Locality from Circuit Lower Bounds

Matthew Anderson, Dieter van Melkebeek, Nicole Schweikardt, Luc Segoufin

Locality from Circuit Lower Bounds*

Matthew Anderson†
University of Wisconsin-Madison, USA
mwa@cs.wisc.edu

Dieter van Melkebeek†
University of Wisconsin-Madison, USA
dieter@cs.wisc.edu

Nicole Schweikardt
Goethe-Universität Frankfurt am Main, Germany
schweika@informatik.uni-frankfurt.de

Luc Segoufin
INRIA and ENS-Cachan, LSV, France
http://www-rocq.inria.fr/~segoufin

August 23, 2012

We study the locality of an extension of first-order logic that captures graph queries computable in AC^0, i.e., by families of polynomial-size constant-depth circuits. The extension considers first-order formulas over relational structures which may use arbitrary numerical predicates in such a way that their truth value is independent of the particular interpretation of the numerical predicates. We refer to such formulas as Arb-invariant first-order.

We consider the two standard notions of locality, Gaifman and Hanf locality. Our main result gives a Gaifman locality theorem: An Arb-invariant first-order formula cannot distinguish between two tuples that have the same neighborhood up to distance $(\log n)^c$, where n represents the number of elements in the structure and c is a constant depending on the formula. When restricting attention to string structures, we achieve the same quantitative strength for Hanf locality. In both cases we show that our bounds are tight. We also present an application of our results to the study of regular languages.

Our proof exploits the close connection between first-order formulas and the complexity class AC^0, and hinges on the tight lower bounds for parity on constant-depth circuits.

* A preliminary version of this paper appeared in ICALP 2011 [AvMSS11].
† Partially supported by NSF grants 0728809 and 1017597.

Contents

1 Introduction

Expressibility of logics over finite structures plays an important role in various areas of computer science. In *descriptive complexity*, logics over finite structures are used to characterize complexity classes [Imm99]. E.g., existential second-order logic can describe exactly those graph problems that belong to the complexity class NP. Concerning *databases*, common query languages have well-known logical equivalents. In particular, the relational calculus has precisely the power of first-order logic (FO); extensions of FO by aggregation, grouping, arithmetic operations, or recursive definitions, capturing large parts of the database query language SQL, have been identified in the literature (cf., e.g., [Lib03, SSS09]). In *automated verification*, one uses logics as specification languages to describe properties of hardware and software systems, and one needs to balance the expressivity of the logics used with the feasibility of the model checking task (cf., e.g., [CGP99]).

The classical inexpressibility arguments for logics over finite structures (i.e., back-and-forth systems or Ehrenfeucht-Fraïssé games, cf., e.g., [Lib04]) often involve nontrivial combinatorics. The notion of *locality* has been proposed as an alternative that allows one to contain much of the hard work in generic results, and keep the specific applications simple. Roughly speaking, a query is local if one only needs to look at a small, localized part of the structure in order to answer the query. If one can show that every query in a given logic has a certain degree of locality, and the query at hand does not, then one can conclude that the query is not expressible in the logic. For example, one can show that for every first-order query on graphs, there exists a constant r such that the result of the query depends only on the neighborhood up to distance r of the vertices that are part of the query (cf., e.g., the textbook [Lib04]). On the other hand, it is easy to see that the connectivity of two vertices in a graph is not determined solely by the restriction to those neighborhoods. Therefore, connectivity is not expressible in first-order logic. Apart from inexpressibility proofs, locality is also used as a tool for obtaining algorithmic meta theorems, i.e., results stating that if a problem is expressible in a certain logic on a certain class of structures, then it can be solved algorithmically within certain resource bounds (c.f., [GK11] for a recent overview on this topic).

In this paper we show how to use *circuit lower bounds* to establish *upper bounds on the locality radius* of certain logics. In particular, we consider a logic that corresponds to the complexity class AC^0 of all languages that can be decided by nonuniform families of polynomial-size constant-depth circuits. By exploiting the known lower bounds for parity and related problems on constant-depth circuits [Hås86], we obtain an upper bound for the locality radius of queries expressible in that logic. We also present an application of this result to the study of regular languages, and we give examples showing that our upper bounds are essentially tight.

The logic we consider is the extension of order-invariant first-order logic with arbitrary numerical predicates. The notion of *order-invariance* was introduced a while ago to capture the data independence principle in databases, cf., [AHV95, Lib04]: An implementation of a database query may exploit the order in which the elements are stored on a disk, but only in such a way that the result of the query does not depend on this order. Order-invariant first-order queries are exactly those first-order queries that can make use of an order predicate $<$ but such that the answer is independent of the interpretation of $<$ as a linear order on the domain of the structure. In our extension, on top of the order predicate $<$, we also allow the use of *arbitrary numerical predicates* that are induced by the order. We require that the result of a query not depend on the actual

choice of the linear order when all numerical predicates are interpreted consistent with the linear order. We denote this logic[1] as *Arb-invariant* FO. In terms of graph queries, Arb-invariant FO expresses precisely those computable in the complexity class AC^0. We refer to Section 1.3 for more background.

1.1 Results

In order to state our results, we need to introduce the two standard notions of locality, known as *Gaifman locality* and *Hanf locality*. Both are based on the distance measure on the elements of a structure when viewed as the vertices of a graph in which two elements are connected by an edge whenever they appear together in a tuple of one of the structure's relations. The latter graph is referred to as the *Gaifman graph* of the structure. In a nutshell, Gaifman locality means that a query cannot distinguish between two tuples having the same neighborhood type in a given structure, while Hanf locality means that a query cannot distinguish between two structures having the same (multi-)set of neighborhood types. Here, the neighborhood type of a tuple refers to the equivalence class under isomorphism of the substructure induced by the elements up to distance r from the tuple, where r is a parameter. It is known that Hanf locality implies Gaifman locality, modulo a constant factor loss in the distance parameter r. We refer to Section 2 for the formal treatment of these notions.

A well-known result (c.f., e.g., [Lib04]) shows that first-order logic exhibits constant locality w.r.t. both notions, i.e., every FO query is Gaifman and Hanf local with a constant parameter r depending on the query. In the presence of an extra linear order that is part of the structure, all neighborhoods of positive radius degenerate to the entire domain, so all queries are trivially 1-local. Locality becomes meaningful again in *order-invariant* FO, where the formulas can make use of an order, but the structure does not contain the order and the semantics are independent of the order. It is shown in [GS00] that order-invariant FO queries are Gaifman local with a constant parameter r depending on the query. The status of Hanf locality for order-invariant FO is still open in general; it is only known for structures like strings and trees [BS09b].

When we allow arbitrary numerical predicates, constant locality no longer holds, even if we require Arb-invariance. In fact, we show that the level of Gaifman locality of Arb-invariant FO queries can be polylogarithmic in the number of elements of the structure, but no worse than that: Arb-invariant FO is Gaifman $(\log n)^{O(1)}$-local in the following sense.

Theorem 1.1. *Every Arb-invariant* FO *formula is Gaifman* $(\log n)^c$*-local for some constant* c *depending on the formula, and for every constant* c *there exists an Arb-invariant* FO *formula that is not Gaifman* $(\log n)^c$*-local.*

The upper bound in Theorem 1.1 means that for any query in Arb-invariant FO and any large enough number n, if a structure has n elements and if two tuples of that structure have the same neighborhood up to distance $(\log n)^c$, then they cannot be distinguished by the query. The lower bound part of Theorem 1.1 is realized by variations of the connectivity example mentioned before.

As easy consequence of the upper bound in Theorem 1.1 one obtains, e.g., that the following graph queries are not computable in AC^0: Does a node x lie on a cycle? Are two nodes x and y

[1] Strictly speaking, Arb-invariant FO is a "logical system" rather than a "logic", as the syntax is undecidable (cf., e.g., [Lib04]).

connected by a path? Do nodes x and y have the same distance to node z? Does node x belong to a connected component that is acyclic?

Theorem 1.1 provides an essentially complete picture of the Gaifman locality of Arb-invariant FO. Similar to the case of order-invariant FO, the Hanf locality of Arb-invariant FO queries is still open in general, but if we restrict our attention to structures that represent strings, we can establish Hanf locality with the same bounds as in Theorem 1.1. In the following statement, Arb-invariant FO(*Succ*) refers to Arb-invariant queries over string structures.

Theorem 1.2. *Every Arb-invariant* FO(*Succ*) *formula is Hanf* $(\log n)^c$*-local for some constant* c *depending on the formula, and for every constant* c *there exists an Arb-invariant* FO(*Succ*) *formula that is not Hanf* $(\log n)^c$*-local.*

The upper bound in Theorem 1.2 means the following, where we use r to denote $(\log n)^c$: For any Arb-invariant FO query over strings and any large enough number n, if two strings of length n have the same prefix of length $2r$, the same suffix of length $2r$, and the same multiset of substrings of length $2r+1$, then they cannot be distinguished by the query. We believe that the upper bound in Theorem 1.2 generalizes to structures over trees (c.f., the discussion in Section 7 for more details). Since Hanf locality implies Gaifman locality, the lower bound in Theorem 1.2 can be viewed as a strengthening of the lower bound part of Theorem 1.1.

We also present an application of our locality results to the study of regular languages. It is known that the class of definable regular languages does not grow when we move from FO to order-invariant FO [BS09b], but does grow when we proceed to addition-invariant FO, i.e., Arb-invariant FO where the only numerical predicate used is addition [SS10]. The larger class coincides with FO(*Succ*, *lm*), i.e., the extension of FO(*Succ*) with predicates determining the length of a string modulo some constant [SS10]. Based on our locality results, we can show that the class does not grow further when we allow the use of arbitrary numerical predicates, i.e., when we proceed from addition-invariant FO to Arb-invariant FO.

Theorem 1.3. *A regular language is definable in Arb-invariant* FO(*Succ*) *iff it is definable in* FO(*Succ*, *lm*).

1.2 Techniques

Our proof of the *upper bound* on Gaifman locality in Theorem 1.1 exploits the tight connection between Arb-invariant FO formulas and the complexity class AC^0: Given an Arb-invariant FO formula φ that distinguishes two points of the universe whose neighborhoods up to distance r are of the same type, we construct a circuit on $2m = \Theta(r)$ bits that distinguishes inputs with exactly m ones from inputs with exactly $m+1$ ones. In the special case where the neighborhoods of the two points are disjoint the circuit actually computes parity. The depth of the circuit is a constant depending on φ, and its size is polynomial in n. The known exponential circuit lower bounds [Hås86] then imply that r is bounded by a polylogarithmic function in n. This argument establishes the upper bound in Theorem 1.1 for the case of formulas with a single free variable. In order to handle an arbitrary number k of free variables, we show how to reduce any case with $k > 1$ free variables to one with fewer variables in a way that is conceptually similar to (but technically different from) [GS00]. See Section 1.3 for a more detailed discussion of this related work.

As mentioned before, we do not know how to extend the upper bound of Theorem 1.1 to the stronger notion of Hanf locality in general, but we can establish it in Theorem 1.2 for the special case of strings. The reason the latter case is simpler is because on strings being Hanf local is equivalent to closure under swapping substrings whose endpoints have the same neighborhood type — a condition that has much of a Gaifman locality flavor. In fact, to prove that closure under such swaps holds for Arb-invariant FO($Succ$) formulas, we use a reduction to the upper bound for Gaifman locality from Theorem 1.1.

The *lower bounds* in Theorems 1.1 and 1.2 follow because arithmetic predicates like addition and multiplication allow one to define a bijection between the elements of a first-order definable set S of polylogarithmic size and an initial segment of the natural numbers [DLM07]. Thus, the binary representation of a single element of the entire domain can be used to represent a list of elements of S. By exploiting this, Arb-invariant FO can express, e.g., reachability between two nodes in S by a path of polylogarithmic length.

For the proof of Theorem 1.3 we use a characterization from [SS10] stating that a regular language is definable in addition-invariant FO iff it is definable in FO($Succ, lm$) iff it is closed under two operations called "swap" and "transfer". By applying a pumping argument we obtain that Hanf locality and regularity imply closure under "swaps". Furthermore, a reduction from circuit lower bounds, similar to the one used for the upper bound proof of Theorem 1.1, along with a pumping argument shows that regular languages definable in Arb-invariant FO are closed under "transfers".

1.3 Related Work

We now give a brief overview of related work.

Invariant logics The expressiveness of order-invariant FO was considered in various places, cf., e.g., [AHV95, Lib04, EF99, GS00, BS09b]. Logics allowing invariant uses of predicates weaker than the linear order were considered in [Ros07, Ott00], concentrating on successor-invariant FO and epsilon-invariant FO, respectively. Logics allowing invariant uses of arbitrary numerical predicates were formally introduced in [Mak97], pointing out, in particular, that the graph properties definable in Arb-invariant FO are precisely the graph properties computable in AC^0. Similarly, [Mak98] showed that the graph properties definable in Arb-invariant least fixed-point logic coincide with the graph properties computable in P/poly. By results of [Imm87, Imm86, Var82] it is known that $(+, \times)$-invariant FO (i.e., Arb-invariant FO where the formulas only use the numerical predicates $+$ and $\times$) and order-invariant least fixed-point logic precisely capture the graph properties computable in uniform AC^0 and in polynomial time, respectively.

Quite a number of articles in the circuit complexity literature and the finite model theory literature concentrated on graph properties (or queries) computable in AC^0 or definable in Arb-invariant FO (or variants thereof), without explicitly mentioning the notion of Arb-invariance. For example, [Raz85, And85, AB87] showed an exponential lower bound on the size of monotone circuits computing the k-clique problem on n-vertex graphs. Recently, [Ros08, Ros10] established a strong lower bound on the size of constant-depth circuits computing the k-clique problem, and applied this to show that the bounded variable hierarchy inside FO is strict on the class of finite ordered graphs and on the class of finite graphs enriched by arbitrary numerical predicates. [Ajt89] showed that the query selecting all pairs (x, y) of nodes in a graph that are connected by a path of length

at most $f(n)$, where n is the size of the graph and f is an unbounded function, is not definable in Arb-invariant FO. In [Ajt83] it was shown that the class of graphs having an even number of edges is not definable in the Arb-invariant version of the extension of FO called existential monadic second-order logic (EMSO). [FSV95, Sch96] proved that connectivity of graphs is not definable in EMSO with numerical predicates of moderate degree.

Locality The notions of Hanf and Gaifman locality were introduced in [FSV95, HLN99], going back to results from [Han65, Gai82]. Showing that a logic is Hanf or Gaifman local provides insight into the limitations of its expressiveness and constitutes a high-level tool for proving that certain properties or queries cannot be expressed by formulas of this logic. Hanf and Gaifman locality results have been obtained for FO and for various extensions of FO (e.g., by counting quantifiers). For an overview on locality results and their applications in complexity theory we refer to [LN00]. Most locality results obtained in the literature deal with locality radii of constant size (cf., the example on FO mentioned at the beginning of the introduction, and the results mentioned in [HLN99, LN00, GS00, Lib04]). In their concluding sections, the articles [HLN99, GS00], however, proposed to also consider notions of locality where the radius of the neighborhoods grows with the size of the structures — this is what we do in the present paper. As pointed out in [HLN99, GS00], an analogue of our Theorem 1.1 for order-invariant first-order logic with counting quantifiers would lead to a separation of the complexity classes TC^0 and LOGSPACE.

The notion of locality in logic has a somewhat similar flavor to the notion of sensitivity in circuit complexity. The sensitivity of a Boolean function f at an input x is the number of bit positions i in x such that if we flip the ith bit in x, then the value of f changes. The average sensitivity of every function f in AC^0 over all inputs of length n is known to be polylogarithmically bounded in n [LMN93]. The latter result is closely related to the exponential lower bounds for parity on constant-depth circuits [Hås86]. Rather than going through sensitivity, our argument for proving Theorems 1.1 and 1.2 directly uses those circuit lower bounds to establish a polylogarithmic upper bound on the locality of Arb-invariant FO.

Comparison with [GS00] Our Theorem 1.1 can be viewed as an analogue of the main result of [GS00]. While their result states that *order*-invariant FO queries are Gaifman local with a *constant* locality radius r (depending on the query), our result states that *Arb*-invariant FO queries are Gaifman local with a locality radius $(\log n)^c$ where c is a constant (depending on the query) and n is the size of the underlying structure. Our proof of the upper bound in Theorem 1.1 has the same overall structure as the proof of [GS00]: It first considers queries of arity $k = 1$ for the case of disjoint neighborhoods, then for the case of overlapping neighborhoods, and afterwards it gives a reduction from queries of arbitrary arity $k > 1$ to queries of arity $k-1$. Our method for handling the case of overlapping neighborhoods uses techniques from [GS00]; however, our overall argument for treating queries of arity $k = 1$ gives a reduction to lower bounds in circuit complexity, while the argument of [GS00] relies on Ehrenfeucht-Fraïssé games. Our proof for the arity reduction from $k > 1$ to $k-1$ is conceptually similar to the proof of [GS00], but involves substantial technical differences. On the one hand, the notion of Arb-invariance allows us to give a non-uniform reduction (while the order-invariance of [GS00] requires uniformity). On the other hand, Arb-invariance requires us to construct a reduction that does not change the size of the universe of the structures considered (while the reduction of [GS00] changes the size of the universe and builds on the fact that this

preserves order-invariance).

1.4 Organization

In Section 2 we present general preliminaries concerning first-order logic, locality, and circuit complexity. In Section 3 we formally introduce our notion of Arb-invariance and describe its connection to AC^0. Section 4 contains our results for Gaifman locality, Section 5 our results for Hanf locality on strings, and Section 6 our application to regular languages. We end in Section 7 with some suggestions for further research.

2 Preliminaries

In this section we briefly review relevant background material on structures and queries, first-order logic, Gaifman locality, Hanf locality, circuit complexity, and how to encode structures and queries as strings.

Structures and queries A *relational schema* is a set of relation symbols each with an associated arity. A *structure* M over a relational schema τ is a *finite* set $dom(M)$, the *domain*, containing all the *elements* of M, together with an interpretation $R^M \subseteq dom(M)^k$ of each relation symbol $R \in \tau$ of arity k. We call a structure over a relational schema τ a *τ-structure*. The *size* of a structure M is the cardinality of its domain $dom(M)$.

If U is a set of elements of M, then $M_{|U}$ denotes the *induced substructure of M on U*. That is, $M_{|U}$ is the structure whose domain is U and whose relations are the relations of M restricted to those tuples containing only elements in U.

We say that two τ-structures M and M' are *isomorphic*, $M \cong M'$, if there exists a bijection $\pi : dom(M) \to dom(M')$ such that for each k-ary relation symbol $R \in \tau$, $(a_1, a_2, \ldots, a_k) \in R^M$ iff $(\pi(a_1), \pi(a_2), \ldots, \pi(a_k)) \in R^{M'}$. We write $\pi : M \cong M'$ to indicate that π is an isomorphism that maps M to M'. If $\bar{a}$ and $\bar{b}$ are tuples (of the same length) of elements of $dom(M)$ and $dom(M')$, respectively, then we write $(M, \bar{a}) \cong (M', \bar{b})$ to indicate that there is an isomorphism $\pi : M \cong M'$ which maps $\bar{a}$ to $\bar{b}$. All *classes* of structures considered in this paper are closed under isomorphisms.

A *k-ary query on τ-structures* is a mapping q that associates with each τ-structure M a relation $q(M) \subseteq dom(M)^k$, and that is closed under isomorphism in the following sense: If $(M, \bar{a}) \cong (M', \bar{b})$, then $\bar{a} \in q(M)$ iff $\bar{b} \in q(M')$.

First-order logic We denote by $FO(\tau)$ the first-order logic with respect to the schema τ. This is the set of logical formulas whose atoms are formed based on the relation symbols in τ, the equality symbol $=$, and an infinite sequence of variables $(x_1, x_2, \ldots)$, and that is closed under Boolean connectives ($\wedge, \vee$, and $\neg$) and existential and universal quantifications ($\exists$ and $\forall$). We use the standard syntax and semantics for FO (cf., e.g., [Lib04]). If ϕ is a formula, we write $\phi(\bar{x})$ to denote that $\bar{x}$ is a list of the free variables of ϕ. The alternation depth of a formula is the maximum, over all paths from the root of a formula to its atoms, of the number of alternating blocks of quantifiers along the path. We write $M \models \phi(\bar{a})$ or $(M, \bar{a}) \models \phi(\bar{x})$ to express that the tuple $\bar{a}$ of elements in $dom(M)$ makes the formula $\phi(\bar{x})$ true in M. A formula $\phi(\bar{x})$ with k free variables defines the k-ary query that associates with every τ-structure M the set of k-tuples $\bar{a} \in dom(M)^k$ for which

$M \models \phi(\bar{a})$. Sometimes, we will say that $\phi(\bar{x})$ is a *k-ary formula*. A *sentence* is a formula that has no free variables.

Neighborhoods To each structure M we associate an undirected graph $G(M)$, known as the *Gaifman graph* of M, whose vertices are the elements of the domain of M and whose edges relate two elements of M whenever there exists a tuple in one of the relations of M in which both appear. For example, consider a relational schema τ consisting of one binary relation symbol E. Each τ-structure M is then a directed graph in the standard sense, and $G(M)$ coincides with M when ignoring the orientation. Given two elements u and v of a structure M, we denote as $dist^M(u, v)$ the distance between u and v in M, which is defined as their distance in the Gaifman graph $G(M)$. If $\bar{a}$ and $\bar{b}$ are tuples of elements of M, then $dist^M(\bar{a}, \bar{b})$ denotes the minimum distance between any pair of elements (one from $\bar{a}$ and one from $\bar{b}$).

For every $r \in \mathbb{N}$ and tuple $\bar{a} \in dom(M)^k$, the *r-ball around* $\bar{a}$ *in* M is the set

$$N_r^M(\bar{a}) \quad := \quad \{v \in dom(M) : dist^M(\bar{a}, v) \leq r\},$$

and the *r-neighborhood around* $\bar{a}$ *in* M is the structure

$$\mathcal{N}_r^M(\bar{a}) \quad := \quad \big(M_{|N_r^M(\bar{a})}\,,\, \bar{a}\big).$$

That is, $\mathcal{N}_r^M(\bar{a})$ is the induced substructure of M on $N_r^M(\bar{a})$ with k distinguished elements $\bar{a}$. Two neighborhoods $\mathcal{N}_r^M(\bar{a})$ and $\mathcal{N}_r^{M'}(\bar{b})$ are isomorphic, if there is an isomorphism $\pi : M_{|N_r^M(\bar{a})} \cong M'_{|N_r^{M'}(\bar{b})}$ that maps $\bar{a}$ to $\bar{b}$.

Locality Let $\phi(\bar{x})$ be a logical formula with k free variables. We consider two notions of locality of $\phi(\bar{x})$ (the precise definitions are basically taken from [Lib04]). We first define the notions with respect to *fixed* structures and then with respect to *all* structures.

Definition 2.1 (Gaifman Locality). *A formula $\phi(\bar{x})$ is* Gaifman r-local *with respect to a τ-structure M, if for all tuples $\bar{a}, \bar{b} \in dom(M)^k$ we have*

$$\mathcal{N}_r^M(\bar{a}) \cong \mathcal{N}_r^M(\bar{b}) \quad \Longrightarrow \quad M \models \phi(\bar{a}) \;\; \textit{iff} \;\; M \models \phi(\bar{b}). \tag{1}$$

For any two τ-structures M, M' and any tuples $\bar{a} \in dom(M)^k$ and $\bar{b} \in dom(M')^k$ we write $(M, \bar{a}) \equiv_r (M', \bar{b})$ if there is a bijection $h : dom(M) \to dom(M')$ such that for every element c in the domain of M, $\mathcal{N}_r^M(c\bar{a}) \cong \mathcal{N}_r^{M'}(h(c)\bar{b})$. Equivalently, the various *isomorphism types* (i.e., distinct local neighborhoods of radius r) occur with the same cardinality in M and M'.

Definition 2.2 (Hanf Locality). *A formula $\phi(\bar{x})$ is* Hanf r-local *with respect to a pair of τ-structures (M, M'), if for all tuples $\bar{a} \in dom(M)^k$ and $\bar{b} \in dom(M')^k$*

$$(M, \bar{a}) \equiv_r (M', \bar{b}) \quad \Longrightarrow \quad M \models \phi(\bar{a}) \;\; \textit{iff} \;\; M' \models \phi(\bar{b}). \tag{2}$$

For either notion of locality and every function $r : \mathbb{N} \to \mathbb{R}_{\geq 0}$, we call a formula $\phi(\bar{x})$ *$r(n)$-local* if there exists a constant n_ϕ such that $\phi(\bar{x})$ is $r(n)$-local with respect to all τ-structures of size $n \geq n_\phi$.

As for the relationship between the two notions of locality, there are two differences: (i) Hanf locality considers two structures that can be different, whereas Gaifman locality considers only one structure, and (ii) Hanf locality requires the existence of a global bijection, whereas Gaifman locality does not. Difference (i) makes Hanf locality a more powerful notion. In particular, Hanf locality is meaningful for sentences, whereas Gaifman locality for sentences trivially holds. When considering a *single* structure M, difference (ii) seems to make Hanf locality weaker than Gaifman locality but this is not the case (modulo a small loss in the distance parameter r). Intuitively, a global bijection can be constructed from an isomorphism between a pair of large-radius neighborhoods and the trivial global isomorphism between two identical structures in such a way that the isomorphism types up to some smaller distance are preserved. One can formalize this argument to show that if a formula is Hanf r-local w.r.t. (M, M) then it is Gaifman $(3r+1)$-local w.r.t. M [HLN99].

Circuit complexity Given a string of Boolean variables $x := x_1 x_2 \cdots x_m$, a Boolean *circuit* over x is a rooted directed acyclic graph whose vertices without outgoing edges are called *input gates* and are labeled either with 0, 1, x_i or $\neg x_i$ for some i, whose internal vertices are called *gates* and are labeled with either $\wedge$ or $\vee$, and whose root is called the *output gate*. A circuit naturally defines a function from $\{0,1\}^m$ to $\{0,1\}$. The *depth* of a circuit is the length of a longest path from the root of the circuit to one of its inputs. The *size* of a circuit is the number of gates it contains.

A *family of circuits* is a sequence $(C_m)_{m\in\mathbb{N}}$ such that for all $m \in \mathbb{N}$, C_m is a circuit over m input variables. We say that a language $L \subseteq \{0,1\}^*$ is accepted by a family of circuits $(C_m)_{m\in\mathbb{N}}$ if for all $m \in \mathbb{N}$ and for all binary strings w of length m, $C_m(w) = 1$ iff $w \in L$. A language L is in AC^0 if there exists a family of circuits accepting L that have constant depth and size polynomial in the input length.

Our locality bounds hinge on the well-known exponential size lower bounds for constant-depth circuits that compute parity [Ajt83, FSS84, Yao85, Hås86]. In fact, we use the following somewhat stronger promise version. For a binary string $w \in \{0,1\}^*$, let $|w|_1$ denote the number of $1s$ in w.

Theorem 2.3 (Implicit in [Hås86, Theorem 5.1]). *For any $d \in \mathbb{N}$, there are constants $\alpha > 0$ and $m_0 > 0$ such that for all $m \geq m_0$ there is no circuit of depth d and size $2^{\alpha m^{1/(d-1)}}$ that accepts all inputs $w \in \{0,1\}^{2m}$ with $|w|_1 = m$ and rejects all inputs with $|w|_1 = m+1$.*

Representing structures and queries as strings In order to enable circuits to act on structures and compute queries, we need to specify how to represent a τ-structure M and a k-tuple $\bar{a} \in dom(M)^k$ as a bit-string. Our results are robust with respect to the details of the encoding. For concreteness, we use the following scheme based on characteristic sequences.

Let $<$ be a linear order on $dom(M)$. Let $R_1, \ldots, R_s$ be a list of the relation symbols in τ and let $r_1, \ldots, r_s$ be the arities of these symbols. For each $R_i \in \tau$, we denote by $enc_<(R_i^M)$ the bit-string of length $|dom(M)|^{r_i}$ whose jth bit is 1 iff the jth smallest element in $dom(M)^{r_i}$ w.r.t. the lexicographic order associated with $<$ belongs to the relation R_i^M. Similarly, for each component a_i of the k-tuple $\bar{a}$, we let $enc_<(a_i)$ be the bit-string of length $|dom(M)|$ whose jth bit is 1 iff a_i is the jth smallest element of $dom(M)$ w.r.t. $<$. Finally, we let

$$enc_<(M, \bar{a}) \;:=\; enc_<(R_1^M) \cdots enc_<(R_s^M)\, enc_<(a_1) \cdots enc_<(a_k)$$

be the *binary encoding of* $(M, \bar{a})$ *w.r.t.* $<$.

The above encoding presumes a linear order $<$ on $dom(M)$. For ordered structures, i.e., structures with an associated order on their domain, the choice of $<$ is fixed. For unordered structures – the ones we care about – we consider all possible linear orders and let

$$\mathrm{Rep}(M, \bar{a}) \ := \ \{enc_<(M, \bar{a}) \ : < \text{ is a linear order on } dom(M)\}$$

denote the set of all binary encodings of $(M, \bar{a})$. Note that $\mathrm{Rep}(M, \bar{a}) = \mathrm{Rep}(M', \bar{b})$ iff $(M, \bar{a}) \cong (M', \bar{b})$.

For a circuit family $F = (C_m)_{m\in\mathbb{N}}$ to compute a k-ary query q on τ-structures, we require that it produces the correct result for all possible representations. In other words, for all τ-structures M, all k-tuples $\bar{a} \in dom(M)^k$, and all strings $\Gamma \in \mathrm{Rep}(M, \bar{a})$, we have that $C_{|\Gamma|}(\Gamma) = 1$ iff $\bar{a} \in q(M)$. Note that for every fixed M and $\bar{a}$, all representations in $\mathrm{Rep}(M, \bar{a})$ have the same length, so the same circuit of the family F acts on all of them.

3 Arb-Invariant First-Order Logic

In this section we introduce our notion of Arb-invariance and give a precise statement of the strong connection between Arb-invariant FO and the queries computable in AC^0.

Arb-invariance We fix an infinite schema σ_{arb}, containing a binary symbol $<$ together with a symbol for each numerical predicate (the "arb" in σ_{arb} comes from allowing arbitrary numerical predicates). For instance, σ_{arb} contains a symbol $+$ for addition, $*$ for multiplication, and so on. Each numerical predicate is implicitly associated, for every $n \in \mathbb{N}$, with a specific interpretation as a relation of the appropriate arity over the domain $[n] := \{1, 2, \ldots, n\}$. For instance, $+$ is associated with the classical relation of addition over $\mathbb{N}$ restricted to $[n]$. Conversely, for each such family of relations, σ_{arb} contains an associated predicate symbol.

Let M be a τ-structure and $n = |dom(M)|$. An *Arb-expansion* of M is a structure M' over the schema consisting of the disjoint union of τ and σ_{arb} such that $dom(M) = dom(M')$, M and M' agree on all relations in τ, and $<$ is interpreted as a linear order over $dom(M)$. This interpretation induces a bijection between $dom(M)$ and $[n]$, identifying each element of M' with its index relative to $<$. All the numerical predicates are then interpreted over $dom(M')$ via this bijection and their associated interpretation over $[n]$. For instance, $+$ is the ternary relation containing all tuples (a, b, c) of $dom(M')^3$ such that $i + j = k$, where a, b, and c are respectively the i^{th}, j^{th} and k^{th} elements of $dom(M')$ relative to $<$. Note that M' is completely determined by M and the choice of the linear order $<$ on $dom(M)$.

We denote by $\mathrm{FO}(\tau, \mathrm{Arb})$ the set of first-order formulas using the schema $\tau \cup \sigma_{\mathrm{arb}}$. A k-ary formula $\phi(\bar{x})$ of $\mathrm{FO}(\tau, \mathrm{Arb})$ is said to be *Arb-invariant with respect to a finite τ-structure M*, if for any k-tuple $\bar{a}$ of elements of M, and any two Arb-expansions M' and M'' of M we have

$$M' \models \phi(\bar{a}) \iff M'' \models \phi(\bar{a}). \tag{3}$$

When $\phi(\bar{x})$ is Arb-invariant with respect to all finite structures M over a schema, we simply say that $\phi(\bar{x})$ is *Arb-invariant.* Note that this is a semantic property which is not decidable (cf., e.g., [Lib04]).

A k-ary Arb-invariant formula defines a k-ary query over τ-structures as follows. When $\phi(\bar{x})$ is an Arb-invariant formula of $\mathrm{FO}(\tau, \mathrm{Arb})$ on M, we write $M \models \phi(\bar{a})$ whenever there is an Arb-expansion M' of M such that $M' \models \phi(\bar{a})$. Hence we view Arb-invariant formulas as formulas over τ-structures, and so we consider the Gaifman graph of M' to contain edges derived only from the relations in τ (i.e., $G(M') = G(M)$). We denote by *Arb-invariant* $\mathrm{FO}(\tau)$ the set of Arb-invariant formulas of $\mathrm{FO}(\tau, \mathrm{Arb})$, or simply *Arb-invariant* FO if τ is clear from the context. When the formula uses only the predicate $<$ of σ_{arb}, we have the classical notion of order-invariant FO (cf., e.g., [GS00, Lib04]).

Arb-invariance and AC^0 There is a strong connection between AC^0, $\mathrm{FO}(\tau, \mathrm{Arb})$, and Arb-invariant FO. For ordered structures, AC^0 and $\mathrm{FO}(\tau, \mathrm{Arb})$ are equivalent, i.e., they can describe exactly the same sets of bit-strings [Imm87]. This means that for every circuit family $F = (C_m)_{m\in\mathbb{N}}$ of constant depth and polynomial size there is a $\mathrm{FO}(\tau, \mathrm{Arb})$-sentence ϕ_F (over a schema τ that uses a unary relation specifying the positions of the string that carry the letter 1) that is satisfied by exactly those bit-strings that are accepted by F. And vice versa, for every $\mathrm{FO}(\tau, \mathrm{Arb})$-sentence ϕ there exists a corresponding AC^0 circuit family F_ϕ.

For unordered τ-structures, a query is computable in AC^0 iff it is definable in Arb-invariant $\mathrm{FO}(\tau, \mathrm{Arb})$. Recall that, by definition, a k-ary query q on τ-structures is computable in AC^0 iff there is a circuit family $(C_m)_{m\in\mathbb{N}}$ of constant depth and polynomial size such that for all τ-structures M, all $\bar{a} \in dom(M)^k$, and all $\Gamma \in \mathrm{Rep}(M, \bar{a})$: $C_{|\Gamma|}(\Gamma) = 1$ iff $\bar{a} \in q(M)$. We only need one direction of this equivalence, namely the one that is implied by the following theorem.

Theorem 3.1 (Implicit in [Imm87]). *For each k-ary $\mathrm{FO}(\tau, Arb)$ formula $\phi(\bar{x})$ with alternation depth d, there exists a family of depth-$(d+3)$ polynomial-size circuits $(C_m)_{m\in\mathbb{N}}$ such that for each τ-structure M, for each linear order $<$ on M and the Arb-expansion M' of M induced by $<$, for each tuple $\bar{a} \in dom(M)^k$, and for the string $\Gamma = enc_<(M, \bar{a})$,*

$$C_{|\Gamma|}(\Gamma) = 1 \quad \Longleftrightarrow \quad M' \models \phi(\bar{a}).$$

Note that for a circuit family $F = (C_m)_{m\in\mathbb{N}}$ to compute the query defined by the k-ary formula ϕ over τ-structures, it has to be the case that for all τ-structures M and all $\bar{a} \in dom(M)^k$, $C_{|\Gamma|}(\Gamma)$ is the same for every $\Gamma \in \mathrm{Rep}(M, \bar{a})$. The latter condition exactly corresponds to the formula ϕ in Theorem 3.1 being Arb-invariant.

4 Gaifman Locality

We now prove the main result of the paper – the upper bound in Theorem 1.1. Recall, our theorem claims that every Arb-invariant FO formula is Gaifman $(\log n)^c$-local, for some constant c which depends only on the formula. In fact, we prove the following slightly stronger version.

Theorem 4.1. *For each $\mathrm{FO}(\tau, Arb)$ formula $\phi(\bar{x})$ with alternation depth d and any constant $c > d+2$, there exists a constant $n_{\phi,c}$ such that if $\phi(\bar{x})$ is Arb-invariant with respect to a τ-structure M with $n := |M| \geq n_{\phi,c}$, then $\phi(\bar{x})$ is Gaifman $(\log n)^c$-local with respect to M.*

We now briefly sketch the overall proof of Theorem 4.1. Suppose we have two tuples, $\bar{a}$ and $\bar{b}$, on a τ-structure M, with domain size n, such that their r-neighborhoods, $\mathcal{N}_r^M(\bar{a})$ and $\mathcal{N}_r^M(\bar{b})$, are

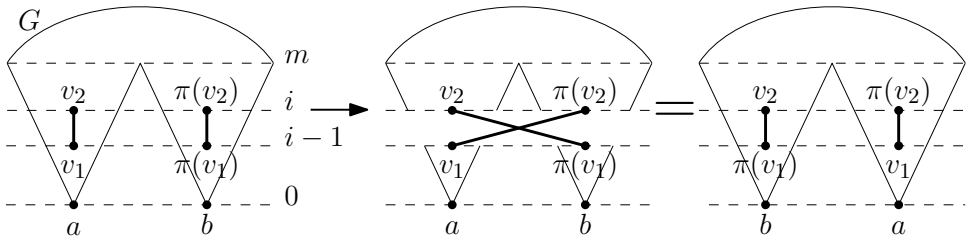

Figure 1: Diagram for swapping the neighborhoods of a and b of radius i, conditioned on $w_i = 1$.

isomorphic (for some big enough r). Further suppose that there is an $\mathrm{FO}(\tau, \mathrm{Arb})$ formula $\phi(\bar{x})$ which is able to distinguish between $\bar{a}$ and $\bar{b}$ on M while being Arb-invariant with respect to M. Using the link between Arb-invariant $\mathrm{FO}(\tau, \mathrm{Arb})$ formulas and AC^0 circuits from Theorem 3.1, we can view the formula $\phi(\bar{x})$ as a small constant-depth circuit C.

Using the hypothesis that $\phi(\bar{x})$ is Arb-invariant and distinguishes between $\bar{a}$ and $\bar{b}$ on M, we can construct from the circuit C and structure M another circuit $\tilde{C}$ that for a $(2m)$-length binary string w distinguishes between the cases where w contains m occurrences of 1 and $m+1$ occurrences, for some m depending on r. This is the *key step* in our argument. If this happens for large enough m, we get a small circuit computing the promise problem described in Theorem 2.3. We can argue that $\tilde{C}$ has size polynomial in n and depth a constant d' depending only on the alternation depth of $\phi(\bar{x})$. Therefore, if $m > b(\log n)^{d'-1}$ for a large enough constant b, the circuit $\tilde{C}$ we construct violates Theorem 2.3, hence $\phi(\bar{x})$ cannot distinguish between tuples which have isomorphic r-neighborhoods. Our construction is such that m is linearly related to r and therefore $\phi(\bar{x})$ is Gaifman $(\log n)^c$-local for any constant $c > d' - 1$ and sufficiently large n.

4.1 Upper Bound for Unary Formulas

In this subsection we consider only unary FO formulas $\phi(x)$. For didactic reasons we first assume that the r-neighborhoods of the elements a and b are disjoint. We argue that we can perform the key step in this setting, and consider the general unary case afterward.

For clarity we describe the intuition with respect to structures that are graphs. Let M be a graph $G = (V, E)$ and take two vertices $a, b \in V$ such that $\pi : \mathcal{N}_r^G(a) \cong \mathcal{N}_r^G(b)$. Suppose, for the sake of contradiction, that there is a unary FO formula $\phi(x)$ which is Arb-invariant with respect to G and such that $G \models \phi(a) \wedge \neg\phi(b)$. Applying Theorem 3.1 to ϕ gives us a circuit C which, for any vertex $c \in V$, outputs the same value for all strings in $\mathrm{Rep}(G, c)$, and distinguishes $\mathrm{Rep}(G, a)$ from $\mathrm{Rep}(G, b)$.

4.1.1 Disjoint Neighborhoods

Let us assume that $\mathcal{N}_r^G(a) \cap \mathcal{N}_r^G(b) = \emptyset$. In this setting it turns out we can pick $m = r$. The neighborhood isomorphism, $\pi : \mathcal{N}_r^G(a) \cong \mathcal{N}_r^G(b)$, implies that the balls of radius $i < r$ around a and b are isomorphic and disjoint in G. Consider the following procedure, depicted in Figure 1. For some $i \in [m]$, cut all the edges linking nodes at distance $i-1$ from a or b to nodes at distance i. Now, swap the positions of the $(i-1)$-neighborhoods around a and b and reconnect the edges in

a way that respects the isomorphism π. The resulting graph is isomorphic to G, but the relative positions of a and b have swapped.

Using this intuition we construct a new graph G_w from G, a, and b that depends on a string of m Boolean variables $w := w_1 w_2 \cdots w_m$. We construct G_w so that for each variable w_i, we swap the relative positions of the $(i-1)$-radius balls around a and b iff w_i is 1. The number of such swaps is $|w|_1$. The m-neighborhood isomorphism between a and b implies that $G_w \cong G$. When $|w|_1$ is even, $(G_w, a) \cong (G, a)$, and when $|w|_1$ is odd, $(G_w, a) \cong (G, b)$.

Using the above construction of G_w we derive a circuit $\tilde{C}$ from C that computes parity on m bits. The circuit $\tilde{C}$ first computes a representation $\Gamma_w \in \mathrm{Rep}(G_w, a)$, and then simulates C on input Γ_w. The above distinguishing property then implies that $\tilde{C}$ accept m-bit strings with even parity. To construct Γ_w we start with a fixed string in $\mathrm{Rep}(G, a)$ and transform it into an element of $\mathrm{Rep}(G_w, a)$ by modifying the edges to switch between the shells in the manner suggested above. Observe that the presence of each edge in G_w depends on at most a single bit of w. This property implies that Γ_w consists of constants, and variables in w or their negations. This means that $\tilde{C}$ is no larger or deeper than C.

We formalize this intuition for general structures and obtain the following lemma.

Lemma 4.2. *Let $m \in \mathbb{N}$. Let M be a τ-structure. Let $a, b \in dom(M)$ such that $dist^M(a,b) > 2m$ and $\mathcal{N}_m^M(a) \cong \mathcal{N}_m^M(b)$. Let C be a circuit that accepts all strings in $\mathrm{Rep}(M,a)$, and rejects all strings in $\mathrm{Rep}(M,b)$. There is a circuit $\tilde{C}$ with the same size and depth as C that computes parity on m bits.*

Proof. For every $r \in \mathbb{N}$ and $a \in dom(M)$, the *r-shell around a in M* is the set

$$S_r^M(a) \quad := \quad \{v \in dom(M) : dist^M(a,v) = r\}.$$

Let π be an isomorphism from $\mathcal{N}_m^M(a)$ to $\mathcal{N}_m^M(b)$. Extend π to take $\mathcal{N}_m^M(b)$ back to $\mathcal{N}_m^M(a)$ (that is, extend the domain of the map to the elements of $N_m^M(b)$ and act as π^{-1} for those elements). This is well-defined because $dist^M(a,b) > 2m$ and the m-neighborhoods are disjoint. Note that, in particular, $\pi(a) = b$, and for all $i \in [m]$, $\pi\big(S_i^M(a)\big) = S_i^M(b)$. Let $S_i := S_i^M(a) \cup S_i^M(b)$ for $i \leq m$. Note, that $S_0 = \{a, b\}$.

Let $w := w_1 w_2 \cdots w_m$ be a string of m Boolean variables. We design a structure M_w that has the following property:

$$\begin{array}{c} \text{If } |w|_1 \text{ is even, then } (M_w, a) \cong (M, a). \\ \text{If } |w|_1 \text{ is odd, then } (M_w, a) \cong (M, b). \end{array}$$

When $|w|_1$ is even, $C(M_w, a)$ accepts, because $(M_w, a) \cong (M, a)$. Similarly, when $|w|_1$ is odd $C(M_w, a)$ rejects because $(M_w, a) \cong (M, b)$. Thus, the above property is sufficient to claim that $C(M_w, a)$ accepts iff the parity of w is even. We show how to construct M_w.

Consider a tuple $\bar{v} = (v_1, ..., v_k)$ that belongs to a relation R^M, for some symbol $R \in \tau$ of arity k, that intersects the shells S_{i-1} and S_i for some $i \in [m]$. (Note that other tuples are wholly contained in single shells, because the pairwise distances between the elements of $\bar{v}$ are at most one.) For clarity we reorder the components of $\bar{v}$ so that $\bar{v} := (\bar{v}_1, \bar{v}_2)$ where $\bar{v}_1 \subseteq S_{i-1}$ and $\bar{v}_2 \subseteq S_i$. Each tuple $\bar{v}$ of this type in R^M induces a set of two potential tuples in R^{M_w}: $\bar{v}$ and $(\bar{v}_1, \pi(\bar{v}_2))$. If $w_i = 0$ we copy the tuple $\bar{v}$ from R^M into R^{M_w}; if $w_i = 1$ we add the tuple $(\bar{v}_1, \pi(\bar{v}_2))$ to R^{M_w}.

Observe that, by construction, for any $i \in [m]$, when $w_i = 1$, each tuple intersecting $S_{i-1}^M(a)$ and $S_i^M(a)$ is replaced with a tuple intersecting $S_{i-1}^M(a)$ and $\pi(S_i^M(a)) = S_i^M(b)$, and each tuple intersecting $S_{i-1}^M(b)$ and $S_i^M(b)$ is replaced with a tuple intersecting $S_{i-1}^M(b)$ and $\pi(S_i^M(b)) = S_i^M(a)$. Note this is general because the m-neighborhoods around a and b are disjoint (hence, "cross tuples" are not present in M). Further, for every i where $w_i = 1$, the construction interchanges the roles of the elements in $N_{i-1}^M(a)$ with their images under π in $N_{i-1}^M(b)$. This implies the relative positions of the elements a and b themselves are swapped once for each bit of w that is one. This argument also implies that $M_w \cong M$. Therefore, when the parity of w is odd $(M_w, a) \cong (M, b)$ because a and b swap positions an odd number of times. When the parity of w is even $(M_w, a) \cong (M, a)$ by the same token. This is the property claimed. We conclude the proof by constructing a Boolean circuit $\tilde{C}$, using M_w and C, which computes parity.

Fix an arbitrary string $\Gamma \in \mathrm{Rep}(M, a)$. We derive a new input string Γ_w, with $|\Gamma_w| = |\Gamma|$, from w and Γ. We want Γ_w to be a binary representation of the structure M_w paired with the element a. With this in mind, we copy the encoding of the distinguished element a from Γ into Γ_w. It remains to determine the encoding of M_w in Γ_w.

For each $i \in [m]$, each relation $R \in \tau$, and each tuple $\bar{v} \in R^M$ crossing between shells S_{i-1} and S_i, we encode the corresponding tuple in Γ_w in the following way: Set the bit of Γ_w corresponding to the tuple $\bar{v}$ and relation R to $\neg w_i$, and the bit of Γ_w corresponding to the tuple $(\bar{v}_1, \pi(\bar{v}_2))$ and relation R to w_i. That is, we modify the encoding so that $\bar{v} \in R^{M_w}$ when $w_i = 0$ and $(\bar{v}_1, \pi(\bar{v}_2)) \in R^{M_w}$ when $w_i = 1$. For all other bits in Γ specifying relations, we copy them verbatim from Γ into Γ_w. Observe that the bits of Γ_w are drawn from $\{0, 1, w_i, \neg w_i\}$. This completes the construction of $\Gamma_w \in \mathrm{Rep}(M_w, a)$.

Finally, define the circuit $\tilde{C}(w) := C(\Gamma_w)$. Observe that $\tilde{C}$ is an m-input circuit that has size and depth no more than C because each component of Γ_w is either a constant or a literal of w. □

4.1.2 General Neighborhoods

We now develop the transformation corresponding to Lemma 4.2 for the general unary case, where the r-neighborhoods around a and b may overlap. As before, we describe the intuition in terms of structures that are graphs.

Consider the iterated application of the isomorphism π to a. We distinguish between two cases. The first case occurs when this iteration travels far from a. That is, for some $t \in \mathbb{N}$, $\pi^t(a)$ is a point c that is far from a. Suppose r is large enough that the isomorphism π implies that a large neighborhood around c is isomorphic to the neighborhood around a. By the triangle inequality, since a is far from c, either (i) b is far from a, or else (ii) c is far from a and b (see Figure 2(i),(ii)). We claim that in each case there is a pair of vertices that are distinguished by C, and whose neighborhoods are isomorphic and disjoint. In case (i), a and b are such a pair; in case (ii), C must distinguish either a and c, or b and c, so a and c, or b and c form such a pair. For this pair of vertices, we are in the disjoint case and Lemma 4.2 can be applied to produce a small circuit that computes parity.

The other case occurs when the iterated application of π to a stays close to a (and b). Let S_0 be the orbit of a under π (i.e., $S_0 := \{\pi^z(a) | z \in \mathbb{N}\}$) (see Figure 2(iii)), and let S_i be the vertices at distance i from S_0, for $i \in [2m]$. Because $\pi(S_0) = S_0$ and π is a partial isomorphism on G, the shells S_i are closed under π.

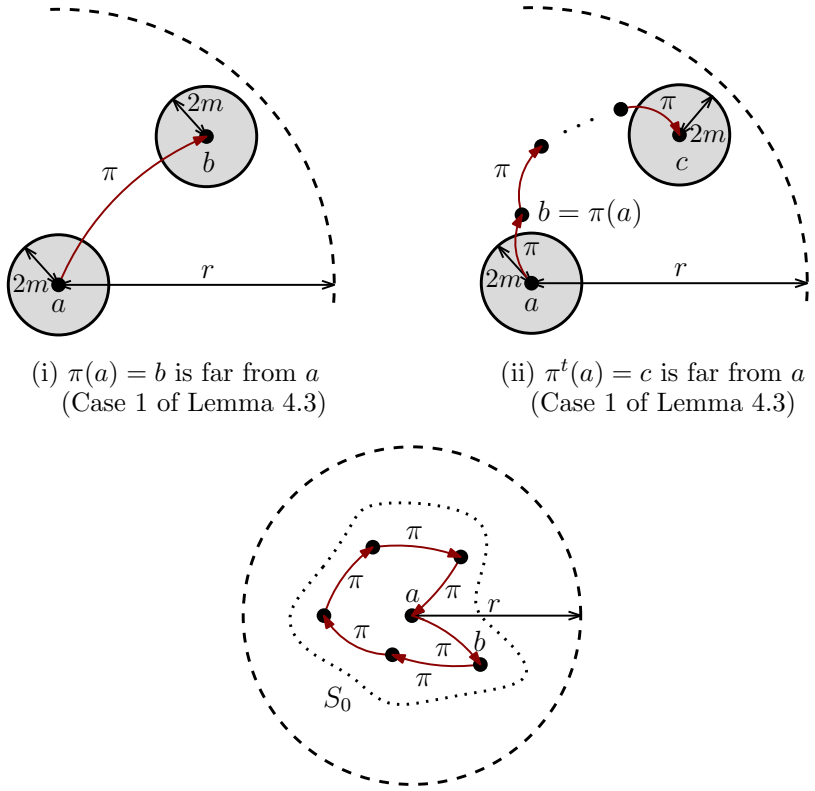

Figure 2: Diagram for the general unary case. (r is the radius of the domain of π.)

We now play a game similar to the disjoint case. Consider the following procedure, depicted in Figure 3. For some $i \in [2m]$ cut all edges between the shells S_{i-1} and S_i. "Rotate" the radius $i-1$ ball around S_0 by π relative to S_i, and reconnect the edges. Because the shells are closed under π, the resulting graph is isomorphic to G. Further, the positions of a and b have shifted relative to an application of π.

As before, we encode this behavior into a modified graph G_w depending on a string of $2m$ Boolean variables $w := w_1 w_2 \cdots w_{2m}$. When $w_i = 0$, we preserve the edges between the shells S_{i-1} and S_i. When $w_i = 1$ we rotate the edges by π. That is, an edge $(v_1, v_2) \in (S_{i-1} \times S_i) \cap E$ becomes the edge $(v_1, \pi(v_2))$ in G_w. The neighborhood isomorphism between a and b implies that $G \cong G_w$. We can argue that

$$(G_w, a) \cong (G, \pi^{|w|_1}(a)). \tag{4}$$

We define the circuit $\tilde{C}$ to simulate C on an input $\Gamma_w \in \mathrm{Rep}(G_w, a)$. The above distinguishing property implies that $\tilde{C}$ distinguishes between $|w|_1 \equiv 0 \bmod |S_0|$ and $|w|_1 \equiv 1 \bmod |S_0|$. (Note, this is non-trivial because $|S_0| \geq 2$ since a and b are distinct and in S_0.) This is not quite the promise problem defined in Theorem 2.3. For this reason we modify the construction to shift a by m applications of π^{-1} in Γ_w. This means that $\Gamma_w \in \mathrm{Rep}(G_w, \pi^{-m}(a))$ and $\tilde{C}$ can distinguish between

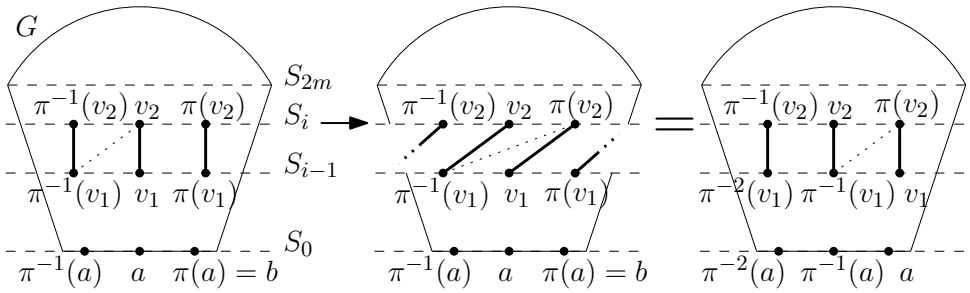

Figure 3: Diagram for rotating the shell of radius i around S_0 when $w_i = 1$.

$|w|_1 \equiv m \bmod |S_0|$ and $|w|_1 \equiv m+1 \bmod |S_0|$. This is ruled out by Theorem 2.3, completing the argument.

For general structures, the idea is formalized in the following lemma, where we achieve $r = 10m$.

Lemma 4.3. *Let $m \in \mathbb{N}$. Let M be a τ-structure. Let $a, b \in dom(M)$ such that $\mathcal{N}^M_{10m}(a) \cong \mathcal{N}^M_{10m}(b)$. Let C be a circuit that accepts all strings in $\mathrm{Rep}(M, a)$ and rejects all strings in $\mathrm{Rep}(M, b)$, and for each $c \in dom(M)$, C has the same output for each string in $\mathrm{Rep}(M, c)$. There is a circuit $\tilde{C}$ with the same size and depth as C that distinguishes $|w|_1 = m$ and $|w|_1 = m+1$ for $w \in \{0,1\}^{2m}$.*

Proof. Let π be an isomorphism between $\mathcal{N}^M_{10m}(a)$ and $\mathcal{N}^M_{10m}(b)$. There are two cases:

Case 1. *The iterated isomorphism takes a far from a.*

More specifically, there exists $t \in \mathbb{N}$ such that

$$dist^M(a, \pi^t(a)) > 8m. \tag{5}$$

Let t be the minimal value such that (5) holds. Let $c := \pi^t(a)$. Hence $dist^M(a, c) > 8m$. Since t is minimal, $dist^M(a, \pi^j(a)) \leq 8m$ for all $j < t$. Because the isomorphism π preserves neighborhoods contained in $N^M_{10m}(a)$, it follows that for all $j < t$, $N^M_{2m}(\pi^j(a)) \subseteq N^M_{8m+2m}(a)$, and π induces an isomorphism from $\mathcal{N}^M_{2m}(\pi^j(a))$ to $\mathcal{N}^M_{2m}(\pi^{j+1}(a))$. This implies that $\mathcal{N}^M_{2m}(a) \cong \mathcal{N}^M_{2m}(b) \cong \mathcal{N}^M_{2m}(c)$.

As $dist^M(a, c) > 8m$, the triangle inequality implies that either $dist^M(a, b) > 4m$ or $dist^M(b, c) > 4m$. In the former case, we complete by applying Lemma 4.2 and observing that parity on $2m$ bits distinguishes between inputs with m ones and inputs with $m+1$ ones. In the latter case, depending on whether C accepts $\mathrm{Rep}(M, c)$ or not, we can proceed either with the pair b and c or the pair a and c. In either case, from the above we see that this pair of points have isomorphic $2m$-neighborhoods and are more than $4m$ apart. Therefore Lemma 4.2 again suffices to reach the required conclusion.

Case 2. *The iterated isomorphism keeps a close to a.*

More specifically, for all $t \in \mathbb{N}$, $dist^M(a, \pi^t(a)) \leq 8m$.

Let $S_0 \subseteq dom(M)$ be the orbit of a under π. Note that $\pi(S_0) = S_0$ and $b \in S_0$. We define S_i as the set of elements of M at distance i from S_0. Because π is an isomorphism from $\mathcal{N}^M_{10m}(a)$ to $\mathcal{N}^M_{10m}(b)$, each S_i is also closed under π for $i \leq 2m$.

Let $w := w_1 w_2 \cdots w_{2m}$ be a string of $2m$ Boolean variables. We proceed similarly to the proof of Lemma 4.2 when constructing a structure M_w and representation Γ_w. We construct M_w and a distinguished element a' so that the following property holds:

$$\text{If } |w|_1 \equiv m \bmod |S_0|, \text{ then } (M_w, a') \cong (M, a).$$
$$\text{If } |w|_1 \equiv m+1 \bmod |S_0|, \text{ then } (M_w, a') \cong (M, b).$$

When $|w|_1 = m$, $C(M_w, a')$ accepts, because $(M_w, a') \cong (M, a)$. Similarly, when $|w|_1 = m+1$, $C(M_w, a')$ rejects, because $(M_w, a') \cong (M, b)$. This property is sufficient to claim that $C(M_w, a')$ distinguishes between $|w|_1 = m$ and $|w|_1 = m+1$, because $|S_0| > 1$ since a and b are distinct and in S_0. We now show how to construct M_w and a'.

Let the structure M_i be the result of performing the construction from Lemma 4.2 only for the tuples intersecting shells S_j and S_{j+1}, for all $j < i$. Note $M_0 = M$ and $(M_0, v) = (M, v)$ for all $v \in S_0$. When $w_i = 0$, we have $M_{i-1} = M_i$, hence $(M_{i-1}, v) = (M_i, v)$ for any $v \in S_0$, because the construction makes no modifications to the structure between shell S_{i-1} and S_i in this case. When $w_i = 1$, we are rotating the neighborhood below shell S_i by π^{-1} relative to S_i. Since the shells are closed under the action of π, we can conclude that $M_i \cong M_{i-1}$ for $i \in [2m]$. This also implies that when $w_i = 1$, $(M_i, v) \cong (M_{i-1}, \pi(v))$. It follows that for any $v \in S_0$ and $i \in [2m]$: $(M_i, v) \cong (M_{i-1}, \pi^{w_i}(v))$. By applying this fact $2m$ times we reach the conclusion that for all $v \in S_0$,

$$(M_w, v) := (M_{2m}, v) \cong (M_0, \pi^{|w|_1}(v)) = (M, \pi^{|w|_1}(v)).$$

The length of the orbit of a with respect to π is $|S_0|$. Define $a' := \pi^{-m}(a)$. Since $a' \in S_0$, it follows that when $|w|_1 \equiv 0 \bmod |S_0|$, $(M_w, a') \cong (M, a')$ and when $|w|_1 \equiv 1 \bmod |S_0|$, $(M_w, a') \cong (M, \pi(a'))$. Observe this implies that when $|w|_1 \equiv m \bmod |S_0|$, $(M_w, a') \cong (M, \pi^m(\pi^{-m}(a))) = (M, a)$ and when $|w|_1 \equiv m+1 \bmod |S_0|$, $(M_w, a') \cong (M, \pi^{m+1}(\pi^{-m}(a))) = (M, b)$. This is the property claimed. It remains to construct the circuit $\tilde{C}$.

We construct the string Γ_w in the same way as in the proof of Lemma 4.2 with respect to M, w, π, and the shells $\{S_i\}_{i \leq 2m}$ defined above. Note that in the case where the construction would assign both w_i and $\neg w_i$ to a bit of Γ_w corresponding to some tuple in a relation of M_w we instead set the corresponding bit of Γ_w to 1. The string Γ_w represents the pairing of the structure M_w with the element a. Form Γ'_w from Γ_w by replacing the encoding of distinguished element a with an encoding of a'. Note that $\Gamma'_w \in \mathrm{Rep}(M_w, a')$. Setting $\tilde{C}(w) := C(\Gamma'_w)$ completes the proof. □

Notice that the idea behind the proof of this lemma is quite similar to the disjoint case. When the neighborhoods are disjoint, the above construction gives $S_0 = \{a, b\}$. In this case the "rotation" by π becomes a swap. Further, since $|S_0| = 2$, the promise problem we solve is distinguishing between $|w|_1 \equiv m \bmod 2$, and $|w|_1 \equiv m+1 \bmod 2$ – this is exactly parity! Thus, in the case of disjoint neighborhoods, the construction in the proof of Lemma 4.3 reduces to the one from Lemma 4.2.

With Lemma 4.3 in hand, we are ready to finish the proof of Theorem 4.1 in the unary case.

Proof of Theorem 4.1 for the case $k = 1$. Assume that $\phi(x)$ is a unary formula of $\mathrm{FO}(\tau, \mathrm{Arb})$ with alternation depth d that is Arb-invariant with respect to a τ-structure M with $n := |M|$. Since $\phi(x)$ is $\mathrm{FO}(\tau, \mathrm{Arb})$, it is computable by a family of circuits in AC^0 (cf. Theorem 3.1). That is, there are a constant β and a circuit C with depth $d+3$ and size n^β such that C computes $\phi(x)$

on size n τ-structures. Since $\phi(x)$ is Arb-invariant with respect to M, for each fixed $a \in dom(M)$, C has the same output for all strings in $\mathrm{Rep}(M, a)$.

Now, for the sake of contradiction, suppose $\phi(x)$ is *not* Gaifman $(\log n)^c$-local with respect to M, for some constant $c > d+2$.This implies that $\phi(x)$ distinguishes between two elements $a, b \in dom(M)$ having isomorphic $(\log n)^c$-neighborhoods.

Let $m := \lfloor \frac{(\log n)^c}{10} \rfloor$. Therefore $\mathcal{N}^M_{10m}(a) \cong \mathcal{N}^M_{10m}(b)$. The circuit C then satisfies the assumptions of Lemma 4.3. From the lemma, we obtain a circuit $\tilde{C}$ of depth $d+3$ and size n^β that distinguishes between $|w|_1 = m$ and $|w|_1 = m+1$ for $w \in \{0,1\}^{2m}$.

From Theorem 2.3 we obtain that $n^\beta > 2^{\alpha m^{1/(d+3-1)}}$, which is equivalent to $\beta \log n > \alpha m^{1/(d+2)}$. The latter condition is violated if we set $m = (\log n)^c$ whenever c is a constant larger than $d+2$ and n is sufficiently large (depending on ϕ and c). This yields the required contradiction, completing the proof. None

4.2 Reducing the Arity

To argue Theorem 4.1 in the case of formulas with an arbitrary number of free variables, we prove the following reduction. Given a k-ary FO(Arb) formula ϕ that is Arb-invariant with respect to the structure M and distinguishes two k-tuples $\bar{a}$ and $\bar{b}$ that have isomorphic r-neighborhoods, we produce, for some $k' < k$, a k'-ary FO(Arb) formula ϕ' that is Arb-invariant with respect to an extended structure M' and distinguishes between two k'-tuples $\bar{a}'$ and $\bar{b}'$ that have isomorphic r'-neighborhoods. Furthermore, r' is only slightly smaller than r.

Repeated application of this idea transforms a distinguishing k-ary formula into a distinguishing unary formula with slightly weaker parameters. For large enough initial radius r this is sufficient to contradict the Gaifman locality of unary formulas.

We first give an intuitive description of the reduction argument. Let $\phi(\bar{x})$ be a k-ary formula as above, and let $\pi : \mathcal{N}^M_r(\bar{a}) \cong \mathcal{N}^M_r(\bar{b})$ denote a neighborhood isomorphism. The main idea is to transfer some of the information present in the initial k-tuples $\bar{a}$ and $\bar{b}$ into a new marking relation R such that we can recover $\bar{a}$ and $\bar{b}$ from $k' < k$ of their components $\bar{a}'$ and $\bar{b}'$ as extensions of $\bar{a}'$ and $\bar{b}'$ that satisfy R. In that case, the formula

$$\phi'(\bar{y}) = (\exists \bar{z})\, R \wedge \phi(\bar{y}, \bar{z}) \tag{6}$$

has arity $k' < k$, and distinguishes the tuples $\bar{a}'$ and $\bar{b}'$ over the extension M' of M with R, where R is a relation of arity at most k evaluated over some of the variables in $\bar{y}$ and $\bar{z}$. The formula ϕ' is Arb-invariant over M' since ϕ is Arb-invariant over M, the domain of M' is the same as of M, and R does not use the Arb relations. Moreover, provided the marking relation R is invariant under π, π also induces a neighborhood isomorphism $\mathcal{N}^{M'}_{r'}(\bar{a}') \cong \mathcal{N}^{M'}_{r'}(\bar{b}')$ in the new structure, albeit possibly for a smaller radius r', e.g., due to the effect of the introduction of R on the Gaifman graph.

We start by considering three situations in which it is relatively simple to obtain a π-invariant marking relation R, and then see how to handle the remaining case. Throughout, we assume without loss of generality that $\bar{a}$ is accepted by ϕ and $\bar{b}$ is rejected by ϕ, and we use the notation $\bar{a} := (a_1, a_2, \ldots, a_k)$ and $\bar{b} := (b_1, b_2, \ldots, b_k)$.

Case 1. *The tuples $\bar{a}$ and $\bar{b}$ have a component in common.*

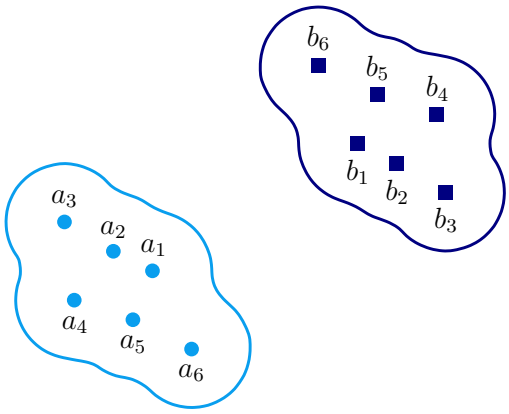

Figure 4: Diagram for Case 3.

Say $a_k = b_k$. Then the fact that ϕ distinguishes $\bar{a}$ and $\bar{b}$ is independent of the last component of the tuples. To exploit this redundancy we mark the element a_k and derive a $(k-1)$-ary formula $\phi'(\bar{y})$ as in (6), where R checks whether $\bar{z}$ equals a_k. Since $a_k = b_k$ and π has to map a_k to b_k, a_k is a fixed point of π, which guarantees that the marking relation is invariant under π. In this case the original isomorphism π remains a neighborhood isomorphism with the same radius $r' = r$.

Case 2. *The elements in the orbit of $\bar{a}$ stay well within the isomorphism neighborhood of $\bar{a}$.*

The *orbit* of $\bar{a}$ is the set of tuples $\pi^t(\bar{a})$ for all $t \in \mathbb{N}$. We now use the relation R to mark all tuples in the orbit of $\bar{a}$. The relation R is π-invariant because the entire orbit stays within the domain of the neighborhood isomorphism π.

This marking allows us to reduce the arity as follows. First, note that we can apply Case 1 whenever there is a pair of marked tuples that is distinguished by ϕ and has a component in common. Therefore, we can assume that ϕ does not distinguish any marked tuples that share a component.

This observation allows us to recover the tuples $\bar{a}$ and $\bar{b}$ (or equivalent ones) from their first components only. Let $\bar{a}' := (a_1)$ and $\bar{b}' := (b_1)$. Since ϕ accepts $\bar{a}$ and $\bar{a}$ is marked, existentially guessing the remaining components consistent with $\bar{a}$ and R shows that ϕ' defined by (6) accepts $\bar{a}'$. On the other hand, since ϕ rejects $\bar{b}$, ϕ rejects all marked tuples that share the first component with $\bar{b}$. Therefore, no marked tuples with b_1 as first component are accepted by ϕ, and hence ϕ' rejects $\bar{b}'$. This establishes the required distinguishing property of ϕ'.

We already mentioned that the marking relation R is π-invariant. The fact that the entire orbit of $\bar{a}$ stays *well within* the isomorphism neighborhood guarantees that the new isomorphism radius r' is not too much smaller than the original radius r.

Case 3. *All components of $\bar{a}$ are close to each other.*

Because of Case 2, we only need to consider the situation where the iterates of π take some component of $\bar{a}$ far from $\bar{a}$. Without loss of generality we can assume that $\bar{b}$ has a component that is far from $\bar{a}$. Also, since π preserves distances, we know that all components of $\bar{b}$ are close to each other. This allows us to choose a relatively large $r' \leq r$ such that the r'-neighborhoods of $\bar{a}$ and $\bar{b}$ do not intersect. So, the situation is as sketched in Figure 4.

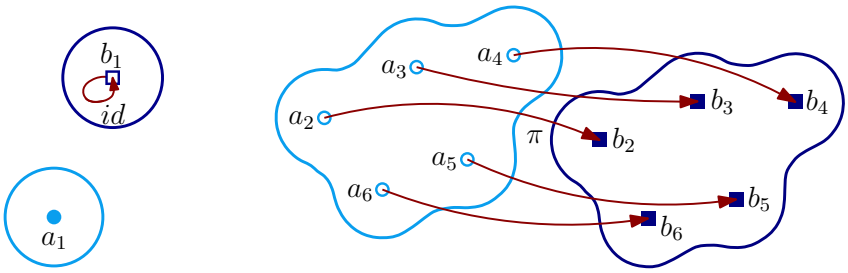

Figure 5: Diagram for the hybrid isomorphism from Case 4 that maps $\bar{h} = (b_1, a_2, \ldots, a_k)$ to $\bar{b} = (b_1, b_2, \ldots, b_k)$.

In this case, simply marking the tuples $\bar{a}$ and $\bar{b}$ yields a relation R that is invariant under π on $N^M_{r'}(\bar{a})$ for the relatively large radius r'. The π-invariance follows from the fact that $\bar{a}$ and $\bar{b} = \pi(\bar{a})$ are far apart, as the range of π on $N^M_{r'}(\bar{a})$ falls entirely outside of $N^M_{r'}(\bar{a})$, so no tuple other than $\bar{a}$ needs to be marked in $N^M_{r'}(\bar{a})$ in order for the marking to be π-invariant.

With this marking, knowledge of one component of $\bar{a}$ and $\bar{b}$ suffices to recover the full tuples, so we can reduce the arity to $k' = 1$ following (6).

Case 4. *Hybrid case.*

In the remaining case we can assume without loss of generality that some component of $\bar{b}$, say b_1, is far from $\bar{a}$, and that a_1 is far from some other component of $\bar{a}$. Due to the isomorphism π, the latter is equivalent to b_1 being far from some other component of $\bar{b}$.

In this case, we do not know how to apply the π-invariant marking strategy to the given tuples $\bar{a}$ and $\bar{b}$. However, we can construct a "hybrid" tuple $\bar{h}$ that has some components in common with $\bar{a}$ and some with $\bar{b}$ such that Case 1 applies to either $\bar{a}$ and $\bar{h}$, or to $\bar{h}$ and $\bar{b}$.

For simplicity, let us first consider the situation where b_1 is far from *all* other components of $\bar{b}$. Recall that b_1 is also far from $\bar{a}$. These two facts imply that for a large radius the neighborhood around $\bar{b}$ is isomorphic to the neighborhood around the tuple $\bar{h} := (b_1, a_2, \ldots, a_k)$. To see this, consider the map which acts as the identity map on the elements near b_1 and acts as π on the elements near $a_2, \ldots, a_k$. Figure 5 illustrates the construction. The distance between b_1, and both $\bar{a}$ and the rest of $\bar{b}$ ensures that no tuples that are in a relation of M straddle the neighborhood of b_1 as well as the neighborhood of some other component. Therefore, on such a tuple our map either acts like the identity on all components, or like π on all components. Since both the identity and π preserve the relations of M, so does our map.

By transitivity, we also have a neighborhood isomorphism between $\bar{a}$ and $\bar{h}$. We also know that ϕ distinguishes $\bar{h}$ from one of $\bar{a}$ and $\bar{b}$ because those tuples are themselves distinguished by ϕ. Thus, there is some pair of tuples which have large isomorphic neighborhoods, are distinguished by ϕ, and share components (since $\bar{h}$ is a hybrid of $\bar{a}$ and $\bar{b}$). We conclude by applying Case 1 to reduce the arity of the formula, while only slightly decreasing the radius of the neighborhood isomorphism.

Finally, consider the case where b_1 is close to some components of $\bar{b}$ and far from others. We iteratively group the components of $\bar{b}$ closest to b_1 to form the set $\bar{b}_I$, until all remaining components are far from $\bar{b}_I$. So, by construction the elements of $\bar{b}_I$ are far from the other components of $\bar{b}$ and far from $\bar{a}$, since b_1 is far from $\bar{a}$. Viewing the component $\bar{b}_I$ as a single element allows us to apply

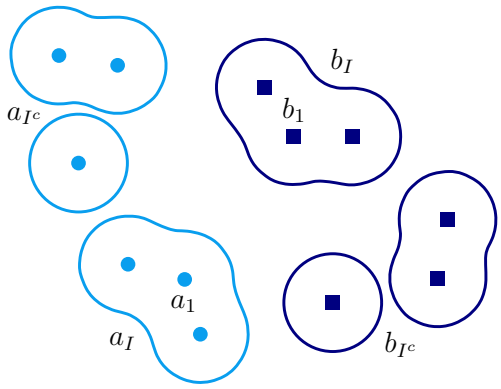

Figure 6: Diagram for Case 4 when b_1 is close to some component of $\bar{b}$.

the argument from the previous paragraph to reduce the instance, albeit with some further loss in the isomorphism radius. This loss is caused by the distortion in distance from grouping elements in this way. See Figure 6 for a diagram of this case.

These ideas are formalized in the proof of the following lemma.

Lemma 4.4. *Let $k, d, r \in \mathbb{N}$ and τ be a schema. Let M be a τ-structure with tuples $\bar{a}, \bar{b} \in dom(M)^k$. Let $\phi(\bar{x})$ be a k-ary $\mathrm{FO}(\tau, Arb)$ formula with alternation depth $d > 0$ which is Arb-invariant with respect to M. Suppose:*

1. *$M \models \phi(\bar{a}) \wedge \neg\phi(\bar{b})$, and*
2. *$\pi : \mathcal{N}_r^M(\bar{a}) \cong \mathcal{N}_r^M(\bar{b})$.*

There is a $k' < k$, a schema $\tau' \supseteq \tau$, a τ'-structure M' with tuples $\bar{a}', \bar{b}' \in dom(M')^{k'}$ and a k'-ary $\mathrm{FO}(\tau', Arb)$ formula $\phi'(\bar{y})$ with alternation depth d which is Arb-invariant with respect to M' such that:

1′. *$M' \models \phi'(\bar{a}') \wedge \neg\phi'(\bar{b}')$, and*

2′. *$\pi' : \mathcal{N}_{r'}^{M'}(\bar{a}') \cong \mathcal{N}_{r'}^{M'}(\bar{b}')$,*

where

$$r' = \frac{r}{9k}. \tag{7}$$

Proof. Since $d > 0$, we can assume without loss of generality that the first quantifier of ϕ is existential, otherwise, we can work with the formula $\neg\phi$ instead and swap the labels of $\bar{a}$ and $\bar{b}$.

Let $\bar{a} := (a_1, a_2, \cdots, a_k)$ and $\bar{b} := (b_1, b_2, \cdots, b_k)$. There are three main cases. In each of the cases, the resulting formula $\phi'(\bar{y})$ is of the form

$$\phi'(\bar{y}) = (\exists \bar{z} \in dom(M')^{k-k'})\ R \wedge \phi(\bar{y}, \bar{z}),$$

where R is a new relation on some subset of the variables $\bar{y}$ and $\bar{z}$ added to the structure M to form M' that does not depend on the order or the arbitrary numerical predicates. It follows that

since ϕ is Arb-invariant with respect to M, ϕ' is Arb-invariant with respect to M'. The form of ϕ' also implies that ϕ' has alternation depth d since ϕ begins with an existential quantifier. We use two distance parameters ℓ and s, in addition to r', which we establish conditions on in the course of the proof. We optimize their value at the end. It remains to show that properties $1'$ and $2'$ hold for radius r' in all cases.

Case 1. *There exists $i \in [k]$ such that $a_i = b_i$.*

Assume without loss of generality that $i = k$. Expand the structure M to M' by adding a new unary predicate $R \notin \tau$ where R is satisfied only by the element a_k. Construct a new formula:

$$\phi'(y_1, y_2, \ldots, y_{k-1}) := (\exists z_k) \;\; R(z_k) \wedge \phi(y_1, y_2, \ldots, y_{k-1}, z_k).$$

Let $\bar{a}' := (a_1, a_2, \ldots, a_{k-1})$ and $\bar{b}' := (b_1, b_2, \ldots, b_{k-1})$. Property $1'$ holds because of Property 1. To see that Property $2'$ holds, observe that the isomorphism π is a bijection between $\cup_{j<k} N_r^M(a_j)$ and $\cup_{j<k} N_r^M(b_j)$. For all relations in τ, π is an isomorphism between these two sets. Further π preserves R on these sets because π maps a_k to itself. From this, it follows that $\pi : \mathcal{N}_{r'}^{M'}(\bar{a}') \cong \mathcal{N}_{r'}^{M'}(\bar{b}')$ and Property $2'$ holds for any radius $r' \leq r$.

In summary, the isomorphism radius of this case satisfies $r' \leq r$, the isomorphism π is not modified, the structure gains one new relation, and the arity of the formula is reduced by one.

Case 2. *For all $t \in \mathbb{N}$ and $i \in [k]$, $dist^M(\bar{a}, \pi^t(a_i)) \leq 2\ell$.*

For all $t \in \mathbb{N}$, $\pi^t(\bar{a})$ is a tuple in $N_{2\ell}^M(\bar{a})$. It follows that $N_{r'}^M(\pi^t(\bar{a})) \subseteq N_{2\ell+r'}^M(\bar{a})$. If

$$r \geq 2\ell + r', \tag{8}$$

these r'-neighborhoods of $\pi^t(\bar{a})$ are contained in the domain of π and we have a chain of r'-neighborhood isomorphisms resulting in $\mathcal{N}_{r'}^M(\bar{a}) \cong \mathcal{N}_{r'}^M(\pi^t(\bar{a}))$.

Suppose that there is $t \in \mathbb{N}$ and $i \in [k]$ such that $b_i = \pi^t(b_i)$, and $M \models \phi(\pi^t(\bar{b}))$. In this case, we finish via the argument in Case 1, because ϕ distinguishes the tuples $\bar{b}$ and $\pi^t(\bar{b})$, these tuples share a component, and $\mathcal{N}_{r'}^M(\bar{b}) \cong \mathcal{N}_{r'}^M(\pi^t(\bar{b}))$. Thus, assume otherwise, i.e., for all $t \in \mathbb{N}$ and $i \in [k]$,

$$b_i = \pi^t(b_i) \Rightarrow M \models \neg\phi(\pi^t(\bar{b})). \tag{9}$$

We expand the structure M to M' by adding a new k-ary relation $R \notin \tau$ containing the tuples $\cup_{t\in\mathbb{N}}\{\pi^t(\bar{b})\}$. Define a new formula:

$$\phi'(y_1) := (\exists z_2, z_3, \ldots, z_k) \;\; R(y_1, z_2, \ldots, z_k) \wedge \phi(y_1, z_2, \ldots, z_k).$$

Let $\bar{a}' := (a_1)$ and $\bar{b}' := (b_1)$. We now establish Property $1'$. First, observe that $M' \models \phi'(\bar{a}')$ via the witness $(a_2, a_3, \ldots, a_k)$. We now argue that $M' \models \neg\phi'(\bar{b}')$. Suppose the contrary, that $M' \models \phi'(\bar{b}')$, then there exists $(c_2, c_3, \ldots, c_k) \in dom(M)^{k-1}$ and $t \in \mathbb{N}$ such that $M \models \phi(b_1, c_2, c_3, \ldots, c_k)$ and $\pi^t(\bar{b}) = (b_1, c_2, c_3, \ldots, c_k)$. This contradicts (9). Therefore $M' \models \phi'(\bar{a}') \wedge \neg\phi'(\bar{b}')$, hence Property $1'$ holds.

We now establish Property $2'$. Observe that $R \subseteq (N_{2\ell}^M(\bar{a}))^k$. This implies that for all $t \in \mathbb{N}$, $N_{r'}^{M'}(\pi^t(\bar{a}')) \subseteq N_{2\ell+r'}^M(\bar{a})$ and further that $N_{r'}^{M'}(\pi^t(\bar{a}'))$ is within the domain of π. Hence when

π acts on $N_{r'}^{M'}(\pi^t(\bar{a}'))$ all relations in τ are preserved. The mapping π also preserves R on $N_{r'}^{M'}(\pi^t(\bar{a}'))$ because R is exactly the orbit of $\bar{a}$ under π. From this, it follows that for all $t \in \mathbb{N}$, $\mathcal{N}_{r'}^{M'}(\bar{a}') \cong \mathcal{N}_{r'}^{M'}(\pi^t(\bar{a}'))$. In particular, $\mathcal{N}_{r'}^{M'}(\bar{a}') \cong \mathcal{N}_{r'}^{M'}(\bar{b}')$ and Property 2′ holds with $r' \leq r - 2\ell$ (see (9)).

In summary, the isomorphism radius is reduced to $r' \leq r - 2\ell$, the structure gains a new relation, and the arity is reduced to one.

Cases 3 & 4. *There exists $t \in \mathbb{N}$ and $i \in [k]$ such $dist^M(\bar{a}, \pi^t(a_i)) > 2\ell$.*

Select t minimal and assume without loss of generality that $i = 1$. Let $\bar{c} := \pi^t(\bar{a})$. Thus, $dist^M(\bar{a}, c_1) > 2\ell$. We argue that we can assume without loss of generality that we have a pair of tuples $\bar{a}^*$ and $\bar{b}^*$ such that for a large distance s (to be determined later):

(i) $\phi \models \phi(\bar{a}^*) \wedge \neg\phi(\bar{b}^*)$,

(ii) $\mathcal{N}_s^M(\bar{a}^*) \cong \mathcal{N}_s^M(\bar{b}^*)$, and

(iii) $dist^M(\bar{a}^*, b_1^*) > \ell$.

Suppose $dist^M(\bar{b}, c_1) \leq \ell$. Since $dist^M(\bar{a}, c_1) > 2\ell$ it follows that $dist^M(\bar{a}, b_j) > \ell$, for some $j \in [k]$. Therefore $\bar{a}$ and $\bar{b}$ satisfy condition (iii) with coordinate j permuted to 1. Conditions (i), and (ii) with

$$s \leq r \tag{10}$$

follow by properties 1 and 2 in the hypothesis of the lemma.

Otherwise, $dist^M(\bar{b}, c_1) > \ell$. Because t is selected minimally, for all $j < t$, $N_s^M(\pi^j(\bar{a})) \subseteq N_{2\ell+s}^M(\bar{a})$. If

$$r \geq 2\ell + s, \tag{11}$$

these s-neighborhoods of $\pi^j(\bar{a})$ are contained in the domain of π and we have a chain of s-neighborhood isomorphisms resulting in $\mathcal{N}_s^M(\bar{a}) \cong \mathcal{N}_s^M(\bar{c})$. Since ϕ distinguishes $\bar{a}$ and $\bar{b}$, ϕ must be able to distinguish between either $\bar{a}$ and $\bar{c}$, or $\bar{b}$ and $\bar{c}$. Therefore, for some pair, all three conditions (i), (ii), and (iii) are met.

Thus, we have a pair of tuples $\bar{a}^*$ and $\bar{b}^*$ that satisfy conditions (i), (ii), and (iii) with $s \leq r - 2\ell$. Let π relabel the isomorphism between the s-neighborhoods for this new pair. There are two subcases.

Case 3. *For all $j \in [k]$, $dist^M(b_1^*, b_j^*) \leq s$.*

See Figure 7 for a diagram of this case. Because of property (ii) and since isomorphisms preserve distance, for all $j \in [k]$, $dist^M(a_1^*, a_j^*) \leq s$.

Expand the structure M to form M' by introducing a new k-ary relation $R \notin \tau$ containing only the tuples $\bar{a}^*$ and $\bar{b}^*$. Let $\bar{a}' := (a_1^*)$ and $\bar{b}' := (b_1^*)$. Construct a new formula:

$$\phi'(y_1) := (\exists z_2, z_3, \ldots, z_k) \;\; R(y_1, z_2, \ldots, z_k) \wedge \phi(y_1, z_2, \ldots, z_k).$$

By Property (iii), a_1^* and b_1^* are distinct. This means that $\bar{a}'$ and $\bar{b}'$ correspond with the distinct elements of R, and, with Property (i), we determine that $M' \models \phi'(\bar{a}') \wedge \neg\phi'(\bar{b}')$, hence Property 1′ holds.

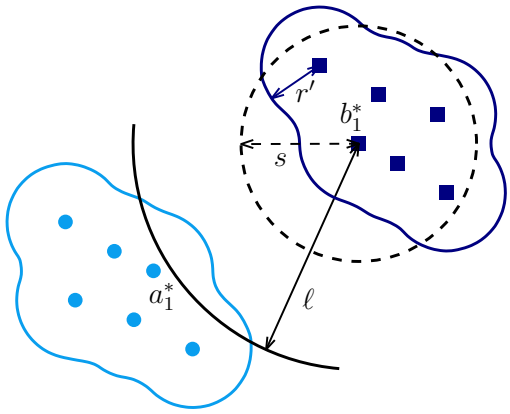

Figure 7: Diagram for Case 3.

To establish Property 2′, consider radius r', with

$$r' \leq s. \tag{12}$$

The tuple $\bar{a}^* \in R$ is fully contained in $N_{r'}^{M'}(\bar{a}')$, because, due to the addition of R when going from M to M', the distance between any two distinct points in $\bar{a}^*$ is reduced to 1. Thus, if the sets $N_{r'}^{M'}(\bar{a}')$ and $N_{r'}^{M'}(\bar{b}')$ are disjoint, there is no intersection between $\bar{a}^*$ and $N_{r'}^{M'}(\bar{b}')$ (similarly for the tuple $\bar{b}^*$ and $N_{r'}^{M'}(\bar{a}')$). The fact that $R = \{\bar{a}^*, \bar{b}^*\}$, Property (ii) holds, and (12) imply that π preserves R on the domain $\mathcal{N}_{r'}^{M'}(\bar{a}')$. Hence, $\pi : \mathcal{N}_{r'}^{M'}(\bar{a}') \cong \mathcal{N}_{r'}^{M'}(\bar{b}')$. Thus Property 2′ holds for r'. It remains to establish a sufficient condition for such disjointness.

First, observe that $dist^M(\bar{a}^*, \bar{b}^*) > \ell - s$, because all elements of $\bar{b}^*$ are within s of b_1^* and $dist^M(\bar{a}^*, b_1^*) > \ell$. This implies that $dist^{M'}(\bar{a}^*, \bar{b}^*) > \ell - s$, because the tuples in R cannot contribute an edge in a shortest path between $\bar{a}^*$ and $\bar{b}^*$. Further, since $\bar{a}'$ and $\bar{b}'$ are elements in $\bar{a}^*$ and $\bar{b}^*$, $dist^{M'}(\bar{a}', \bar{b}') > \ell - s$. Therefore, if we select

$$r' \leq \frac{\ell - s}{2}, \tag{13}$$

the r'-neighborhoods of $\bar{a}'$ and $\bar{b}'$ are disjoint in M'.

In summary, the structure gets one new relation, the isomorphism radius reduces to $r' \leq \min(\frac{\ell - s}{2}, s)$, and the arity reduces to one.

Case 4. *There exists $j \in [k]$, such that $dist^M(b_1^*, b_j^*) > s$.*

See Figure 8 for a diagram of this case. In this case we can construct a hybrid tuple $\bar{h}$ from $\bar{a}^*$ and $\bar{b}^*$ such that ϕ distinguishes $\bar{h}$ from one of $\bar{a}^*$ or $\bar{b}^*$ and the r'-neighborhoods of all three tuples are isomorphic, where the term "hybrid" means that $\bar{h}$ has components from both $\bar{a}^*$ and $\bar{b}^*$. As this pair of tuples shares some common components we can apply Case 1 to conclude.

For an index set $I \subseteq [k]$, let the tuple $\bar{b}_I$ consist of only the components of $\bar{b}^*$ with indices in I. Start with $I := \{1\}$. While there is an $i \in [k] \backslash I$ such that $dist^M(b_i^*, \bar{b}_I) \leq 2r' + 1$, add i to I. If

$$(k-1)(2r'+1) \leq s, \tag{14}$$

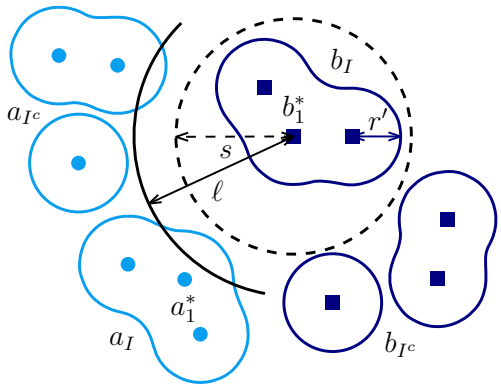

Figure 8: Diagram for Case 4.

$\bar{b}_I$ cannot contain every component of $\bar{b}^*$ (by the hypothesis of this case). Let I^c be the complement of I (i.e., $I^c := [k] \backslash I$) and define $\bar{h} := (\bar{b}_I, \bar{a}_{I^c})$. We argue that we can construct an isomorphism

$$\rho : \mathcal{N}_{r'}^M(\bar{b}^*) = \mathcal{N}_{r'}^M(\bar{b}_I, \bar{b}_{I^c}) \cong \mathcal{N}_{r'}^M(\bar{b}_I, \bar{a}_{I^c}) = \mathcal{N}_{r'}^M(\bar{h}).$$

This implies that

$$\rho \circ \pi : \mathcal{N}_{r'}^M(\bar{a}^*) \cong \mathcal{N}_{r'}^M(\bar{h}).$$

Together this implies that the r'-neighborhoods of $\bar{a}^*$, $\bar{b}^*$ and $\bar{h}$ are isomorphic. Since one of $M \models \phi(\bar{h})$ or $M \models \neg\phi(\bar{h})$ holds we can conclude by applying Case 1 with $\bar{h}$ and one of $\bar{a}^*$ and $\bar{b}^*$. This establishes properties $1'$ and $2'$ for radius r'.

It remains to argue the isomorphism ρ exists. If $N_{r'}^M(\bar{b}_I)$ is at least distance two from both $N_{r'}^M(\bar{b}_{I^c})$ and $N_{r'}^M(\bar{a}_{I^c})$, it suffices to define ρ to be identity map on the domain $N_{r'}^M(\bar{b}_I)$ and act as π^{-1} on the domain $N_{r'}^M(\bar{b}_{I^c})$, since no tuple in a relation can straddle both parts of the domain. By construction $dist^M(\bar{b}_I, \bar{b}_{I^c}) > 2r' + 1$. This implies that $dist^M(N_{r'}^M(\bar{b}_I), N_{r'}^M(\bar{b}_{I^c})) > 1$. It remains to argue $dist^M(\bar{b}_I, \bar{a}_{I^c}) > 2r' + 1$. Suppose otherwise, $dist^M(\bar{b}_I, \bar{a}_{I^c}) \leq 2r' + 1$. Then $dist^M(b_1^*, \bar{a}^*) \leq (k-2)(2r'+1) + dist^M(\bar{b}_I, \bar{a}_{I^c}) \leq (k-1)(2r'+1)$. This contradicts Property (iii) as long as we choose,

$$\ell \geq (k-1)(2r'+1). \tag{15}$$

Therefore ρ exists and the case is concluded.

In summary, during Case 4 the isomorphism π and the structure are modified, and the isomorphism radius is reduced to r' satisfying (10), (11), (14), and (15).

This completes the case analysis.

Choosing $\ell = k(2r'+1)$ and $s = (k-1)(2r'+1)$ we see that the conditions (8), (10), (11), (12), (13), (14), and (15) are satisfied for $r \geq (3k-1)(2r'+1)$. Since the lemma holds trivially when $r' < 1$, selecting $r = 9kr' = (3k)(3r') \geq (3k-1)(2r'+1)$ suffices. This is (7) in the statement of the lemma which completes the proof. □

4.3 Upper Bound for General Formulas

In this section we prove the general case of Theorem 4.1, which implies the upper bound in Theorem 1.1. The critical cases are the ones with positive alternation depth. In those cases, the idea is to iteratively apply Lemma 4.4 to reduce to the unary version of Theorem 4.1.

Suppose that a k-ary formula ϕ with alternation depth $d > 0$ is Arb-invariant and not $(\log n)^c$-local with respect to a structure M, where c is some positive constant. This means that there exists a violation in M to the $(\log n)^c$-locality of ϕ. Iteratively applying Lemma 4.4 yields a violation of the $(\gamma_k \cdot (\log n)^c)$-locality of some unary formula ϕ' of alternation depth d on some structure M' with $|M'| = |M| = n$, where γ_k only depends on k. For c' a constant such that $d + 2 < c' < c$, and n sufficiently large such that $(\gamma_k \cdot (\log n)^c) \geq (\log n)^{c'}$, we obtain a contradiction with the unary version of Theorem 4.1 as long as $n \geq n_{\phi',c'}$. Thus, if we can upper bound the values $n_{\phi',c'}$ that can arise in the reduction from ϕ, we are done.

The upper bound follows because the number of different formulas ϕ' that can arise from ϕ is bounded. This is because in each case of the proof of Lemma 4.4, the resulting formula ϕ' consists of (i) an existential quantification over the marking relation to construct a tuple, and (ii) an evaluation of ϕ on the quantified tuple. The number of such formulas depends only on how the free variables are situated. This means that in the end the reduction produces only a bounded number of unary formulas, depending on ϕ.

We now formalize this argument.

Proof of Theorem 4.1. We first remark that the case $d = 0$ in Theorem 4.1 trivially holds. To see this, consider a k-ary quantifier-free formula $\phi(\bar{x})$ of $\mathrm{FO}(\tau, \mathrm{Arb})$ that is Arb-invariant on a τ-structure M. We show that $\phi(\bar{x})$ is 0-local with respect to M. This implies that $\phi(\bar{x})$ is $(\log n)^c$-local with respect to M for any constant c. Let $n := |M|$. Assume that $M \models \phi(\bar{a})$ and consider a tuple $\bar{b}$ such that $\mathcal{N}_0^M(\bar{a}) \cong \mathcal{N}_0^M(\bar{b})$. Consider any Arb-expansion M' of M. By Arb-invariance we have $M' \models \phi(\bar{a})$. Recall that the linear order of M' induces a bijection h' between $dom(M)$ and $[n]$. Let h'' be any bijection between $dom(M)$ and $[n]$ such that $h'(\bar{a}) = h''(\bar{b})$. This bijection induces a new linear order on M and a new Arb-expansion M'' of M. We claim that $M'' \models \phi(\bar{b})$. By Arb-invariance this implies $M \models \phi(\bar{b})$ as desired. From $\mathcal{N}_0^M(\bar{a}) \cong \mathcal{N}_0^M(\bar{b})$ we get that each atom of ϕ involving a relation in τ is true on M (and therefore on M' and M'') for $\bar{a}$ iff it is true for $\bar{b}$. From $h'(\bar{a}) = h''(\bar{b})$ we get that each atom of ϕ involving a numerical predicate is true for $\bar{a}$ on M' iff it is true for $\bar{b}$ on M''. As $\phi(\bar{x})$ is quantifier free we conclude that $M' \models \phi(\bar{a})$ iff $M'' \models \phi(\bar{b})$, which finishes the quantifier-free case.

Consider now the case of alternation depth $d > 0$. Suppose that $\phi(\bar{x})$ is a k-ary alternation-depth-d formula of $\mathrm{FO}(\tau, \mathrm{Arb})$ that is Arb-invariant with respect to a τ-structure M. Further suppose that $\phi(\bar{x})$ is not $(\log n)^c$-Gaifman local with respect to M, where $n := |M| \geq n_{\phi,c}$ ($n_{\phi,c}$ will be determined later). The non-locality of ϕ is witnessed by two tuples $\bar{a}$ and $\bar{b}$ on M. To ϕ and the witness $(M, \bar{a}, \bar{b})$ we can apply Lemma 4.4 at most $k - 1$ times to produce a unary formula ϕ' with alternation depth d which is Arb-invariant and non-local with respect to a structure M'. This non-locality is witnessed by the elements a' and b' distinguished by ϕ', and an isomorphism between the $\lfloor \frac{(\log n)^c}{(9k)^{k-1}} \rfloor$-neighborhoods of a' and b'. Let c' be any constant such that $d + 2 < c' < c$.

If we select $n_{\phi,c}$ satisfying $\lfloor \frac{(\log n_{\phi,c})^c}{(9k)^{k-1}} \rfloor \geq (\log n_{\phi,c})^{c'}$ and $n_{\phi,c} \geq n_{\phi',c'}$ we have a structure M', with $|M'| = n \geq n_{\phi',c'}$, where ϕ' is Arb-invariant on M', but not $(\log n)^{c'}$-local with respect to M'. This contradicts the unary version of this theorem. Therefore, it suffices to pick $n_{\phi,c}$ to be

the maximum of $2^{(9k)^{\frac{k-1}{c-c'}}}$ and $n_{\phi',c'}$ for each ϕ' that may result from the iterated applications of Lemma 4.4. We now argue that the number of such formulas ϕ' is bounded by a constant.

Claim 4.5. *For a given formula ϕ, the number of different formulas ϕ' that can be produced by Lemma 4.4 for different choices of M, $\bar{a}$, and $\bar{b}$ is upper bounded by a function of k only.*

Proof. Consider each case of the proof of Lemma 4.4 and the formula produced. In all cases except Case 1, the formula is of the form

$$\phi'(\bar{y}) = (\exists \bar{z} \in dom(M')^{k-k'})\ R(\bar{y}, \bar{z}) \wedge \phi(\bar{y}, \bar{z}).$$

This induces at most $k-1$ different formulas because the range of k' is $1 \leq k' \leq k-1$. Note that Case 1 is slightly different in that R is a unary predicate, not a k-ary relation. So, taken together we have at most k basic formula types. However, we often (implicitly) relabeled the free variables for convenience of notation. These variations may induce distinct formulas as well. This relabeling can increase the number of distinct formulas by a factor of at most $k!$ (one for each ordering of the variables). We conclude that there are at most $k \cdot k!$ different formulas that the proof may produce. Note that bound is independent of M, $\bar{a}$, $\bar{b}$, the Arb relations and even $|M|$. □

Since the number of possible ϕ' is bounded by a constant (depending on ϕ), $n_{\phi,c}$ can be selected to be such a constant as well. This concludes the proof. None

4.4 Lower Bound

For $c = 1$, the lower bound of Theorem 1.1 is implicit in [DLM07, Corollary 2]. For generalizing the result to arbitrary $c \geq 1$, the proof idea is as follows: We consider graphs represented as τ_E-structures, where τ_E is the schema consisting of a binary relation symbol E. We construct an Arb-invariant formula $\phi_c(y)$ which, when evaluated in a graph G, expresses that (i) G has less than $(\log n)^{c+1}$ non-isolated nodes (where n denotes the total number of nodes of G), and (ii) y is reachable from a node that lies on a triangle. To note that ϕ_c is *not* Gaifman $(\log n)^c$-local, consider, for a sufficiently large n, the graph G that consists of the disjoint union of

- a triangle, connected to a path of length $(\log n)^c + 1$,
- a path of length $(\log n)^c + 1$, and
- enough isolated nodes such that the total number of nodes of G is exactly n.

Let b and b' be the last nodes on the two paths present in G. Obviously, their $(\log n)^c$-neighborhoods are isomorphic. But b is reachable from a node that lies on a triangle, and b' is not. Since n is sufficiently large, the total number of non-isolated nodes is less than $(\log n)^{c+1}$. Thus, $G \models \phi_c(b)$ and $G \not\models \phi_c(b')$. In summary, ϕ_c is not Gaifman $(\log n)^c$-local.

For the construction of the formula ϕ_c, we use the following lemma. This lemma will also be used later on, in Section 5, for the proof of the lower bound of Theorem 1.2.

Lemma 4.6. *Let τ_{ES} be the schema consisting of a binary relation symbol E and a unary relation symbol S. For every integer $d \geq 1$ there is an Arb-invariant* $\mathrm{FO}(\tau_{ES}, Arb)$*-formula $reach_d(x, y)$*

such that the following is true for all finite τ_{ES}-structures M, all elements a, b in S^M, and $n := |dom(M)|$:

$$M \models reach_d(a,b) \iff \begin{array}{l} |S^M| < (\log n)^d \text{ and} \\ \text{there is a path from } a \text{ to } b \text{ in the induced subgraph} \\ \text{of } G := (dom(M), E^M) \text{ on } S^M. \end{array}$$

Proof. For the proof, we use the following technical result of [DLM07].

Lemma 4.7 (Corollary 1 in [DLM07]). *Let τ_S be the schema consisting of a unary relation symbol S. For every integer $d \geq 1$ there is a $\mathrm{FO}(\tau_S, Arb)$-formula $bij_d(x,y)$ such that the following is true for all τ_S-structures M, all Arb-expansions M' of M, all elements a, b in $dom(M)$, and $n := |dom(M)|$:*

$$M' \models bij_d(a,b) \iff \begin{array}{l} |S^M| < (\log n)^d \text{ and} \\ a \text{ is the } i\text{th largest element w.r.t. } <^{M'} \text{ in } S^M, \\ \text{where } i \text{ is the index of } b \text{ in } dom(M') \text{ w.r.t. } <^{M'}. \end{array}$$

Note that the formula $bij_d(x,y)$ of Lemma 4.7 constitutes a bijection from the set S^M to an initial set of elements of $dom(M')$ (initial, with respect to the linear order present in the structure M'), and thus to the natural numbers $1, 2, \ldots, |S^M|$. This enables us to represent elements of S^M by natural numbers of size $\leq |S^M| < (\log n)^d$. Using its binary representation, we encode each such number by a binary string of length (exactly) $d \log\log n$.

Hence a sequence of elements of S^M of size bounded by $\ell(n) := \frac{\log n}{d \log\log n}$ can be represented using $\log n$ bits, i.e., a natural number $< n$ or, equivalently, an element of M. Given an element of M, using the appropriate numerical predicates present in the Arb-expansion M' of M, we can extract from this element any of its blocks of length $d \log\log n$ and, using $bij_d(x,y)$, retrieve the corresponding element of S^M.

We can use this to construct an $\mathrm{FO}(\tau_{ES}, \mathrm{Arb})$-formula $\varrho(x,y)$ which, when evaluated in M', expresses that x and y are elements in S^M such that there is a path of length at most $\ell(n)$ in S^M from x to y: The formula $\varrho(x,y)$ simply guesses the path by existentially quantifying over the element of M representing its sequence and then checks that there is indeed an edge between any two consecutive nodes of this path. From the discussion above, this can be expressed in $\mathrm{FO}(\tau_{ES}, \mathrm{Arb})$.

In summary, $\varrho(x,y)$ is an $\mathrm{FO}(\tau_{ES}, \mathrm{Arb})$ formula such that the following is true for all finite τ_{ES}-structures M, all Arb-expansions M' of M, all elements a, b in S^M, and $n := |dom(M)|$:

$$M' \models \varrho(a,b) \iff \begin{array}{l} |S^M| < (\log n)^d \text{ and} \\ \text{there is a directed path from } a \text{ to } b \text{ of length} \leq \ell(n) \\ \text{in the} \\ \text{induced subgraph of } G := (dom(M), E^M) \text{ on } S^M. \end{array}$$

Thus, obviously, $\varrho(x,y)$ is Arb-invariant.

We iterate $\varrho(x,y)$ for a suitable number of times in order to obtain a formula for reachability $reach_d(x,y)$ by paths of length up to $(\log n)^d$: Let $\psi_1(x,y) := \varrho(x,y)$, and for $i \geq 2$, let $\psi_i(x,y)$ be the formula obtained from $\varrho(x,y)$ by replacing every atom of the form $E(z,z')$ by the formula

$\psi_{i-1}(z, z')$. It is straightforward to see that $\psi_i(x, y)$ states that there is a path of length at most $\ell(n)^i$ in S^M from x to y.

For $i := d + 1$, there exists an n_0 such that for all $n > n_0$ we have $\ell(n)^i \geq (\log n)^d > |S^M|$. Therefore, we can choose $\mathit{reach}_d(x, y)$ to be the formula stating that either $|\mathit{dom}(M)| > n_0$ and $\psi_i(x, y)$ holds, or, for some $\ell \in \{1, \ldots, n_0\}$, we have $|\mathit{dom}(M)| = \ell$ and $|S^M| \leq (\log \ell)^d$, and x and y are nodes in S^M such that y is reachable from x by a path of length $\leq \ell$ that uses only nodes in S^M (note that for each fixed ℓ this can be expressed in $\mathrm{FO}(\tau_{ES})$). This concludes the proof of Lemma 4.6. □

We are now ready for the proof of the lower bound of Theorem 1.1.

Proof of Theorem 1.1 – Lower Bound. Let τ_E be the schema consisting of a binary relation symbol E, let $d := c+1$, and let $\mathit{reach}_d(x, y)$ be the Arb-invariant formula provided by Lemma 4.6. Let $\varrho(x, y)$ be the formula obtained from $\mathit{reach}_d(x, y)$ by replacing every atomic formula of the form $S(z)$ by a formula stating that z is a non-isolated node (i.e., by the formula $\exists z'\big(E(z, z') \vee E(z', z)\big)$). Clearly, when evaluated in a τ_E-structure M, the formula $\varrho(x, y)$ states that there are less than $(\log n)^d$ non-isolated nodes in M, x and y are non-isolated, and there is a path from x to y.

Let $\phi_c(y)$ be the formula

$$\exists x \exists x_1 \exists x_2 \big(E(x, x_1) \wedge E(x_1, x_2) \wedge E(x_2, x) \wedge \varrho(x, y)\big).$$

Obviously, ϕ_c is Arb-invariant, since ϱ is Arb-invariant. Furthermore, when evaluated in a τ_E-structure M, the formula ϕ_c expresses that (i) there are less than $(\log n)^d$ non-isolated nodes (where n denotes the size of $\mathit{dom}(M)$), and (ii) y is reachable from a node that lies on a triangle.

By the reasoning given at the beginning of Section 4.4, ϕ_c is not Gaifman $(\log n)^c$-local. This completes the proof of the lower bound in Theorem 1.1. None

Remark 4.8. *We point out that addition $+$ and multiplication $\times$ are the only numerical predicates occurring in the formula $\mathit{bij}_d(x, y)$ of Lemma 4.7. Furthermore, all the constructions in the proof of Lemma 4.6 can be realized with those numerical predicates only (cf., e.g., [Imm99, Sch05]). Thus, the formula $\mathit{reach}_d(x, y)$ provided by Lemma 4.6 is an Arb-invariant* FO*-formula that only uses the numerical predicates $+$ and $\times$. Consequently, the lower bound of Theorem 1.1 already holds for such formulas.*

5 Hanf Locality for String Structures

In Section 4 we showed that Arb-invariant FO formulas are Gaifman $(\log n)^{O(1)}$-local. We are not able to prove that Arb-invariant FO formulas are also Hanf $(\log n)^{O(1)}$-local in general but are able to do so in the special case when the structures represent strings.

Fix a finite alphabet A and consider structures over the schema τ_s containing one unary predicate per element of A and one binary predicate E. Let $\mathcal{S}$ be the class of τ_s-structures M that interpret E as a successor relation and where each element of M belongs to exactly one of the unary predicates in τ_s. Each structure in $\mathcal{S}$ represents a string in the obvious way and we blur the distinction between a string w and its actual representation as a structure. We then consider $\mathrm{FO}(\tau_s \cup \sigma_{\mathrm{arb}})$ formulas that are Arb-invariant over all structures in $\mathcal{S}$ and denote the corresponding set of formulas by

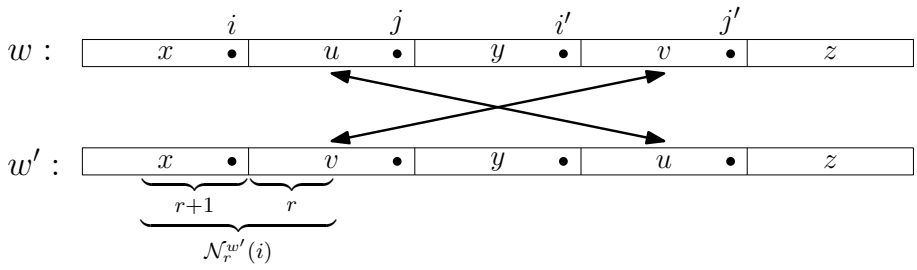

Figure 9: r-swapping $w = xuyvz$ to $w' = xvyuz$.

Arb-invariant FO(*Succ*). We say that a language $L \subseteq A^*$ is definable in Arb-invariant FO(*Succ*) if there is a sentence of Arb-invariant FO(*Succ*) whose set of models in $\mathcal{S}$ is exactly L.

The goal of this section is to prove Theorem 1.2. The lower bound part will be proved in Section 5.4. For the upper bound part we actually show the following result:

Theorem 5.1. *Arb-invariant* FO(*Succ*) *formulas with alternation depth d are Hanf $(\log n)^c$-local for any constant $c > d + 2$.*

The crux of Theorem 5.1 is the case where the formula is a sentence. For that reason, we only consider sentences in Sections 5.1 and 5.2. We return to the general case in Section 5.3.

The proof of Theorem 5.1 for sentences consists of two parts. In Section 5.1 we introduce a closure property of languages allowing to swap substrings inside a string without affecting membership in the language as long as the neighborhoods around the endpoints of the substrings look similar. We then show that a language being closed under swaps is equivalent to the language being Hanf local, where the size of the boundary neighborhoods is essentially the isomorphism radius. In Section 5.2 we show that languages definable in Arb-invariant FO(*Succ*) are closed under this swap operation for boundary neighborhoods of radius $(\log n)^{O(1)}$. We conclude in Section 5.3 by combining the two previous results and derive Theorem 5.1.

5.1 Connection with Closure under Swaps for Sentences

In this section we introduce the key notion of a swap. It is an operation that exchanges two substrings inside a string as long as the neighborhoods around the endpoints of the substrings look similar. Our notion of a swap is somewhat related to a similar notion that was introduced in [TW85] for regular languages (see also [BP89, BS09a]).

Let $w \in A^*$, $i, j \in \mathbb{N}$, define $w[i, j]$ to be the substring of w starting at position i of w and ending at position j. Let $n = |w|$ and $r > 0$, then the *r-suffix* of w is $w[n - r + 1, n]$ and the *r-prefix* of w is $w[1, r]$. Notice that if i is the last position of u in the string $w = uv$ then $\mathcal{N}_r^w(i)$ is the concatenation of the $(r + 1)$-suffix of u with the r-prefix of v.

Let $r \in \mathbb{N}$ and $w \in A^*$. A string $w' \in A^*$ is obtained from w by a *r-swap operation* if $w = xuyvz$, $\mathcal{N}_r^w(i) \cong \mathcal{N}_r^w(i')$ and $\mathcal{N}_r^w(j) \cong \mathcal{N}_r^w(j')$ where i, j, i', and j' are, respectively, the positions in w immediately before the substrings u, y, v, and z, and $w' = xvyuz$. See Figure 9 for a diagram.

Let $r : \mathbb{N} \to \mathbb{R}_{\geq 0}$. A language L is said to be *closed under $r(n)$-swaps* if there exists a $n_0 \in \mathbb{N}$ such that for all strings $w, w' \in A^*$, with $|w| = n > n_0$, if w' is obtained from w by a $r(n)$-swap

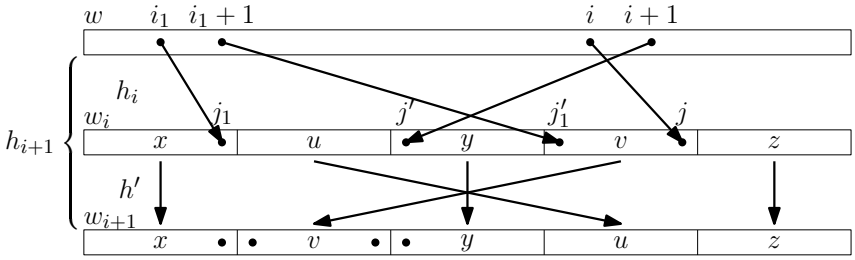

Figure 10: Constructing h_{i+1} from h_i and w_i.

operation then we have:

$$w \in L \qquad \text{iff} \qquad w' \in L.$$

Informally, a language is closed under swaps if the language is unable to distinguish the relative order of substrings whose local neighborhoods look the same.

There are tight connections between closure under swaps and Hanf locality. The first one is that an r-swap operation does not change the $\equiv_r$-class of a string. This follows from the observation that an r-swap preserves r-neighborhoods. The second one concerns the opposite direction: If two strings are in a same $\equiv_r$-class then there is a sequence of $(r-1)$-swap operations transforming one into the other. Intuitively this is shown as follows. Assuming that $w \equiv_r w'$, we transform w' into w using $(r-1)$-swaps to embed larger and larger prefixes of w within the transformed string. Here an embedding is a partial function on string positions preserving r-neighborhoods and the string order (i.e., the relative ordering of the positions in the prefix of w is preserved by the embedding). Eventually, the entirety of w embeds into the transformed string, and because $|w| = |w'|$ this implies that the transformed string coincides with w. Thus, we have transformed w' into w via a sequence of $(r-1)$-swaps.

We now sketch the transformation procedure. See Figure 10 for a diagram of this construction. In the i^{th} step of the procedure, we consider the i-prefix of w that we assume embeds into the string $w_i \equiv_r w$ via the embedding h_i. Our goal is to extend the embedding to the $(i+1)$-prefix of w while preserving the $\equiv_r$-class. Let $j := h_i(i)$. Because $w \equiv_r w_i$, there is a position j' in w_i, outside the image of h_i, that has the same r-neighborhood as $i+1$. If $j' > j$, mapping $i+1$ to j' preserves the ordering of w and extends the embedding h_i to the $(i+1)$-prefix of w. Otherwise, we have $j' < j$. Let i_1 be the maximal position in the i-prefix of w such that $h_i(i_1) < j'$. By maximality of i_1, we have the following relative positions within w_i: $h_i(i_1) < j' < h_i(i_1+1) \leq j$. The key observation is that because i and $j = h_i(i)$ have the same r-neighborhood then $j'-1$ and j have the same $(r-1)$-neighborhood. As the same consequence can be derived for $h_i(i_1)+1$ and $h_i(i_1+1)$, we can $(r-1)$-swap the substrings of w_i with these endpoints (i.e., substrings u and v in Figure 10). We then observe that the $(i+1)$-prefix of w embeds into the resulting string w_{i+1} and that $w_{i+1} \equiv_r w_i$, so w_{i+1} remains in the same $\equiv_r$-class as w. Initializing $w_1 = w'$ and $h_1 : w \equiv_r w'$ establishes the conditions required to start the procedure.

Lemma 5.2. *Let $r \in \mathbb{N}$, and $w, w' \in A^*$.*

1. *If w' is obtained from w by a r-swap operation then $w \equiv_r w'$.*
2. *If $w \equiv_r w'$ then there is a finite sequence of $(r-1)$-swap operations transforming w into w'.*

Proof. Fix $r \in \mathbb{N}$.

Part 1.

Assume $w = xuyvz$, $w' = xvyuz$, and for i, j, i' and j' which are, respectively, the positions immediately before u, y, v, and z in w, we have $\mathcal{N}_r^w(i) \cong \mathcal{N}_r^w(i')$ and $\mathcal{N}_r^w(j) \cong \mathcal{N}_r^w(j')$. Let h be a bijection from w to w' that sends each block x, u, y, v, z to its corresponding block in w'. In other words, h sends the first letter of x in w to the first letter of x in w' and so on; h acts on the other substrings u, y, v, and z in a similar fashion. We show that h preserves r-neighborhoods. It is enough to show this for the harder cases, i.e., the boundary cases. By symmetry we only need to consider the boundaries of y.

Recall that j is the position immediately before y, i.e., the last position of u. Let $k := h(j)$ be the last position of u in w'. We want to show $\mathcal{N}_r^{w'}(k) \cong \mathcal{N}_r^w(j)$. First consider the right part of the respective neighborhoods. In w this is the r-prefix of yvz while in w' it is the r-prefix of z. By hypothesis, $\mathcal{N}_r^w(j) \cong \mathcal{N}_r^w(j')$, and the r-prefix of z is the same as the r-prefix of yvz, so we are done.

Now consider the left part of the respective neighborhoods. In w it is xu while in w' it is $xvyu$. We need to show that they have the same $(r+1)$-suffix. From $\mathcal{N}_r^w(i) \cong \mathcal{N}_r^w(i')$ we know that x and xuy have the same $(r+1)$-suffix; this implies that xv and $xuyv$ have the same $(r+1)$-suffix. From $\mathcal{N}_r^w(j) \cong \mathcal{N}_r^w(j')$ we get that $xuyv$ and xu have the same $(r+1)$-suffix. By combining these two facts we see that xu and xv have the same $(r+1)$-suffix. Therefore xuy and xvy have the same $(r+1)$-suffix. Hence x and xvy have the same $(r+1)$-suffix, and thus xu and $xvyu$ do as well, as desired.

The other boundary of y, position i', is treated similarly.

Part 2.

Let w and w' be two strings of A^* such that $w \equiv_r w'$ and $|w| = n$. Let h be a bijection witnessing $w \equiv_r w'$.

We construct by induction a sequence of strings $w_1, \cdots, w_n$ such that (i) $w_1 = w'$, (ii) for $i \leq n$, $w \equiv_r w_i$ via a bijection h_i verifying $\forall j < j' \leq i, \quad h_i(j) < h_i(j')$, and (iii) w_{i+1} is either w_i or is obtained from w_i via a $(r-1)$-swap operation. Note that (ii) implies that h_n is the identity and therefore $w_n = w$. Properties (i) and (iii) imply that $w = w_n$ is obtained from $w' = w_1$ by a finite sequence of $(r-1)$-swap operations, proving the result.

The base case is immediate by setting $w_1 := w'$ and $h_1 := h$.

Suppose we have constructed w_i and h_i satisfying the inductive properties (i), (ii), and (iii) up to i. Let $j := h_i(i)$ and $j' := h_i(i+1)$. If $j' > j$, (ii) is already satisfied and we are done. Assume now that $j' < j$. Let $i_1 < i$ be the position in w such that $h_i(i_1) < j' < h_i(i_1 + 1)$, and let $j_1 := h_i(i_1)$ and $j_1' := h_i(i_1 + 1)$. Note an index i_1 such that $h_i(i_1) < j'$ exists because h_i preserves r-neighborhood-types, and therefore must map the element of w with index 1 to the element of w_i with index 1 (i.e., $1 = h_i(1) < j'$). See Figure 10 for a diagram of the construction.

Now, notice that because h_i preserves r-neighborhood-types and j_1, j_1' are the images of consecutive positions in w, $\mathcal{N}_{(r-1)}^{w_i}(j_1) \cong \mathcal{N}_{(r-1)}^{w_i}(j_1' - 1)$. For the same reason, $\mathcal{N}_{(r-1)}^{w_i}(j' - 1) \cong \mathcal{N}_{(r-1)}^{w_i}(j)$.

Hence our string w_i can be decomposed as $xuyvz$ where (in the case where $j' = j_1 + 1$, u is the

empty string):

$$\begin{aligned} x &:= w_i[1, j_1], \\ u &:= w_i[j_1 + 1, j' - 1], \\ y &:= w_i[j', j'_1 - 1], \\ v &:= w_i[j'_1, j], \\ z &:= w_i[j + 1, n] \end{aligned}$$

and the conditions for a $(r-1)$-swap hold. Note that this swap induces a permutation h' on the positions of w_i.

We set $w_{i+1} := xvyuz$ and condition (iii) holds. We now set $h_{i+1} := h' \circ h_i$, the composition of h_i and h'. We have the following claim.

Claim 5.3. $h_{i+1} : w \equiv_r w_{i+1}$

Proof. We show that $h' : w_i \equiv_r w_{i+1}$. The claim then follows because $h_i : w \equiv_r w_i$. We argue that the r-neighborhoods of the substrings x, u, y, v, and z are identical in both w_i and w_{i+1}, hence $w_i \equiv_r w_{i+1}$.

We start by deriving a few identities from our hypothesis. For a string s, we use the notation $P(s)$ to denote the r-prefix of s and $S(s)$ to denote the r-suffix of s. Consider $S(x)$. Because $\mathcal{N}_r^w(i_1) \cong \mathcal{N}_r^{w_i}(j_1)$, $S(x) = S(w[1, i_1])$. Moreover, as $\mathcal{N}_r^w(i_1 + 1) \cong \mathcal{N}_r^{w_i}(j'_1)$, we have $S(w[1, i_1]) = S(xuy)$. Hence $S(x) = S(xuy)$. Similarly the known neighborhood isomorphisms give us the following facts, where the text in the square brackets indicates which neighborhoods the identities address, e.g., "left nbh of v" means that the identity shows that the strings of length r preceding v in w and w' are the same.

$$S(x) = S(xuy) \qquad \text{[left nbh of } v\text{]} \qquad (16)$$
$$S(xu) = S(xuyv) \qquad (17)$$
$$P(z) = P(yvz) \qquad \text{[right nbh of } u\text{]} \qquad (18)$$
$$P(vz) = P(uyvz) \qquad (19)$$

We can derive a number of implications using (16)-(19).

$$\begin{aligned} (16) \wedge (17) &\Rightarrow S(xv) = S(xuyv) = S(xu) && \text{[left nbh of } y\text{]} && (20) \\ &\Rightarrow S(xuy) = S(xvy) \Rightarrow_{(16)} S(x) = S(xvy) && \text{[left nbh of } u\text{]} && (21) \\ &\Rightarrow S(xu) = S(xvyu) \Rightarrow_{(17)} S(xuyv) = S(xvyu) && \text{[left nbh of } z\text{]} \end{aligned}$$

$$\begin{aligned} (18) \wedge (19) &\Rightarrow P(uz) = P(uyvz) = P(vz) && \text{[right nbh of } y\text{]} && (22) \\ &\Rightarrow P(yvz) = P(yuz) \Rightarrow_{(18)} P(z) = P(yuz) && \text{[right nbh of } v\text{]} \\ &\Rightarrow P(vz) = P(vyuz) \Rightarrow_{(19)} P(uyvz) = P(vyuz) && \text{[right nbh of } x\text{]} && (23) \end{aligned}$$

With these facts in hand we can argue that the r-neighborhoods of each substring x, u, y, v, z are identical in w_i and w_{i+1}. As before we only prove it for the boundary cases.

Consider first x and its last position j_1. Notice that $h'(j_1) = j_1$. In order to show that $\mathcal{N}_r^{w_i}(j_1) \cong \mathcal{N}_r^{w_{i+1}}(j_1)$ it remains to show that the r-prefix of $uyvz$ is the same as the r-prefix of $vyuz$. This is (23).

Now consider u. If u is empty, h' trivially preserves the r-neighborhoods of the elements in u. Otherwise, assume that u is not empty and consider its first position $j_1 + 1$ and let $k := h'(j_1 + 1)$. In order to show that $\mathcal{N}_r^{w_i}(j_1 + 1) \cong \mathcal{N}_r^{w_{i+1}}(k)$ we need to show that $S(x) = S(xvy)$ and that the $(r + 1)$-prefix of $uyvz$ is the same as the $(r + 1)$-prefix of uz. The former is (21) and the latter is immediate from (18) as u is not empty. Consider now the last position $j' - 1$ of u and let $k' := h'(j' - 1)$. In order to show that $\mathcal{N}_r^{w_i}(j' - 1) \cong \mathcal{N}_r^{w_{i+1}}(k')$ we need to show that the $(r + 1)$-suffix of xu is the same as the $(r + 1)$-suffix of $xvyu$ and that $P(yvz) = P(z)$. The latter is (18) while the former is immediate from (21) as u is not empty.

Finally consider the first position j' of y. Let $k := h'(j')$. In order to show that $\mathcal{N}_r^{w_i}(j') \cong \mathcal{N}_r^{w_{i+1}}(k)$ we need to show that $S(xu) = S(xv)$ and that the $(r+1)$-prefix of yvz is the same as the $(r+1)$-prefix of yuz. The former is (20) while the latter is immediate from (22) as y is not empty.

The other cases are treated similarly by symmetry. This completes the proof of Claim 5.3. □

Observe that for all $k \leq i_1$, $h_i(k)$ maps into x, and for all $k \in [i_1 + 1, i]$, $h_i(k)$ maps into v. Since $h_i(k)$ is monotone for $k \leq i$ and $h_{i+1}(i + 1)$ maps onto j', $h_{i+1}(k)$ is monotone for $k \leq i + 1$. Combining this fact with the claim implies that we have extended property (ii) to $i + 1$. With all properties satisfied the induction step is complete. □

5.2 Closure under Swaps

We now show that a language definable by a sentence of Arb-invariant FO(*Succ*) is closed under $(\log n)^c$-swaps for some constant c.

Lemma 5.4. *If L is a language definable by an Arb-invariant* FO(*Succ*) *sentence with alternation depth d then L is closed under $(\log n)^c$-swaps for any constant $c > d + 2$.*

To prove this lemma we observe that it suffices to consider swaps where the various neighborhoods are disjoint. This is because (i) when the isomorphic neighborhoods involved have substantial overlap, the swap operation has no effect and trivially preserves membership to L, and (ii) when the isomorphic neighborhoods have small overlap, restricting their radius slightly yields isomorphic neighborhoods that are disjoint. In Section 5.2.1 we show that Arb-invariant FO(*Succ*) is closed under disjoint swaps, and in Section 5.2.2 we formalize (i) and (ii) by showing that closure under disjoint swaps implies closure under general swaps modulo a small constant factor increase in the isomorphism radius. We combine these two steps to prove Lemma 5.4.

5.2.1 Disjoint Swaps

We weaken the condition for closure under r-swaps slightly by only considering neighborhoods $\mathcal{N}_r^w(i)$, $\mathcal{N}_r^w(i')$, $\mathcal{N}_r^w(j)$, and $\mathcal{N}_r^w(j')$ which are pairwise disjoint. We call this closure under *disjoint* r-swaps. We argue that languages definable in Arb-invariant FO(*Succ*) are closed under disjoint $(\log n)^c$-swaps, for some constant c depending on the alternation-depth of the language. One way to prove this fact is by mimicking our proof of Gaifman locality for the special case of string structures. Alternatively, we can use our Gaifman locality result as a blackbox. We follow the

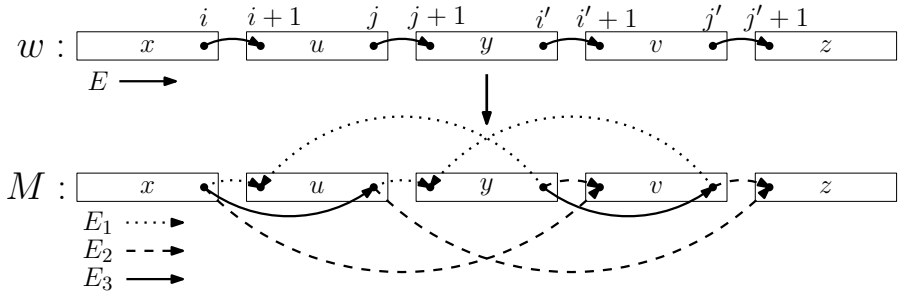

Figure 11: Constructing the structure M from the string w.

latter approach. The idea is that given a pair of strings w and w' of length n which witness the violation of the closure-under-$(\log n)^c$-swaps property for a sentence ϕ, we can derive: (i) a τ-structure M of size n with two tuples that have isomorphic neighborhoods up to distance $\Omega((\log n)^c)$, and (ii) an FO(τ, Arb) formula ψ distinguishing these tuples that is Arb-invariant with respect to M. We instantly conclude by applying our Gaifman locality theorem (Theorem 4.1) to produce a contradiction.

Proposition 5.5. *If L is a language definable by an Arb-invariant* FO(*Succ*) *sentence with alternation depth d then L is closed under* disjoint $(\log n)^c$*-swaps for any constant $c > d + 2$.*

Proof. Let ϕ be an Arb-invariant FO(*Succ*) sentence with alternation depth d defining L. Suppose that L is not closed under disjoint r-swaps, where $r := (\log n)^c$ and c is a large enough constant depending only on ϕ that will become apparent during the proof. Then there exists an infinite class of equal-length string pairs $\mathcal{W}$, such that for every pair $\langle w, w' \rangle \in \mathcal{W}$, the conditions for disjoint r-swaps are satisfied for the pair, but $w \in L$ and $w' \notin L$.

Consider one such pair $\langle w, w' \rangle \in \mathcal{W}$ and let $n := |w| = |w'|$. Let u, v, x, y, z be as in the definition of r-swaps, with $w = xuyvz$ and $w' = xvyuz$. Let i, i', j, j' denote the positions in w supplied by the definition of r-swaps. Using the assumed disjointness property, the neighborhoods $\mathcal{N}_r^w(i)$, $\mathcal{N}_r^w(i')$, $\mathcal{N}_r^w(j)$, and $\mathcal{N}_r^w(j')$ are all disjoint, and i, i', j, and j' are distinct positions in w.

Recall that w and w' can be seen as labeled graphs where the edge relation is called E. Consider the schema containing three extra binary relations E_1, E_2, and E_3. We construct a structure M from w over this extended schema by slightly modifying E. This construction is diagrammed in Figure 11. The purpose of the relations E_1, E_2, and E_3 is to mark the boundary vertices of u and v so that the boundaries can easily be recovered, and at the same time ensure that the neighborhoods around u and v in M appear identical. Note that the E-edges leaving i, j, i', and j' in w are eliminated in M, and all other E-edges of w are unchanged. M has the following property.

Claim 5.6. $\mathcal{N}_{r-1}^M(i, j, i', j') \cong \mathcal{N}_{r-1}^M(i', j', i, j)$.

Proof. To show this, we describe a witnessing isomorphism π. Let π act as the identity on $(r-1)$-prefixes of u, y, v, and z. Let π take the r-suffix of x to the r-suffix of y and *vice versa*. Let π take the r-suffix of u to the r-suffix of v and *vice versa*. This mapping is well-defined because the r-neighborhoods around i, j, i', and j' are all disjoint. It is now routine to verify that π is an isomorphism as claimed using the facts that $\mathcal{N}_r^w(i) \cong \mathcal{N}_r^w(i')$, $\mathcal{N}_r^w(j) \cong \mathcal{N}_r^w(j')$, and that these neighborhoods are disjoint. □

We can use the Arb-invariant FO(*Succ*) sentence ϕ to construct a formula ψ with four free variables x_1, y_1, x_2, y_2 that does the following: When evaluated in M, $\psi(i,j,i',j')$ simulates ϕ on w while $\psi(i',j',i,j)$ simulates ϕ on w', and for all other tuples ψ rejects. The formula ψ is constructed from ϕ as follows. In ϕ, replace all atoms $E(x,y)$ with

$$\begin{aligned}\theta(x,y,x_1,y_1,x_2,y_2) := E(x,y) \vee ((x = x_1 \vee x = y_1) \wedge E_1(x,y)) \\ \vee ((x = x_2 \vee x = y_2) \wedge E_2(x,y)).\end{aligned}$$

In order to ensure that on M ψ rejects when the tuple is not (i,j,i',j') or (i',j',i,j), we explicitly test for these inputs using the E_3-edge and reject if not found.

$$\begin{aligned}\psi(x_1,y_1,x_2,y_2) := \; & E_3(x_1,y_1) \wedge E_3(x_2,y_2) \wedge x_1 \neq x_2 \\ & \wedge \phi_{E(x,y) \leftarrow \theta(x,y,x_1,y_1,x_2,y_2)}(x_1,y_1,x_2,y_2).\end{aligned}$$

Here the notation indicates that we are replacing all occurrences of the relation $E(x,y)$ in ϕ by the relation $\theta(x,y,x_1,y_1,x_2,y_2)$. ψ is Arb-invariant with respect to M. To see this, observe that when the input tuple is not (i,j,i',j') or (i',j',i,j), the formula always rejects. In the other case ϕ is effectively evaluated on either w or w', which are strings, hence the action of ϕ is Arb-invariant by hypothesis.

Observe that ψ is a 4-ary formula that is defined only with respect to ϕ, and has the same alternation depth as ϕ. Applying Theorem 4.1 to ψ we see that for any constant $c' > d+2$ and for $n \geq n_{\psi,c'}$, ψ is Gaifman $(\log n)^{c'}$-local for structures for which it is Arb-invariant. Now, the infinite class of r-swap closure violations $\mathcal{W}$ with respect to ϕ induces an infinite class of structures $\mathcal{M}$ with Gaifman $(r-1)$-locality violations with respect to the formula ψ, where each violation is of the form $\langle M, \bar{a} := (i,j,i',j'), \bar{b} := (i',j',i,j)\rangle$, and ψ is Arb-invariant with respect to the structures in $\mathcal{M}$. If we pick $c' < c$, this infinite class of violations allows us to select an input length n which is at least $n_{\psi,c'}$ and large enough to make $r - 1 = (\log n)^c - 1 \geq (\log n)^{c'}$. This violates Theorem 4.1, and completes the proof. □

5.2.2 From Disjoint Swaps to General Swaps

We now argue that languages which are closed under disjoint swaps are also closed under swaps. Consider a language L which is closed under disjoint $r'(n)$-swaps and a pair of sufficiently long strings $w = xuyvz$ and $w' = xvyuz$ which satisfy the isomorphism conditions of an $r(n)$-swap. It suffices to argue that L does not distinguish w and w'.

We observe that when the neighborhoods of the substrings overlap a large amount the neighborhood isomorphisms induce periodic behavior within the $r(n)$-neighborhoods, so much so that the substrings uyv and vyu become identical. This implies that $w = w'$, and hence $w \in L$ iff $w' \in L$. This takes care of the case of large neighborhood overlap. We then focus on the case where the $r(n)$-neighborhoods of the substrings only overlap a small amount and show that there is freedom to select slightly smaller neighborhoods (of radius $r'(n)$) that are pairwise disjoint, though still induce the same effective swapping. This allows us to apply the closure of L under disjoint $r'(n)$-swaps to conclude that L does not distinguish w and w'. We now formalize this approach.

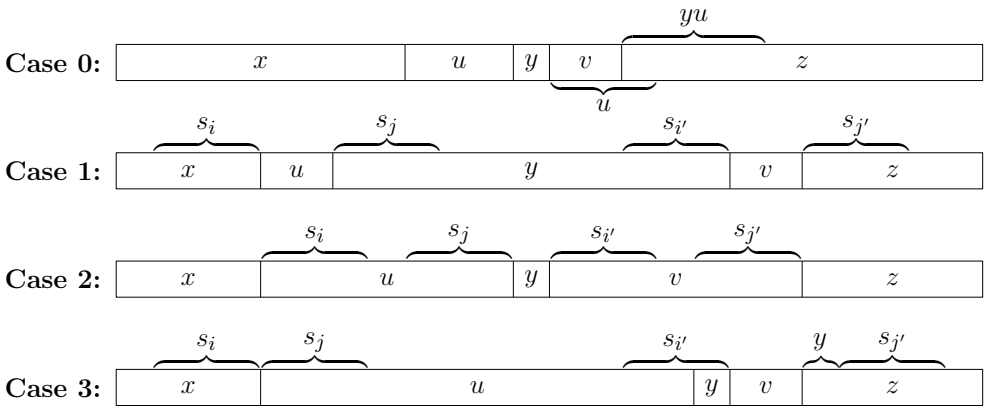

Figure 12: The cases in the proof of Lemma 5.4.

Proposition 5.7. *Let L be a language and r be a function $\mathbb{N} \to \mathbb{R}_{\geq 0}$. If L is closed under disjoint $\frac{r(n)}{14}$-swaps then L is closed under $r(n)$-swaps.*

Proof. Suppose that L is closed under disjoint $r'(n)$-swaps, where $r'(n) = r(n)/c$ for some constant c to be determined later. Let n_0 be the associated constant. Fix a length $n \geq n_0$, and a pair of length n strings $w := xuyvz$ and $w' := xvyuz$ whose substrings satisfy the isomorphism conditions for a $r(n)$-swap. To prove the lemma it suffices to argue that $w \in L$ iff $w' \in L$. We drop the parameter to r' and r in what follows.

The isomorphism conditions of r-swaps imply it suffices to consider $|x| \geq r$. Otherwise, to satisfy the conditions it must be that $|u| = |y| = |v| = 0$, and hence $w = xz = w'$, and we trivially conclude $w \in L$ iff $w' \in L$. Analogously, $|z| \geq r$.

We proceed by case analysis on the sizes of u, y, and v relative to r'. If the lengths of u, y, and v are all short relative to r', then their respective neighborhoods overlap and u, y, and v are present in each neighborhood. The neighborhood isomorphisms drag these three substrings around implying that $uyv = vyu$, and hence that $w = w'$. This is Case 0.

In the three remaining cases we use the closure of L under disjoint r'-swaps to conclude that L does not distinguish w and w'. In each of these cases, we determine a set of disjoint substrings $s_i, s_j, s_{i'}, s_{j'}$ occurring in this order in w, each of length $2r'$. We show that $s_i = s_{i'}$ and $s_j = s_{j'}$. Let i, j, i', j' be the respective middle positions of these substrings, then the r'-neighborhoods of i and i', and of j and j' are isomorphic. We argue that the result of swapping the substrings $w[i+1, j]$ and $w[i'+1, j']$ in w yields w', and thus w and w' are separated by a disjoint r'-swap. Since L is closed under disjoint r'-swaps and $n \geq n_0$, $w \in L$ iff $w' \in L$, concluding the proof in each case. See Figure 12 for a diagram of the four cases.

Case 0. $|u|, |y|, |v| < 4r'$.

Let i, i', j, and j' be the positions in w preceding the substrings u, y, v, and z, respectively. We claim that if $r \geq 12r'$ then $uyv = vyu$. This implies that $w = w'$ and hence $w \in L$ iff $w' \in L$. To see the claim, notice first that because $r \geq 4r'$ and $|u| < 4r'$, the right part of the r-neighborhood around i starts with u. Therefore, as $\mathcal{N}_r^w(i) \cong \mathcal{N}_r^w(i')$, u is a prefix of vz. A similar reasoning around j and j', using the fact that $r \geq 8r'$, shows that z and yvz have the same prefix of length

$8r'$. As u is a prefix of vz and $|yu| < 8r'$, this implies that yu is a prefix of z. Therefore vyu is a prefix of vz. Now, because $r \geq 12r'$ and $\mathcal{N}_r^w(i) \cong \mathcal{N}_r^w(i')$, $uyvz$ and vz have the same prefix of length $12r'$. Because $|vyu| < 12r'$, this implies that vyu is a prefix of $uyvz$ and therefore $vyu = uyv$.

We now consider the cases with less overlap between neighborhoods.

Case 1. $|y| \geq 4r'$.

Let s_i be the $2r'$-suffix of x and $s_{i'}$ be the $2r'$-suffix of y. Let s_j be the $2r'$-prefix of y and $s_{j'}$ be the $2r'$-prefix of z. Since $2r' \leq |y|$, s_j and $s_{i'}$ do not overlap. Since $|x|, |z| \geq r > 2r'$, s_i and $s_{j'}$ are fully realized. The given neighborhood isomorphisms imply that $s_i = s_{i'}$ and $s_j = s_{j'}$. Inspection shows that swapping $w[i+1, j]$ with $w[i'+1, j']$ produces w', completing the case.

Case 2. $|u|, |v| \geq 4r'$.

Let s_i be the $2r'$-prefix of u and $s_{i'}$ be the $2r'$-prefix of v. Let s_j be the $2r'$-suffix of u and $s_{j'}$ be the $2r'$-suffix of v. Since $|u|, |v| \geq 4r'$ these substrings $s_i, s_{i'}, s_j, s_{j'}$ are pairwise disjoint. Further, using the known neighborhood isomorphisms, $s_i = s_{i'}$ and $s_j = s_{j'}$. Inspection shows that swapping $w[i+1, j]$ with $w[i'+1, j']$ produces w', completing the case.

Case 3. $|v|, |y| < 4r', |u| \geq 4r'$ (analogously, $|u|, |y| < 4r', |v| \geq 4r'$).

Let s_i be the $2r'$-suffix of x and $s_{i'}$ be the $2r'$-suffix of uy. Using the given neighborhood isomorphisms and the fact $|u| \geq 4r'$, s_i and $s_{i'}$ are disjoint and equal. Let s_j be the $2r'$-prefix of u. Since $|u| \geq 4r'$, s_j and $s_{i'}$ are disjoint. Let $s_{j'} := z[|y|, |y| + 2r']$. Since $|z| \geq r$ and $|y| < 4r'$, $s_{j'}$ is fully realized if $6r' < r$. We need to argue that $s_j = s_{j'}$.

Observe that the last element of u and the first element of z are within $8r'$ of each other. Without loss of generality $|yv| > 0$, because otherwise $w = xuz = w'$. The fact that r-neighborhoods around the point at the end of u and the point immediately before z are isomorphic implies that the neighborhood following v contains a sequence of many repetitions of the string yv. This string yv repeats up to distance at least $r - 8r'$ into z. Therefore, if $14r' \leq r$, the string $s_{j'}$ starts with v followed by repetitions of yv for the entire length of $s_{j'}$, and is the same as $2r'$-prefix of vz. Hence the neighborhood isomorphism between the last point of x and the point preceding v implies that $s_j = s_{j'}$.

Write $z = yz'$, then $w = xuyvyz'$. Swapping the substring vy with the empty string between x and u produces $xvyuyz' = xvyuz = w'$. This completes the case.

Choosing $c = 14$ suffices to satisfy the assumptions made in each case and completes the proof.□

Combining Proposition 5.5 and Proposition 5.7 yields a proof of Lemma 5.4.

Proof of Lemma 5.4. Let L be a language definable by an Arb-invariant FO(*Succ*) sentence with alternation depth d. For any constant $c' > d+2$ we have, by Proposition 5.5, that L is closed under disjoint $(\log n)^{c'}$-swaps. By Proposition 5.7, L is closed under $14(\log n)^{c'}$-swaps. If we pick $c' < c$, then $14(\log n)^{c'} < (\log n)^c$ for n sufficiently large. It follows that L is closed under $(\log n)^c$-swaps. None

5.3 Upper Bound

We are now ready to prove Theorem 5.1. It is essentially a combination of Lemma 5.4 and Lemma 5.2 with a reduction from general formulas to sentences.

Proof of Theorem 5.1. We first prove the case of sentences. Let L be a language definable by an Arb-invariant FO(*Succ*) sentence with alternation depth d. Let $c > c' > d + 2$. By Lemma 5.4 L is closed under $(\log n)^{c'}$-swaps. Consider now a pair of strings w, w' such that $|w| = |w'| = n$ for a sufficiently large n. Assume that $w \equiv_{(\log n)^c} w'$. By Lemma 5.2 there is a sequence of $((\log n)^c - 1)$-swaps that turns w into w'. For large enough n, $(\log n)^c - 1 \geq (\log n)^{c'}$, and therefore L is closed under such swaps. For this sufficiently large input length none of these swaps affect membership in L, hence $w \in L$ iff $w' \in L$ and the theorem is proved for the case of sentences.

For the general case, we can mark the free variables by new unary predicates, one per free variable. To the initial formula we can associate a sentence quantifying existentially over these elements and then evaluating the initial formula. For any r, the initial query is Hanf r-local if its associated sentence is also Hanf r-local. Also, if the initial query is Arb-invariant then so is its associated sentence. We may assume that the quantifier depth of the initial formula is at least one; otherwise, the formula is trivially 0-local (see the proof of Theorem 4.1). Observe that a formula is Hanf r-local and Arb-invariant iff the negation of the formula has the same property. This allows us to further assume without loss of generality that the initial formula begins with an existential quantifier (otherwise, we take the negation), and hence the resulting sentence has the same alternation depth as the initial formula. Theorem 5.1 follows from case of sentences we have just proved. None

We believe that the techniques developed in this section extend to trees and that Hanf $(\log n)^{O(1)}$-locality holds for Arb-invariant FO sentences over trees. See Section 7 for more details.

5.4 Lower Bound

We use a similar idea as in the proof of the lower bound of Theorem 1.1 to show the Hanf locality lower bound for Arb-invariant FO(*Succ*) claimed in Theorem 1.2.

Proof of Theorem 1.2 - Lower Bound. Fix a constant $c > 0$ and consider the alphabet $A := \{a, b, e, f\}$. We will construct an Arb-invariant sentence ϕ_c that is satisfied by exactly those strings w of the form $(a|b|e)^* f^*$, where (i) the total number of a's, b's, and e's in w is less than $(\log n)^{c+1}$ (where n denotes the length of w), and (ii) the first occurrence of a in w is somewhere to the left of the first occurrence of a b in w.

To note that ϕ_c is *not* Hanf $(\log n)^c$-local, consider, for a sufficiently large n, the string w of length n that, for $m := 2(\log n)^c + 1$, is of the form $e^m a e^m b e^m f^*$. Furthermore, let w' be the string obtained from w by swapping the letters a and b. Note that the strings w and w' are chosen in such a way that

$$w \equiv_{(\log n)^c} w'.$$

Furthermore, w and w' are both of the form $(a|b|e)^* f^*$ and satisfy the following: Since n is sufficiently large, the total number of a's, b's, and e's in w as well as in w' is less than $(\log n)^{c+1}$. In w, the first occurrence of the letter a is somewhere to the left of the first occurrence of the letter b; however, in w' this is not the case. In summary, we thus obtain that $w \models \phi_c$ and $w' \not\models \phi_c$. Hence, the strings w and w' witness that ϕ_c is not Hanf $(\log n)^c$-local.

To conclude the proof it remains to show how to construct the formula ϕ_c. For this, we use the Arb-invariant formula $reach_d(x, y)$ from Lemma 4.6 for $d := c + 1$. Let $\varrho(x, y)$ be the formula obtained from $reach_d(x, y)$ by replacing every occurrence of an atomic formula of the form $S(z)$ by a formula stating that position z carries one of the letters a, b, or e. Choosing ϕ_c to be the

sentence stating that there exist positions x and y that carry the letters a and b, respectively, such that $\varrho(x, y)$ is satisfied, we obtain an Arb-invariant sentence that, when evaluated in a string w of the form $(a|b|e)^*f^*$, states that (i) the total number of a's, b's, and e's in w is less than $(\log n)^{c+1}$ (where n denotes the length of w), and (ii) the first occurrence of a in w is somewhere to the left of the first occurrence of a b in w. This concludes the proof for the lower bound part of Theorem 1.2. None

Remark 5.8. *By Remark 4.8 the formula ϕ_c constructed in the above proof only uses the numerical predicates of addition $+$ and multiplication $\times$. Thus, the lower bound of Theorem 1.2 already holds for such formulas.*

We point out an alternate route for proving the lower bound in Theorem 1.2, namely by establishing the lower bound in Theorem 1.1 for strings. The former follows from the latter because if a formula is Hanf r-local w.r.t. (M, M) then it is Gaifman $(3r + 1)$-local w.r.t. M (recall Definitions 2.1 & 2.2) [HLN99]. This alternate route yields an Arb-invariant *formula*, rather than *sentence*, that is not Hanf $(\log n)^c$-local for a given constant c, but the formula can be transformed into a sentence using the translation given in the proof of Theorem 5.1. We did not follow this route because establishing the lower bound in Theorem 1.1 for strings would require the schema to have both a unary and a binary predicate, whereas our proof of the lower bound in Theorem 1.1 only needs the minimal requirement of a binary predicate.

6 Implications for Regular Languages

In this section we show that the locality results we proved in the previous sections have nice consequences for regular languages. It is shown in [BS09b] that each order-invariant sentence of FO(*Succ*) has an equivalent FO(τ_s) formula over strings. In other words, over strings, a linear order used in an order-invariant way does not bring any new expressive power. This is no longer the case when arithmetic is allowed. For instance, in the presence of addition the logic can express parity of the length of a string. To see this, consider the following Arb-invariant sentence which expresses that the string has even length.

$$\exists x, y \quad y = x + x \ \wedge \ \neg(\exists z \ E(y, z))$$

If addition is the only numerical predicate allowed, then it is shown in [SS10] that addition-invariant FO(*Succ*) definable regular languages are exactly those expressible in FO(τ_s, lm), where lm is the family of predicates testing the length of a string modulo some fixed number. We now show that adding any other numerical predicate does not allow new regular languages to be defined, i.e., we prove Theorem 1.3.

In order to do so, we make use of an equivalent characterization of definability of regular languages in FO(τ_s, lm) in terms of closure under certain operations. We first introduce those operations, which are themselves based on the notion of idempotence for a regular language.

Let L be a regular language, a string e of A^* is said to be *idempotent* (for L, but we will omit L when it is understood from the context) if it is not the empty string and for all $u, v \in A^*, uev \in L$ iff $ueev \in L$. Let $\omega \in \mathbb{N}$ be the smallest positive integer such that for all $u \in A^+$, u^ω is idempotent. Note that ω is well-defined.

A regular language L is *closed under swaps* if for all $x, u, y, v, z, e, f \in A^*$ such that e, f are idempotent we have:

$$xeufyevfz \in L \quad \text{iff} \quad xevfyeufz \in L. \tag{24}$$

A regular language L is *closed under transfers* if for all $x, u, y, v, z \in A^*$ such that $|u| = |v|$ we have:

$$xu^\omega uyv^\omega z \in L \quad \text{iff} \quad xu^\omega yvv^\omega z \in L. \tag{25}$$

The following result was shown in [SS10].

Lemma 6.1 ([SS10]). *Let L be a regular language. Then L is definable in $\mathrm{FO}(\tau_s, lm)$ iff L is closed under transfers and under swaps.*

Lemma 6.1 allows us to prove Theorem 1.3 by arguing that the regular languages definable in Arb-invariant FO(*Succ*) are closed under transfers and swaps. In Section 6.1 we consider a generalization of the notion of closure under transfers for an arbitrary language L, namely closure under r-transfers, where $r : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is a function. We prove that Arb-invariant FO(*Succ*) is closed under $(\log n)^{O(1)}$-transfers. In Section 6.2 we use a pumping argument to show that for a regular language L, closure under r-transfers for any function r implies closure under transfers. Using a similar pumping argument, we show that for regular languages, Hanf locality implies closure under swaps. The upper bound in Theorem 1.2 then concludes the proof of Theorem 1.3.

6.1 Closure under Transfers

Let $r \in \mathbb{N}$ and $w \in A^*$. A string $w' \in A^*$ is obtained from w by an *r-transfer operation* if $w = xu^r uyv^r z$ and $w' = xu^r yv^r vz$ for some $x, u, y, v, z \in A^*$ with $|u| = |v| \neq 0$.

Let $r : \mathbb{N} \to \mathbb{R}_{\geq 0}$. A language L is said to be *closed under $r(n)$-transfers* if there exists a $n_0 \in \mathbb{N}$ such that for all strings $w, w' \in A^*$, with $|w| = n > n_0$, if w' is obtained from w by a $r(n)$-transfer operation then we have:

$$w \in L \quad \text{iff} \quad w' \in L.$$

We begin by proving a lemma similar to Lemma 4.2, but specialized to the task at hand. We show that an Arb-invariant FO(*Succ*) sentence that can distinguish strings separated by a transfer can be used to solve the hard promise problem from Theorem 2.3.

Lemma 6.2. *Let $m \in \mathbb{N}$. Let x, u, y, v, z be strings over A with $|u| = |v|$. Let $w := xu^{3m+1}yv^{3m+1}z$ and $w' := xu^{3m}yv^{3m+2}z$. Suppose C is a circuit that accepts all strings in $\mathrm{Rep}(w)$ and rejects all strings in $\mathrm{Rep}(w')$, then there is a circuit $\tilde{C}$ with the same size and depth as C that distinguishes $|b|_1 = m$ and $|b|_1 = m + 1$ for $b \in \{0,1\}^{2m}$.*

Proof. Let $b := b_1 b_2 \dots b_{2m}$ be a string of $2m$ Boolean variables. We design a string w_b that is easy to compute from b and has the following property:

$$\begin{aligned} &\text{If } |b|_1 = m, \text{ then } w_b \cong w. \\ &\text{If } |b|_1 = m + 1, \text{ then } w_b \cong w'. \end{aligned}$$

Once we have such a string w_b, we argue as follows. Consider any binary encoding $\Gamma_b \in \mathrm{Rep}(w_b)$. When $|b|_1 = m$, $C(\Gamma_b)$ accepts, because $w_b \cong w$. Similarly, when $|b|_1 = m + 1$, $C(\Gamma_b)$ rejects,

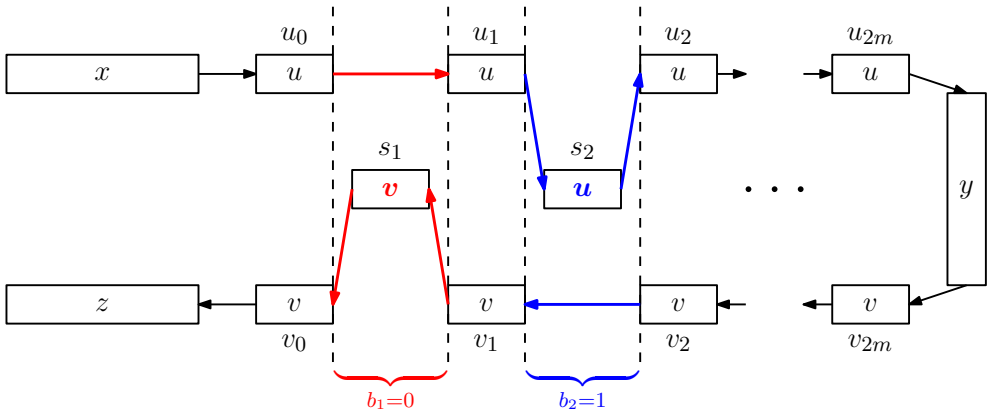

Figure 13: Constructing the string w_b from b. (Note that the strings v and z are drawn right to left.)

because $w_b \cong w'$. Thus, $\tilde{C}(b) := C(\Gamma_b)$ yields the required circuit provided Γ_b is sufficiently easy to compute from b. We construct the string w_b and a representation $\Gamma_b \in \mathrm{Rep}(w_b)$ as follows. Let

$$w_b = xu^{4m+1-|b|_1}yv^{2m+1+|b|_1}z. \tag{26}$$

We construct w_b as in Figure 13.

Add the strings x, y and z to w_b (this includes the vertices, internal edges and labels). Construct $2m+1$ copies of the strings u and v and add them to w_b. Call these copies $u_0, u_1, ..., u_{2m}$, and $v_0, v_1, ..., v_{2m}$, respectively. Add edges from x to u_0, u_{2m} to y, y to v_{2m}, and v_0 to z, i.e., connect the vertex at the end of the former string to the vertex at the beginning of the latter. Construct $2m$ strings of length $|u| = |v|$ with no labels and add them to w_b. Call these strings $s_1, ..., s_{2m}$. Observe that w_b contains exactly $|w| = |w'|$ vertices thus far, though w_b itself is not yet a string because not all edges or labels have been set.

For each $i \in [2m]$:

1. If $b_i = 0$, connect v_i to s_i, s_i to v_{i-1}, and u_{i-1} to u_i, and label s_i as v.
2. If $b_i = 1$, connect u_{i-1} to s_i, s_i to u_i and v_i to v_{i-1}, and label s_i as u.

Note that the w_b constructed is a string of length $|w| = |w'|$ and can be written as in (26). Since the presence of each edge and the value of each label depends on at most one bit of b, the string w_b can be encoded in binary $\Gamma_b \in \mathrm{Rep}(w_b)$ so that each bit of the encoding is either a constant or a literal from b (see the proof of Lemma 4.2 for more details). This allows us to define the circuit $\tilde{C}(b) := C(\Gamma_b)$ which has depth and size no larger than C, and completes the proof. □

With this lemma we can prove that Arb-invariant FO(*Succ*) is closed under $(\log n)^{O(1)}$-transfers following the same approach as in the proof of Theorem 1.1.

Lemma 6.3. *If L is a language definable in Arb-invariant* FO(*Succ*) *then there is a $c \in \mathbb{N}$ such that L is closed under $(\log n)^c$-transfers.*

Proof. Immediate from Lemma 6.2 and Theorem 2.3 via the same type of manipulation that proves Theorem 1.1. □

6.2 Definability under Arb-Invariance

We now use Lemma 6.3 to show that a regular language is definable in Arb-invariant FO(*Succ*) iff it is definable in FO(*Succ*, *lm*). The "only if" direction is straightforward as any predicate of the form *lm* can be expressed in Arb-invariant FO(*Succ*) (actually only addition is needed). For the "if" direction we show that a pumping argument combined with Hanf $(\log n)^{O(1)}$-locality and $(\log n)^{O(1)}$-transfers implies closure under swaps and transfers, and we can conclude using Lemma 6.1.

Proof of Theorem 1.3. As L is definable in Arb-invariant FO(*Succ*), by Theorem 5.1 there is a constant $c \in \mathbb{N}$ such that L is Hanf $(\log n)^c$-local. Hence there is a constant n_0 such that for all strings w, w' with $|w| = |w'| > n_0$, $w \equiv_{(\log n)^c} w'$ implies $w \in L$ iff $w' \in L$.

Consider x, u, y, v, z, e, and f as in the hypothesis for closure under swaps. As e and f are idempotents we have for all $i \in \mathbb{N}$, $xe^iuf^iye^ivf^iz \in L$ iff $xeufyevfz \in L$. Take i large enough so that $i \cdot |e|, i \cdot |f| \geq (\log |xe^{2i}uf^{2i}ye^{2i}vf^{2i}z|)^c \geq (\log n_0)^c$. Notice that $xe^{2i}uf^{2i}ye^{2i}vf^{2i}z \equiv_{(\log n)^c} xe^{2i}vf^{2i}ye^{2i}uf^{2i}z$ via the bijection sending respectively xe^i, e^iyf^i, e^iuf^i, e^ivf^i and f^iz to their corresponding substring in the other string. Hence by Theorem 5.1 we have $xe^{2i}uf^{2i}ye^{2i}vf^{2i}z \in L$ iff $xe^{2i}vf^{2i}ye^{2i}uf^{2i}z \in L$. But again, as e and f are idempotents, this implies $xeufyevfz \in L$ iff $xeufyevfz \in L$. Therefore L is closed under swaps.

The pumping argument for closure under transfers is identical. By Lemma 6.3 there are constants c and n_0 such that L is closed under $(\log n)^c$-transfers for strings of length bigger than n_0.

Consider x, u, y, v, z as in the hypothesis for closure under transfers. As u^ω and v^ω are idempotents we have for all $i \in \mathbb{N}$, $xu^{\omega \cdot i}uyv^{\omega \cdot i}z \in L$ iff $xu^\omega uyv^\omega z \in L$. Take i large enough so that $i \cdot \omega \geq (\log |xu^{\omega \cdot i}uyv^{\omega \cdot i}z|)^c \geq (\log n_0)^c$. Hence we can apply closure under $(\log n)^c$-transfers and by Lemma 6.3 we have $xu^{\omega \cdot i}uyv^{\omega \cdot i}z \in L$ iff $xu^{\omega \cdot i}yvv^{\omega \cdot i}z \in L$. But again, as u^ω and v^ω are idempotents, this implies $xu^\omega uyv^\omega z \in L$ iff $xu^\omega yvv^\omega z \in L$. Therefore L is closed under transfers.

This concludes the proof of Theorem 1.3. None

7 Further Research

We end with a few suggestions for further research.

We have established the precise level of locality of Arb-invariant FO formulas for the Gaifman notion of locality. As pointed out in [Gro09]: "It would be interesting to see a small complexity class like uniform AC^0 [...] captured by a logic." This remains an open problem – recall that although Arb-invariant FO does capture AC^0, it does not have an effective syntax. Note that *over regular languages*, we do have an effective syntax as we have shown that Arb-invariant FO(*Succ*) has exactly the same expressive power as FO(τ_s, *lm*).

Recall that order-invariant FO queries are Gaifman local with a *constant* locality radius [GS00], whereas our Theorem 1.1 shows that Arb-invariant FO queries are Gaifman local with a polylogarithmic locality radius. We also constructed, for any constant c, formulas with locality radius larger than $(\log n)^c$ only using the numerical predicates of addition $+$ and multiplication $\times$. (cf. Remark 4.8). It is an open question whether multiplication is really necessary for constructing such

formulas. The authors of [GS00] conjectured that this is indeed the case — in fact, they conjectured that Arb-invariant FO queries that only use the numerical predicate of addition + (so-called addition-invariant FO queries) are Gaifman local with a *constant* locality radius.

A further open question is whether our upper bound for Gaifman locality also holds for Hanf locality on arbitrary structures. We have established this bound for string structures only. We believe that a similar argument should also work for tree structures. We end with a sketch of that argument.

Hanf locality for trees We indicate here how the proof of Section 5 could be adapted to the tree case. We would introduce two kinds of swaps, a "vertical swap" and an "horizontal swap". The *vertical r-swap* resembles the r-swap operation for strings. It is based on four nodes of the tree, all aligned on a path from its root to one of its leaf and such that the r-neighborhood of the first node (resp. second node) is identical to the r-neighborhood of the third node (resp. fourth node). It then swaps the corresponding parts of the tree. The *horizontal r-swap* considers two nodes having the same r-neighborhoods and such that the subtrees rooted at those nodes do not intersect. It then swaps the two subtrees.

That sets of trees defined by Arb-invariant sentences are closed under vertical and horizontal $(\log n)^{O(1)}$-swaps would be proved exactly as for Lemma 5.4 by first showing that it is enough to consider disjoint neighborhoods and solving the disjoint neighborhoods case using a reduction to Gaifman locality.

The link between these swap operations and Hanf locality could be proved using the same ideas as for the proof of Lemma 5.2.

As before, it follows almost by definition that horizontal and vertical r-swap operations do not change the $\equiv_r$-class of a tree as each of these swaps preserves r-neighborhoods.

For the opposite direction we would prove that if two trees are in a same $\equiv_r$-class then there is a sequence of $(r-1)$-swap operations transforming one into the other using the same inductive argument as in the proof of Lemma 5.2. We recall the sketch of Lemma 5.2 and adapt it to trees to illustrate the respective role of the two swap operations. Assuming that $t \equiv_r t'$, we would transform by induction t' using $(r-1)$-swap operations in order to embed larger and larger prefixes of t within t'. Over trees, a *prefix* is a set of nodes closed under the parent function and an embedding is a function on nodes preserving r-neighborhoods and the ancestor relationship. Eventually t itself will embed into t' and because $|t| = |t'|$ this implies that $t = t'$ as desired. For the induction step, consider the largest prefix u of t that can embed in t' via some embedding h. Let i be any boundary node of u, i.e. a node of u having a child i' not in t, and let $j := h(i)$. Because $t \equiv_r t'$, there is an node j' in t', outside the image of h, that has the same r-neighborhood as i'. Using the maximality of u, which rules out the case $j < j'$, we distinguish two cases.

In the first case we assume that $j' < j$, i.e., j' is an ancestor of j. Let i_1 be the node of u such that $h(i_1) < j'$ and no descendant of i_1 has this property. As $h(i_1) < j' < j$, by maximality of i_1 there is a child i'_1 of i_1 such that in t' we have $h(i_1) < j' < h(i'_1) \leq j$. As in the string case we derive, from the known child/parent relationships and the fact that an embedding preserves r-neighborhoods, equalities between suitable $(r-1)$-neighborhoods and can apply a vertical $(r-1)$-swap. We would then observe that the resulting tree embeds a prefix of t strictly bigger than u (it now also encompass i') and that it remains in the same $\equiv_r$-class. The induction can continue from here.

The second case is when j and j' are not related by the ancestor relationship. But then j' and

a child of j have the same $(r-1)$-neighborhood and their corresponding subtrees do not intersect. We can therefore apply a horizontal $(r-1)$-swap operation. If the resulting tree now embeds i', and all its ancestors, and remains in the same $\equiv_r$-class, it may not embed a prefix of t strictly bigger than u. This situation happens only when the subtree rooted at j' contains a node in the image of h. We cope with this situation as follows. Let i_1 be the node of u such that $h(i_1) < j'$ and no descendant of i_1 has this property. Then all the children of i_1 that are in the domain of h have their image under h inside the subtree of j'. Let k be a child of i_1. Notice that its $(r-1)$-neighborhood is the same as the one of the children of i_1. Therefore, after the initial horizontal $(r-1)$-swap between j' and some child of j, we can now apply another horizontal $(r-1)$-swap between k and the corresponding child of i_1, putting the subtree rooted in k below i_1 preserving the descendant relationship for this part of u. All children of i_1 could be treated this way independently because their subtrees do not intersect. Eventually, we would obtain a tree embedding at least u and i' and remaining in the same $\equiv_r$-class. The induction can continue from here.

References

[AB87] Noga Alon and Ravi B. Boppana. The monotone circuit complexity of boolean functions. *Combinatorica*, 7(1):1–22, 1987.

[AHV95] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[Ajt83] Miklos Ajtai. Σ^1_1-formulae on finite structures. *Annals of Pure and Applied Logic*, 24(1):1–48, 1983.

[Ajt89] Miklos Ajtai. First-order definability on finite structures. *Annals of Pure and Applied Logic*, 45(3):211–225, 1989.

[And85] Alexander E. Andreev. On a method for obtaining lower bounds for the complexity of individual monotone functions. *Soviet Math. Dokl.*, 31(3):530–534, 1985.

[AvMSS11] Matthew Anderson, Dieter van Melkebeek, Nicole Schweikardt, and Luc Segoufin. Locality of queries definable in invariant first-order logic with arbitrary built-in predicates. In *Proc. Intl. Colloq. on Automata, Languages and Programming (ICALP)*, volume 6756 of *Lecture Notes in Computer Science*, pages 368–379. Springer, 2011.

[BP89] Danièle Beauquier and Jean-Éric Pin. Factors of words. In *Proc. Intl. Colloq. on Automata, Languages and Programming (ICALP)*, volume 372 of *Lecture Notes in Computer Science*, pages 63–79. Springer, 1989.

[BS09a] Michael Benedikt and Luc Segoufin. Regular tree languages definable in FO and FO_{mod}. *ACM Transactions on Computational Logic*, 11(1), 2009.

[BS09b] Michael Benedikt and Luc Segoufin. Towards a characterization of order-invariant queries over tame structures. *Journal of Symbolic Logic*, 74(1):168–186, 2009.

[CGP99] Edmund M. Clarke, Orna Grumberg, and Doron A. Peled. *Model Checking*. MIT Press, 1999.

[DLM07] Arnaud Durand, Clemens Lautemann, and Malika More. Counting results in weak formalisms. In *Circuits, Logic, and Games*, number 06451 in Dagstuhl Seminar Proc. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany, 2007.

[EF99] Heinz-Dieter Ebbinghaus and Jörg Flum. *Finite model theory*. Springer, 1999.

[FSS84] Merrick Furst, James Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984.

[FSV95] Ronald Fagin, Larry J. Stockmeyer, and Moshe Y. Vardi. On monadic NP vs. monadic co-NP. *Information and Computation*, 120(1):78–92, 1995.

[Gai82] Haim Gaifman. On local and non-local properties. In J. Stern, editor, *Proc. Herbrand Symposium, Logic Colloquium 1981*, pages 105–135, 1982.

[GK11] Martin Grohe and Stephan Kreutzer. Methods for algorithmic meta theorems. In Martin Grohe and Johann Makowsky, editors, *Model Theoretic Methods in Finite Combinatorics*, volume 558 of *Contemporary Mathematics*. American Mathematical Society, 2011.

[Gro09] Martin Grohe. Fixed-point definability and polynomial time. In *Proc. Conf. for Computer Science Logic (CSL)*, volume 5771 of *Lecture Notes in Computer Science*, pages 20–23. Springer, 2009.

[GS00] Martin Grohe and Thomas Schwentick. Locality of order-invariant first-order formulas. *ACM Transactions on Computational Logic*, 1(1):112–130, 2000.

[Han65] William Hanf. Model-theoretic methods in the study of elementary logic. In J. W. Addison, L. Henkin, and A. Tarski, editors, *The Theory of Models*, pages 132–145. North-Holland, 1965.

[Hås86] Johan Håstad. *Computational limitations for small-depth circuits*. PhD thesis, MIT, 1986.

[HLN99] Lauri Hella, Leonid Libkin, and Juha Nurmonen. Notions of locality and their logical characterizations over finite models. *Journal of Symbolic Logic*, 64(4):1751–1773, 1999.

[Imm86] Neil Immerman. Relational queries computable in polynomial time. *Information and Control*, 68(1-3):86–104, 1986.

[Imm87] Neil Immerman. Languages that capture complexity classes. *SIAM Journal on Computing*, 16(4):760–778, 1987.

[Imm99] Neil Immerman. *Descriptive Complexity*. Springer-Verlag, 1999.

[Lib03] Leonid Libkin. Expressive power of SQL. *Theoretical Computer Science*, 296(3):379–404, 2003.

[Lib04] Leonid Libkin. *Elements of Finite Model Theory*. Springer, 2004.

[LMN93] Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, Fourier transform, and learnability. *Journal of the ACM*, 40(3):607–620, 1993.

[LN00] Leonid Libkin and Juha Nurmonen. Counting and locality over finite structures: A survey. In *Generalized Quantifiers and Computation, 9th European Summer School in Logic, Language, and Information (ESSLLI'97)*, volume 1754 of *Lecture Notes in Computer Science*, pages 18–50. Springer, 2000.

[Mak97] Johann A. Makowsky. Invariant definability (extended abstract). In *Computational Logic and Proof Theory, Proc. 5th Kurt Gödel Colloquium (KGC'97)*, volume 1289 of *Lecture Notes in Computer Science*, pages 186–202. Springer, 1997.

[Mak98] Johann A. Makowsky. Invariant definability and P/*poly*. In *Proc. Conf. for Computer Science Logic (CSL)*, volume 1584 of *Lecture Notes in Computer Science*, pages 142–158. Springer, 1998.

[Ott00] Martin Otto. Epsilon-logic is more expressive than first-order logic over finite structures. *Journal of Symbolic Logic*, 65(4):1749–1757, 2000.

[Raz85] Alexander A. Razborov. Lower bounds on the monotone complexity of some boolean functions. *Soviet Math. Dokl.*, 31:354–357, 1985.

[Ros07] Benjamin Rossman. Successor-invariant first-order logic on finite structures. *Journal of Symbolic Logic*, 72(2):601–618, 2007.

[Ros08] Benjamin Rossman. On the constant-depth complexity of k-clique. In *Proc. Symp. on the Theory of Computing (STOC)*, pages 721–730. ACM, 2008.

[Ros10] Benjamin Rossman. *Average-Case Complexity of Detecting Cliques*. PhD thesis, MIT, 2010.

[Sch96] Thomas Schwentick. On winning Ehrenfeucht games and monadic NP. *Annals of Pure and Applied Logic*, 79(1):61–92, 1996.

[Sch05] Nicole Schweikardt. Arithmetic, first-order logic, and counting quantifiers. *ACM Transactions on Computational Logic*, 6(3):634–671, 2005.

[SS10] Nicole Schweikardt and Luc Segoufin. Addition-invariant FO and regularity. In *Proc. Symp. on Logic in Computer Science (LICS)*, pages 273–282. IEEE Computer Society, 2010.

[SSS09] Nicole Schweikardt, Thomas Schwentick, and Luc Segoufin. Database theory: Query languages. In M. J. Atallah and M. Blanton, editors, *Algorithms and Theory of Computation Handbook*. CRC Press, 2nd edition, 2009. Chapter 19.

[TW85] Denis Thérien and Alex Weiss. Graph congruences and wreath products. *Journal of Pure and Applied Algebra*, 36:205–215, 1985.

[Var82] Moshe Vardi. The complexity of relational query languages. In *Proc. Symp. on the Theory of Computing (STOC)*, pages 137–146. ACM, 1982.

[Yao85] Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles (Preliminary Version). In *Proc. Symp. on Foundations of Computer Science (FOCS)*, pages 1–10. IEEE Computer Society, 1985.