Near-Optimal Lower Bounds on Quantifier Depth and Weisfeiler–Leman Refinement Steps

Christoph Berkholz
Humboldt-Universität zu Berlin
berkholz@informatik.hu-berlin.de

Jakob Nordström
KTH Royal Institute of Technology
jakobn@kth.se

Abstract

We prove near-optimal trade-offs for quantifier depth versus number of variables in first-order logic by exhibiting pairs of n-element structures that can be distinguished by a k-variable first-order sentence but where every such sentence requires quantifier depth at least $n^{\Omega(k/\log k)}$. Our trade-offs also apply to first-order counting logic, and by the known connection to the k-dimensional Weisfeiler–Leman algorithm imply near-optimal lower bounds on the number of refinement iterations. A key component in our proof is the hardness condensation technique recently introduced by [Razborov '16] in the context of proof complexity. We apply this method to reduce the domain size of relational structures while maintaining the quantifier depth required to distinguish them.

Categories and Subject Descriptors F.4.1 [*Mathematical Logic*]: Computational Logic, Model theory; F.2.3 [*Tradeoffs between Complexity Measures*]

Keywords First-order logic, first-order counting logic, bounded variable fragment, quantifier depth, Weisfeiler–Leman, refinement iterations, lower bounds, trade-offs, hardness condensation, XORification

1. Introduction

The k-variable fragment of first-order logic L^k consists of those first-order sentences that use at most k different variables. A simple example is the L^2 sentence

$$\exists x \exists y (Exy \wedge \exists x (Eyx \wedge \exists y (Exy \wedge \exists x Eyx))) \qquad (1)$$

stating that there exists a directed path of length 4 in a digraph. Extending L^k with counting quantifiers $\exists^{\geq i} x$ yields C^k, which can be more economical in terms of variables. As an illustration, the L^8 sentence

$$\exists x \exists y_1 \cdots \exists y_7 \big(\textstyle\bigwedge_{i \neq j} y_i \neq y_j \wedge \bigwedge_i Exy_i \big) \qquad (2)$$

stating the existence of a vertex of degree at least 7 in a graph can be written more succinctly as the C^2 sentence

$$\exists x \exists^{\geq 7} y Exy \ . \qquad (3)$$

Bounded variable fragments of first order logic have found numerous applications in finite model theory and related areas (see [19] for a survey). Their importance stems from the fact that the model checking problem (given a finite relational structure $\mathcal{A}$ and a sentence φ, does $\mathcal{A}$ satisfy φ?) can be decided in polynomial time [25, 35]. Moreover, the equivalence problem (given two finite relational structures $\mathcal{A}$ and $\mathcal{B}$, do they satisfy the same sentences?) for L^k and C^k can be decided in time $n^{O(k)}$ [26], i.e., polynomial for constant k.

Quantifier Depth If $\mathcal{A}$ and $\mathcal{B}$ are not equivalent in L^k or C^k, then there exists a sentence φ that defines a distinguishing property, i.e., such that $\mathcal{A} \models \varphi$ and $\mathcal{B} \not\models \varphi$, which certifies that the structures are non-isomorphic. But how complex can such a sentence be? In particular, what is the minimal quantifier depth of an L^k or C^k sentence that distinguishes two n-element relational structures $\mathcal{A}$ and $\mathcal{B}$? The best upper bound for the quantifier depth of L^k and C^k is n^{k-1} [26], while to the best of our knowledge the strongest lower bounds have been only linear in n [13, 20, 16]. In this paper we present a near-optimal lower bound of $n^{\Omega(k/\log k)}$.

Theorem 1.1. *There are $\varepsilon > 0$, $K_0 \in \mathbb{N}$ such that for all k, n with $K_0 \leq k \leq n^{1/12}$ there is a pair of n-element $(k-1)$-ary relational structures $\mathcal{A}_n, \mathcal{B}_n$ that can be distinguished in k-variable first-order logic but satisfy the same L^k and C^k sentences up to quantifier depth $n^{\varepsilon k/\log k}$.*

Note that any two non-isomorphic n-element σ-structures $\mathcal{A}$ and $\mathcal{B}$ can always be distinguished by a simple n-variable first-order sentence of quantifier depth n, namely

$$\exists x_1 \cdots \exists x_n \Bigg(\bigwedge_{i \neq j} x_i \neq x_j \ \wedge \bigwedge_{\substack{R \in \sigma, \\ (v_{i_1}, \ldots, v_{i_r}) \in R^{\mathcal{A}}}} R x_{i_1}, \ldots, x_{i_r} \wedge \bigwedge_{\substack{R \in \sigma, \\ (v_{i_1}, \ldots, v_{i_r}) \notin R^{\mathcal{A}}}} \neg R x_{i_1}, \ldots, x_{i_r} \Bigg) \ . \qquad (4)$$

Since our $n^{\Omega(k/\log k)}$ lower bound for k-variable logics grows significantly faster than this trivial upper bound n on the quantifier depth as the number of variables increases, Theorem 1.1 also describes a trade-off in the super-critical regime above worst-case investigated by Razborov [32]: If one reduces one complexity measure (the number of variables), then the other complexity parameter (the quantifier depth) increases sharply even beyond its worst-case upper bound.

The equivalence problem for C^{k+1} is known to be closely related to the *k-dimensional Weisfeiler–Leman algorithm* (k-WL) for testing non-isomorphism of graphs and, more generally, relational structures. It was shown by Cai, Fürer, and Immerman [13] that two structures are distinguished by k-WL if and only if there exists a C^{k+1} sentence that differentiates between them. Moreover, the

LICS '16, July 05-08, 2016, New York, NY, USA
Copyright © 2016 ACM 978-1-4503-4391-6/16/07. . . $15.00
DOI: http://dx.doi.org/10.1145/2933575.2934560

quantifier depth of such a sentence also relates to the complexity of the WL algorithm in that the number of iterations k-WL needs to tell $\mathcal{A}$ and $\mathcal{B}$ apart coincides with the minimal quantifier depth of a distinguishing C^{k+1} sentence. Therefore, Theorem 1.1 also implies a near-optimal lower bound on the number of refinement steps required in the Weisfeiler–Leman algorithm. We discuss this next.

The Weisfeiler–Leman Algorithm The Weisfeiler–Leman algorithm, independently introduced by Babai in 1979 and by Immerman and Lander in [26] (cf. [13] and [2] for historic notes), is a hierarchy of methods for isomorphism testing that iteratively refine a partition (or colouring) of the vertex set, ending with a *stable colouring* that classifies *similar vertices*. Since no isomorphism can map non-similar vertices to each other, this reduces the search space. Moreover, if two structures end up with different stable colourings, then we can immediately deduce that the structures are non-isomorphic. The 1-dimensional Weisfeiler–Leman algorithm, better known as *colour refinement*, initially colours the vertices according to their degree (clearly, no isomorphism identifies vertices of different degree). The vertex colouring is then refined based on the colour classes of the neighbours. For example, two degree-5 vertices get different colours in the next step if they have a different number of degree-7 neighbours. This refinement step is repeated until the colouring stays stable (i.e., every pair of equally coloured vertices have the same number of neighbours in every other colour class). This algorithm is already quite strong and is extensively used in practical graph isomorphism algorithms.

In k-dimensional WL this idea is generalized to colourings of k-*tuples* of vertices. Initially the k-tuples are coloured by their isomorphism type, i.e., two tuples $\vec{v} = (v_1, \ldots, v_k)$ and $\vec{w} = (w_1, \ldots, w_k)$ get different colours if the mapping $v_i \mapsto w_i$ is not an isomorphism on the substructures induced on $\{v_1, \ldots, v_k\}$ and $\{w_1, \ldots, w_k\}$. In the refinement step, we consider for each k-tuple $\vec{v} = (v_1, \ldots, v_k)$ and every vertex v the colours of the tuples $\vec{v}_j := (v_1, \ldots, v_{j-1}, v, v_{j+1}, \ldots, v_k)$, where v is substituted at the jth position in the tuple $\vec{v}$. We refer to the tuple $(c(\vec{v}_1), \ldots, c(\vec{v}_k))$ of these k colours as the *colour type* $\mathtt{t}(\vec{v}, v)$ and let v be a $\mathtt{t}$-neighbour of $\vec{v}$ if $\mathtt{t} = \mathtt{t}(\vec{v}, v)$. Now two tuples $\vec{v}$ and $\vec{w}$ get different colours if they are already coloured differently, or if there exists a colour type $\mathtt{t}$ such that $\vec{v}$ and $\vec{w}$ have a different number of $\mathtt{t}$-neighbours. The refinement step is repeated until the colouring stays stable. Since in every round the number of colour classes grows, the process stops after at most n^k steps. The colour names can be chosen in such a way that the stable colouring is canonical, which means that two isomorphic structures end up with the same colouring, and such a canonical stable colouring can be computed in time $n^{O(k)}$.

This simple combinatorial algorithm is surprisingly powerful. Grohe [21] showed that for every non-trivial graph class that excludes some minor (such as planar graphs or graphs of bounded treewidth) there exists some k such that k-WL computes a different colouring for all non-isomorphic graphs, and hence solves graph isomorphism in polynomial time on that graph class. Weisfeiler–Leman has also been used as a subroutine in algorithms that solve graph isomorphism on all graphs. As one part of his very recent graph isomorphism algorithm, Babai [2] applies k-WL for polylogarithmic k to relational (k-ary) structures and makes use of the quasi-polynomial running time of this algorithm.

Given the importance of the Weisfeiler–Leman procedure, it is a natural question to ask whether the trivial n^k upper bound on the number of refinement steps is tight. By the correspondence between the number of refinement steps of k-WL and the quantifier depth of C^{k+1} [13], our main result implies a near-optimal lower bound even up to polynomial, but still sublinear, values of k (i.e., $k = n^\delta$ for small enough constant δ).

Theorem 1.2. *There are $\varepsilon > 0$, $K_0 \in \mathbb{N}$ such that for all k, n with $K_0 \leq k \leq n^{1/12}$ there is an n-element k-ary relational structure $\mathcal{A}_n$ for which the k-dimensional Weisfeiler–Leman algorithm needs $n^{\varepsilon k / \log k}$ refinement steps to compute the stable colouring.*

In addition to the near-optimal lower bounds for a specific dimension (or number of variables) k, we also obtain the following trade-off between the dimension and the number of refinement steps: If we fix two parameters ℓ_1 and ℓ_2 (possibly depending on n) satisfying $\ell_1 \leq \ell_2 \leq n^{1/6}/\ell_1$, then there are n-element structures such that k-WL needs $n^{\Omega(\ell_1/\log \ell_2)}$ refinement steps for all $\ell_1 \leq k \leq \ell_2$. A particularly interesting choice of parameters is $\ell_1 = \log^c n$ for some constant $c > 1$ and $\ell_2 = n^{1/7}$. This implies the following quasi-polynomial lower bound on the number of refinement steps for Weisfeiler–Leman from polylogarithmic dimension all the way up to dimension $n^{1/7}$.

Theorem 1.3. *For every $c > 1$ there is a sequence of n-element relational structures $\mathcal{A}_n$ for which the k-dimensional Weisfeiler–Leman algorithm needs $n^{\Omega(\log^{c-1} n)}$ refinement steps to compute the stable colouring for all k with $\log^c n \leq k \leq n^{1/7}$.*

Previous Lower Bounds In their seminal work [13], Cai, Fürer and Immerman showed that there exist non-isomorphic n-vertex graphs that cannot be distinguished by any first-order counting sentence with $o(n)$ variables. Since every pair of non-isomorphic n-element structures can be distinguished by a C^n (or even L^n) sentence (as shown in (4) above), this result also implies a linear lower bound on the quantifier depth of C^k if $k = \Omega(n)$. For all constant $k \geq 2$, a linear $\Omega(n)$ lower bound on the quantifier depth of C^k follows implicitly from an intricate construction of Grohe [20], which was used to show that the equivalence problems for L^k and C^k are complete for polynomial time. An explicit linear lower bound based on a simplified construction was subsequently presented by Fürer [16].

For the special case of C^2, Krebs and Verbitsky [29] recently obtained an improved $(1 - o(1))n$ lower bound on the quantifier depth, nearly matching the upper bound n. In contrast, Kiefer and Schweitzer [27] showed that if two n-vertex graphs can be distinguished by a C^3 sentence, then there is always a distinguishing sentence of quantifier depth $O(n^2/\log n)$. Hence, the trivial n^2 upper bound is not tight in this case.

As far as we are aware, the current paper presents the first lower bounds that are super-linear in the domain size n.

Discussion of Techniques The hard instances we construct are based on propositional XOR (exclusive or) formulas, which can alternatively be viewed as systems of linear equations over $\mathrm{GF}(2)$. There is a long history of using XOR formulas for proving lower bounds in different areas of theoretical computer science such as, e.g., finite model theory, proof complexity, and combinatorial optimization/hardness of approximation. Our main technical insight is to combine two methods that, to the best of our knowledge, have not been used together before, namely Ehrenfeucht-Fraïssé games on structures based on XOR formulas and hardness amplification by variable substitution as used in proof complexity.

More than three decades ago, Immerman [24] presented a way to encode an XOR formula into two graphs that are isomorphic if and only if the formula is satisfiable. This can then be used to show that the two graphs cannot be distinguished by a sentence with few variables or low quantifier depth using Ehrenfeucht-Fraïssé games. Arguably the most important application of this method is the result in [13] establishing that a linear number of variables is needed to distinguish two graphs in first-order counting logic. Graph constructions based on XOR formulas have also been used to prove lower bounds on the quantifier depth of C^k [24, 16].

We remark that for our result we have to use a slightly different encoding of formulas into relational structures rather than graphs.

In proof complexity, various flavours of XOR formulas (usually called *Tseitin formulas* when used to encode the *handshaking lemma* saying that the sum of all vertex degrees in an undirected graph has to be an even number) have been used to obtain lower bounds for proof systems such as resolution [34], polynomial calculus [12], and bounded-depth Frege [5]. Such formulas have also played an important role in many lower bounds for the Positivstellensatz/sums-of-squares proof system [18, 28, 33] corresponding to the Lasserre semidefinite programming hierarchy, which has been the focus of much recent interest in the context of combinatorial optimization.[1] Another use of XOR in proof complexity has been for hardness amplification, where one takes a (typically non-XOR) formula that is moderately hard with respect to some complexity measure, substitutes all variables by exclusive ors over pairwise distinct sets of variables, and then shows that the new *XORified* formula must be very hard with respect to some other (more important) complexity measure. This technique was perhaps first made explicit in [6] and has later appeared in, e.g., [11, 7, 8, 4, 15].

An even more crucial role in proof complexity is played by well-connected so-called *expander graphs*. For instance, given a formula in conjunctive normal form (CNF) one can look at its bipartite clause-variable incidence graph (CVIG), or some variant of the CVIG derived from the combinatorial structure of the formula, and prove that if this graph is an expander, then this implies that the formula must be hard for proof systems such as resolution [9] and polynomial calculus [1, 30].

In a striking recent paper [32], Razborov combines XORification and expansion in a simple (with hindsight) but amazingly powerful way. Namely, instead of replacing every variable by an XOR over new, fresh variables, he recycles variables from a much smaller pool, thus decreasing the total number of variables. This means that the hardness amplification proofs no longer work, since they crucially use that all new substitution variables are distinct. But here expansion come into play. If the pattern of variable substitutions is described by a strong enough bipartite expander, it turns out that locally there is enough "freshness" even among the recycled variables to make the hardness amplification go through over a fairly wide range of the parameter space. And since the formula has not only become harder but has also had the number of variables decreased, this can be viewed as a kind of *hardness compression* or *hardness condensation*.

What we do in this paper is to first revisit Immerman's old quantifier depth lower bound for first-order counting logic [24] and observe that the construction can be used to obtain an improved scalable lower bound for the k-variable fragment. We then translate Razborov's hardness condensation technique [32] into the language of finite variable logics and use it—perhaps somewhat amusingly applied to XORification of XOR formulas, which is usually not the case in proof complexity—to reduce the domain size of relational structures while maintaining the minimal quantifier depth required to distinguish them.

Outline of This Paper The rest of this paper is organized as follows. In Section 2 we describe how to translate XOR formulas to relational structures and play combinatorial games on these structures. This then allows us to state our main technical lemmas in Section 3 and show how these lemmas yield our results. Turning to the proofs of these technical lemmas, in Section 4 we present a version of Immerman's quantifier depth lower bound for XOR formulas, and in Section 5 we apply Razborov's hardness condensation technique to these formulas. Finally, in Section 6 we make some concluding remarks and discuss possible directions for future research. Due to space constraints, we omit some of the more standard technical proofs in this conference version, referring the reader to the upcoming full-length version for the missing details.

[1] No proof complexity is needed in this paper, and so readers unfamiliar with these proof systems need not worry—this is just an informal overview.

2. From XOR Formulas to Relational Structures

In this paper all structures are finite and defined over a relational signature σ. We use the letters X, E, and R for unary, binary, and r-ary relation symbols, respectively, and let $X^{\mathcal{A}}$, $E^{\mathcal{A}}$, and $R^{\mathcal{A}}$ be their interpretation in a structure $\mathcal{A}$. We write $V(\mathcal{A})$ to denote the domain of the structure $\mathcal{A}$. The *k-variable fragment of first-order logic* L^k consists of all first-order formulas that use at most k different variables (possibly re-quantifying them as in Equation (1)). We also consider *k-variable first-order counting logic* C^k, which is the extension of L^k by counting quantifiers $\exists^{\geq i} x\varphi(x)$, stating that there exist at least i elements $u \in V(\mathcal{A})$ such that $(\mathcal{A}, u) \models \varphi(x)$. For a survey of finite variable logics and their applications we refer the reader to, e.g., [19].

An *ℓ-XOR clause* is a tuple $(x_1, \ldots, x_\ell, a)$ consisting of ℓ Boolean variables and a Boolean value $a \in \{0, 1\}$. We refer to ℓ as the *width* of the clause. An assignment α *satisfies* $(x_1, \ldots, x_\ell, a)$ if $\alpha(x_1) + \cdots + \alpha(x_\ell) \equiv a \pmod 2$. An ℓ-XOR formula F is a conjunction of XOR clauses of width at most ℓ and is satisfied by an assignment α if α satisfies all clauses in F.

For every ℓ-XOR formula F on n variables we can define a pair of $2n$-element structures $\mathcal{A} = \mathcal{A}(F)$ and $\mathcal{B} = \mathcal{B}(F)$ that are isomorphic if and only if F is satisfiable. The domain of the structures contains two elements x_i^0 and x_i^1 for each Boolean variable x_i. There is one unary predicate X_i for every variable x_i identifying the corresponding two elements x_i^0 and x_i^1. Hence these unary relations partition the domain of the structures into two-element sets, i.e., $X_i^{\mathcal{A}} = X_i^{\mathcal{B}} = \{x_i^0, x_i^1\}$. To encode the XOR clauses, we introduce one m-ary relation R_m for every $1 \leq m \leq \ell$ and set

$$R_m^{\mathcal{A}} = \left\{ \left(x_{i_1}^{a_1}, \ldots, x_{i_m}^{a_m}\right) \,\middle|\, (x_{i_1}, \ldots, x_{i_m}, a) \in F, \textstyle\sum_i a_i \equiv 0 \right\} \quad \text{(5a)}$$

and

$$R_m^{\mathcal{B}} = \left\{ \left(x_{i_1}^{a_1}, \ldots, x_{i_m}^{a_m}\right) \,\middle|\, (x_{i_1}, \ldots, x_{i_m}, a) \in F, \textstyle\sum_i a_i \equiv a \right\} \quad \text{(5b)}$$

(where the sums are taken $\bmod\ 2$). Every bijection β between the domains of $\mathcal{A}(F)$ and $\mathcal{B}(F)$ that preserves the unary relations X_i can be translated to an assignment α for the XOR formula via the correspondence $\alpha(x_i) = 0 \Leftrightarrow \beta(x_i^0) = x_i^0 \Leftrightarrow \beta(x_i^1) = x_i^1$ and $\alpha(x_i) = 1 \Leftrightarrow \beta(x_i^0) = x_i^1 \Leftrightarrow \beta(x_i^1) = x_i^0$. It is not hard to show that such a bijection defines an isomorphism between $\mathcal{A}(F)$ and $\mathcal{B}(F)$ if and only if the corresponding assignment satisfies F.

This kind of encodings of XOR formulas into relational structures have been very useful for proving lower bounds for finite variable logics in the past. Our transformation of XOR clauses of width ℓ into ℓ-ary relational structures resembles the way Gurevich and Shelah [22] encode XOR formulas as hypergraphs. It is also closely related to the way Cai, Fürer, and Immerman [13] obtain two non-isomorphic graphs $\mathcal{G}$ and $\mathcal{H}$ from an unsatisfiable 3-XOR formula F in the sense that $\mathcal{G}$ and $\mathcal{H}$ can be seen to be the incidence graphs of our structures $\mathcal{A}(F)$ and $\mathcal{B}(F)$.

In order to prove our main result, we make use of the combinatorial characterization of quantifier depth of finite-variable logics in terms of pebble games for L^k and C^k, which are played on two given relational structures. Since in our case the structures are based on XOR formulas, for convenience we consider a simplified combinatorial game that is played directly on the formulas rather than on their structure encodings. We first describe this game and then show in Lemma 2.1 that this yields an equivalent characterization.

The *r-round k-pebble game* is played on an XOR formula F by two players, whom we will refer to as Player 1 and Player 2.

A position in the game is a partial assignment α of at most k variables of F and the game starts with the empty assignment. In each round, Player 1 can delete some variable assignments from the current position (he chooses some $\alpha' \subseteq \alpha$). If the current position assigns values to exactly k variables, then Player 1 has to delete at least one variable assignment. Afterwards, Player 1 chooses some currently unassigned variable x and asks for its value. Player 2 answers by either 0 or 1 (independently of any previous answers to the same question) and adds this assignment to the current position.

A winning position for Player 1 is an assignment falsifying some clause from F. Player 1 wins the r-round k-pebble game if he has a strategy to win every play of the k-pebble game within at most r rounds. Otherwise, we say that Player 2 wins (or survives) the r-round k-pebble game. Player 1 *wins the k-pebble game* if he wins the r-round k-pebble game within a finite number of rounds r. Note that if Player 1 wins the k-pebble game, then he can always win the k-pebble within $2^k n^{k+1}$ rounds, because there are at most $\sum_{i=0}^{k} 2^i \binom{n}{i} \leq 2^k n^{k+1}$ different positions with at most k pebbles on n-variable XOR formulas. We say that Player 1 *can reach a position β* from a position α within r rounds, if he has a strategy such that in every play of the r-round k-pebble game starting from position α he either wins or ends up with position β.

Let us now show that the game described above is equivalent to the pebble game for L^k and to the bijective pebble game for C^k played on the structures $\mathcal{A}(F)$ and $\mathcal{B}(F)$.

Lemma 2.1. *Let k, p, r be integers such that $r > 0$ and $k \geq p$ and let F be a p-XOR formula giving rise to structures $\mathcal{A} = \mathcal{A}(F)$ and $\mathcal{B} = \mathcal{B}(F)$ as described in the paragraph preceding* (5a)–(5b). *Then the following statements are equivalent:*

(a) Player 1 wins the r-round k-pebble game on F.
(b) There is a k-variable first-order sentence $\varphi \in \mathsf{L}^k$ of quantifier depth r such that $\mathcal{A}(F) \models \varphi$ and $\mathcal{B}(F) \not\models \varphi$.
(c) There is a k-variable sentence in first-order counting logic $\varphi \in \mathsf{C}^k$ of quantifier depth r such that $\mathcal{A}(F) \models \varphi$ and $\mathcal{B}(F) \not\models \varphi$.
(d) The $(k-1)$-dimensional Weisfeiler–Leman procedure can distinguish between $\mathcal{A}(F)$ and $\mathcal{B}(F)$ within r refinement steps.

Proof sketch. Let us start by briefly recalling known characterizations in terms of Ehrenfeucht-Fraïssé games of L^k [3, 25] and C^k [13, 23]. In both cases the game is played by two players, referred to as Spoiler and Duplicator, on the two structures $\mathcal{A}$ and $\mathcal{B}$. Positions in the games are partial mappings $p = \{(u_1, v_1), \ldots, (u_i, v_i)\}$ from $V(\mathcal{A})$ to $V(\mathcal{B})$ of size at most k. The games start from the empty position and proceed in rounds. At the beginning of each round in both games, Spoiler chooses $p' \subseteq p$ with $|p'| < k$.

- In the L^k-game, Spoiler then selects either some $u \in V(\mathcal{A})$ or some $v \in V(\mathcal{B})$ and Duplicator responds by choosing an element $v \in V(\mathcal{B})$ or $u \in V(\mathcal{A})$ in the other structure.
- In the C^k-game, Duplicator first selects a global bijection $f : V(\mathcal{A}) \to V(\mathcal{B})$ and Spoiler chooses some pair $(u, v) \in f$. (If $|V(\mathcal{A})| \neq |V(\mathcal{B})|$, Spoiler wins the C^k-game immediately.)

The new position is $p' \cup \{(u, v)\}$. Spoiler wins the r-round L^k / C^k game if he has a strategy to reach within r rounds a position that does not define an isomorphism on the induced substructures. Both games characterize equivalence in the corresponding logics: Spoiler wins the r-round L^k / C^k game if and only if there is a depth-r sentence $\varphi \in \mathsf{L}^k$ / C^k such that $\mathcal{A} \models \varphi$ and $\mathcal{B} \not\models \varphi$.

When these games are played on the two structures $\mathcal{A}(F)$ and $\mathcal{B}(F)$ obtained from an XOR formula F, it is not hard to verify that both games are equivalent to the k-pebble game on F. To see this, we identify Spoiler with Player 1, Duplicator with Player 2, and partial mappings $p = \{(x_i^{a_i}, x_i^{b_i}) \mid i \leq \ell\}$ with partial assignments $\alpha = \{x_i \mapsto a_i \oplus b_i \mid i \leq \ell\}$. Because of the X_i-relations, we can assume that partial assignments of any other form will not occur as they are losing positions for Duplicator.

If Spoiler asks for some x_i^0 or x_i^1 in the L^k-game, which corresponds to a choice by Player 1 of $x_i \in \mathit{Vars}(F)$, the only meaningful action for Duplicator is to choose either x_i^0 or x_i^1 in the other structure, corresponding to an assignment to x_i by Player 2. With any other choice Duplicator would lose immediately because of the unary relations X_i. Thus, there is a natural correspondence between strategies in the L^k-game and the k-pebble game.

The players in the k-pebble game can be assumed to have perfect knowledge of the strategy of the other player. This means that at any given position in the game, without loss of generality we can think of Player 1 as being given a complete truth value assignment to the remaining variables, out of which he can pick one variable assignment. By the correspondence discussed above we see that this can be translated to a bijection f chosen by Duplicator in the C^k-game (which has to preserve the X_i relations). Therefore, Spoiler picking some pair of the form (x_i^a, x_i^b) from f can be viewed as Player 1 asking about the assignment to x_i and getting a response from Player 2 in the game on F (again using the above-mentioned correspondence between partial mappings p and partial assignments α). Finally, we observe that by design a partial mapping that preserves the X_i-relations defines a local isomorphism if and only if the corresponding α does not falsify any XOR clause.

Formalizing the proof sketch above, it is not hard to show that statements (a)–(c) are all equivalent. The equivalence between (c) and (d) was proven in [13]. The lemma follows. ☐

3. Proofs of Main Theorems

To prove our lower bounds of the quantifier depth of finite variable logics in Theorem 1.1 and the number of refinement steps of the Weisfeiler–Leman algorithm in Theorems 1.2 and 1.3, we utilize the characterization in Lemma 2.1 and show that there are n-variable XOR formulas on which Player 1 is able to win the k-pebble game but cannot do so in significantly less than $n^{k/\log k}$ rounds. The next lemma states this formally and also provides a trade-off as the number of pebbles increases.

Lemma 3.1 (Main technical lemma). *There is an absolute constant $K_0 \in \mathbb{N}^+$ such that for k_{lo}, k_{hi}, and n satisfying $K_0 \leq k_{\mathrm{lo}} \leq k_{\mathrm{hi}} \leq n^{1/6}/k_{\mathrm{lo}}$ there is an XOR formula F with n variables such that Player 1 wins the k_{lo}-pebble game on F, but does not win the k_{hi}-pebble game within $n^{k_{\mathrm{lo}}/(10 \log k_{\mathrm{hi}})-1/5}$ rounds.*

Let us see how this lemma yields the theorems in Section 1.

Proof of Theorem 1.1. This theorem can be seen to follow immediately from Lemmas 2.1 and 3.1, but let us write out the details for clarity. By setting $k_{\mathrm{lo}} = k_{\mathrm{hi}} = k$ in Lemma 3.1, we can find XOR formulas with n variables such that Player 1 wins the k-pebble game on F_n but needs more than $n^{\varepsilon k/\log k}$ rounds in order to do so (provided we choose $\varepsilon < 1/10$ and K_0 large enough). We can then plug these XOR formulas into Lemma 2.1 to obtain n-element structures $\mathcal{A}_n = \mathcal{A}(F_n)$ and $\mathcal{B}_n = \mathcal{B}(F_n)$ that can be distinguished in the k-variable fragments of first-order logic L^k and first-order counting logic C^k, but where this requires sentences of quantifier depth at least $n^{\varepsilon k/\log k}$. ☐

Proof of Theorem 1.2. If we let F_n be the XOR formula from Lemma 3.1 for $k_{\mathrm{lo}} = k_{\mathrm{hi}} = k$, then by Lemma 2.1 it holds that the structures $\mathcal{A}(F_n)$ and $\mathcal{B}(F_n)$ will be distinguished by the k-dimensional Weisfeiler–Leman algorithm, but only after $n^{\varepsilon k/\log k}$

refinement steps. Hence, computing the stable colouring of either of these structures requires at least $n^{\varepsilon k/\log k}$ refinement steps (since they would be distinguished earlier if at least one of the computations terminated earlier). □

Proof of Theorem 1.3. This is similar to the proof of Theorem 1.2, but setting $k_{\mathrm{lo}} = \lfloor \log n^c \rfloor$ and $k_{\mathrm{hi}} = \lceil n^{1/7} \rceil$ in Lemma 3.1. □

The proof of Lemma 3.1 splits into two steps. We first establish a rather weak lower bound on the number of rounds in the pebble game played on suitably chosen m-variable XOR formulas for $m \gg n$. We then transform this into a much stronger lower bound for formulas over n variables using hardness condensation. To help the reader keep track of which results are proven in which setting, in what follows we will write ℓ_{lo} and ℓ_{hi} to denote parameters depending on m and k_{lo} and k_{hi} to denote parameters depending on n.

To implement the first step in our proof plan, we use tools developed by Immerman [24] to establish a lower bound as stated in the next lemma. It gives non-trivial lower bounds for the ℓ_{hi}-pebble game on m-variable formulas for ℓ_{hi} up to $2^{\sqrt{\log m}}$.

Lemma 3.2. *For all $\ell_{\mathrm{hi}}, m \geq 3$ there is an m-variable 3-XOR formula $F_m^{\ell_{\mathrm{hi}}}$ on which Player 1*

(a) wins the 3-pebble game, but
(b) does not win the $\left(\frac{1}{\lceil \log \ell_{\mathrm{hi}} \rceil} m^{1/(1+\lceil \log \ell_{\mathrm{hi}} \rceil)} - 2\right)$-round ℓ_{hi}-pebble game.

We defer the proof of Lemma 3.2 to Section 4, but at this point an expert reader might wonder why we would need to prove this lower bound at all, since a much stronger $\Omega(m)$ bound on the number of rounds in the pebble game on 4-XOR formulas was already obtained by Fürer [16]. The reason is that in Fürer's construction Player 1 cannot win the game with $o(\ell_{\mathrm{hi}})$ pebbles. However, it is crucial for the second step of our proof, where we boost the lower bound but also significantly increase the number of pebbles that are needed to win the game, that Player 1 is able to win the original game with very few pebbles.

The second step in the proof of our main technical lemma is carried out by using the techniques developed by Razborov [32] and applying them to the XOR formulas in Lemma 3.2. Roughly speaking, if we set $k_{\mathrm{lo}} = k_{\mathrm{hi}} = k$ for simplicity, then the number of variables decreases from m to $n \approx m^{1/k}$, whereas the $m^{1/\log k}$ round lower bound for the k-pebble game stays essentially the same and hence becomes $n^{k/\log k}$ in terms of the new number of variables n. The properties of hardness condensation are summarized in the next lemma, which we prove in Section 5. To demonstrate the flexibility of this tool we state the lemma in its most general form—readers who want to see an example of how to apply it to the XOR formulas in Lemma 3.2 can mentally fix $p = 3$, $\ell_{\mathrm{lo}} = 3$, $\ell_{\mathrm{hi}} = k_{\mathrm{hi}}$, $r \approx m^{1/\log k_{\mathrm{hi}}}$, and $\Delta \approx k_{\mathrm{hi}}/3$ when reading the statement of the lemma below.

Lemma 3.3 (Hardness condensation lemma). *There is an absolute constant Δ_0 such that the following holds. Let F be an m-variable p-XOR formula and suppose that we can choose parameters $\ell_{\mathrm{lo}} > 0$, $\ell_{\mathrm{hi}} \geq \Delta_0 \ell_{\mathrm{lo}}$ and $r > 0$ such that Player 1*

(a) has a winning strategy for the ℓ_{lo}-pebble game on F, but
(b) does not win the ℓ_{hi}-pebble game on F within r rounds.

Then for any Δ satisfying $\Delta_0 \leq \Delta \leq \ell_{\mathrm{hi}}/\ell_{\mathrm{lo}}$ and $(2\ell_{\mathrm{hi}}\Delta)^{2\Delta} \leq m$ there is an (Δp)-XOR formula H with $\lceil m^{3/\Delta} \rceil$ variables such that Player 1

(a') has a winning strategy for the $(\Delta \ell_{\mathrm{lo}})$-pebble game on H, but
(b') does not win the ℓ_{hi}-pebble game on H within $r/(2\ell_{\mathrm{hi}})$ rounds.

Taking Lemmas 3.2 and 3.3 on faith for now, we are ready to prove our main technical lemma yielding an $n^{\Omega(k/\log k)}$ lower bound on the number of rounds in the k-pebble game.

Proof of Lemma 3.1. Let Δ_0 be the constant from Lemma 3.3. We let $K_0 \geq 3\Delta_0 + 9$ be an absolute constant to be determined later. We are given k_{hi}, k_{lo}, and n satisfying the conditions from Lemma 3.1. In order to apply Lemma 3.3 to the XOR formulas provided by Lemma 3.2 we fix the parameters $p := 3$, $\ell_{\mathrm{lo}} := 3$, $\ell_{\mathrm{hi}} := k_{\mathrm{hi}}$, $\Delta := 3\lfloor k_{\mathrm{lo}}/9 \rfloor$, and $m := n^{\lfloor k_{\mathrm{lo}}/9 \rfloor}$.

By assumption we have $\ell_{\mathrm{lo}} \geq 3$ and therefore we can appeal to Lemma 3.2 to obtain an m-variable 3-XOR formula on which Player 1 wins the 3-pebble game but cannot win the ℓ_{hi}-pebble game within $r := \frac{1}{\lceil \log \ell_{\mathrm{hi}} \rceil} m^{1/(1+\lceil \log \ell_{\mathrm{hi}} \rceil)} - 2$ rounds. Now we apply hardness condensation to this formula and have to check that the chosen parameters satisfy the conditions. By definition we have $\ell_{\mathrm{lo}} > 0$ and furthermore $\ell_{\mathrm{hi}} = k_{\mathrm{hi}} \geq K_0 \geq 3\Delta_0 = \Delta_0 \ell_{\mathrm{lo}}$. Statements (a) and (b) are satisfied by Lemma 3.2. To verify the bounds on Δ note that $\Delta_0 \leq 3\lfloor K_0/9 \rfloor \leq 3\lfloor k_{\mathrm{lo}}/9 \rfloor = \Delta \leq k_{\mathrm{lo}}/3 \leq k_{\mathrm{hi}}/3 = \ell_{\mathrm{hi}}/\ell_{\mathrm{lo}}$. We proceed with checking the condition $(2\ell_{\mathrm{hi}}\Delta)^{2\Delta} \leq m$. Because $\Delta \leq k_{\mathrm{lo}}/3$ and $\ell_{\mathrm{hi}} = k_{\mathrm{hi}} \leq n^{1/6}/k_{\mathrm{lo}}$ we get $(2\ell_{\mathrm{hi}}\Delta)^{2\Delta} \leq (\frac{2}{3} n^{1/6})^{2\Delta} \leq n^{\Delta/3} = m$. Finally, we verify that $n = n^{\lfloor k_{\mathrm{lo}}/9 \rfloor 3/\Delta} = m^{3/\Delta}$. Now Lemma 3.3 provides us with an n-variable k_{lo}-XOR formula on which (a') Player 1 has a winning strategy for the (3Δ)-pebble game and hence also for the game with $k_{\mathrm{lo}} \geq 9\lfloor k_{\mathrm{lo}}/9 \rfloor = 3\Delta$ pebbles. Furthermore, by (b') Player 1 needs $r/(2k_{\mathrm{hi}})$ rounds to win the k_{hi}-pebble game. To complete the proof we note that (since $k_{\mathrm{hi}} \leq n^{1/6}$)

$$r/(2k_{\mathrm{hi}}) = \frac{1}{2k_{\mathrm{hi}} \lceil \log k_{\mathrm{hi}} \rceil} n^{\lfloor k_{\mathrm{lo}}/9 \rfloor/(1+\lceil \log k_{\mathrm{hi}} \rceil)} - 2 \tag{6}$$

$$\geq n^{k_{\mathrm{lo}}/(10 \log k_{\mathrm{hi}}) - 1/5} \tag{7}$$

where the inequality holds for large enough $k_{\mathrm{lo}}, k_{\mathrm{hi}}, n \geq K_0$ and we choose the global constant K_0 such that the parameters are large enough for this inequality to hold. □

4. XOR Formulas over Pyramids

We now proceed to establish the k-pebble game lower bound stated in Lemma 3.2. Our XOR formulas will be constructed over directed acyclic graphs (DAGs) as described in the following definition.

Definition 4.1. Let $\mathcal{G}$ be a DAG with sources S and a unique sink z. The XOR formula $xor(\mathcal{G})$ contains one variable v for every vertex $v \in V(\mathcal{G})$ and consists of the following clauses:

(a) $(s, 0)$ for every source $s \in S$,
(b) $(v, w_1, \ldots, w_\ell, 0)$ for all non-sources $v \in V(\mathcal{G}) \setminus S$ with in-neighbours $N^-(v) = \{w_1, \ldots, w_\ell\}$,
(c) $(z, 1)$ for the unique sink z.

Note that the formula $xor(\mathcal{G})$ is always unsatisfiable, since all sources are forced to 0 by (a), which forces all other vertices to 0 in topological order by (b), contradicting (c) for the sink. Incidentally, these formulas are somewhat similar to the *pebbling formulas* defined in [9], which have been very useful in proof complexity (see the survey [31] for more details). The difference is that pebbling formulas state that a vertex v is true if and only if all of its in-neighbours are true, whereas $xor(\mathcal{G})$ states that v is true if and only if the parity of the number of true in-neighbours is odd.

It is clear that one winning strategy for Player 1 is to ask first about the sink z, for which Player 2 has to answer 1 (or lose immediately) and then about all the in-neighbours of the sink until the answer for one vertex v is 1 (if there is no such vertex, Player 2 again loses immediately). At this point Player 1 can forget all other vertices and then ask about the in-neighbours of v until a 1-labelled vertex w is found, and then continue in this way to trace a path of

1-labelled vertices backwards through the DAG until some source s is reached, which contradicts the requirement that s should be labelled 0. Formalizing this as an induction proof on the depth of $\mathcal{G}$ shows that if the in-degree is bounded, then Player 1 can win the pebble game on $xor(\mathcal{G})$ with few pebbles as stated next.

Lemma 4.2. *Let $\mathcal{G}$ be a DAG with a unique sink and maximal in-degree d. Then Player 1 wins the $(d+1)$-pebble game on $xor(\mathcal{G})$.*

As a warm-up for the proof of Lemma 3.2, let us describe a very weak lower bound from [24] for the complete binary tree of height h (with edges directed from the leaves to the root), which we will denote $\mathcal{T}_h$. By the lemma above, Player 1 wins the 3-pebble game on $xor(\mathcal{T}_h)$ in $O(h)$ steps by propagating 1 from the root down to some leaf. On the other hand, Player 2 has the freedom to decide on which path she answers 1. Hence, she can safely respond 0 for a vertex v as long as there is some leaf with a pebble-free path leading to the lowest pebble labelled 1 without passing v. In particular, if Player 2 is asked about vertices at least ℓ layers below the lowest pebbled vertex for which the answer 1 was given, then she can answer 0 for $2^{\ell}-1$ queries. It follows that the height h provides a lower bound on the number of rounds Player 1 needs to win the game, even if he has an infinite amount of pebbles. We remark that this proof in terms of pebble-free paths is somewhat reminiscent of an argument by Cook [14] for the so-called black pebble game corresponding to the pebbling formulas in [9] briefly discussed above.

The downside of this lower bound is that the height is only logarithmic in the number of vertices and thus too weak for us as we are shooting for a lower bound of the order of $n^{1/\log k}$. To get a better bound for the black pebble game Cook instead considered so-called pyramid graphs as in Figure 1. These will not be sufficient to obtain strong enough lower bounds for our pebble game, however. Instead, following Immerman we consider a kind of high-dimensional generalization of these graphs, for which the lower bound on the number of rounds in the k-pebble game is still linear in the height h while the number of vertices is roughly $h^{\log k}$.

Definition 4.3 ([24]). For $d \geq 1$ we define the $(d+1)$-*dimensional pyramid of height h*, denoted by $\mathcal{P}_h^d$, to be the following layered DAG. We let L, $0 \leq L \leq h$ be the *layer number* and set $q_d(L) := \lfloor L/d \rfloor$ and $r_d(L) := L \pmod{d}$. Hence, for any L we have $L = q_d(L) \cdot d + r_d(L)$. For integers $x_i \geq 0$ the vertex set is

$$V\big(\mathcal{P}_h^d\big) = \big\{(x_0, \ldots, x_{d-1}, L) \mid L \leq h; \\ x_i \leq q_d(L)+1 \text{ if } i<r_d(L); x_i \leq q_d(L) \text{ if } i \geq r_d(L)\big\} \ , \tag{8a}$$

where we say that L is the *layer* of the vertex $(x_0, \ldots, x_{d-1}, L)$. The edge set $E\big(\mathcal{P}_h^d\big)$ consists of the pair of edges

$$\big((x_0, \ldots, x_{d-1}, L+1), (x_0, \ldots, x_{d-1}, L)\big) \ , \\ \big((x_0, \ldots, x_{r_d(L)}+1, \ldots, x_{d-1}, L+1), (x_0, \ldots, x_{d-1}, L)\big) \tag{8b}$$

for all vertices $(x_0, \ldots, x_{d-1}, L) \in V(\mathcal{P}_h^d)$ and layers $L < h$, so that every vertex in layer L has exactly two in-neighbours from layer $L+1$.

We refer the reader to Figures 1 and 2 for illustrations of 2-dimensional and 3-dimensional pyramids (where all the edges in the figures are assumed to be directed upwards). The vertex $(0, \ldots, 0)$ at the top of the pyramid is the unique sink and all vertices at the bottom layer h are sources.

As high-dimensional pyramids have in-degree 2, Lemma 4.2 implies that Player 1 wins the 3-pebble game on $\mathcal{P}_h^d$. Recall that, as discussed in the proof sketch of the lemma, Player 1 starts his winning strategy in the 3-pebble game by pebbling the sink of the pyramid and its two in-neighbours. One of them has to be

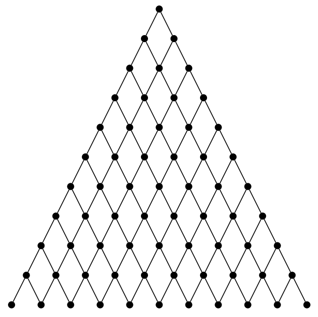
Figure 1. 2D pyramid

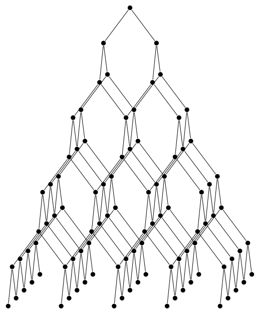
Figure 2. 3D pyramid

labelled 1. Then he picks up the two other pebbles and pebbles the two in-neighbours of the vertex marked with 1 and so on. Continuing this strategy, he is able to "move" the 1 all the way to the bottom, reaching a contradiction, in a number of rounds that is linear in the height of the pyramid. This strategy turns out to be nearly optimal in the sense that in order to move a 1 from the top to the bottom in $\mathcal{P}_h^d$, it makes no sense for Player 1 at any point in the game to pebble a vertex that is d or more levels away from the lowest level containing a pebble.

The next lemma states a key property of pyramids in this regard. In order to state it, we need to make a definition.

Definition 4.4. We refer to a partial assignment $\mathcal{M}$ of Boolean values to the vertices of a DAG $\mathcal{G}$ as a *labelling* or *marking* of $\mathcal{G}$. We say that $\mathcal{M}$ is *consistent* if no clause of type (b) or (c) in the XOR formula $xor(\mathcal{G})$ in Definition 4.1 is falsified by $\mathcal{M}$.

That is, a consistent labelling does not violate any constraint on any non-source vertex, but source constraints (a) may be falsified. Such labellings are easy to find for high-dimensional pyramids.

Lemma 4.5 ([24]). *Let $\mathcal{M}$ be a consistent labelling of all vertices in a pyramid $\mathcal{P}_h^d$ from layer 0 to layer L. Then for every set S of 2^d-1 vertices on or below layer $L+d$ there is a consistent labelling of the entire pyramid that extends $\mathcal{M}$ and labels all vertices in S with 0.*

To get some intuition why Lemma 4.5 holds, note that the d-dimensional pyramids are constructed in such a way that they locally look like binary trees. In particular, every vertex $v \in V\big(\mathcal{P}_h^d\big)$ together with all its predecessors at distance at most d form a complete binary tree. By the same argument as for the binary trees above, it follows that if v is labelled with 1, Player 2 can safely answer 0 up to 2^d-1 times when asked about vertices d layers below v. However, the full proof of Lemma 4.5 is more challenging and requires some quite subtle reasoning. We refer the reader to [24] (or the upcoming full-length version of this paper) for the details.

In [24] $\log n$-dimensional pyramids (where n is the number of vertices) are used to prove a $\Omega\big(2^{\sqrt{\log n}}\big)$ lower bound on the quantifier depth of full first-order counting logic. The next lemma shows that if we instead choose the dimension to be logarithmic in the number of variables (i.e., pebbles) in the game, we get an improved quantifier depth lower bound for the k-variable fragment.

Lemma 4.6. *For every $d \geq 2$ and height h, Player 1 does not win the 2^d-pebble game on $xor\big(\mathcal{P}_h^d\big)$ within $\lfloor h/d \rfloor - 1$ rounds.*

Proof. We show that Player 2 has a counter-strategy to answer consistently for at least $\lfloor h/d \rfloor - 1$ rounds. Starting at the top layer $L_1 = 0$, she maintains the invariant that at the start of round r she has a consistent labelling of all vertices from layer 0 to layer L_r with the property that there is no pebble on layers L_r+1 to L_r+d.

Whenever Player 1 places a pebble on a layer above L_r, Player 2 responds according to the labelling and whenever Player 1 puts a pebble on or below layer $L_r + d$, she answers 0, and in both cases sets $L_{r+1} = L_r$. If Player 1 places a pebble between layer $L_r + 1$ and $L_r + d$, Player 2 first extends her labelling to the first layer $L_{r+1} > L_r$ such that there is no pebble on layers $L_{r+1} + 1$ to $L_{r+1} + d$ and then answers according to the new labelling. The existence of such an extension is guaranteed by Lemma 4.5. Note that when Player 2 skips downward from layer L_r to layer L_{r+1} she might jump over a lot of layers in one go, but if so there is at least one pebble for every dth layer forcing such a big jump.

We see that following this strategy Player 2 survives for at least $\lfloor h/d \rfloor - 1$ rounds, and this establishes the lemma. $\square$

Putting the pieces together, we can now present the lower bound for the k-pebble game in Lemma 3.2.

Proof of Lemma 3.2. Recall that we want to prove that for all $\ell_{\mathrm{hi}} \geq 3$ and $m \geq 3$ there is an m-variable 3-XOR formula F on which Player 1 wins the 3-pebble game but cannot win the ℓ_{hi}-pebble game within $\frac{1}{\lceil \log \ell_{\mathrm{hi}} \rceil} m^{1/(1+\lceil \log \ell_{\mathrm{hi}} \rceil)} - 2$ rounds.

We choose the formula to be $F = xor\left(\mathcal{P}_h^d\right)$ for parameters $d = \lceil \log k \rceil$ and $h = \left\lceil m^{1/(d+1)} \right\rceil$. Since the graph $\mathcal{P}_h^d$ has indegree 2, Lemma 4.2 says that Player 1 wins the 3-pebble game as claimed in Lemma 3.2(a). The lower bound for the ℓ_{hi}-pebble game in Lemma 3.2(b) follows from Lemma 4.6 and the fact that $\mathcal{P}_h^d$ contains less than h^{d+1} vertices. $\square$

5. Hardness Condensation

In this section we establish Lemma 3.3, which shows how to convert an XOR formula into a harder formula over fewer variables. As discussed in the introduction, this part of our construction relies heavily on Razborov's recent paper [32]. We follow his line of reasoning closely below, but translate it from proof complexity to a pebble game argument for bounded variable logics.

A key technical concept in the proof is graph expansion. Let us define the particular type of expander graphs we need and then discuss some crucial properties of these graphs. We use standard graph notation, letting $\mathcal{G} = (U \,\dot\cup\, V, E)$ denote a bipartite graph with left vertex set U and right vertex set V. We let $N^{\mathcal{G}}\left(U'\right) = \left\{v \,\middle|\, \{u,v\} \in E(\mathcal{G}), u \in U'\right\}$ denote the right neighbours of a left vertex subset $U' \subseteq U$ (and vice versa for right vertex subsets).

Definition 5.1 (Boundary expander). A bipartite graph $\mathcal{G} = (U \,\dot\cup\, V, E)$ is an $m \times n$ (s,c)*-boundary expander graph* if $|U| = m$, $|V| = n$, and for every set $U' \subseteq U$, $|U'| \leq s$, it holds that $\left|\partial^{\mathcal{G}}(U')\right| \geq c|U'|$, where the *boundary* $\partial^{\mathcal{G}}(U')$ is the set of all $v \in N^{\mathcal{G}}(U')$ having a unique neighbour in U', meaning that $\left|N^{\mathcal{G}}(v) \cap U'\right| = 1$. An (s,Δ,c)-boundary expander is an (s,c)-boundary expander where additionally $\left|N^{\mathcal{G}}(u)\right| \leq \Delta$ for all $u \in U$, i.e., the graph has left degree bounded by Δ.

In what follows, we will omit $\mathcal{G}$ from the notation when the graph is clear from context.

In any (s,c)-boundary expander with $c > 0$ it holds that any left vertex subset $U' \subseteq U$ of size $|U'| \leq s$ has a partial matching into V where the vertices in U' can be ordered in such a way that every vertex $u_i \in U'$ is matched to a vertex outside of the neighbourhood of the preceding vertices $u_1, \ldots, u_{i-1}$. The proof of this fact is sometimes referred to as a *peeling argument*.

Lemma 5.2 (Peeling lemma). *Let $\mathcal{G} = (U \,\dot\cup\, V, E)$ be an (s,c)-boundary expander with $s \geq 1$ and $c > 0$. Then for every set $U' \subseteq U$, $|U'| = t \leq s$ there is an ordering $u_1, \ldots, u_t$ of its vertices and a sequence of vertices $v_1, \ldots, v_t \in V$ such that $v_i \in N(u_i) \setminus N(\{u_1, \ldots, u_{i-1}\})$.*

Proof sketch. Fix any $v_t \in \partial(U')$ and let $u_t \in U'$ be the unique vertex such that $\left|N(v_t) \cap U'\right| = \{u_t\}$. Then it holds that $v_t \in N(u_t) \setminus N\left(U' \setminus \{u_t\}\right)$. By induction we can now find sequences $u_1, \ldots, u_{t-1}$ and $v_1, \ldots, v_{t-1}$ for $U' \setminus \{u_t\}$ such that $v_i \in N(u_i) \setminus N(\{u_1, \ldots, u_{i-1}\})$, to which we can append u_t and v_t at the end. The lemma follows. $\square$

For a right vertex subset $V' \subseteq V$ in $\mathcal{G} = (U \,\dot\cup\, V, E)$ we define the *kernel* $\mathrm{Ker}\left(V'\right) \subseteq U$ to be the set of all left vertices whose entire neighbourhood is contained in V', i.e.,

$$\mathrm{Ker}\left(V'\right) = \left\{u \in U \,\middle|\, N(u) \subseteq V'\right\} \ . \tag{9}$$

We let $\mathcal{G} \setminus V'$ denote the subgraph of $\mathcal{G}$ induced on $\left(U \setminus \mathrm{Ker}(V')\right) \dot\cup \left(V \setminus V'\right)$. Thus, we obtain $\mathcal{G} \setminus V'$ from $\mathcal{G}$ by first deleting V' and afterwards all isolated vertices from U.

The next lemma states that for any small enough right vertex set V' in an expander $\mathcal{G}$ we can find a *closure* $\gamma\left(V'\right) \supseteq V'$ with a small kernel such that $\mathcal{G} \setminus \gamma(V')$ has good expansion. The proof of this lemma (albeit with slightly different parameters) can be found in [32] and is also provided in the full-length version of this paper.

Lemma 5.3 ([32]). *Let $\mathcal{G}$ be an $(s,2)$-boundary expander. Then for every $V' \subseteq V$ with $|V'| \leq s/2$ there exists a subset $\gamma\left(V'\right) \subseteq V$ with $\gamma(V') \supseteq V'$ such that $\left|\mathrm{Ker}\left(\gamma\left(V'\right)\right)\right| \leq \left|V'\right|$ and the induced subgraph $\mathcal{G} \setminus \gamma(V')$ is an $(s/2,1)$-boundary expander.*

In order for Lemmas 5.2 and 5.3 to be useful, we need to know that there exist good expanders. This can be established by a standard probabilistic argument. A proof of the next lemma is given in [32] and can also be found in the full version of this paper.

Lemma 5.4. *There is a constant Δ_0 such that for all Δ, s, m satisfying $\Delta \geq \Delta_0$ and $(s\Delta)^{2\Delta} \leq m$ there exist $m \times \left\lceil m^{3/\Delta} \right\rceil$ $(s,\Delta,2)$-boundary expanders.*

We will use such expanders $\mathcal{G} = (U \,\dot\cup\, V, E)$ when we do XOR substitution in our formulas as described in the next definition. In words, variables in the XOR formula are identified with left vertices U in $\mathcal{G}$, the pool of new variables is the right vertex set V, and every variable $u \in U$ in an XOR clause is replaced by an exclusive or $\bigoplus_{v \in N(u)} v$ over its neighbours $v \in N(u)$.

Definition 5.5 (XOR substitution with recycling). Let F be an XOR formula with $Vars(F) = U$ and let $\mathcal{G} = (U \,\dot\cup\, V, E)$ be a bipartite graph. For every clause $C = (u_1, \ldots, u_t, a)$ in F we let $C[\mathcal{G}]$ be the clause $(v_1^1, \ldots, v_1^{z_1}, \ldots, v_t^1, \ldots, v_t^{z_t}, a)$, where $N(u_i) = \{v_i^1, \ldots, v_i^{z_i}\}$ for all $1 \leq i \leq t$. Taking unions, we let $F[\mathcal{G}]$ be the XOR formula $F[\mathcal{G}] = \{C[\mathcal{G}] \mid C \in F\}$.

When using an $m \times m^{3/\Delta}$ $(s,\Delta,2)$-boundary expander as in Lemma 5.4 for substitution in an m-variable XOR formula F as described in Definition 5.5, we obtain a new XOR formula $F[\mathcal{G}]$ where the number of variables have decreased significantly to $m^{3/\Delta}$. The next lemma, which is at the heart of our logic-flavoured version of hardness condensation, states that a round lower bound for the k-pebble game on F implies a round lower bound for the k-pebble game on $F[\mathcal{G}]$.

Lemma 5.6. *Let k be a positive integer and let $\mathcal{G}$ an $m \times n$ $(2k,2)$-boundary expander. Then if F is an XOR formula over m variables such that Player 2 wins the r-round k-pebble game on F, she also wins the $r/(2k)$-round k-pebble game on $F[\mathcal{G}]$.*

Before embarking on a formal proof of Lemma 5.6, which is rather technical and will take the rest of this section, let us discuss the intuition behind it. The main idea to obtain a good strategy for Player 2 on the substituted formula $F[\mathcal{G}]$ is to think of the game as being played on F and simulate the survival strategy there for as long as possible (which is where expansion comes into play).

Let $\mathcal{G} = (U \dot\cup V, E)$ be an $(s, 2)$-boundary expander as stated in the lemma. We have $\mathit{Vars}(F) = U$ and $\mathit{Vars}(F[\mathcal{G}]) = V$. Given a strategy for Player 2 in the r-round k-pebble game on F, we want to convert this into a winning strategy for Player 2 for the $r/(2k)$-round k-pebble game on $F[\mathcal{G}]$. The first idea (which will not quite work) is the following.

While playing on the substituted formula $F[\mathcal{G}]$, Player 2 simulates the game on F. For every position β in the game on $F[\mathcal{G}]$, she maintains a corresponding position α on F, which is defined on all variables whose entire neighbourhood in the expander is contained in the domain of β, i.e., $\mathit{Vars}(\alpha) = \mathrm{Ker}(\mathit{Vars}(\beta))$. The assignments of α should be defined in such a way that they are *consistent with* β, i.e., $\alpha(u) = \bigoplus_{v \in N(u)} \beta(v)$. It follows from the definition of XORification that α falsifies an XOR clause of F if and only if β falsifies an XOR clause of $F[\mathcal{G}]$.

Now Player 2 wants to play in such a way that if β changes to β' in one round of the game on $F[\mathcal{G}]$, then the corresponding position α also changes to α' in one round of the game on F. Intuitively, this should be done as follows. Suppose that starting from a position β, Player 1 asks for a variable $v \in V$. If v is not the last free vertex in a neighbourhood of some $u \in U$, i.e., $\mathrm{Ker}(\mathit{Vars}(\beta)) = \mathrm{Ker}(\mathit{Vars}(\beta) \cup \{v\})$, then Player 2 can make an arbitrary choice as $\alpha = \alpha'$ is consistent with both choices. If v is the last free vertex in the neighbourhood of exactly one vertex u, i.e., $\{u\} = \mathrm{Ker}(\mathit{Vars}(\beta) \cup \{v\}) \setminus \mathrm{Ker}(\mathit{Vars}(\beta))$, then Player 2 assumes that she was asked for u in the simulated game on F. If in her strategy for the r-round k-pebble game on F she would answer with an assignment $a \in \{0, 1\}$ which would yield the new position $\alpha' = \alpha \cup \{u \mapsto a\}$, then in the game on $F[\mathcal{G}]$ she now sets v to the right value $b \in \{0, 1\}$ so that the new position $\beta' = \beta \cup \{v \mapsto b\}$ satisfies the consistency property $\alpha'(u) = \bigoplus_{v \in N(u)} \beta'(v)$. Following that strategy, the number of rounds Player 2 survives the game on $F[\mathcal{G}]$ is lower-bounded by the number of rounds she survives in the game on F.

The gap in this intuitive argument is how to handle the case when the queried variable v completes the neighbourhood of two (or more) vertices u_1, u_2 at the same time, i.e., if we have $\{u_1, u_2\} \subseteq \mathrm{Ker}(\mathit{Vars}(\beta) \cup \{v\}) \setminus \mathrm{Ker}(\mathit{Vars}(\beta))$. This would indeed be a problem, as u_1 and u_2 could guide to two different ways of assigning v, implying that for the new position β' there will be no consistent assignment α' of $\mathrm{Ker}(\mathit{Vars}(\beta'))$.

To circumvent this problem and implement the proof idea above, we will use the boundary expansion of $\mathcal{G}$ to ensure that this problematic case does not occur. For instance, suppose that the graph $\mathcal{G}' = \mathcal{G} \setminus \mathit{Vars}(\beta)$, which is the induced subgraph of $\mathcal{G}$ on $U \setminus \mathit{Vars}(\alpha)$ and $V \setminus \mathit{Vars}(\beta)$, has boundary expansion at least 1. Then the bad situation described above with two variables u_1, u_2 having neighbourhood $N^{\mathcal{G}'}(u_1) = N^{\mathcal{G}'}(u_2) = \{v\}$ in $\mathcal{G}'$ cannot arise, since this would imply $\partial^{\mathcal{G}'}(\{u_1, u_2\}) = \emptyset$, contradicting the expansion properties of $\mathcal{G}'$. Unfortunately, we cannot ensure boundary expansion of $\mathcal{G} \setminus \mathit{Vars}(\beta)$ for every position β, but we can apply Lemma 5.3 and extend the current position to a larger one that is defined on $\gamma(\mathit{Vars}(\beta))$ and has the desired expansion property. Since Lemma 5.3 ensures that $\mathrm{Ker}(\gamma(\mathit{Vars}(\beta)))$, the domain of the consistent α, is bounded by $|\alpha| \leq |\beta| \leq k$, this is still good enough.

We now proceed to present a formal proof. When doing so, it turns out to be convenient for us to prove the contrapositive of the statement discussed above. That is, instead of transforming a strategy for Player 2 in the r-round k-pebble game on F to a strategy for the $r/(2k)$-round k-pebble game on $F[\mathcal{G}]$, we will show that a winning strategy for Player 1 in the game on $F[\mathcal{G}]$ can be used to obtain a winning strategy for Player 1 in the game on F.

Suppose that β is a position in the k-pebble game on $F[\mathcal{G}]$, i.e., a partial assignment of variables in V. Since $|\beta| \leq k$, we can apply Lemma 5.3 to obtain a superset $\gamma(\mathit{Vars}(\beta)) \supseteq \mathit{Vars}(\beta)$ such that $|\mathrm{Ker}(\gamma(\mathit{Vars}(\beta)))| \leq |\mathit{Vars}(\beta)|$ and the induced subgraph $\mathcal{G} \setminus \gamma(\mathit{Vars}(\beta))$ is an $(s/2, 1)$-boundary expander. For the rest of this section, fix a minimal such set $\gamma(V')$ for for every $V' = \mathit{Vars}(\beta)$ corresponding to a position β in the k-pebble game. This will allow us to define formally what we mean by *consistent* positions in the two games on F and $F[\mathcal{G}]$ as described next.

Definition 5.7. Let α be a partial assignment of variables in U and β be a partial assignment of variables in V. We say that α *is consistent with* β if there exists an extension $\beta_{\mathrm{ext}} \supseteq \beta$ with $\mathit{Vars}(\beta_{\mathrm{ext}}) = N(\mathit{Vars}(\alpha)) \cup \mathit{Vars}(\beta)$ such that for all $u \in \mathit{Vars}(\alpha)$ it holds that $\alpha(u) = \bigoplus_{v \in N(u)} \beta_{\mathrm{ext}}(v)$.

For every position β in the k-pebble game on the XOR-substituted formula $F[\mathcal{G}]$ we let $\mathit{Cons}(\beta)$ be the set of all positions α with $\mathit{Vars}(\alpha) = \mathrm{Ker}(\gamma(\mathit{Vars}(\beta)))$ that are consistent with β.

Observe that by Lemma 5.3 it holds that $|\alpha| \leq |\beta|$ for all $\alpha \in \mathit{Cons}(\beta)$. The next claim states the core inductive argument.

Claim 5.8. Let β be a position on $F[\mathcal{G}]$ and suppose that Player 1 wins the i-round k-pebble game on $F[\mathcal{G}]$ from position β. Then Player 1 has a strategy to win the k-pebble game on F within $2ki$ rounds from every position $\alpha \in \mathit{Cons}(\beta)$.

We note that this claim is just a stronger (contrapositive) version of Lemma 5.6.

Proof of Lemma 5.6 assuming Claim 5.8. We apply Claim 5.8 with parameters $\beta = \emptyset$ and $i = r/(2k)$. Since $\mathit{Cons}(\emptyset) = \{\emptyset\}$, we directly get the contrapositive statement of Lemma 5.6 that if Player 1 wins the $r/(2k)$-round k-pebble game on $F[\mathcal{G}]$, then he wins the r-round k-pebble game on F. □

All that remains for us to do now is to establish Claim 5.8, after which the hardness condensation lemma will follow easily.

Proof of Claim 5.8. The proof is by induction on i. For the base case $i = 0$ we have to show that if β falsifies an XOR clause in $F[\mathcal{G}]$, then every assignment $\alpha \in \mathit{Cons}(\beta)$ falsifies an XOR clause in F. But if β falsifies a clause of $F[\mathcal{G}]$, which by construction has the form $C[\mathcal{G}]$ for some clause C from F, then by Definitions 5.5 and 5.7 it holds that every $\alpha \in \mathit{Cons}(\beta)$ falsifies C.

For the induction step, suppose that the statement holds for $i-1$ and assume that Player 1 wins the i-round k-pebble game on $F[\mathcal{G}]$ from position β. Note that a move of Player 1 from position β consists of two steps:

1. Player 1 first chooses a subassignment $\beta' \subseteq \beta$.
2. He then asks for the value of one variable $v \in V$, to which Player 2 has to choose an assignment $a \in \{0, 1\}$ yielding the new position $\beta' \cup \{v \mapsto a\}$.

As Player 1 has a strategy to win from β within i rounds, it follows that he can win from both $\beta' \cup \{v \mapsto 0\}$ and $\beta' \cup \{v \mapsto 1\}$ within $i - 1$ rounds. By the inductive assumption we then immediately obtain the following statement for the set of assignments

$$\mathit{Cons}(\beta' * v) := \bigcup_{a \in \{0,1\}} \mathit{Cons}(\beta' \cup \{v \mapsto a\}) \qquad (10)$$

consistent with either $\beta' \cup \{v \mapsto 0\}$ or $\beta' \cup \{v \mapsto 1\}$.

Subclaim 5.9. Player 1 can win the k-pebble game on F within $2k(i-1)$ rounds from all positions in $\mathit{Cons}(\beta' * v)$.

Note that a position is in $\mathit{Cons}\big(\beta' * v\big)$ if it is consistent with either $\beta' \cup \{v \mapsto 0\}$ or $\beta' \cup \{v \mapsto 1\}$. Therefore, $\mathit{Cons}\big(\beta' * v\big)$ is the set of all positions over $\mathrm{Ker}\big(\gamma\big(\beta'\big) \cup \{v\}\big)$ that are consistent with β'. What remains to show is that from every position $\alpha \in \mathit{Cons}(\beta)$ Player 1 can reach some position in $\mathit{Cons}\big(\beta' * v\big)$ within $2k$ rounds. We split the proof into two steps, corresponding to the two steps in the move of Player 1 from position β.

Subclaim 5.10. From every position $\alpha \in \mathit{Cons}(\beta)$ Player 1 can reach some position in $\mathit{Cons}\big(\beta'\big)$ for $\beta' \subseteq \beta$ within k rounds.

Subclaim 5.11. From every position $\alpha \in \mathit{Cons}\big(\beta'\big)$ Player 1 can reach some position in $\mathit{Cons}\big(\beta' * v\big)$ within k rounds.

We now establish Subclaim 5.11. The proof of Subclaim 5.10 is similar and deferred to the full-length version of the paper.

Proof of Subclaim 5.11. Player 1 starts with some assignment $\alpha_{\mathrm{start}} \in \mathit{Cons}\big(\beta'\big)$ defined over $U_{\mathrm{start}} = \mathrm{Ker}\big(\gamma\big(\mathit{Vars}\big(\beta'\big)\big)\big)$, and wants to reach some assignment $\alpha_{\mathrm{end}} \in \mathit{Cons}\big(\beta' * v\big)$ defined over the variables $U_{\mathrm{end}} = \mathrm{Ker}\big(\gamma\big(\mathit{Vars}\big(\beta'\big) \cup \{v\}\big)\big)$. To do this, he first deletes all assignments of variables in $U_{\mathrm{start}} \setminus U_{\mathrm{end}}$ from α_{start}. Afterwards, he asks for all remaining variables $U' = U_{\mathrm{end}} \setminus U_{\mathrm{start}}$. The difficult part is to ensure that final position is consistent with β' and here the Peeling lemma comes into play.

As discussed above, by our choice of the closure $\gamma\big(\mathit{Vars}\big(\beta'\big)\big)$ (which used Lemma 5.3) we know that the bipartite graph $\mathcal{G}' = \mathcal{G} \setminus \gamma\big(\mathit{Vars}\big(\beta'\big)\big)$ is an $(s/2, 1)$-boundary expander and furthermore that for $U' = U_{\mathrm{end}} \setminus U_{\mathrm{start}}$ it holds that $|U'| \le |U_{\mathrm{end}}| \le \big|\mathit{Vars}\big(\beta'\big) \cup \{v\}\big| \le s/2$. Hence, we can apply the peeling argument in Lemma 5.2 to $\mathcal{G}'$ and U' to obtain an ordered sequence $u_1, \ldots, u_t$ satisfying $N^{\mathcal{G}'}(u_i) \setminus N^{\mathcal{G}'}(\{u_1, \ldots, u_{i-1}\}) \neq \emptyset$. We will think of Player 1 as querying the (at most k) vertices in U' in this order, after which he ends up with a position α_{end} defined on the variables U_{end}. We want to argue that independently of how Player 2 answers, the position α_{end} obtained in this way is consistent with β', and hence contained in $\mathit{Cons}\big(\beta' * v\big)$. We argue this by showing inductively that all positions encountered during the transition from α_{start} to α_{end} are consistent with β'. This clearly holds for the starting position α_{start} by assumption, and hence also for the position obtained from α_{start} by deleting all assignments of variables in $U_{\mathrm{start}} \setminus U_{\mathrm{end}}$. For the induction step, let $i \ge 0$ and assume inductively that the current position α_i over

$$U_i := (U_{\mathrm{start}} \cap U_{\mathrm{end}}) \cup \{u_j \mid 1 \le j \le i\} \tag{11}$$

is consistent with β'. Now Player 1 asks about the variable u_{i+1} and Player 2 answers with a value a_{i+1}. Since α_i is consistent with β', there is an assignment $\beta_{\mathrm{ext}} \supseteq \beta'$ that sets the variables $v \in N(\mathit{Vars}(\alpha_i))$ to the right values such that $\alpha_i(u) = \bigoplus_{v \in N(u)} \beta_{\mathrm{ext}}(v)$ for all $u \in \mathit{Vars}(\alpha_i)$. By our ordering of $U' = \{u_1, \ldots, u_t\}$ chosen above we know that u_{i+1} has at least one neighbour on the right-hand side V that is neither contained in $N^{\mathcal{G}}(U_i) = N^{\mathcal{G}}(\mathit{Vars}(\alpha_i))$ nor in the domain of β'. Hence, regardless of which value a_{i+1} Player 2 chooses for her answer we can extend the assignment β_{ext} to the variables $N^{\mathcal{G}}(u_{i+1}) \setminus \big(N^{\mathcal{G}}\big(\mathit{Vars}(\alpha_i)\big) \cup \mathit{Vars}\big(\beta'\big)\big)$ in such a way that $\bigoplus_{v \in N(u_{i+1})} \beta_{\mathrm{ext}}(v) = a_{i+1}$. This shows that α_{i+1} defined over $U_{i+1} = (U_{\mathrm{start}} \cap U_{\mathrm{end}}) \cup \{u_j \mid 1 \le j \le i+1\}$ is consistent with β'. Subclaim 5.11 now follows by the induction principle. $\dashv$

Combining Subclaims 5.9, 5.10 and 5.11, we conclude that Player 1 wins from every position $\alpha \in \mathit{Cons}(\beta)$ within $2ki$ rounds. This concludes the proof of Claim 5.8. $\square$

We are finally in a position to give a formal proof of Lemma 3.3.

Proof of Lemma 3.3. We let Δ_0 be the constant in Lemma 5.4. Suppose we are given an m-variable p-XOR formula F and parameters ℓ_{lo}, ℓ_{hi}, r, Δ that satisfy the conditions in Lemma 3.3. We set $s := 2\ell_{\mathrm{hi}}$. By the requirements on Δ we have $(s\Delta)^{2\Delta} \le m$ and $\Delta \ge \Delta_0$. Hence we can apply Lemma 5.4 to obtain an $m \times \lceil m^{3/\Delta} \rceil$ $(s, \Delta, 2)$-boundary expander $\mathcal{G} = (U \mathbin{\dot\cup} V, E)$, and applying XORification with respect to $\mathcal{G}$ we construct the formula $H := F[\mathcal{G}]$.

For the upper bound in Lemma 3.3(a'), we recall that Player 1 has a winning strategy in the ℓ_{lo}-pebble game on F by assumption (a) in the lemma. He uses this strategy to win the $(\Delta \cdot \ell_{\mathrm{lo}})$-pebble game on H as follows. Whenever his strategy tells him to ask for a variable $u \in U = \mathit{Vars}(F)$, he instead asks for the at most Δ variables in $N(u) \subseteq V = \mathit{Vars}(H)$ and assigns to u the value that corresponds to the parity of the answers Player 2 assigns to $N(u)$. In this way, he can simulate his strategy on F until he reaches an assignment that contradicts an XOR clause C from F. As the corresponding assignment of the variables $\{v \mid v \in N(u), u \in \mathit{Vars}(C)\}$ falsifies the constraint $C[\mathcal{G}] \in H$, at this point Player 1 wins the $(\Delta \cdot \ell_{\mathrm{lo}})$-pebble game on H.

The lower bound in Lemma 3.3(b') follows immediately from Lemma 5.6. Since by assumption (b) in Lemma 3.3 Player 1 does not win the ℓ_{hi}-pebble game on F within r rounds, Lemma 5.6 implies that he does not win the ℓ_{hi}-pebble game on H within $r/(2\ell_{\mathrm{hi}})$ rounds either. $\square$

6. Concluding Remarks

In this paper we prove an $n^{\Omega(k/\log k)}$ lower bound on the minimal quantifier depth of L^k and C^k sentences that distinguish two finite n-element relational structures, nearly matching the trivial n^{k-1} upper bound. By the known connection to the k-dimensional Weisfeiler–Leman algorithm, these results imply near-optimal $n^{\Omega(k/\log k)}$ lower bounds also on the number of refinement steps of this algorithm.

An obvious open problem is to improve the lower bound. One way to achieve this would be to strengthen the lower bound on the number of rounds in the k-pebble game on 3-XOR formulas in Lemma 3.2 from $n^{\log^{-1} k}$ to n^{δ} for some $\delta \gg \log^{-1} k$. By the hardness condensation lemma this would directly improve our lower bound from $n^{\Omega(k/\log k)}$ to $n^{\Omega(\delta k)}$.

The structures on which our lower bounds hold are n-element relational structures of arity $\Theta(k)$ and size $n^{\Theta(k)}$. We would have liked to have this results also for structures of bounded arity, such as graphs. However, the increase of the arity is inherent in the construction as the method of substitution decreases the number of variables in an XOR formula, while the number of clauses remains unchanged. An optimal lower bound of $n^{\Omega(k)}$ on the quantifier depth required to distinguish two n-vertex graphs has been obtained by the first author in an earlier work [10] for the *existential-positive fragment* of L^k. Determining the quantifier rank of full L^k and C^k on n-vertex graphs remains an open problem.

Another open question concerns the complexity of finite variable equivalence for non-constant k. What is the complexity of deciding, given two structures and a parameter k, whether the structures are equivalent in L^k or C^k? As this problem can be solved in time $(\|\mathcal{A}\| + \|\mathcal{B}\|)^{O(k)}$, it is in $\mathsf{EXPTIME}$ (if k is part of the input). It has been conjectured that this problem is $\mathsf{EXPTIME}$-complete [17], but it is not even known whether it is NP-hard. Note that the quantifier depth is connected to the computational complexity of the equivalence problem by the fact that an upper bound of the form $n^{O(1)}$ on n-element structures would have implied that testing equivalence is in PSPACE. Hence, our lower bounds on the quantifier depth can be seen as a necessary requirement for establishing $\mathsf{EXPTIME}$-hardness of the equivalence problem.

Acknowledgments

We are very grateful to Alexander Razborov for patiently explaining his hardness condensation technique during numerous and detailed discussions. We also want to thank Neil Immerman for helping us find relevant references to previous works and explaining some of the finer details in [24]. The first author wishes to acknowledge useful feedback from the participants of Dagstuhl Seminar 15511 *The Graph Isomorphism Problem*. Finally, we are most indebted to the anonymous *LICS* reviewers for very detailed comments that helped improve the manuscript considerably.

Part of the work of the first author was performed while at KTH Royal Institute of Technology supported by a fellowship within the Postdoc-Programme of the German Academic Exchange Service (DAAD). The research of the second author was supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007–2013) / ERC grant agreement no. 279611 and by Swedish Research Council grants 621-2010-4797 and 621-2012-5645.

References

[1] M. Alekhnovich and A. A. Razborov. Lower bounds for polynomial calculus: Non-binomial case. *Proc. Steklov Institute of Mathematics*, 242:18–35, 2003.

[2] L. Babai. Graph isomorphism in quasipolynomial time. In *Proc. 48th ACM Symposium on Theory of Computing (STOC '16)*, 2016. To appear.

[3] J. Barwise. On Moschovakis closure ordinals. *J. Symbolic Logic*, 42(2):292–296, 1977.

[4] C. Beck, J. Nordström, and B. Tang. Some trade-off results for polynomial calculus. In *Proc. 45th ACM Symposium on Theory of Computing (STOC '13)*, pp. 813–822, 2013.

[5] E. Ben-Sasson. Hard examples for the bounded depth Frege proof system. *Computational Complexity*, 11(3-4):109–136, 2002.

[6] E. Ben-Sasson. Size space tradeoffs for resolution. In *Proc. 34th ACM Symposium on Theory of Computing (STOC '02)*, pp. 457–464, 2002.

[7] E. Ben-Sasson and J. Nordström. Short proofs may be spacious: An optimal separation of space and length in resolution. In *Proc. 49th IEEE Symposium on Foundations of Computer Science (FOCS '08)*, pp. 709–718, 2008.

[8] E. Ben-Sasson and J. Nordström. Understanding space in proof complexity: Separations and trade-offs via substitutions. In *Proc. 2nd Symposium on Innovations in Computer Science (ICS '11)*, pp. 401–416, 2011.

[9] E. Ben-Sasson and A. Wigderson. Short proofs are narrow—resolution made simple. *J. ACM*, 48(2):149–169, 2001.

[10] C. Berkholz. The propagation depth of local consistency. In *Proc. 20th International Conference on Principles and Practice of Constraint Programming (CP '14)*, pp. 158–173, 2014.

[11] J. Buresh-Oppenheim and T. Pitassi. The complexity of resolution refinements. *J. Symbolic Logic*, 72(4):1336–1352, 2007.

[12] S. R. Buss, D. Grigoriev, R. Impagliazzo, and T. Pitassi. Linear gaps between degrees for the polynomial calculus modulo distinct primes. *J. Computer and System Sciences*, 62(2):267–289, 2001.

[13] J.-Y. Cai, M. Fürer, and N. Immerman. An optimal lower bound on the number of variables for graph identifications. *Combinatorica*, 12(4):389–410, 1992.

[14] S. A. Cook. An observation on time-storage trade off. *J. Computer and System Sciences*, 9(3):308–316, 1974.

[15] Y. Filmus, M. Lauria, M. Mikša, J. Nordström, and M. Vinyals. Towards an understanding of polynomial calculus: New separations and lower bounds (Extended abstract). In *Proc. 40th International Colloquium on Automata, Languages and Programming (ICALP '13)*, pp. 437–448. 2013.

[16] M. Fürer. Weisfeiler–Lehman refinement requires at least a linear number of iterations. In *Proc. 28th International Colloquium on Automata, Languages, and Programming (ICALP '01)*, pp. 322–333. 2001.

[17] E. Grädel, P. G. Kolaitis, L. Libkin, M. Marx, J. Spencer, M. Y. Vardi, Y. Venema, and S. Weinstein. *Finite Model Theory and Its Applications*. Springer, 2007.

[18] D. Grigoriev. Linear lower bound on degrees of Positivstellensatz calculus proofs for the parity. *Theoretical Computer Science*, 259(1–2):613–622, 2001.

[19] M. Grohe. Finite variable logics in descriptive complexity theory. *Bulletin of Symbolic Logic*, 4(4):345–398, 1998.

[20] M. Grohe. Equivalence in finite-variable logics is complete for polynomial time. *Combinatorica*, 19(4):507–532, 1999.

[21] M. Grohe. Fixed-point definability and polynomial time on graphs with excluded minors. *J. ACM*, 59(5):27:1–27:64, 2012.

[22] Y. Gurevich and S. Shelah. On finite rigid structures. *J. Symbolic Logic*, 61(2):549–562, 1996.

[23] L. Hella. Logical hierarchies in PTIME. *Information and Computation*, 129:1–19, 1996.

[24] N. Immerman. Number of quantifiers is better than number of tape cells. *J. Computer and System Sciences*, 22(3):384–406, 1981.

[25] N. Immerman. Upper and lower bounds for first order expressibility. *J. Computer and System Sciences*, 25(1):76–98, 1982.

[26] N. Immerman and E. Lander. Describing graphs: a first-order approach to graph canonization. In *Complexity Theory Retrospective: In Honor of Juris Hartmanis on the Occasion of His Sixtieth Birthday*, pp. 59–81. Springer, 1990.

[27] S. Kiefer and P. Schweitzer. Upper bounds on the quantifier depth for graph differentiation in first order logic. In *Proc. 31st ACM/IEEE Symposium on Logic in Computer Science (LICS '16)*, 2016. To appear.

[28] A. Kojevnikov and D. Itsykson. Lower bounds of static Lovász–Schrijver calculus proofs for Tseitin tautologies. In *Proc. 33rd International Colloquium on Automata, Languages and Programming (ICALP '06)*, pp. 323–334. 2006.

[29] A. Krebs and O. Verbitsky. Universal covers, color refinement, and two-variable counting logic: Lower bounds for the depth. In *Proc. 30th ACM/IEEE Symposium on Logic in Computer Science (LICS '15)*, pp. 689–700, 2015.

[30] M. Mikša and J. Nordström. A generalized method for proving polynomial calculus degree lower bounds. In *Proc. 30th Computational Complexity Conference (CCC '15)*, pp. 467–487, 2015.

[31] J. Nordström. Pebble games, proof complexity and time-space trade-offs. *Logical Methods in Computer Science*, 9:15:1–15:63, 2013.

[32] A. A. Razborov. A new kind of tradeoffs in propositional proof complexity. *J. ACM*, 63:16:1–16:14, 2016.

[33] G. Schoenebeck. Linear level Lasserre lower bounds for certain k-CSPs. In *Proc. 49th IEEE Symposium on Foundations of Computer Science (FOCS '08)*, pp. 593–602, 2008.

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

[35] M. Y. Vardi. On the complexity of bounded-variable queries (Extended abstract). In *Proc. 14th ACM Symposium on Principles of Database Systems (PODS '95)*, pp. 266–276, 1995.