Computationally Hard Problems Are Hard for QBF Proof Systems Too

Agnes Schleitzer, Olaf Beyersdorff

Institut für Informatik, Friedrich-Schiller-Universität Jena, Germany
agnes.schleitzer@uni-jena.de, olaf.beyersdorff@uni-jena.de

Abstract

There has been tremendous progress in the past decade in the field of quantified Boolean formulas (QBF), both in practical solving as well as in creating a theory of corresponding proof systems and their proof complexity analysis. Both for solving and for proof complexity, it is important to have interesting formula families on which we can test solvers and gauge the strength of the proof systems. There are currently few such formula families in the literature.

We initiate a general programme on how to transform computationally hard problems (located in the polynomial hierarchy) into QBFs hard for the main QBF resolution systems Q-Res and QU-Res that relate to core QBF solvers. We illustrate this general approach on three problems from graph theory and logic. This yields QBF families that are provably hard for Q-Res and QU-Res (without any complexity assumptions).

1 Introduction

The primary goal of *proof complexity* is to examine the size of proofs within various formal proof systems. Originating from computational complexity (Cook and Reckhow 1979), proof complexity has significant connections to other domains, especially logic (Krajíček 2019; Cook and Nguyen 2010) and solving techniques (Buss and Nordström 2021). Proof complexity serves as the main theoretical framework to evaluate the strength of modern solving methods.

A key challenge in proof complexity is to establish *lower bounds* on proof size and to obtain *separations* between different calculi. This requires witnesses in terms of *specific formula families*. In propositional proof complexity, particularly for propositional resolution – well-studied also due to its close ties with SAT solving (Buss and Nordström 2021; Pipatsrisawat and Darwiche 2011; Atserias, Fichte, and Thurley 2011; Beame, Kautz, and Sabharwal 2004) – there is extensive literature on difficult formulas from various fields such as combinatorics (e.g., Haken 1985; Bonet et al. 2000), graph theory (Urquhart 1987), logic (Krajíček 1995), random formulas (Beame and Pitassi 1996), and many more (Krajíček 2019; Segerlind 2007).

In contrast, *proof complexity of quantified Boolean formulas* (QBF) is relatively nascent. While there are several QBF proof systems, Q-Resolution (Kleine Büning, Karpinski, and Flögel 1995) and QU-Resolution (Van Gelder 2012) are most notable. These systems extend propositional resolution by a universal reduction rule that allows for the elimination of specific universal variables from clauses.

Similar to SAT, QBF resolution systems are deeply intertwined with QBF solving methods (cf. Beyersdorff et al. 2021a, for a recent overview), with Q-Res and its extension long-distance Q-Resolution (LD-Q-Res Balabanov and Jiang 2012) corresponding to quantified conflict-driven clause learning (QCDCL, cf. Beyersdorff et al. 2021a; Zhang and Malik 2002; Beyersdorff and Böhm 2021; Lonsing, Egly, and Gelder 2013).

However, compared to the plethora of hard formulas for propositional resolution, there is a scarcity of interesting QBF families suitable for proof-theoretic analysis. Only few families (and their variations) have been utilised for lower bounds and separations in the QBF literature. Most notable among these are the KBKF formulas introduced in the initial Q-Res paper (Kleine Büning, Karpinski, and Flögel 1995), equality formulas (Beyersdorff, Blinkhorn, and Hinde 2019), parity formulas (Beyersdorff, Chew, and Janota 2019), and CR formulas (Janota and Marques-Silva 2015). These comprise the primary toolkit for QBF proof complexity and are used for nearly all known separations. We recently introduced a method to construct new hard QBF families (Schleitzer and Beyersdorff 2023) – this method can also be used to generate the KBKF and equality formulas – but they are highly structured and handcrafted.

Thus there is a strong need for more interesting and especially natural QBFs that are challenging for Q-Res or QU-Res. Such QBFs could not only advance proof complexity but also serve as benchmarks in solving, aiding in the comparison of different solving techniques.[1]

It is also not straightforward to use the large pool of hard propositional formulas for QBF purposes. Although the existentially quantified version of any CNF hard for propositional resolution is also hard for Q-Res and QU-Res, our focus is on 'genuine' QBF hardness arising from quantifier alternations rather than the propositional base system.[2]

[1] A track of crafted formulas was introduced into QBFEval 2020, and a tool to generate the mentioned QBF families was presented by Beyersdorff et al. (2021b).

[2] A formal framework for 'genuine' QBF hardness was intro-

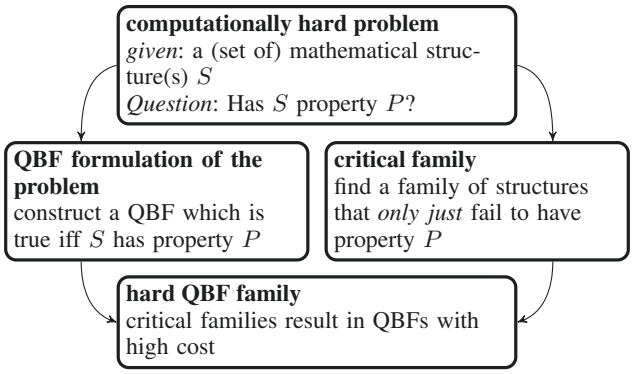

Figure 1: The basic procedure we use to construct families of hard QBFs from mathematical problems.

Our Contributions. We present a general method for constructing families of hard QBFs from computationally complex mathematical problems.

Specifically, we consider three problems, two from graph theory and one from logic, convert them into QBFs and show that these QBFs require exponential-size proofs in QU-Res (thereby also in Q-Res). The first problem `Succinct k-Radius` asks whether a graph, represented succinctly as a circuit, has radius $\leq k$ for fixed k. The problem is known to be complete for the third level Σ_3^p of the polynomial hierarchy PH. The second graph problem `k-Clique Colouring` is complete for Σ_2^p. We complement this with a problem from logic `ALL-EQUAL`$\exists\forall$`3SAT`, also complete for Σ_2^p. As the problems are in PH, they can be expressed naturally as sequences of QBFs.

Our *main technical work* goes into showing that these formulas are hard for the QBF resolution systems QU-Res and Q-Res. For this we use the semantic size-cost lower-bound technique for QBF calculi, developed by Beyersdorff, Blinkhorn, and Hinde (2019). This involves constructing explicit 'critical' instances of graphs or formulas on which the universal player demonstrably needs a large winning strategy. The workflow for our construction is depicted in Fig. 1.

We view these three explicit problems and hardness construction as *case studies* towards the far more general claim that *any* computationally hard problem can be shown to give rise to hard QBFs via our method (though specific problems will likely require individual, hand-crafted instances for the argument, cf. the discussion in Section 7).

We mention that the hardness results actually extend to more powerful QBF proof systems, namely all systems with bounded capacity, cf. (Beyersdorff, Blinkhorn, and Hinde 2019). In particular, our QBF families are hard not only for QU-Res, but also for the QBF versions of polynomial calculus and cutting planes (Beyersdorff, Blinkhorn, and Hinde 2019; Beyersdorff et al. 2018).

The idea to use computationally complex problems for hard QBFs is quite natural. Indeed, we can take any PSPACE-complete problem L (or a problem complete for some level Σ_k^p with $k \geq 2$), express it as QBFs (this is possible as deciding validity of QBFs is PSPACE complete as well), and these QBFs will be hard for *any* QBF proof system Q *assuming* NP $\neq$ PSPACE (or NP $\neq \Sigma_k^p$, resp.).[3]

While using PSPACE-hard problems such as games for QBF instances is a natural idea (Fraenkel and Goldschmidt 1987; Shaik et al. 2023; He, Saffidine, and Thielscher 2024), such formulas have never been shown to be formally hard for QBF proof systems, a task that appears daunting due to the syntactically complex QBF translations (Shaik et al. 2023). Our main achievement here is that our hardness results hold *unconditionally* without assuming any unproven conjectures from computational complexity.

We highlight that similar results are *not known in propositional proof complexity*. The *clique* formulas are a famous example (Beyersdorff et al. 2012). They express the NP-complete problem that a given graph contains a k-clique. Hence by the same argument as above, they will give rise to hard CNFs (for suitably chosen graphs) for any propositional proof system *assuming* NP $\neq$ co-NP. To show this *unconditionally* is a very hard problem in propositional proof complexity. So far the claim has been shown unconditionally for tree-like resolution (Beyersdorff, Galesi, and Lauria 2013) and, in a break-through result, for regular resolution (Atserias et al. 2021), while the case of general resolution is wide open. In contrast, our method allows to show such results for QBF proof systems very elegantly.

Organisation. We start in Section 2 with preliminaries on QBF and relevant proof systems. Section 3 contains our generic construction of hard QBFs from mathematical problems. Sections 4-6 each present a mathematical problem, the QBF translations, and the hardness argument. We conclude in Section 7 with a discussion.

2 Preliminaries

A *Conjunctive Normal Form (CNF)* is a conjunction of *clauses*, where each clause is a disjunction of literals. A *literal* l is a propositional variable x or its negation $\overline{x}$, we denote this by $\mathsf{vars}(l) = x$. We also write CNFs as sets of clauses; the size of a CNF is the number of its clauses.

QBFs. A *Quantified Boolean Formula (QBF)* in *closed prenex form* $\phi = \mathcal{P} \cdot \varphi$ consists of a *quantifier prefix* $\mathcal{P}$ and a propositional formula φ, referred to as the *matrix*. The prefix is a sequence of quantifiers $Q \in \{\forall, \exists\}$, each followed by a set of variables. For a *closed* QBF (which is our focus), $\mathcal{P}$ quantifies exactly the variables appearing in φ. Consequently, for $\mathcal{P} = Q_1 X_1 Q_2 X_2 \dots Q_n X_n$, the matrix φ is expressed in terms of the variables $\bigcup_{i\in[n]} X_i$ and we write $\mathsf{vars}(\mathcal{P} \cdot \varphi) = \mathsf{vars}(\varphi) = \bigcup_{i\in[n]} X_i$. Since a closed QBF has no free variables, it is either *true* or *false*. We denote the set of *existential variables* (associated with $\exists$) in $\mathcal{P} \cdot \varphi$ by $\mathsf{vars}_\exists(\varphi)$, the set of *universal variables* (associated with $\forall$) by $\mathsf{vars}_\forall(\varphi)$. A QBF with a CNF matrix is termed a *QCNF*.

duced by Beyersdorff, Hinde, and Pich (2020). All the aforementioned QBF examples – KBKF, equality, and parity – are genuinely hard in this sense.

[3]This follows as short Q-proofs for QBF translations of L could be guessed non-deterministically, thus placing $L \in$ NP and implying NP $=$ PSPACE (or NP $= \Sigma_k^p$, respectively).

Axiom	$\frac{}{\overline{C}}$	C is a non-tautologous clause in the matrix φ.
Q-Res	$\frac{C_1 \cup \{x\} \quad C_2 \cup \{\overline{x}\}}{C_1 \cup C_2}$	$C_1 \cup C_2$ is non-tautologous; $x \in \mathsf{vars}_\exists(\phi)$.
QU-Res	$\frac{C_1 \cup \{x\} \quad C_2 \cup \{\overline{x}\}}{C_1 \cup C_2}$	$C_1 \cup C_2$ is non-tautologous.
∀Red	$\frac{C \cup \{u\}}{C}$	$u \in \mathsf{vars}_\forall(\phi)$ and quantified right of each existential variable in C regarding $\mathcal{P}$.

Figure 2: Rules of the QBF proof systems Q-Res and QU-Res for a QBF $\phi = \mathcal{P}.\varphi$.

An *assignment* is a mapping of variables to (Boolean) truth values in $\{0, 1\}$. Sometimes, we represent an assignment as a set of variable-value pairs. We write $\langle V \rangle$ for the set of all possible assignments to V.

Closed QBFs can be viewed as a two-player game between an existential and a universal player, who assign truth values to all variables in the order dictated by the quantifier prefix. The existential player assigns values to existential variables, while the universal player assigns values to universal ones. The existential player wins if the resulting assignment satisfies the matrix; otherwise, the universal player wins. For any closed QBF, one of the players has a *winning strategy*. This game is known as the *assignment game*.

A *countermodel* is a winning strategy for the universal player. While countermodels are often described as a collection of functions (one for each universal variable), we prefer to consider them as a single function that outputs an assignment to the universal variables (cf. e.g. (Beyersdorff, Blinkhorn, and Mahajan 2020)). The range of a countermodel is the number of distinct assignments to the universal variables that can be produced under the strategy. The range of a countermodel on a single universal block is the number of different assignments to the variables of that block.

Proof systems. *Resolution* (Res) is a refutational proof system for propositional formulas with two inference rules. For an input CNF χ, any clause $C \in \chi$ can be used as an axiom. Additionally, from two clauses $C_1 \cup x$ and $C_2 \cup \overline{x}$, the resolvent $C_1 \cup C_2$ can be derived by resolving over the pivot x.

Q-Res (Kleine Büning, Karpinski, and Flögel 1995) lifts the resolution method to QBFs. The system adapts the resolution rule to Q-Res, whereby only existential pivots are permitted and tautological resolvents are prohibited. Universal variables are eliminated using universal reduction (∀Red). The rules are shown in Figure 2.

QU-Res (Van Gelder 2012) extends the weaker system Q-Res by allowing resolution over universal pivots.

The *size* $|\pi|$ of a proof π is defined as the number of clauses in π.

3 From Hard Problems to Hard QBFs

Our aim is to construct hard formulas for QU-Res from difficult mathematical problems. The hardness should be intuitively easy to understand. For this purpose, we use the lower bound technique via cost introduced by Beyersdorff, Blinkhorn, and Hinde (2019). We start by recalling it.

Definition 1 (Beyersdorff, Blinkhorn, and Hinde 2019)**.** *We consider all countermodels for a false QBF ϕ and determine for each of them the largest range on a single universal block. The minimum over these cardinalities is the* cost *of ϕ.*

For Σ_3^b formulas (i.e., with only one universal block), cost coincides with the minimum cardinality of the range of a countermodel for ϕ. Cost is an absolute lower bound for proof size in QU-Res (and Q-Res):

Theorem 2 (Beyersdorff, Blinkhorn, and Hinde 2019)**.** *Let ϕ be a false QCNF. Then QU-Res refutations of ϕ have size at least* $\mathrm{cost}(\phi)$.

Figure 1 shows the basic procedure we use to construct new families of hard QBFs. At this point, it is not yet clear exactly what it means that a 'critical' structure *only just* fails to have a property. This will become clear later and has to do with the fact that the non-fulfilment of the property must not be too obvious, i.e. there must be numerous ways of *almost* proving the property (and failing finally).

In the following, we will analyse selected hard problems in more detail. We use problems from the polynomial hierarchy, Schaefer and Umans (2002) provide a good overview.

4 Succinct k-Radius

First we deal with the radius of graphs – more precisely with the question of whether there is a vertex in a given (directed) graph from which every other vertex can be reached in at most k steps. We will define this problem formally soon, but first let us consider representations of graphs. It is not difficult to verify that the described problem is easy if the graph is given as an adjacency matrix. We therefore consider succinct representations, one of which is the following:

Definition 3. *Given a directed graph $G = (V_G, E_G)$ with $V = \{0, 1\}^n$, we call a circuit C a* Galperin-Wigderson representation *of G iff C has exactly $2n$ input gates and one output gate and it holds that $C(x, y) = 1 \leftrightarrow (x, y) \in E$ for any $x, y \in V$. Note that there can be more than one Galperin-Wigderson representation of a graph.*

With the help of this circuit representation it is possible to represent graphs with 2^n vertices much more succinctly, possibly even by circuits of polynomial size in n.

In order to formally describe the problem, we still need to define some concepts related to k-radius and reachability:

Definition 4. *With respect to a vertex $v \in V$, we call a vertex $u \in V, u \neq v$ k-*reachable *if there is a path of length at most k between v and u in $G = (V, E)$. Otherwise we refer to u as k-*isolated *(with respect to v).*

*For a graph $G = (V, E)$, a vertex $v \in V$ is called a k-*center *if any $u \in V$, $u \neq v$ is k-reachable from v.*

The radius *of a graph $G = (V, E)$ is the smallest $k \in \mathbb{N}$ for which there exists a k-center of G.*

We now define the problem we want to analyse:

Definition 5 (Hemaspaandra et al. 2010; Galperin and Wigderson 1983)**.** *Given a Galperin-Wigderson representation C of a directed graph $G = (V_G, E_G)$ and an integer k. Succinct k-Radius(G) asks, whether G has radius at most k.*

This problem is Σ_3^p-complete for any fixed $k \geq 2$ (Hemaspaandra et al. 2010).

4.1 QBF Encodings of `Succinct k-Radius`

We will now construct QBFs describing `Succinct k-Radius`. We first need variables representing vertices from G. For $|V_G| = n$ we use variables $P_i = \{p_1^i, \dots, p_{\log n}^i\}$ for $i \in [0, k]$, which encode vertices $p_0, \dots, p_k \in V_G$. We want the matrix to state that $p_0 \dots p_k$ is a path in G. We must also consider the case where the length of the path is smaller than k. We will therefore allow $p = p_0 \dots p_k$ to repeat the same vertex multiple times (even without the corresponding loop being contained in G).

Let $\varphi_{\text{edge}}(i, j)$ be a propositional formula that encodes the circuit C with input P_i, P_j. Please note that this formula is not necessarily a CNF. We will address this problem later, for now the encoding as a propositional formula suffices (allowing us to express C without additional variables).

Let $\varphi_{\text{equal}}(i, j) := \bigwedge_{k \in \log n} \left((p_k^i \vee \overline{p_k^j}) \wedge (\overline{p_k^i} \vee p_k^j) \right)$ be a formula checking the encodings of p_i, p_j (which are represented by P_i, P_j) for bitwise equality. Now it is quite easy to assemble prefix and matrix:

$$\exists P_0 \forall P_k \exists P_1 \dots P_{k-1} \cdot \bigwedge_{i \in [k]} (\varphi_{\text{equal}}(i-1, i) \vee \varphi_{\text{edge}}(i-1, i)) .$$

The size of this formula is $\mathcal{O}(k(\log n + |\varphi_{\text{edge}}|))$.

As noted above, the formula presented so far is not necessarily a CNF. Although the bitwise equality checks have already been formulated as CNF, this does not apply to the encoding of the circuit C (for edge checking) and therefore the entire matrix. However, a Tseitin transformation (TTF in the formula below) allows us to construct a satisfiability-equivalent formula whose size is linear in the size of the original formula, i.e. $\mathcal{O}(\log n + |\varphi_{\text{edge}}|)$. This involves inserting additional variables V_{Tseitin}, which we simply add to the last existential block.

This yields the following formula:

$$\text{SR}_k(G) := \exists P_0 \forall P_k \exists P_1 \dots P_{k-1} V_{\text{Tseitin}} \cdot \text{TTF}\left(\bigwedge_{i \in [k]} (\varphi_{\text{equal}}(i-1, i) \vee \varphi_{\text{edge}}(i-1, i)) \right) .$$

It should be intuitively clear that the constructed QBFs express the graph problem:

Lemma 6. *$\text{SR}_k(G)$ is true iff G has radius at most k.*

4.2 Constructing a Critical Graph Family

To show hardness of the SR()-formulas, we need a family of directed graphs which can be succinctly represented by small circuits and whose instances do not have radius at most k, but a lot of vertices, which are *almost* k-centers (this is what we called a *critical family* in Fig. 1). This should translate to large strategy size resp. cost of the formulas, which allows us to show hardness via Theorem 2. To better describe the graphs and our requirements for them, we define the concept of an *almost* k-center.

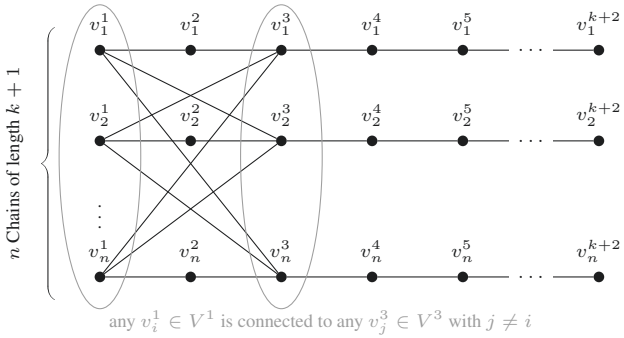

Figure 3: Graph G_n^k with radius $> k$ and n almost-k-centers v_i^1 with corresponding corruptors v_i^{k+2}. Note that the subgraph induced by $V^1 = \{v_i^1 \mid i \in [n]\}$ and $V^3 = \{v_i^3 \mid i \in [n]\}$ is *almost* a complete bipartite graph $K_{\{n,n\}}$, missing only one edge $\{v_i^1, v_i^3\}$ per vertex.

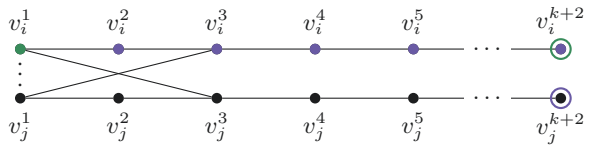

Figure 4: G_n^k has radius $> k$ for $n \geq 2, k > 2$. Vertex v_i^1 (●) is k-isolated from v_i^{k+2} (◉). All vertices $v_i^2, \dots, v_i^{k+2}$ (●) are k-isolated from v_j^{k+2} (◉). i, j are arbitrary with $i \neq j$.

Definition 7. *We call a vertex $v \in V$ an* almost-k-center *of $G = (V, E)$, if there is a vertex $u \in V, u \neq v$ which is k-isolated from v, but any vertex in $V \setminus \{u, v\}$ is k-reachable from v. We call u the* corruptor *of v.*

Note that (according to the definition of `Succinct k-Radius`) we actually consider directed graphs. In the following we will construct a family of undirected graphs, all previously defined concepts are easily transferable. To finally obtain undirected graphs, all edges can easily be realised in both directions.

The idea is to use chains of length $k+1$ (i.e. they consist of $k+2$ vertices each), such that the last vertex of a chain is not k-reachable from the first one (and is thus its corruptor). Now, to make the first vertices almost-k-centers, we need to combine the chains such that the last vertices of other chains *are* k-reachable. This can be done via connections, which, in a sense, skip a vertex. We realise this by connecting the first vertex of a chain with the third vertex of each *other* chain. The resulting graph G_n^k is shown in Figure 3.

The next claim is easy to verify with this intuition:

Lemma 8. *The graph G_n^k constructed as described above has radius $> k$ and n almost-k-centers with pairwise different corruptors for $n \geq 2, k > 2$.*

The constructed graphs have $|V| = (k+2) \cdot n \in \mathcal{O}(n)$ vertices and $|E| = (k+1) \cdot n + n \cdot (n-1) = n^2 + kn \in \mathcal{O}(n^2)$ edges (since k is a constant).

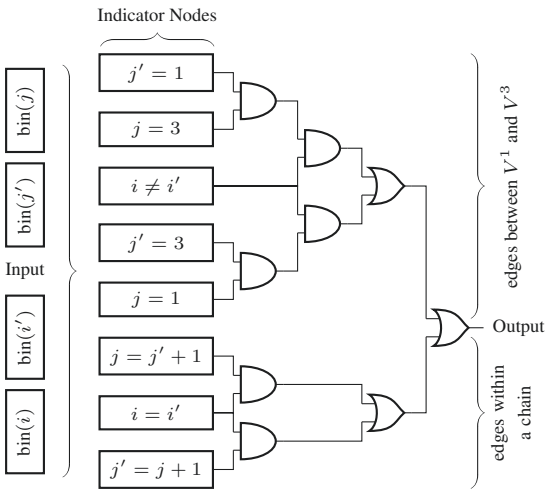

Figure 5: Sketch of a Boolean circuit that calculates whether or not an edge exists between two input vertices using the indicator nodes presented. i, i' are indices of chains, j, j' indices within a chain.

As we want to use the technique from Theorem 2, it is important that the graphs can be represented succinctly by circuits. We will specify small circuits of size $\mathcal{O}(\log n)$ that receive two vertices as input and determine the (non-)existence of an edge between these vertices. Note that the circuits include the above-mentioned step of realising each undirected edge in both directions, thus representing directed graphs as desired.

Lemma 9. *The family of G_n^k graphs as described in this section can be represented by circuits of logarithmic size.*

Proof. We have $\log(n \cdot (k+2))$ bits available for the encoding of a single vertex. As the vertices in our graphs are arranged in a grid (see Fig. 3), we encode the coordinates i and j for v_i^j (this is an easy task because $i \in [n]$, $j \in [k+2]$, $\log(n \cdot (k+2)) = \log(n) + \log(k+2)$). For input i, j, i', j', the circuit must now provide indicator nodes for the following situations:

- $j = 1, j' = 1$,
- $j = 3, j' = 3$,
- $i = i', i \neq i'$
- $j' = j + 1, j = j' + 1$.

So overall we need a binary incrementer, testing for equality (between two n-bit numbers as well as between an n-bit number and a constant) and testing for inequality (between two n-bit numbers). Any of these calculations can be performed in linear size (in the input length). Using these indicator nodes, we can now easily construct circuits that output 1 if there is an edge between the input vertices and 0 otherwise (see Fig. 5). These circuits obviously have size $\mathcal{O}(\log n)$ as desired. □

4.3 Hardness of the QBFs for QU-Res

The number of almost-k-centers and their pairwise different corruptors in G_n^k as stated in Lemma 8 together with Lemma 9 imply that the formulas constructed from this graph family have exponential cost:

Theorem 10. $\mathsf{cost}(\mathrm{SR}_k(G_n^k)) \geq n$ *for* $n \geq 2, k > 2$.

Proof. Let G_n^k and $\mathrm{SR}_k(G_n^k)$ be constructed according to the descriptions in Sections 4.2 and 4.1. Then $\mathrm{SR}_k(G_n^k)$ has size $\mathcal{O}(\log n)$ because G_n^k can be represented by circuits of logarithmic size according to Lemma 9. By Lemma 8, G_n^k has n almost-k-centers with pairwise different corruptors. A winning strategy for the universal player must take all these corruptors into account, as the existential player can specify the corresponding almost-k-centers in his moves for the first existential block. $\mathrm{SR}_k(G_n^k)$ therefore has cost $\geq n$, which is exponential in the size of the formula. □

From Theorems 2 and 10 we infer:

Corollary 11. $\mathrm{SR}_k(G_n^k)$ *require* QU-Res *proofs of size at least* 2^n.

5 k-Clique Colouring

For our next problem, we will deal with cliques in graphs as well as with special colourings. Let us first introduce some definitions concerning different types of colourings that will be important in this section.

Definition 12. *A* graph-colouring $c : V \to C$ *colours the vertices of a graph $G = (V, E)$. It is called* proper *if for any $e = (u, v) \in E$, $u \neq v$ the vertices u, v are coloured differently, i.e. $c(u) \neq c(v)$. A* k-graph-colouring *uses at most k colours, thus $|c(V)| \leq k$. A graph G is* k-colourable *if it has a proper k-colouring.*

A (k-)clique-colouring *of G is a (k-)graph-colouring of G such that for any maximal clique $C \subseteq V$ there are two vertices u, v with $\{u, v\} \subseteq C$ and $c(u) \neq c(v)$ (i.e. there are no monochromatic maximal cliques).*

We can now define the problem of interest:

Definition 13 (Marx 2011)**.** *Given a graph $G = (V, E)$ and an integer k,* `k-Clique Colouring` *asks, whether there is a k-clique-colouring for G.*

This problem is Σ_2^p-complete for any $k \geq 2$ (Marx 2011).

5.1 QBF Encodings of k-Clique Colouring

The QBFs we will construct from this question shall express the following process: The existential player chooses a k-colouring of the vertices. The universal player then selects a (maximal) clique of the graph. Finally, the existential player selects two vertices from the clique with different colours. If the existential player has a winning strategy, the graph is k-clique-colourable and the QBF is true. If, on the other hand, the universal player has a winning strategy (i.e. he finds a monochromatic maximum clique for each colouring), the graph is not k-clique-colourable and the QBF is false.

To realise this, we first introduce a few variables. We assume G to have vertex set $V = \{v_1, \ldots, v_n\}$.

- $b_{ij}, i \in [n], j \in [k]$: v_i is coloured with j.
- $c_i, i \in [n]$: v_i is contained in the considered clique.
- $t_i, i \in [n]$: t_i is chosen as witness for non-monochromaticity.

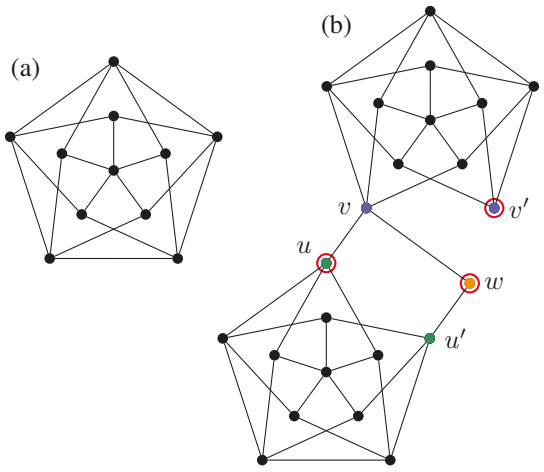

Figure 6: (a) Grötzsch graph, the smallest non 3-colourable graph and (b) how to build an independent set of size 3 (marked red) whose vertices must be coloured pairwise differently in any 3-Clique-Colouring. We call (b) the G_3^{ips}-graph.

It is immediately clear that this enumeration already anticipates the division of the variables into the quantifier blocks: The prefix will be $\exists B \forall C \exists T$, where $B = \{b_{ij} \mid i \in [n], j \in [k]\}$, $C = \{c_i \mid i \in [n]\}$, $T = \{t_i, i \in [n]\}$.

We divide the construction of the matrix into individual sub-statements in order to increase clarity: ① B represents a k-colouring. ② C represents a clique (we identify C with this clique as well). ③ C is maximal. ④ The vertices selected by T are in C. ⑤ There are at most two vertices selected by T. ⑥ The vertices selected by T are coloured differently.

The exact formulas that express the statements ① - ⑥ are easy to construct.At this point, it is sufficient for us to know that, from a graph $G = (V, E)$ with $|V| = n$ and a number k of colours, we obtain a formula $\mathrm{CC}_k(G) = \exists B \forall C \exists T \cdot ① \wedge [(② \wedge ③) \rightarrow (④ \wedge ⑤ \wedge ⑥)]$ whose size is cubic in n (even if it should be necessary to apply Tseitin transformation to convert the matrix into CNF).

5.2 Constructing a Critical Graph Family

To find a graph family that allows us to assess the `k-Clique Colouring` formulas in terms of proof complexity using Theorem 2, let us examine the relationship between k-colourability and k-clique-colourability: Obviously, k-colourability coincides with k-clique-colourability if we consider triangle-free graphs. A triangle-free graph that is not k-colourable is therefore also not k-clique-colourable. The smallest interesting example of such graphs is the Grötzsch graph (Mycielski 1955; Chvátal 1974; see Figure 6). The construction can be extended arbitrarily for $k > 3$ (as Mycielsky graphs, Mycielski 1955). However, we will just use the case $k = 3$.

For our purposes, we do not just need a single graph that is not 3-clique-colourable, but a whole family. In addition, it should be possible to force the universal player to select certain cliques by skilful moves of the existential player (selection of special colourings).

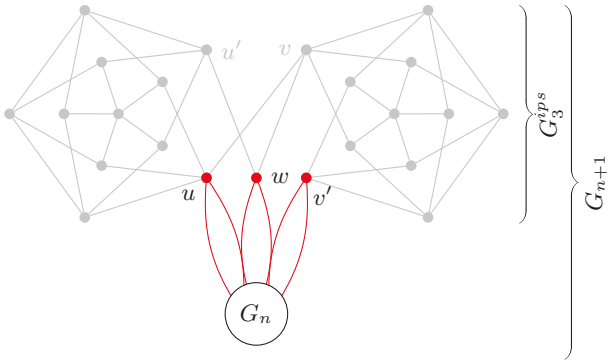

Figure 7: From G_n to G_{n+1}. The double red edges indicate that all vertices in G_n are adjacent with u, v' and w.

Definition 14. *We call a maximal clique $C \subseteq V$ within a graph $G = (V, E)$ k-colour-selectable, if we can find a k-colouring of V, which colours C monochromatically, but any other maximal clique is not monochromatic.*

We need exponentially many 3-colour-selectable cliques in our graphs. To generate such graphs inductively, we use the following idea: Given a non-3-clique-colourable graph with m 3-colour-selectable cliques. How can we modify the graph by adding a constant number of vertices so that the new graph has, for example, $3m$ 3-colour-selectable cliques? The idea is to add three vertices which build an independent set, but must be coloured pairwise differently in each clique colouring. If we then connect each vertex of the original graph to each vertex of the independent set, we get three new (maximal) cliques from each old (maximal) clique – and in the case of a monochromatic clique, exactly one of the three resulting new cliques must be monochromatic again.

Conveniently, the Grötzsch graph is useful to us in this construction in several ways: On the one hand, it serves as the start of the induction, as it cannot be 3-clique-coloured, but the result after removing any edge can (which means that any edge is 3-colour-selectable).On the other hand, we use it to construct the three vertices that must necessarily be coloured differently (see Fig. 6(b) for more details).

Let us take a look at the inductive construction of $G_n = (V_n, E_n)$. For $n = 1$ we choose $G_n = G_1$ to be the Grötzsch graph, which has 11 vertices and 20 edges. Each of these edges is 3-colour-selectable, so we have $m = 20$ 3-colour-selectable cliques. We generate G_{n+1} as follows (see also Figure 7):

- Take G_n and add G_3^{ips} shown in Figure 6(b) (You may need to rename vertices of G_n to avoid duplication).
- Connect each vertex from G_n with each vertex from the independent set $\{u, v', w\}$ in G_3^{ips}.
- The result is G_{n+1}.

Note that the maximal cliques resulting from the connection of G_n and G_3^{ips} are $n+2$-cliques, while the maximal cliques inside G_3^{ips} (as a subgraph of G_{n+1}) are still edges.

The number of vertices of these graphs is linear, the number of edges quadratic in n.

The following statement is shown by induction, working along the composition of the graphs.

Lemma 15. *The graph G_n constructed as described above has at least $3^{n-1} \cdot 20 \in \Omega(3^n)$ 3-colour-selectable cliques and is not 3-clique-colourable.*

5.3 Hardness of the QBFs for QU-Res

Since any 3-colour-selectable clique must be part of any winning strategy for the universal player, their exponential number (Lemma 15) immediately implies that the formulas constructed from G_n have exponential cost:

Theorem 16. $\mathsf{cost}(\mathrm{CC}_3(G_n)) \geq 20 \cdot 3^{n-1}$ *for* $n \in \mathbb{N}$.

From Theorem 16 and Theorem 2 we obtain:

Corollary 17. $\mathrm{CC}_3(G_n)$ *require proofs of exponential size in QU-Res.*

The construction is also possible for more colours: The graphs G_k^{ips} become larger for $k > 3$, but retain constant size. The Mycielsky graphs then serve as starting points of the construction instead of the Grötzsch graph.

6 ALL-EQUAL∃∀3SAT

Finally, let us analyse a problem from logic:

Definition 18. *Given a 3-CNF formula $\varphi(X, Y)$, where (X, Y) is a partition of $\mathsf{vars}(\varphi)$, ALL-EQUAL∃∀3SAT asks, whether there is an assignment to the variables in X such that for each assignment to the variables in Y there is at least one clause that contains only true or only false literals.*

ALL-EQUAL∃∀3SAT is Σ_2^p-complete, since it is the complement of NOT-ALL-EQUAL∀∃3SAT (Eiter and Gottlob 2000) which we know to be Π_2^p-complete.

Constructing QBFs from ALL-EQUAL∃∀3SAT is straightforward. The X-variables form the first existential block and the Y-variables the following universal block. We add $\log n$ selector variables $C = \{c_1, \ldots, c_{\log n}\}$ in a second existential block to select a clause.

The matrix is intended to express: If clause $C_i \in \varphi(X, Y)$ was selected using the C-variables, all literals in C_i should be identically true or false. Thus, we need the following subformulas: χ_i to express, that clause $C_i \in \varphi(X, Y)$ was selected using the C-variables and ψ_i to express, that all literals in C_i are identically true or false. Now, given a 3-CNF $\varphi(X, Y) = \bigwedge_{i=1}^{n} C_i$ with $C_i = (l_1^i, l_2^i, l_3^i)$, the matrix can be written as a simple conjunction of implications:

$$\mathrm{AE}(\varphi(X, Y)) = \exists X \forall Y \exists C \cdot \bigwedge_{i \in [n]} \chi_i \to \psi_i,$$

which gives us formulas whose size is linear in n.

It is quite simple to construct corresponding (propositional) formulas that allow us to prove hardness using Theorem 2. We therefore define $\varphi_n(X_n, Y_n)$ for $n \in \mathbb{N}$ as following: Let $X_n = \{x_i^a, x_i^b \mid i \in [n]\}$ and $Y_n = \{y_i \mid i \in [n]\}$. Let $C_i = \{x_i^a, x_i^b, \overline{y_i}\}$ for $i \in [n]$ and

$$\varphi_n(X_n, Y_n) = \bigwedge_{i \in [n]} C_i.$$

The special feature of these formulas is that certain assignments from $\langle X_n \rangle$ force certain assignments from $\langle Y_n \rangle$ to ensure that each clause contains both true and false literals. We capture this in a definition:

Definition 19. *We call an assignment $\beta \in \langle Y_n \rangle$* non-monotonicity-enforceable[4] *with respect to $\varphi_n(X_n, Y_n)$, iff there is an assignment $\alpha \in \langle X_n \rangle$ such that α together with β results in every clause containing both true and false literals, but α combined with any assignment from $\langle Y_n \rangle \setminus \{\beta\}$ results in at least one clause containing only true or only false literals.*

It is easy to see that any assignment $\beta \in \langle Y_n \rangle$ is enforceable by assigning $\alpha(x_i^a) = \alpha(x_i^b) = \overline{\beta(y_i)}$ for $i \in [n]$:

Lemma 20. $\varphi_n(X_n, Y_n)$ *has 2^n non-monotonicity-enforceable assignments to the variables in Y_n.*

The findings from Lemma 20 immediately lead to the conclusion, that $\mathrm{AE}(\varphi_n(X_n, Y_n))$ has high cost, since any non-monotonicity-enforceable assignment needs to be part of a winning strategy for the universal player:

Theorem 21. $\mathsf{cost}(\mathrm{AE}(\varphi_n(X_n, Y_n))) \geq 2^n$.

From Theorem 21 and Theorem 2 we infer:

Corollary 22. $\mathrm{AE}(\varphi_n(X_n, Y_n))$ *require proofs of exponential size in QU-Res.*

7 Discussion

The contribution of the article lies less in the concrete formulas presented, but rather in the demonstration of a *general approach:* Generating hard QBFs from difficult mathematical problems. We know a lot of hard mathematical problems and quantifier alternations naturally appear in many of them. The potential of our method therefore appears to be large and it should be possible to construct numerous new hard QBF families. In particular, it becomes clear that for the hardness of the formulas, a family of the underlying mathematical structures must be found, which has special properties in each case. If, on the other hand, families that do not completely fulfil these conditions (or even random families) are used for the construction, formulas may be created that could be well suited as benchmarks for solver competitions. Also, it appears interesting to generate formulas from such mathematical structures that make them just *not* easy, i.e. intermediate between short (polynomial-size) proofs and exponential hardness. These could lead to interesting test cases for different solvers.

The special feature of our formulas lies in their naturalness – the selection of mathematical problems from the polynomial hierarchy results in very natural quantifier alternations in the constructed QBFs. The close link to mathematical problems is also advantageous for hardness proofs, as the underlying structures can be used to argue for high strategy size without having to analyse syntactic details of the sometimes quite complex QBFs.

[4] Please note that this monotonicity concept differs from the one generally used for clauses, where the polarity of the literals (and not their assignment) is considered.

Acknowledgements

Research was supported by the Carl-Zeiss Foundation and DFG grant BE 4209/3-1.

References

Atserias, A.; Bonacina, I.; de Rezende, S. F.; Lauria, M.; Nordström, J.; and Razborov, A. A. 2021. Clique Is Hard on Average for Regular Resolution. *J. ACM*, 68(4): 23:1–23:26.

Atserias, A.; Fichte, J. K.; and Thurley, M. 2011. Clause-Learning Algorithms with Many Restarts and Bounded-Width Resolution. *J. Artif. Intell. Res.*, 40: 353–373.

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

Beame, P.; Kautz, H. A.; and Sabharwal, A. 2004. Towards Understanding and Harnessing the Potential of Clause Learning. *J. Artif. Intell. Res. (JAIR)*, 22: 319–351.

Beame, P.; and Pitassi, T. 1996. Simplified and improved resolution lower bounds. In *Proc. 37th IEEE Symposium on the Foundations of Computer Science*, 274–281. IEEE Computer Society Press.

Beyersdorff, O.; Blinkhorn, J.; and Hinde, L. 2019. Size, Cost, and Capacity: A Semantic Technique for Hard Random QBFs. *Logical Methods in Computer Science*, 15(1).

Beyersdorff, O.; Blinkhorn, J.; and Mahajan, M. 2020. Hardness Characterisations and Size-Width Lower Bounds for QBF Resolution. In *Proc. ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 209–223. ACM.

Beyersdorff, O.; and Böhm, B. 2021. Understanding the Relative Strength of QBF CDCL Solvers and QBF Resolution. In *12th Innovations in Theoretical Computer Science Conference (ITCS 2021)*, volume 185 of *Leibniz International Proceedings in Informatics (LIPIcs)*, 12:1–12:20.

Beyersdorff, O.; Chew, L.; and Janota, M. 2019. New Resolution-Based QBF Calculi and Their Proof Complexity. *ACM Transactions on Computation Theory*, 11(4): 26:1–26:42.

Beyersdorff, O.; Chew, L.; Mahajan, M.; and Shukla, A. 2018. Understanding cutting planes for QBFs. *Inf. Comput.*, 262: 141–161.

Beyersdorff, O.; Galesi, N.; and Lauria, M. 2013. Parameterized Complexity of DPLL Search Procedures. *ACM Transactions on Computational Logic*, 14(3): 20:1–20:21.

Beyersdorff, O.; Galesi, N.; Lauria, M.; and Razborov, A. 2012. Parameterized Bounded-Depth Frege is Not Optimal. *ACM Transactions on Computation Theory*, 4(3).

Beyersdorff, O.; Hinde, L.; and Pich, J. 2020. Reasons for Hardness in QBF Proof Systems. *ACM Transactions on Computation Theory*, 12(2).

Beyersdorff, O.; Janota, M.; Lonsing, F.; and Seidl, M. 2021a. Quantified Boolean Formulas. In Biere, A.; Heule, M.; van Maaren, H.; and Walsh, T., eds., *Handbook of Satisfiability*, Frontiers in Artificial Intelligence and Applications, 1177–1221. IOS Press.

Beyersdorff, O.; Pulina, L.; Seidl, M.; and Shukla, A. 2021b. QBFFam: A Tool for Generating QBF Families from Proof Complexity. In Li, C.; and Manyà, F., eds., *Theory and Applications of Satisfiability Testing (SAT)*, volume 12831 of *Lecture Notes in Computer Science*, 21–29. Springer.

Bonet, M. L.; Esteban, J. L.; Galesi, N.; and Johannsen, J. 2000. On the Relative Complexity of Resolution Refinements and Cutting Planes Proof Systems. *SIAM J. Comput.*, 30(5): 1462–1484.

Buss, S.; and Nordström, J. 2021. Proof Complexity and SAT Solving. In Biere, A.; Heule, M.; van Maaren, H.; and Walsh, T., eds., *Handbook of Satisfiability*, Frontiers in Artificial Intelligence and Applications, 233–350. IOS Press.

Chvátal, V. 1974. The minimality of the mycielski graph. *Graphs and Combinatorics*, 243–246.

Cook, S. A.; and Nguyen, P. 2010. *Logical Foundations of Proof Complexity*. Cambridge University Press.

Cook, S. A.; and Reckhow, R. A. 1979. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44(1): 36–50.

Eiter, T.; and Gottlob, G. 2000. Complexity results for some eigenvector problems. *International Journal of Computer Mathematics*, 76(1): 59–74.

Fraenkel, A. S.; and Goldschmidt, E. 1987. PSPACE-hardness of some combinatorial games. *Journal of Combinatorial Theory, Series A*, 46(1): 21–38.

Galperin, H.; and Wigderson, A. 1983. Succinct representations of graphs. *Information and Control*, 56(3): 183–198.

Haken, A. 1985. The intractability of Resolution. *Theoretical Computer Science*, 39: 297–308.

He, Y.; Saffidine, A.; and Thielscher, M. 2024. Solving Two-player Games with QBF Solvers in General Game Playing. In Dastani, M.; Sichman, J. S.; Alechina, N.; and Dignum, V., eds., *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 807–815. International Foundation for Autonomous Agents and Multiagent Systems / ACM.

Hemaspaandra, E.; Hemaspaandra, L. A.; Tantau, T.; and Watanabe, O. 2010. On the complexity of kings. *Theor. Comput. Sci.*, 411(4-5): 783–798.

Janota, M.; and Marques-Silva, J. 2015. Expansion-based QBF solving versus Q-resolution. *Theor. Comput. Sci.*, 577: 25–42.

Kleine Büning, H.; Karpinski, M.; and Flögel, A. 1995. Resolution for Quantified Boolean Formulas. *Inf. Comput.*, 117(1): 12–18.

Krajíček, J. 1995. *Bounded Arithmetic, Propositional Logic, and Complexity Theory*, volume 60 of *Encyclopedia of Mathematics and Its Applications*. Cambridge: Cambridge University Press.

Krajíček, J. 2019. *Proof complexity*, volume 170 of *Encyclopedia of Mathematics and Its Applications*. Cambridge University Press.

Lonsing, F.; Egly, U.; and Gelder, A. V. 2013. Efficient Clause Learning for Quantified Boolean Formulas via QBF

Pseudo Unit Propagation. In *Proc. International Conference on Theory and Applications of Satisfiability Testing (SAT)*, 100–115.

Marx, D. 2011. Complexity of clique coloring and related problems. *Theor. Comput. Sci.*, 412(29): 3487–3500.

Mycielski, J. 1955. Sur le coloriage des graphs. In *Colloquium Mathematicae*, volume 3, 161–162.

Pipatsrisawat, K.; and Darwiche, A. 2011. On the power of clause-learning SAT solvers as resolution engines. *Artif. Intell.*, 175(2): 512–525.

Schaefer, M.; and Umans, C. 2002. Completeness in the polynomial-time hierarchy: A compendium. *SIGACT news*, 33(3): 32–49.

Schleitzer, A.; and Beyersdorff, O. 2023. Classes of Hard Formulas for QBF Resolution. *J. Artif. Intell. Res.*, 77: 1455–1487.

Segerlind, N. 2007. The Complexity of Propositional Proofs. *Bulletin of Symbolic Logic*, 13(4): 417–481.

Shaik, I.; Mayer-Eichberger, V.; van de Pol, J.; and Saffidine, A. 2023. Implicit QBF Encodings for Positional Games. In Hartisch, M.; Hsueh, C.; and Schaeffer, J., eds., *Advances in Computer Games - 18th International Conference, ACG 2023, Revised Selected Papers*, volume 14528 of *Lecture Notes in Computer Science*, 133–145. Springer.

Urquhart, A. 1987. Hard examples for resolution. *J. ACM*, 34(1): 209–219.

Van Gelder, A. 2012. Contributions to the Theory of Practical Quantified Boolean Formula Solving. In *Proc. Principles and Practice of Constraint Programming (CP)*, 647–663.

Zhang, L.; and Malik, S. 2002. Conflict driven learning in a quantified Boolean Satisfiability solver. In *Proc. IEEE/ACM International Conference on Computer-aided Design (ICCAD)*, 442–449.