j in $[m]$, either

   (i) $M_i^j = \mathrm{select}(M_r^j, M_s^j)$, or

   (ii) $x \in H_j$ and $M_i^j = \mathrm{merge}(M_r^j, M_s^j, x, i)$;

Suppose that $\sigma(x) = 1$. From the definitions of select (Definition 12.6) and merge (Definition 12.8), it is easy to see that, for each $j \in [m]$

$$h_r^j \text{ is not trivial} \quad \Rightarrow \quad h_i^j(\sigma) = h_r^j(\sigma)$$

Moreover, since σ falsifies C_r, the assignment

$$\sigma \cup \{h_r^j(\sigma\restriction_{H_j}) : i \in [m], M_r^j \text{ is not trivial}\} \tag{12.1}$$

falsifies F, by the inductive hypothesis. Hence

$$\sigma \cup \{h_i^j(\sigma\restriction_{H_j}) : i \in [m]\} \tag{12.2}$$

also falsifies C.

On the other hand, supposing that $\sigma(x) = 1$, we can show that assignment (12.2) falsifies some clause in F with a symmetrical argument.

I Suppose that L_i was derived by instantiation from L_r. Then there exists an integer $r < i$ such that $C_r \subseteq C_i$ and, for each j in $[m]$, either

(i) $M_i^j = M_r^j$, or

(ii) M_r^j is trivial and, for some literal a in $\{\bar{u}, u\}$, $M_i^j = i \mapsto a$.

Now, σ falsifies C_r, which is a subset of C_i. Hence, by the inductive hypothesis, assignment (12.1), falsifies some clause C in F. Since $h_i^j = h_r^j$ whenever M_r^j is not trivial, assignment (12.2) also falsifies C. □

Since all assignments falsify the empty clause, a partial countermodel against the empty clause is a (complete) countermodel. So the soundness of M-Res is an immediate corollary of Lemma 12.13.

Corollary 12.14. *If an H-form DQBF has an* M-Res *refutation, then it is false.*

12.2.2 Completeness

We prove the completeness of Merge Resolution using a 'full binary tree' construction, similar to that which we used to show completeness for Resolution (Fact 2.9).

First, an overview of the construction. Let Q be a false DQBF with a countermodel $\{h_j\}_{j\in[m]}$. For each total existential assignment ε, the assignment

$$\sigma \cup \{h_j(\sigma\restriction_{H_j}) : j \in [m]\}$$

falsifies some clause C_ε in the matrix of Q.

Consider the M-Res line L_ε, whose clause is the negation of ε, and whose merge maps are constant functions computing $h(\varepsilon)$. Each L_ε can be derived in two steps, by weakening the axiom corresponding to C_ε. Moreover, the clauses $\{C_\varepsilon : \varepsilon \in \langle X\rangle\}$ form the leaves of a full binary tree Resolution refutation which can be completed using an arbitrary fixed order of the existential variables.

The merge maps are constructed by merging over the pivot x when, and only when, x is in the dependency set. Otherwise the select operation takes the merge map from the first antecedent, since the full binary tree structure *guarantees* that they are isomorphic.

As the structure of merge maps is bound to the structure of the refutation, it is no surprise that the merge maps in our construction are also full binary trees. This is captured by the following definition.

Definition 12.15 (binary tree merge map)**.** *A* binary tree merge map *for a variable u_j over a sequence of variables $x_1, \ldots, x_n$ is a function M with domain $[2^{n+1} - 1]$ satisfying*

- *$M(i) = (x_{\lfloor \log i \rfloor + 1}, 2i, 2i + 1)$, for $1 \leq i < 2^n$, and*
- *$M(i) \in \{\bar{u}_j, u_j\}$, for $2^n \leq i < 2^{n+1}$.*

As a precursor to the completeness proof, we show the following fact.

Fact 12.16. *Given an H-form DQBF Q, an assignment σ to an existential x, and an* M-Res *refutation of $Q[\sigma]$ with concluding merge maps $\{M_j\}_{j \in [m]}$, one can construct an* M-Res *derivation of*

$$(C, \{M_j\}_{j \in [m]})$$

from Q, where C is the negation of σ.

Proof. Let π be the refutation with the given conclusion. The desired derivation may be obtained from π just by adding the unique literal in C to each clause, applying weakening where necessary, and adjusting the indexing of the merge maps to account for the extra weakening steps. □

Lemma 12.17. *Every false H-form DQBF has an* M-Res *refutation.*

Proof. Let

$$Q := \exists x_1 \cdots \exists x_n \forall u_1(H_1) \cdots \forall u_m(H_m) \cdot F$$

be a false DQBF

A merge map for u_j over H_j is said to be *complete* if it is isomorphic to a binary tree merge map for u_j over the sequence

$$x_{\tau(1)}, \ldots x_{\tau(|H_j|)} \, ,$$

which enumerates H_j in increasing index order; that is, $\tau : [|H_j|] \to [n]$ is the unique function satisfying

(a) $\{x_{\tau(i)} : i \in [|H_j|]\} = H_j$, and

(b) $i < i' \Leftrightarrow \tau(i) < \tau(i')$, for each i, i' in $[|H_j|]$.

By induction on the number n of existential variables, we show that, for each countermodel $\{h_j\}_{j\in[m]}$ for Q, there exists an M-Res refutation whose concluding merge maps are complete, and compute h.

For the base case $n = 0$, observe that each h_j is a constant function, which maps the empty assignment to one of the assignments $u_j \mapsto 1$ or $u_j \mapsto 0$. By definition of countermodel, there exists a clause $C \in F$ which is falsified by the assignment

$$\{h_j(\emptyset) : j \in [m]\}\,.$$

Applying the axiom rule to C, one obtains a derivation of the line $(\emptyset, \{M^j\}_{j\in[m]})$ in which M_j computes the constant function h_j if u_j is in $\text{vars}(C)$, and is trivial otherwise. With a single weakening step, each trivial M_j can be swapped for a merge map isomorphic to $1 \mapsto h_j(\emptyset)$. Then each M^j is trivially complete and computes the constant function h_j.

For the inductive step, let $n \geq 1$. By Lemma 3.6,

$$\{h_j[x_i \mapsto 1]\}_{j\in[m]} \qquad \text{and} \qquad \{h_j[x_i \mapsto 0]\}_{j\in[m]}$$

are countermodels for $Q[x_i \mapsto 0]$ and $Q[x_i \mapsto 1]$, both of which have $i - 1$ existential variables. By Fact 12.16 and the inductive hypothesis, we deduce that there exist M-Res derivations π and π' from Q of the lines

$$L := (\{\bar{x}_n\}, \{M_j\}_{j\in[m]}) \qquad \text{and} \qquad L' := (\{x_n\}, \{M'_j\}_{j\in[m]})\,,$$

in which the M_j and M'_j are complete merge maps computing $h_j[x_n \mapsto 1]$ and $h_j[x_n \mapsto 0]$.

Assume that the lines of π are indexed from 1 to $|\pi|$ and that those of π' are indexed from $|\pi| + 1$ to $|\pi| + |\pi'|$. For each j in $[m]$, the domains of M_j and M'_j are disjoint, so $M_j \bowtie M'_j$. If x_n is not in H_j, then

$$h_j[x_n \mapsto 1] = h_j[x_n \mapsto 0]\,,$$

and we must have $M_j \simeq M'_j$, since complete merge maps computing the same function are isomorphic. It follows that the line

$$L'' := (\emptyset, \{M''_j\}_{j\in[m]})$$

can be derived by resolution from L and L', where

$$M''_j := \begin{cases} \text{merge}(M_j, M'_j, x_i, |\pi| + |\pi'| + 1) & \text{if } x_i \in H_i\,, \\ M_j & \text{if } x_i \notin H_j\,. \end{cases}$$

It is easy to see that the M''_j are complete merge maps computing the h_j. ☐

So, we have proved that M-Res is sound and complete, which leaves only checkability. It is easy to see that the isomorphism and consistency relations can be efficiently checked. From there, it is easy to see that the proof rules of Merge Resolution are polynomial-time checkable.

Theorem 12.18. M-Res *is a proof system for the language of false H-form DQBF.*

12.3 Merge Resolution on QBF

Here we consider how Merge Resolution operates in the smaller realm of QBF. We show that it is exponentially stronger than deferred LDQ-Res.

12.3.1 Simulation of deferred LDQ-Res

In a nutshell, deferred LDQ-Res is like M-Res with a restricted view of isomorphism, namely, isomorphism is restricted to constant functions. In this view, select is only defined when one of the maps is trivial, or both are isomorphic and constant.

For the simulation, we translate the universal literals in the long-distance refutation into merge maps.

Theorem 12.19. M-Res *p-simulates deferred* LDQ-Res*.*

Proof. Let $\pi := C_1, \ldots, C_k, \ldots, C_{k'}$ be a deferred LDQ-Res refutation of a QBF Q with universal variables $\{u_1, \ldots, u_m\}$, where C_k is the final clause that not derived by universal reduction. For each i in $[k]$ and j in $[m]$ in, we define

$$a_i^j := \begin{cases} u_j & \text{if } \bar{u}_j \in C_i\,, \\ \bar{u}_j & \text{if } u_j \in C_i\,, \\ * & \text{if } u_j \notin \text{vars}(C_i)\,. \end{cases}$$

We define a sequence $\rho := L_1, \ldots, L_n$, in which each $L_i := (C'_i, \{M_i^j\}_{j \in [m]})$, and prove that it is an M-Res refutation of Q. For each i in $[k]$, we define C'_i to be the existential subclause of C_i. For each j in $[m]$, the merge maps are defined recursively as follows:

If C_i is an axiom, we take M_i^j as the merge map $i \mapsto a_i^j$. If C_i is derived by resolution, say $C_i = \mathrm{res}(C_r, C_s, x)$ with $r, s < i$, then

$$M_i^j := \begin{cases} \mathrm{select}(M_r^j, M_s^j)\,, & \text{if } \mathrm{select}(M_r^j, M_s^j) \text{ is defined}\,, \\ \mathrm{merge}(M_r^j, M_s^j, x, i)\,, & \text{otherwise}\,. \end{cases}$$

By induction on i in $[k]$, we show the following for each j in $[m]$:

(a) if $\{\bar{u}_j, u_j\} \nsubseteq C_i$, then M_i^j is isomorphic to $1 \mapsto a_i^j$;

(b) L_i can be derived from previous clauses using an M-Res rule.

For the base case $i = 1$, C_i is an axiom, and both (a) and (b) are established trivially. For the inductive step, let $i \geq 2$. The inductive step for an axiom is the same as the base case, so we assume that C_i was derived by resolution. So $C_i = \mathrm{res}(C_r, C_s, x)$ for some $r, s < i$ and some existential variable x.

(a) Suppose that $\{\bar{u}_j, u_j\} \nsubseteq C_i$. By definition of resolution, either (1) $a_i^j = a_r^j = a_s^j$, or (2) exactly one of a_r^j, a_s^j is $*$ (a_s^j, say), the other (a_r^j, say) is equal to a_i^j. In case (1), M_r^j and M_s^j are both isomorphic to $1 \mapsto a_i^j$, by the inductive hypothesis. In case (2), M_r^j is isomorphic to $1 \mapsto a_i^j$ and M_b^j is trivial. Either way

$$M_i^j = \mathrm{select}(M_r^j, M_s^j) = M_r^j$$

is isomorphic to $1 \mapsto a_i^j$.

(b) We show that L_i can be derived by resolution from L_r and L_s. We need only show that $\mathrm{merge}(M_r^j, M_s^j, x, i)$ is defined whenever $\mathrm{select}(M_r^j, M_s^j)$ is not.

Now, if $\{\bar{u}_j, u_j\}$ is not a subset of C_i, then it is not a subset of C_r or C_s. By the inductive hypothesis (a), at least one of M_r^j and M_s^j is isomorphic to $1 \mapsto a_i^j$, and the other is either isomorphic to $1 \mapsto a_i^j$ or trivial. So $\mathrm{select}(M_r^j, M_s^j)$ is defined.

On the other hand, suppose that $\{\bar{u}_j, u_j\}$ is indeed a subset of C_i. Then it is a subset of at least one on C_r and C_s, say C_r. If u_j is not in $\mathrm{vars}(C_s)$, then, by the inductive hypothesis (a), M_s^j is trivial, so $\mathrm{select}(M_r^j, M_s^j)$ is defined. On the other hand, if u_j is in $\mathrm{vars}(C_s)$, then x is in H_j by the definition of long-distance resolution, so $\mathrm{merge}(M_r^j, M_s^j, x, i)$ is defined.

This completes the induction. Since C_k contains only universal variables, C_k' is the empty clause, and ρ is a refutation. $\square$

12.3.2 Short proofs of the squared equality family

Here we construct short M-Res refutations of the squared equality formulas. The approach is as follows.

First, for each i, j in $[n]$, we derive a line $(\{z_{i,j}\}, M_{i,j})$ by resolving the axioms for the four clauses in eq_n^2 that contain $\{z_{i,j}\}$. In the set $M_{i,j}$, the merge map for u_i outputs x_i with a single query, the merge map for v_j outputs y_j with a single query, and all other maps are trivial.

Since all the non-trivial merge maps for a given universal variable are isomorphic, the n^2 unit clauses can all be resolved against the square clause, utilising the select operation. It is this final step which is unavailable in deferred LDQ-Res.

Theorem 12.20. *The squared equality family has $O(n^2)$-size* M-Res *refutations.*

Proof. Let $n \in \mathbb{N}$. We construct a refutation in two stages. In the first stage we explicitly construct an M-Res derivation $\pi := L_1, \ldots, L_k$ from EQ_n^2, where $k = 2n^2$. In the second stage, we show that π can be extended to a refutation with a further $n^2 + 1$ lines.

Stage one. For each $h, i, j \in \mathbb{N}$ we put

$$\delta(h, i, j) := (h-1)n^2 + (i-1)n + j$$

We use $L(h, i, j)$ as an alias for $L_{\delta(h,i,j)}$. Similarly, we let $C(h, i, j)$ be the clause, $U(h, i, j)$ be the merge map for u_i, and $V(h, i, j)$ be the merge map for v_j appearing on line $L(h, i, j)$. All other merge maps in π are trivial.

Letting i, j in $[n]$, we define the first $4n^2$ lines with

$$\begin{aligned}
C(0,i,j) &:= \{x_i, y_j, z_{i,j}\}\,,\\
C(1,i,j) &:= \{\bar{x}_i, y_j, z_{i,j}\}\,,\\
C(2,i,j) &:= \{x_i, \bar{y}_j, z_{i,j}\}\,,\\
C(3,i,j) &:= \{\bar{x}_i, \bar{y}_j, z_{i,j}\}\,,
\end{aligned}$$

$$\begin{aligned}
U(0,i,j) &:= \delta(0,i,j) \mapsto \bar{u}_i & V(0,i,j) &:= \delta(0,i,j) \mapsto \bar{v}_j\,,\\
U(1,i,j) &:= \delta(1,i,j) \mapsto u_i & V(1,i,j) &:= \delta(1,i,j) \mapsto \bar{v}_j\,,\\
U(2,i,j) &:= \delta(2,i,j) \mapsto \bar{u}_i & V(2,i,j) &:= \delta(2,i,j) \mapsto v_j\,,\\
U(3,i,j) &:= \delta(3,i,j) \mapsto u_i & V(3,i,j) &:= \delta(3,i,j) \mapsto v_j\,,
\end{aligned}$$

and observe that each of these lines can be introduced as an axiom.

The next $2n^2$ lines are the result of the natural resolutions over y_j. For each i, j in $[n]$, we define

$$\begin{aligned}
C(4,i,j) &:= \{x_i, z_{i,j}\} & U(4,i,j) &:= U(0,i,j)\,,\\
C(5,i,j) &:= \{\bar{x}_i, z_{i,j}\} & U(5,i,j) &:= U(1,i,j)\,,
\end{aligned}$$

$$
\begin{aligned}
V(4,i,j) \quad &:= \quad \delta(4,i,j) \mapsto (y_j, \delta(0,i,j), \delta(2,i,j)) \\
& \delta(2,i,j) \mapsto v_j \\
& \delta(0,i,j) \mapsto \bar{v}_j\,, \\
V(5,i,j) \quad &:= \quad \delta(5,i,j) \mapsto (y_j, \delta(1,i,j), \delta(3,i,j)) \\
& \delta(3,i,j) \mapsto v_j \\
& \delta(1,i,j) \mapsto \bar{v}_j\,.
\end{aligned}
$$

Each line $L(4,i,j)$ can be derived by resolution from $L(0,i,j)$ and $L(2,i,j)$. To see this, note that $U(0,i,j)$ is clearly isomorphic to $U(2,i,j)$ and $V(0,i,j)$ is trivially consistent with $V(2,i,j)$ (their domains are disjoint), therefore

$$
\begin{aligned}
U(4,i,j) \quad &= \quad \mathrm{select}(U(0,i,j), U(2,i,j))\,, \text{ and} \\
V(4,i,j) \quad &= \quad \mathrm{merge}(V(0,i,j), V(2,i,j), \delta(4,i,j), y_j)\,.
\end{aligned}
$$

A similar argument shows each that $L(5,i,j)$ can be derived by resolution from $L(1,i,j)$ and $L(3,i,j)$.

The final n^2 lines are the result of the natural resolutions over x_i. For each i,j in $[n]$ we define

$$
C(6,i,j) \quad := \quad \{z_{i,j}\} \qquad V(6,i,j) \quad := \quad V(4,i,j)\,,
$$

$$
\begin{aligned}
U(6,i,j) \quad &:= \quad \delta(6,i,j) \mapsto (x_i, \delta(0,i,j), \delta(1,i,j)) \\
& \delta(1,i,j) \mapsto u_i \\
& \delta(0,i,j) \mapsto \bar{u}_i\,.
\end{aligned}
$$

It is easy to see that each $L(6,i,j)$ can be derived by resolution from $L(4,i,j)$ and $L(5,i,j)$, since $V(4,i,j)$ is clearly isomorphic to $V(5,i,j)$ (an isomorphism is $l \mapsto l + n^2$) and $U(0,i,j)$ is trivially consistent with $U(1,i,j)$ (disjoint domains).

Stage two. We now show how π can be extended to a refutation. Let

$$
L_6 := \{L(6,i,j) : i,j \in [n]\}
$$

denote the final n^2 lines of π, in each of which appears some unit clause $\{z_{i,j}\}$. We observe that, for each $r,s,i \in [n]$, $U(6,i,r)$ is isomorphic to $U(6,i,s)$ (an isomorphism is $l \mapsto l + s - r$); that is, amongst the lines L_6, the non-trivial merge maps for u_i are pairwise isomorphic. Similarly, for each j in $[n]$, the non-trivial merge maps for v_j appearing in L_6 are pairwise isomorphic.

Now, a line T, consisting of the clause $\{\bar{z}_{i,j} : i,j \in [n]\}$ and a full set of trivial merge maps, can be introduced as an M-Res axiom in a derivation from EQ_n^2. From T and L_6, in a further n^2 steps we obtain a refutation by successively resolving each line in L_6 against T, removing a literal $\bar{z}_{i,j}$ each time. All such resolution steps are valid, since the merge map for u_i (v_j) in any line can be defined as $\mathrm{select}(M_r, M_s)$, where M_r and M_s are the merge maps for u_i (v_j) appearing in the antecedent lines. $\square$

Combining this upper bound with the lower bound in Theorem 7.11, along with the simulation in Theorem 12.19, we have proved the following.

Theorem 12.21. *On the language* FQBF, *deferred* LDQ-Res $<_p$ M-Res.

Part IV

Conclusions

Chapter 13

Outlook

Now we wrap up the thesis and discuss the import of the results.

13.1 Future perspectives for QBF solving

Models proposed and results proved in this thesis suggest some potential future directions for both QBF and DQBF solving.

Dependency schemes in QBF solving

We showed that the reflexive resolution path dependency scheme can exponentially shorten proofs in $\forall\mathsf{Exp}+\mathsf{Res}$, the proof system underpinning expansion-based QBF solving. This suggests that incorporating dependency schemes into expansion solvers has the potential to solve instances that would otherwise remain intractable. Thus, we endorse the move in this direction mooted at the conclusion of [33].

Moreover, we showed that applying dependency schemes dynamically, on restrictions of the the input QBF, shortens proofs even further. Dynamic application of dependency schemes could be implemented by using dependency detection as a form of inprocessing, rather than preprocessing. Our results validate this approach as at least reasonable in theory, and potentially valuable in practice.

Another avenue of exploration is dependency learning [46], a recently proposed approach to solving QBF that can be combined with dependency schemes [48]. We have shown that dependency schemes are compatible with expansion, which perhaps suggests that one could consider implementing dependency learning there as well.

Solving H-form DQBF

We introduced Merge Resolution, the first sound and complete QCDCL-style proof system for DQBF. In so doing we made two modifications: first, we switched the input format to H-form, and second, we built syntactic representations of partial countermodels into the proofs, allowing the system to reason with them.

It is conceivable that Merge Resolution gives rise to a decision procedure for H-form DQBF. This would take the form of a CDCL-style proof search, which could also be viewed as an exhaustive search for a countermodel.

As far as we are aware, there has been no dedicated research into solving H-form DQBF, and as such there are no known applications. Our work suggests that solving H-form is just as viable as S-form, forming the basis of an unexplored workflow. It remains to be seen which problems lend themselves naturally to H-form encodings, but it is hard to imagine that such problems do not exist to serve present or future industrial applications.

13.2 Generalisations of results

Many of the results presented in this thesis can be extended to more general settings.

Resolution over universal pivots

The system QU-Res extends Q-Res by adding resolution over universal pivots. Since resolution is a logically correct propositional rule of inference which preserves satisfying assignments, allowing resolution over all pivots is perfectly natural. Moreover, any argument which does not use the fact that the pivot is existential will lift immediately to QU-Res.

For example, our lower bound technique for Q-Res (Corollary 5.17) is equally applicable to QU-Res, and hence our concrete lower bounds derived using that technique (e.g. for the equality family, Theorem 5.18) apply also to QU-Res. In fact, the technique and the applications were originally presented for QU-Res in [8].

Also, all the material on dependency schemes in Q-Res appearing in Chapter 11 is applicable to QU-Res, and is presented this way in [9].

Beyond Resolution and conjunctive normal form

Our lower bound technique for Q-Res can actually be applied in a much broader context: P+∀red. The proof system P+∀red adds a universal reduction rule to an appropriate propositional proof system P, lifting it to a QBF proof system.

It turns out that a certain measure on the propositional proof system P (*capacity*) along with a semantic measure on the QBF family (*cost*) is often sufficient to prove a lower bound, by means of the Size-Cost-Capacity Theorem [10].

Theorem ([10]). *The size of a* P+∀red *refutation of a QBF Q is at least*

$$\frac{\text{cost}(Q)}{\text{capacity}(\mathsf{P})}.$$

QU-Res is P+∀red when P is Resolution, and, as it turns out, the capacity of Resolution is 1. Our Q-Res lower bound for the equality family really rests on the fact the family has exponential cost, and our lower bound technique is really only a special case of Size-Cost-Capacity.

Moreover, the propositional proof system Cutting Planes (CP), which p-simulates Resolution, also has capacity 1. Hence the equality formulas require exponential size refutations even in CP+∀red. Further applications to other QBF proof systems were covered in [10].

13.3 Open problems and conjectures

Proof complexity of dependency schemes

All of our separations for models of solving with dependency schemes were based on short proofs with the reflexive resolution path dependency scheme. An obvious question is whether such separations can be made also with the standard dependency scheme.

Open Problem 1. *Does* Q-Res *p-simulate* Q($\mathcal{D}^{\text{std}}$)-Res*?*

Open Problem 2. *Does* ∀Exp+Res *p-simulate* ∀Exp($\mathcal{D}^{\text{std}}$)+Res*?*

Another interesting question which we did not settle is the status of the two expansion systems parameterised by $\mathcal{D}^{\text{rrs}}$.

Open Problem 3. *Does* ∀Exp($\mathcal{D}^{\text{rrs}}$)+Res *p-simulate* IR($\mathcal{D}^{\text{rrs}}$)-calc*?*

Given that we were able to find short proofs of $\mathcal{KB}$ in IR($\mathcal{D}^{\text{rrs}}$)-calc, but not in ∀Exp($\mathcal{D}^{\text{rrs}}$)+Res, we conjecture that the answer is no. Perhaps some DQBF generalisation of our ∀Exp+Res lower bound technique could be investigated as a lead for the ∀Exp($\mathcal{D}^{\text{rrs}}$)+Res lower bound.

H-form DQBF

At the time of writing, interest in solving DQBF, S-form in particular, is growing. Our proposal of Merge Resolution as a natural QCDCL-style proof system for DQBF pushes H-form towards the foreground. Moreover, our feeling is that H-form is not merely a syntactic dual to S-form. With the expansion-reduction hypothesis, we have attempted to explain why, and to clarify the inherent semantic subtleties of S-form versus H-form DQBF.

One interesting and related open problem is the complexity of the decision problem for H-form DQBF.

Open Problem 4. *What is the computational complexity of the language of false H-form DQBFs?*

This question is not a priori related to the complexity of S-form DQBF, and it is not clear whether the proof of NEXP-completeness for S-form can be modified for an answer. Since H-form includes QBF, the decision problem is at least PSPACE-hard, and since there are at most doubly-exponentially-many sets of dependency functions, with a wave of the hand we can claim it is also in NEXP. Hence NEXP-complete seems a fair conjecture.

Bibliography

[1] Valeriy Balabanov, Hui-Ju Katherine Chiang, and Jie-Hong R. Jiang. Henkin quantifiers and Boolean formulae: A certification perspective of DQBF. *Theoretical Computer Science*, 523:86–100, 2014.

[2] Valeriy Balabanov and Jie-Hong R. Jiang. Unified QBF certification and its applications. *Formal Methods in System Design*, 41(1):45–65, 2012.

[3] Valeriy Balabanov, Jie-Hong Roland Jiang, Mikoláš Janota, and Magdalena Widl. Efficient extraction of QBF (counter)models from long-distance resolution proofs. In Blai Bonet and Sven Koenig, editors, *National Conference on Artificial Intelligence (AAAI)*, pages 3694–3701. AAAI Press, 2015.

[4] Eli Ben-Sasson, Russell Impagliazzo, and Avi Wigderson. Near optimal separation of tree-like and general resolution. *Combinatorica*, 24(4):585–603, 2004.

[5] Marco Benedetti and Hratch Mangassarian. QBF-based formal verification: Experience and perspectives. *Journal of Satisfiability, Boolean Modeling and Computation*, 5(1-4):133–191, 2008.

[6] Olaf Beyerdorff, 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.

[7] Olaf Beyersdorff and Joshua Blinkhorn. Dependency schemes in QBF calculi: Semantics and soundness. In Michel Rueher, editor, *International Conference on Principles and Practice of Constraint Programming (CP)*, volume 9892 of *Lecture Notes in Computer Science*, pages 96–112. Springer, 2016.

[8] Olaf Beyersdorff and Joshua Blinkhorn. Genuine lower bounds for QBF expansion. In Rolf Niedermeier and Brigitte Vallée, editors, *International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 96 of *Leib-*

niz International Proceedings in Informatics (LIPIcs), pages 12:1–12:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[9] Olaf Beyersdorff and Joshua Blinkhorn. Dynamic QBF dependencies in reduction and expansion. *ACM Transactions on Computational Logic*, 21(2):1–27, 2019.

[10] Olaf Beyersdorff, Joshua Blinkhorn, and Luke Hinde. Size, cost, and capacity: A semantic technique for hard random QBFs. *Logical Methods in Computer Science*, 15(1), 2019.

[11] Olaf Beyersdorff, Joshua Blinkhorn, and Meena Mahajan. Building strategies into QBF proofs. In Rolf Niedermeier and Christophe Paul, editors, *International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 126 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 14:1–14:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[12] Olaf Beyersdorff, Ilario Bonacina, and Leroy Chew. Lower bounds: From circuits to QBF proof systems. In Madhu Sudan, editor, *ACM Conference on Innovations in Theoretical Computer Science (ITCS)*, pages 249–260. ACM, 2016.

[13] Olaf Beyersdorff, Leroy Chew, and Mikoláš Janota. On unification of QBF resolution-based calculi. In Erzsébet Csuhaj-Varjú, Martin Dietzfelbinger, and Zoltán Ésik, editors, *International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 8635 of *Lecture Notes in Computer Science*, pages 81–93. Springer, 2014.

[14] Olaf Beyersdorff, Leroy Chew, and Mikoláš Janota. Proof complexity of resolution-based QBF calculi. In Ernst W. Mayr and Nicolas Ollinger, editors, *International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 30 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 76–89. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2015.

[15] Olaf Beyersdorff, Leroy Chew, Renate A. Schmidt, and Martin Suda. Lifting QBF resolution calculi to DQBF. In Nadia Creignou and Daniel Le Berre, editors, *International Conference on Theory and Practice of Satisfiability Testing (SAT)*, volume 9710 of *Lecture Notes in Computer Science*, pages 490–499. Springer, 2016.

[16] Nikolaj Bjørner, Mikolás Janota, and William Klieber. On conflicts and strategies in QBF. In Ansgar Fehnker, Annabelle McIver, Geoff Sutcliffe, and Andrei Voronkov, editors, *International Conference on Logic for Programming, Artificial Intelligence and Reasoning - Short Presentations (LPAR)*, volume 35 of *EPiC Series in Computing*, pages 28–41. EasyChair, 2015.

[17] Roderick Bloem, Uwe Egly, Patrick Klampfl, Robert Könighofer, and Florian Lonsing. SAT-based methods for circuit synthesis. In *Conference on Formal Methods in Computer-aided Design (FMCAD)*, pages 31–34. IEEE, 2014.

[18] Samuel R. Buss. Towards NP-P via proof complexity and search. *Annals of Pure and Applied Logic*, 163(7):906–917, 2012.

[19] Vasek Chvátal and Endre Szemerédi. Many hard examples for resolution. *Journal of the ACM*, 35(4):759–768, 1988.

[20] Stephen A. Cook. The complexity of theorem-proving procedures. In Michael A. Harrison, Ranan B. Banerji, and Jeffrey D. Ullman, editors, *ACM Symposium on Theory of Computing (STOC)*, pages 151–158. ACM, 1971.

[21] Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44(1):36–50, 1979.

[22] Martin Davis, George Logemann, and Donald W. Loveland. A machine program for theorem-proving. *Communications of the ACM*, 5(7):394–397, 1962.

[23] Uwe Egly, Florian Lonsing, and Magdalena Widl. Long-distance resolution: Proof generation and strategy extraction in search-based QBF solving. In Kenneth L. McMillan, Aart Middeldorp, and Andrei Voronkov, editors, *International Conference on Logic for Programming, Artificial Intelligence and Reasoning (LPAR)*, volume 8312 of *Lecture Notes in Computer Science*, pages 291–308. Springer, 2013.

[24] Peter Faymonville, Bernd Finkbeiner, Markus N. Rabe, and Leander Tentrup. Encodings of bounded synthesis. In Axel Legay and Tiziana Margaria, editors, *International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 10205 of *Lecture Notes in Computer Science*, pages 354–370. Springer, 2017.

[25] Andreas Fröhlich, Gergely Kovásznai, Armin Biere, and Helmut Veith. iDQ: Instantiation-based DQBF solving. In Daniel Le Berre, editor, *Workshop on Pragmatics of SAT (POS)*, volume 27 of *EPiC Series in Computing*, pages 103–116. EasyChair, 2014.

[26] Allen Van Gelder. Variable independence and resolution paths for quantified Boolean formulas. In Jimmy Ho-Man Lee, editor, *International Conference on Principles and Practice of Constraint Programming (CP)*, volume 6876 of *Lecture Notes in Computer Science*, pages 789–803. Springer, 2011.

[27] Enrico Giunchiglia, Massimo Narizzano, and Armando Tacchella. Clause/term resolution and learning in the evaluation of quantified Boolean formulas. *Journal of Artificial Intelligence Research*, 26:371–416, 2006.

[28] Alexandra Goultiaeva, Allen Van Gelder, and Fahiem Bacchus. A uniform approach for generating proofs and strategies for both true and false QBF formulas. In Toby Walsh, editor, *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 546–553. IJCAI/AAAI, 2011.

[29] Armin Haken. The intractability of resolution. *Theoretical Computer Science*, 39:297–308, 1985.

[30] Marijn J. H. Heule. Schur number five. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *National Conference on Artificial Intelligence (AAAI)*, pages 6598–6606. AAAI Press, 2018.

[31] Franjo Ivancic, Zijiang Yang, Malay K. Ganai, Aarti Gupta, and Pranav Ashar. Efficient SAT-based bounded model checking for software verification. *Theoretical Computer Science*, 404(3):256–274, 2008.

[32] Mikolás Janota. On Q-resolution and CDCL QBF solving. In Nadia Creignou and Daniel Le Berre, editors, *International Conference on Theory and Practice of Satisfiability Testing (SAT)*, volume 9710 of *Lecture Notes in Computer Science*, pages 402–418. Springer, 2016.

[33] Mikoláš Janota, William Klieber, Joao Marques-Silva, and Edmund M. Clarke. Solving QBF with counterexample guided refinement. *Artificial Intelligence*, 234:1–25, 2016.

[34] Mikoláš Janota and João Marques-Silva. Expansion-based QBF solving versus Q-resolution. *Theoretical Computer Science*, 577:25–42, 2015.

[35] Hans Kleine Büning, Marek Karpinski, and Andreas Flögel. Resolution for quantified Boolean formulas. *Information and Computation*, 117(1):12–18, 1995.

[36] Roman Kontchakov, Luca Pulina, Ulrike Sattler, Thomas Schneider, Petra Selmer, Frank Wolter, and Michael Zakharyaschev. Minimal module extraction from DL-lite ontologies using QBF solvers. In Craig Boutilier, editor, *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 836–841. AAAI Press, 2009.

[37] L.A. Levin. Universal sequential search problems. *Problemy Peredachi Informatsii (Problems of Information Transition)*, 9:115–116, 1973.

[38] Jia Hui Liang, Vijay Ganesh, Ed Zulkoski, Atulan Zaman, and Krzysztof Czarnecki. Understanding VSIDS branching heuristics in conflict-driven clause-learning SAT solvers. In Nir Piterman, editor, *Haifa Verification Conference (HVC)*, volume 9434 of *Lecture Notes in Computer Science*, pages 225–241. Springer, 2015.

[39] Andrew C. Ling, Deshanand P. Singh, and Stephen Dean Brown. FPGA logic synthesis using quantified Boolean satisfiability. In Fahiem Bacchus and Toby Walsh, editors, *International Conference on Theory and Practice of Satisfiability Testing (SAT)*, volume 3569 of *Lecture Notes in Computer Science*, pages 444–450. Springer, 2005.

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

[41] Florian Lonsing and Armin Biere. DepQBF: A dependency-aware QBF solver. *Journal of Satisfiability, Boolean Modeling and Computation*, 7(2-3):71–76, 2010.

[42] Florian Lonsing and Uwe Egly. Evaluating QBF solvers: Quantifier alternations matter. In John N. Hooker, editor, *International Conference on Principles and Practice of Constraint Programming (CP)*, volume 11008 of *Lecture Notes in Computer Science*, pages 276–294. Springer, 2018.

[43] Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, and Sharad Malik. Chaff: Engineering an efficient SAT solver. In *Design Automation Conference (DAC)*, pages 530–535. ACM, 2001.

[44] Jakob Nordström. On the interplay between proof complexity and SAT solving. *SIGLOG News*, 2(3):19–44, 2015.

[45] Tomás Peitl, Friedrich Slivovsky, and Stefan Szeider. Long distance Q-resolution with dependency schemes. In Nadia Creignou and Daniel Le Berre, editors, *International Conference on Theory and Practice of Satisfiability Testing (SAT)*, volume 9710 of *Lecture Notes in Computer Science*, pages 500–518. Springer, 2016.

[46] Tomáš Peitl, Friedrich Slivovsky, and Stefan Szeider. Dependency learning for QBF. In Serge Gaspers and Toby Walsh, editors, *International Conference on Theory and Practice of Satisfiability Testing (SAT)*, volume 10491 of *Lecture Notes in Computer Science*, pages 298–313. Springer, 2017.

[47] Tomáš Peitl, Friedrich Slivovsky, and Stefan Szeider. Polynomial-time validation of QCDCL certificates. In Olaf Beyersdorff and Christoph M. Wintersteiger, editors, *International Conference on Theory and Practice of Satisfiability Testing (SAT)*, volume 10929 of *Lecture Notes in Computer Science*, pages 253–269. Springer, 2018.

[48] Tomáš Peitl, Friedrich Slivovsky, and Stefan Szeider. Combining resolution-path dependencies with dependency learning. In *International Conference on Theory and Practice of Satisfiability Testing (SAT)*, pages 306–318, 2019.

[49] Marcus N. Rabe and Leander Tentrup. CAQE: A certifying QBF solver. In Roope Kaivola and Thomas Wahl, editors, *Conference on Formal Methods in Computer-aided Design (FMCAD)*, pages 136–143. IEEE, 2015.

[50] Markus N. Rabe and Sanjit A. Seshia. Incremental determinization. In Nadia Creignou and Daniel Le Berre, editors, *International Conference on Theory and Practice of Satisfiability Testing (SAT)*, volume 9710 of *Lecture Notes in Computer Science*, pages 375–392. Springer, 2016.

[51] Jussi Rintanen. Asymptotically optimal encodings of conformant planning in QBF. In *National Conference on Artificial Intelligence (AAAI)*, pages 1045–1050. AAAI Press, 2007.

[52] John Alan Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1):23–41, 1965.

[53] Marko Samer. Variable dependencies of quantified CSPs. In Iliano Cervesato, Helmut Veith, and Andrei Voronkov, editors, *International Conference on Logic for Programming, Artificial Intelligence and Reasoning (LPAR)*, volume 5330 of *Lecture Notes in Computer Science*, pages 512–527. Springer, 2008.

[54] Marko Samer and Stefan Szeider. Backdoor sets of quantified Boolean formulas. *Journal of Automated Reasoning*, 42(1):77–97, 2009.

[55] Horst Samulowitz, Jessica Davies, and Fahiem Bacchus. Preprocessing QBF. In Frédéric Benhamou, editor, *International Conference on Principles and Practice of Constraint Programming (CP)*, volume 4204 of *Lecture Notes in Computer Science*, pages 514–529. Springer, 2006.

[56] Nathan Segerlind. The complexity of propositional proofs. *Bulletin of Symbolic Logic*, 13(4):417–481, 2007.

[57] Ohad Shacham and Emmanuel Zarpas. Tuning the VSIDS decision heuristic for bounded model checking. In *International Workshop on Microprocessor Test and Verification (MTV)*, page 75. IEEE Computer Society, 2003.

[58] João P. Marques Silva. The impact of branching heuristics in propositional satisfiability algorithms. In Pedro Barahona and José Júlio Alferes, editors, *Portugese Conference on Progress in Artificial Intelligence (EPIA)*, volume 1695 of *Lecture Notes in Computer Science*, pages 62–74. Springer, 1999.

[59] João P. Marques Silva, Inês Lynce, and Sharad Malik. Conflict-driven clause learning SAT solvers. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, pages 131–153. IOS Press, 2009.

[60] João P. Marques Silva and Karem A. Sakallah. GRASP - a new search algorithm for satisfiability. In Rob A. Rutenbar and Ralph H. J. M. Otten, editors, *International Conference on Computer-Aided Design (ICCAD)*, pages 220–227. IEEE Computer Society / ACM, 1996.

[61] Friedrich Slivovsky. *Structure in #SAT and QBF*. PhD thesis, Vienna University of Technology, 2015.

[62] Friedrich Slivovsky and Stefan Szeider. Computing resolution-path dependencies in linear time. In Alessandro Cimatti and Roberto Sebastiani, editors, *International Conference on Theory and Practice of Satisfiability Testing (SAT)*, volume 7317 of *Lecture Notes in Computer Science*, pages 58–71. Springer, 2012.

[63] Friedrich Slivovsky and Stefan Szeider. Soundness of Q-resolution with dependency schemes. *Theoretical Computer Science*, 612:83–101, 2016.

[64] Stefan Staber and Roderick Bloem. Fault localization and correction with QBF. In João Marques-Silva and Karem A. Sakallah, editors, *International Conference on Theory and Practice of Satisfiability Testing (SAT)*, volume 4501 of *Lecture Notes in Computer Science*, pages 355–368. Springer, 2007.

[65] Larry J. Stockmeyer and Albert R. Meyer. Word problems requiring exponential time: Preliminary report. In Alfred V. Aho, Allan Borodin, Robert L. Constable, Robert W. Floyd, Michael A. Harrison, Richard M. Karp, and H. Raymond Strong, editors, *ACM Symposium on Theory of Computing (STOC)*, pages 1–9. ACM, 1973.

[66] Martin Suda and Bernhard Gleiss. Local soundness for QBF calculi. In Olaf Beyersdorff and Christoph M. Wintersteiger, editors, *International Conference on Theory and Practice of Satisfiability Testing (SAT)*, volume 10929 of *Lecture Notes in Computer Science*, pages 217–234. Springer, 2018.

[67] G. S. Tseitin. On the complexity of derivation in propositional calculus. *Studies in Constructive Mathematics and Mathematical Logic, Part 2*, pages 115–125, 1968.

[68] A.M. Turing. On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society*, 42(2):230–265, 1936.

[69] Alasdair Urquhart. Hard examples for resolution. *Journal of the ACM*, 34(1):209–219, 1987.

[70] Moshe Y. Vardi. Boolean satisfiability: Theory and engineering. *Communications of the ACM*, 57(3):5, 2014.

[71] Ralf Wimmer, Christoph Scholl, Karina Wimmer, and Bernd Becker. Dependency schemes for DQBF. In Nadia Creignou and Daniel Le Berre, editors, *International Conference on Theory and Practice of Satisfiability Testing (SAT)*, volume 9710 of *Lecture Notes in Computer Science*, pages 473–489. Springer, 2016.

[72] Lintao Zhang and Sharad Malik. Conflict driven learning in a quantified Boolean satisfiability solver. In Lawrence T. Pileggi and Andreas Kuehlmann, editors, *International Conference on Computer-Aided Design (ICCAD)*, pages 442–449. IEEE Computer Society / ACM, 2002.