Non-definability of languages by generalized first-order formulas over $(\mathbb{N},+)$

Andreas Krebs[1] and Sreejith A V[2]

[1] Wilhelm-Schickard Institut, Universtität Tübingen, Germany
[2] Institute of Mathematical Sciences, Chennai, India

Abstract. We consider first-order logic with monoidal quantifiers over words. We show that all languages with a neutral letter, definable using the addition predicate are also definable with the order predicate as the only numerical predicate. Let $\mathcal{S}$ be a subset of monoids. Let $\mathcal{L}_\mathcal{S}$ be the logic closed under quantification over the monoids in $\mathcal{S}$, and $\mathbf{N}$ be the class of neutral letter languages. Then we show that

$$\mathcal{L}_\mathcal{S}[<,+] \cap \mathbf{N} = \mathcal{L}_\mathcal{S}[<] \cap \mathbf{N}$$

Our result can be interpreted as the Crane Beach conjecture to hold for the logic $\mathcal{L}_\mathcal{S}$[<,+]. As a corollary of our result we get the result of Roy and Straubing that FO+MOD[<,+] collapses to FO+MOD[<]. For cyclic groups, we answer an open question of Roy and Straubing, proving that MOD[<,+] collapses to MOD[<]. Our result also shows that multiplication as a numerical predicate is necessary for Barrington's theorem to hold and also to simulate majority quantifiers.
All these results can be viewed as separation results for highly uniform circuit classes. For example we separate FO[<,+]-uniform CC^0 from FO[<,+]-uniform ACC^0.

1 Introduction

Consider a language with a "neutral letter", i.e. a letter which can be inserted or deleted from any word in the language without changing its membership. The neutral letter concept has turned out to be useful for showing non-expressibility results. It had been used for showing super linear lower bounds for bounded-width branching programs [4], super linear wires in circuit classes [12] and in communication complexity [9]. But it is mostly known in the context of the Crane Beach conjecture proposed in [2]. There it is conjectured that first order logic with arbitrary numerical predicates (denoted as arb) will collapse to first order logic with only linear ordering in the presence of a neutral letter. The idea is that, in the presence of a neutral letter, formulas cannot rely on the precise location of input letters and hence numerical predicates will be of little use. Let $\mathbf{N}$ denote the class of languages with neutral letters. Let $\mathcal{S}$ be a subset of monoids and $\mathcal{L}_\mathcal{S}$ be the logic closed under quantification, where the quantifiers are Lindström quantifiers over some monoid from $\mathcal{S}$. Then the Crane Beach conjecture says that

$$\mathcal{L}_\mathcal{S}[\mathrm{arb}] \cap \mathbf{N} = \mathcal{L}_\mathcal{S}[<] \cap \mathbf{N}.$$

Nevertheless the conjecture was refuted by Barrington et. al [2]. In fact they show that the conjecture does not hold for the logic FO$[<,+,*]$, i.e. first order logic ($\mathcal{S}$ consists of only the existential quantifier) using addition, and multiplication relation. In the same paper, the authors show that the conjecture hold for various other logics. The Boolean closure of the Σ_1-fragment of FO[arb] does satisfy the conjecture. That is $\mathcal{B}(\Sigma_1)[\mathrm{arb}] \cap \mathbf{N} = \mathcal{B}(\Sigma_1)[<] \cap \mathbf{N}$. Lautemann, Tesson and Thérien [16] considered quantifiers which can count modulo a prime p (called MOD_p). They show that $\mathcal{B}(\Sigma_1^{0,p})[\mathrm{arb}] \cap \mathbf{N} = \mathcal{B}(\Sigma_1^{0,p})[<] \cap \mathbf{N}$. This is equivalent to showing that $\mathrm{MOD}_p[\mathrm{arb}] \cap \mathbf{N} = \mathrm{MOD}_p[<] \cap \mathbf{N}$. Benedikt and Libkin [8], in the context of collapse results in database theory, had shown that first order logic with only addition satisfies the Crane Beach conjecture. A different proof of the result can be found in [2].

We show that this can be generalized to any monoid quantifier. Let $\mathcal{S}$ be a subset of monoids. Consider the logic $\mathcal{L}_\mathcal{S}$ where the quantifiers are Lindström quantifiers whose languages are word problems of monoids in S. Our main result (Theorem 2) shows that the Crane Beach conjecture hold for the logic $\mathcal{L}_{\mathcal{S}}[<,+]$. That is:

$$\mathcal{L}_{\mathcal{S}}[<,+] \cap \mathbf{N} = \mathcal{L}_{\mathcal{S}}[<] \cap \mathbf{N}.$$

If S is the U_1 monoid, then the Theorem is equivalent to the result of Benedikt and Libkin. Roy and Straubing [22] (used ideas of Benedikt and Libkin to) show that $\mathrm{FO}+\mathrm{MOD}[<,+]$ in the presence of neutral letters collapse to FO+MOD[<]. In the paper they asked whether $\mathrm{MOD}[<,+]$ satisfy Crane Beach conjecture? This is answered by a corollary of our Theorem.

Our results can also be viewed from the perspective of descriptive complexity of circuit classes. The books [11], [29] show the close connection between logics with monoid quantifiers and circuit classes. We know that the set of languages accepted by uniform-AC^0 circuits are exactly those definable by first order logic which uses order, addition and multiplication relations. Similarly CC^0 (constant depth, polynomial size circuits with MOD-gates) corresponds to $\mathrm{MOD}[<,+,*]$, ACC^0 corresponds to $\mathrm{FO}+\mathrm{MOD}[<,+,*]$, TC^0 corresponds to $\mathrm{MAJ}[<,+,*]$, and NC^1 corresponds to $\mathrm{GROUP}[<,+,*]$ (The "group quantifier" evaluates over a finite group). It is a well known result that AC^0 is separated from ACC^0 [10], but relationships between most other classes are open. For example, we do not know whether CC^0 is different from ACC^0. In fact we do not know whether $\mathrm{MOD}_6[<,+,*]$ contains uniform-AC^0. This explains why the Crane Beach conjecture for prime modulo quantifiers [16], using arbitrary predicates, cannot be easily extended to composite modulo quantifiers.

We look at these separation questions from the descriptive complexity perspective. As a first step, one can ask the question of separating the logics when the multiplication relation is not available. That is, can one separate $\mathrm{MOD}[<,+]$ from $\mathrm{FO}+\mathrm{MOD}[<,+]$? Is $\mathrm{GROUP}[<,+]$ different from $\mathrm{FO}+\mathrm{MOD}[<,+]$?

Behle and Lange [7] give a notion of interpreting $\mathcal{L}_{\mathcal{S}}[<,+]$ as highly uniform circuit classes. Our results therefore can be summarized as, every $\mathrm{FO}[<,+]$ uniform constant depth polynomial size circuit with gates that compute a product in $\mathcal{S}$ and which recognize a language with a neutral letter can be made FO[<]-uniform.

As a consequence of our Theorem 2 we are able to separate these uniform versions of circuit classes. For example: The theorem states that $\mathrm{MOD}[<,+]$ definable languages with a neutral letter are also definable in MOD[<]. Since MOD[<] cannot simulate existential quantifiers [26] we have that $\mathrm{FO}[<,+]$ and $\mathrm{MOD}[<,+]$ are incomparable. In fact we show that no group quantifier can simulate existential quantifier if only addition is available. Our another corollary gives an alternate proof of the known result [22] that $\mathrm{FO}+\mathrm{MOD}_m[<,+]$ cannot count modulo a prime p, which does not divide m. Another corollary shows that the majority quantifier cannot be simulated by group quantifiers if multiplication is not available, thus separating $\mathrm{MAJ}[<,+]$ from $\mathrm{FO}+\mathrm{GROUP}[<,+]$. Barrington's theorem [1] says that word problems over any finite group can be defined by the logic which uses only the S_5 group quantifier (the group whose elements are the set of all permutations over 5 elements) if addition and multiplication predicates are available. Our result show multiplication is necessary for Barrington's theorem to hold. In other words S_5 cannot define word problems over S_6 if only addition is available.

Non expressibility results for various logics which uses addition and a variety of quantifiers has been considered earlier. Lynch [19] showed that $\mathrm{FO}[<,+]$ cannot count modulo any number. Nurmonen [21] and Niwiński and Stolboushkin [25] looked at logics with counting quantifiers equipped with numerical predicates of form $y = px$ and a linear ordering. Ruhl [23], Schweikardt [24], Lautemann et.al. [15], Lange [14] all show the limited expressive power of addition in the presence of majority quantifiers. Behle, Krebs and Reifferscheid [6, 5] show that non-solvable groups are not definable in the two variable fragment of $MAJ[<]$.

For the purpose of proof we work over infinite strings which contain finite number of non-neutral letters. Our general proof strategy is similar to Benedikt and Libkin [8] or Roy and Straubing [22] and consists of three main steps.

1. Given a formula $\phi \in \mathcal{L}_{\mathcal{S}}[<,+]$, we give an infinite set $\mathcal{D}$ and an "active domain formula" $\phi' \in \mathcal{L}_{\mathcal{S}}[<,+]$ such that for all words w whose non neutral positions belong to $\mathcal{D}$ we have $w \vDash \phi \Leftrightarrow w \vDash \phi'$. Active domain formulas quantify only over non-neutral letter positions. Our major contribution (Theorem 14) is showing this step.

2. We give another infinite set $S \subseteq \mathcal{D}$ and an active domain formula $\psi \in \mathcal{L}_{\mathcal{S}}[<]$ such that for all words w whose non neutral positions belong to S we have $w \vDash \phi' \Leftrightarrow w \vDash \psi$. This step follows from an application of Ramsey theory (Theorem 15).
3. Active domain formulas in $\mathcal{L}_{\mathcal{S}}[<]$ with neutral letters will work on every domain. This is an easy observation given by Lemma 16.

Finally using these three steps we prove our main theorem.

The main step is to show that we can build an active domain formula. Hence we need to show how to simulate a quantifier by an active domain formula. In the case of FO$[<,+]$, the quantifiers, considered as Lindström quantifiers, have a commutative and idempotent monoid. Hence neither the order in which the quantifier runs over the positions of the word is important, nor does it matter if positions are queried multiple times. In Roy and Straubing this idea was extended in such a way that in the simulation of the MOD quantifier (again a commutative monoid), every position is taken into account exactly once. In their construction while replacing a MOD quantifier they need to add additional FO quantifiers and hence their construction only allows to replace a MOD$[<,+]$ formula by an active domain FO + MOD$[<,+]$ formula. In this paper, we construct a formula that takes every position into account exactly once and in the correct order. Moreover we do not introduce any new quantifier, but use only the quantifier that is replaced. This enables us to show the Crane Beach conjecture for logics whose quantifiers have a non-commutative monoid or are groups. For example MOD$[<,+]$, GROUP$[<,+]$, and FO + GROUP$[<,+]$.

In contrast to previous work, we do not construct an equivalent active domain formula, but only a formula that is equivalent for certain domains. We show that it is in general sufficient to show this for one infinite domain. We also introduce a combinatorial structure called *Sorting Tree* which can be of interest on its own. Yet another contribution is to use inverse elements of groups to merge two sorted lists of numbers.
We present our main theorem and its corollaries in Section 3 followed by a section with the proof of Theorem 14. Section 5 which is our main contribution shows how to replace group quantifiers by its active domain version.

2 Preliminaries

An alphabet Σ is a finite set of symbols. The set of all finite words over Σ is denoted by Σ^*, the set of all right infinite words is denoted by Σ^ω. Let $\Sigma^\infty = \Sigma^* \cup \Sigma^\omega$. Consider a language $L \subseteq \Sigma^\infty$ and a letter $\lambda \in \Sigma$. We say that λ is a *neutral letter* for L if for all $u, v \in \Sigma^\infty$ we have that $u\lambda v \in L \Leftrightarrow uv \in L$. We denote the set of all languages with a neutral letter by $\mathbf{N}$.

For a word $w \in \Sigma^\infty$ the notation $w(i)$ denotes the i^{th} letter in w, i.e. $w = w(0)w(1)w(2)\ldots$. For a word w in a language L with neutral letter λ, we define the non-neutral positions $\mathrm{nnp}(w)$ of w to be the set of all positions which do not have the neutral letter.

A monoid is a set closed under a binary associative operation and has an identity element. All monoids we consider except for Σ^* and Σ^∞ will be finite. A monoid M and $S \subseteq M$ defines a *word problem.* Its language is composed of words $w \in M^*$, such that when the elements of w are multiplied in order we get an element in S. We say that a monoid M *divides* a monoid N if there exists a submonoid N' of N and a surjective morphism from N' to M. A monoid M *recognizes* a language $L \subseteq \Sigma^*$ if there exists a morphism $h : \Sigma^* \to M$ and a subset $T \subseteq M$ such that $L = h^{-1}(T)$. It is known that finite monoids recognize exactly regular languages [26]. We denote by $\mathcal{M}$ the set of all finite monoids, $\mathcal{G} \subset \mathcal{M}$ the set of all finite groups and MOD the set of all finite cyclic group. We denote by U_1 the monoid consisting of elements $\{0, 1\}$ under multiplication. For a monoid M, the element $1 \in M$ will denote its identity element. We also use the block product of monoids, whose definition can be found in [26].

Given a formula ϕ with free variables $x_1, \ldots, x_k$, we write $w, i_1, \ldots, i_k \models \phi$ if w is a model for the formula ϕ when the free variables x_j is assigned to i_j for $j = 1, \ldots, k$. We abuse notation and let $c \in \Sigma$ also be the unary predicate symbols of the logic we consider. That is $w, i \models c(x)$ iff $w(i) = c$. Let $\mathcal{V}$ be a set of variables, $\mathcal{R}$ be a set of numerical predicates and $\mathcal{S} \subseteq \mathcal{M}$. We define the logic $\mathcal{L}_{\mathcal{S}}[\mathcal{R}]$ to be built from the unary predicate symbols c, where $c \in \Sigma$, the binary predicate $\{=\}$, the predicates in $\mathcal{R}$, the variable symbols $\mathcal{V}$, the Boolean connectives $\{\neg, \vee, \wedge\}$, and the monoid quantifiers Q_M^m, where $M \in \mathcal{S}$ is a monoid and $m \in M$. We also identify the logic class $\mathcal{L}_{\mathcal{S}}[\mathcal{R}]$ with the set of all languages definable in it.

Our definition of monoid quantifiers is a special case of Lindström quantifiers [18]. The formal definition of a monoid quantifier [3] is as follows. Let $M = \{m_1, \ldots, m_K, 1\}$ be a monoid with $K+1$ elements. For an $m \in M$, the quantifier Q_M^m is applied on K formulas. Let x be a free variable and $\phi_1(x), \ldots, \phi_K(x)$ be K formulas. Then $w \models Q_M^m x \langle \phi_1(x), \ldots, \phi_K(x) \rangle$ iff the word u when multiplied gives the element m, i.e. $\prod_i u(i) = m$, where the i^{th} letter of u, $0 \leq i < |w|$, is

$$u(i) = \begin{cases} m_1 & \text{if } w, i \models \phi_1 \\ m_2 & \text{if } w, i \models \neg\phi_1 \wedge \phi_2 \\ \vdots & \\ m_K & \text{if } w, i \models \neg\phi_1 \wedge \cdots \wedge \neg\phi_{K-1} \wedge \phi_K \\ 1 & \text{otherwise} \end{cases}$$

The following "shorthand" notation is used to avoid clutter. We denote by $Q_M^m x \ \phi \ \langle \alpha_1, \ldots, \alpha_K \rangle$, the formula $Q_M^m x \langle \phi \wedge \alpha_1, \ldots, \phi \wedge \alpha_K \rangle$. Informally, this relativizes the quantifier to the positions where ϕ is true, by multiplying the neutral element in all other places.

Consider the monoid U_1. It is easy to see that the word problem defined by U_1 and the set $\{0\}$ defines the regular language $1^*0\{0,1\}^*$. Then $Q_{U_1}^0$ is same as the existential quantifier $\exists$, since any formula $\exists x \phi$ is equivalent to $Q_{U_1}^0 x \ \langle \phi \rangle$. So the logic $\mathcal{L}_{U_1}[<]$ denotes first-order logic, FO$[<]$. Let C_q stand for the cyclic group with q elements. Then the quantifiers $Q_{C_q}^1$ corresponds to modulo quantifiers [28]. Thus $\mathcal{L}_{\text{MOD}}[<]$ corresponds to all regular languages whose syntactic monoids are solvable groups [26]. For a sentence $\phi \in \mathcal{L}_{\mathcal{S}}[\mathcal{R}]$ we define $L(\phi) = \{w \mid w \vDash \phi\}$. The following result gives an algebraic characterization for the logic $\mathcal{L}_{\mathcal{S}}[<]$.

Lemma 1 ([26]). *Let $\mathcal{S} \subseteq \mathcal{M}$. Let S' be the smallest set containing $\mathcal{S}$ and is closed under block products. Let $L \subseteq \Sigma^*$ such that M is the smallest monoid which recognizes L. Then L is definable in $\mathcal{L}_{\mathcal{S}}[<]$ iff M divides a monoid in S'.*

3 Results

Let $\mathcal{S} \subseteq \mathcal{M}$ be any set of monoids. We show that the Crane Beach conjecture is true for the logic $\mathcal{L}_{\mathcal{S}}[<, +]$.

Theorem 2 (Main Theorem). *Let $\mathcal{S} \subseteq \mathcal{M}$. Then*

$$\mathcal{L}_{\mathcal{S}}[<, +] \cap \mathbf{N} = \mathcal{L}_{\mathcal{S}}[<] \cap \mathbf{N}$$

The proof of this theorem is given in Section 4.

The above Theorem give us the following corollaries.

Corollary 3. *All languages with a neutral letter in $\mathcal{L}_{\mathcal{M}}[<, +]$ are regular.*

Proof. By Theorem 2 we know that all languages with a neutral letter in $\mathcal{L}_{\mathcal{M}}[<, +]$ can be defined in $\mathcal{L}_{\mathcal{M}}[<]$ which by Lemma 1 is the set of all regular languages. □

Recall that a monoid M *divides* a monoid N if M is a morphic image of a submonoid of N.

Corollary 4. *Let $\mathcal{S} \subseteq \mathcal{G}$. Let G be a simple group that does not divide any monoid M in $\mathcal{S}$. Then the word problem over G is not definable in $\mathcal{L}_{\mathcal{S}}[<, +]$.*

Proof. The word problem over G has a neutral letter. The result now follows from Theorem 2 and Lemma 1. □

The majority quantifier, Maj x $\phi(x)$ is given as follows.

$$w \vDash \text{Maj } x \ \phi(x) \Leftrightarrow |\{i \mid w \vDash \phi(i),\ i \leq |w|\}| > \frac{|w|}{2}$$

MAJ[<] denotes the logic closed under majority quantifiers. It is known that the majority quantifier can be simulated by the non-solvable group S_5 if both multiplication and addition are available [29]. We show that multiplication is necessary to simulate majority quantifiers.

Corollary 5. MAJ[<] $\nsubseteq \mathcal{L}_{\mathcal{M}}[<,+]$.

Proof. Consider the language $L \subseteq \{a,b,c\}^*$ consisting of all words with an equal number of a's and b's. L can be shown to be definable in MAJ[<]. Also note that c is a neutral element for L. By Corollary 3, and the fact that L is nonregular, we know that L is not definable in $\mathcal{L}_{\mathcal{M}}[<,+]$. □

Barrington's theorem [1] says that the word problem of any finite group can be defined in the logic $\mathcal{L}_{S_5}[<,+,*]$. The following theorem shows that multiplication is necessary for Barrington's theorem to hold.

Corollary 6. *The word problem over the group S_6 is not definable in $\mathcal{L}_{S_5}[<,+]$. Infact there does not exist any one finite monoid M such that all regular languages can be defined in $\mathcal{L}_M[<,+]$.*

Proof. A_6 is a simple subgroup of S_6, which does not divide S_5. From Corollary 4 it follows that the word problem over S_6 is not definable in $\mathcal{L}_{S_5}[<,+]$.
For any finite monoid M, there exists a simple group G such that G does not divide M and hence the word problem over G is not definable in $\mathcal{L}_M[<,+]$. □

Let L_p be the set of all words $w \in \{0,1\}^*$ such that the number of occurrences of 1 in w is equal to 0 (mod p). Then we get the result in [22] that L_p is not definable in FO + MOD$_m[<,+]$, if p is a prime which does not divide m.

Corollary 7 ([22]). *If p is a prime which does not divide m, then L_p is not definable in* FO + MOD$_m[<,+]$.

Proof. Let L_p be definable in FO + MOD$_m[<,+]$. Since 0 is a neutral letter in L_p, Theorem 2 says L_p is also definable in FO + MOD$_m[<]$. Due to Lemma 1 and [26], this is a contradiction. □

It is an open conjecture whether the language 1^* can be accepted by the circuit complexity class CC^0 [26]. It is also known that languages accepted by CC^0 circuits are exactly those which are definable by $\mathcal{L}_{\text{MOD}}[<,+,*]$ formulas [29].

To progress in this direction Roy and Straubing [22] had posed the question of whether one can show that $1^* \notin \mathcal{L}_{\text{MOD}}[<,+]$. Below we show that this is the case.

Corollary 8. $1^* \notin \mathcal{L}_{\text{MOD}}[<,+]$. *In fact* $1^* \notin \mathcal{L}_{\mathcal{G}}[<,+]$.

Proof. The minimal monoid which can accept 1^* is U_1 and clearly the language is in **N**. By Theorem 2 if there is a formula in $\mathcal{L}_{\mathcal{G}}[<,+]$ which can define 1^*, then $\mathcal{L}_{\mathcal{G}}[<]$ can also define 1^*. From Lemma 1 it follows that the monoid U_1 divides a group. But this is a contradiction [26]. □

Behle and Lange [7] give a notion of interpreting $\mathcal{L}_{\mathcal{S}}[<,+]$ as highly uniform circuit classes. As a consequence we can interpret the following results as a separation of the corresponding circuit classes.

Corollary 9. *The following separation results hold, for all $m > 1$*

- FO[<,+] $\nsubseteq$ MOD[<,+].
- MOD$_m[<,+] \nsubseteq$ FO[<,+].
- FO[<,+] $\subsetneq$ FO + MOD$_m[<,+] \subsetneq$ FO + MOD[<,+]
- FO + MOD[<,+] $\subsetneq$ FO + GROUP[<,+]
- MAJ[<,+] $\nsubseteq$ FO + GROUP[<,+]

Let $\mathcal{S}$ be a set of monoids such that, given a monoid M, it is decidable if M divides a block product of monoids in $\mathcal{S}$. Then, given a regular language L, it is decidable if $L \in L_{\mathcal{S}}[<]$. Together with our main theorem we get that it is decidable if $L \in \mathcal{L}_{\mathcal{S}}[<,+]$.

Corollary 10. *Let $\mathcal{S}$ be a set of monoids such that, given a monoid M, it is decidable if M divides a block product of monoids in $\mathcal{S}$. Then, given a regular language L, it is decidable if $L \in L_{\mathcal{S}}[<,+]$.*

For FO+MOD$[<,+]$ this was proved in [22]. Here we show the claim for the special case when $\mathcal{S} = \mathrm{MOD}$.

Corollary 11. *Given a regular language L, the question whether L is definable in* MOD$[<,+]$ *is decidable.*

4 Proof of the Main Theorem

In this section we handle the general proof steps as in Libkin or Roy and Straubing of removing the plus predicate from the formula in the presence of a neutral letter. We show that all these results go through even in the presence of general Lindström quantifiers. The new crucial step is Lemma 12 where we show how to convert a group quantifier to an active domain formula without introducing any other quantifiers. The proof of this lemma is deferred to the next section.

Let $\mathcal{S} \subseteq \mathcal{M}$ be any nonempty set. To prove Theorem 2 we will consider the more general logic, $\mathcal{L}_{\mathcal{S}}[<,+,0,\{\equiv_q\colon q>1\}]$ over the alphabet Σ. In this logic $+$ is a binary function, 0 is a constant, and $a \equiv_q b$ means q divides $b-a$. The reason for introducing these new relations (which are definable using $+$) is to use a quantifier elimination procedure. All languages recognized by this logic are in $\mathcal{L}_{\mathcal{S}}[<,+]$.

The formulas we consider will usually define languages with a neutral letter. Let an *active domain formula* over a letter $\lambda \in \Sigma$ be a formula where all quantifiers are of the form: $Q_M^m x\ \neg\lambda(x)\langle\phi_1,\ldots,\phi_K\rangle$. That is the quantifiers, quantify only over the "active domain", the positions which does not contain the letter λ. For the purpose of the proof we assume that the neutral letter language defined by a formula $\phi \in \mathcal{L}_{\mathcal{S}}[<,+]$ is a subset of $\Sigma^*\lambda^\omega$. The idea is to work with infinite words, where the arguments are easier, since the variable range is not bounded by the word length.

For $r \in \mathbb{N}$ we define the set $\mathcal{D}_r = \{r^i \mid 0 < i \in \mathbb{N}\}$. We say that a formula $\phi(x_1,\ldots,x_t) \in \mathcal{L}_{\mathcal{S}}[<,+]$ *collapses* to ϕ', if ϕ' is an active domain formula in $\mathcal{L}_{\mathcal{S}}[<,+]$ and there exists an $\mathcal{R}_\phi \in \mathbb{N}$ such that for all $r \geq \mathcal{R}_\phi$, $w \in \Sigma^*\lambda^\omega$ with $\mathrm{nnp}(w) \subseteq \mathcal{D}_r$ and for all $a_1,\ldots,a_t \in \mathbb{N}$ we have that

$$w \models \phi(a_1,\ldots,a_t) \Leftrightarrow w \models \phi'(a_1,\ldots,a_t)$$

In the above definition we say that $\mathcal{R}_\phi$ collapse ϕ to ϕ'.

The results by Benedikt and Libkin [8], and Roy and Straubing [22] show that for all formulas $\phi \in \mathcal{L}_{\mathrm{MOD}\cup U_1}[<,+]$ there exists an active domain formula ϕ' in that logic, such that for all words $w \in \Sigma^*\lambda^\omega$, $w \vDash \phi \Leftrightarrow w \vDash \phi'$. They assume no restriction on the non-neutral positions of w. Observe that our collapse result is different from theirs. We show that if we consider only words, whose non-neutral positions are in $\mathcal{D}_r$, then any formula $\phi \in \mathcal{L}_{\mathcal{S}}[<,+]$ is equivalent to the active domain formula $\phi' \in \mathcal{L}_{\mathcal{S}}[<,+]$. That is, we are not concerned about the satisfiability of those words with non-neutral positions not in $\mathcal{D}_r$.

We show that formulas with a group quantifier, $G \in \mathcal{S}$ can be collapsed.

Lemma 12. *Let $\phi = Q_G^m z\langle\phi_1,\ldots,\phi_K\rangle$ be in $\mathcal{L}_{\mathcal{S}}[<,+]$. Assume formulas $\phi_1,\ldots,\phi_K$ collapse. Then ϕ collapses to an active domain formula ϕ'.*

The proof of Lemma 12 will be given in Section 5. Benedikt and Libkin [8] gives a similar theorem for the monoid U_1 (the existential quantifier).

Lemma 13 ([8]). *Let $\phi = Q_{U_1}^m z\langle\phi_1,\ldots,\phi_K\rangle$ be a formula in $\mathcal{L}_{\mathcal{S}}[<,+]$. Let us assume that formulas $\phi_1,\ldots\phi_K$ collapse. Then ϕ collapses to an active domain formula ϕ'.*

Recall the 3 steps for proving the main theorem given in Introduction. The following theorem proves the first step.

Theorem 14. *Let $\phi \in \mathcal{L}_{\mathcal{S}}[<,+]$. Then there exists an active domain formula $\phi' \in \mathcal{L}_{\mathcal{S}}[<,+]$ such that ϕ collapses to ϕ'.*

Proof. Let $\phi \in \mathcal{L}_{\mathcal{S}}[<,+]$. We first claim that we can convert ϕ into a formula which uses only groups and U_1 as quantifiers. This follows from the Krohn-Rhodes decomposition theorem for monoids that every monoid can be decomposed into block products over groups and U_1. This decomposition can then be converted back into a formula using the groups and U_1 as quantifiers [26].

So without loss of generality we can assume ϕ has only group or U_1 quantifiers. The proof is by induction on the quantifier depth. For the base case, let ϕ be a quantifier free formula. It is an active domain formula and therefore the claim holds. Let the claim be true for all formulas with quantifier depth $< d$. Lemma 12 and Lemma 13 show that the claim is true for formulas of type $\phi = Q_M^m z\langle\phi_1, \ldots, \phi_K\rangle$ with quantifier depth d, when M is a group or U_1 respectively. We are now left with showing that the claim is closed under conjunction and negation. So assume that formulas ϕ_1, ϕ_2 collapse to ϕ'_1, ϕ'_2 respectively. That is there exist $\mathcal{R}_{\phi_1}, \mathcal{R}_{\phi_2} \in \mathbb{N}$ such that $\mathcal{R}_{\phi_1}$ collapses ϕ_1 to ϕ'_1 and $\mathcal{R}_{\phi_2}$ collapses ϕ_2 to ϕ'_2. Let $\mathcal{R} = max\{\mathcal{R}_{\phi_1}, \mathcal{R}_{\phi_2}\}$. Then it is easy to see that $\mathcal{R}$ collapses $\phi_1 \wedge \phi_2$ to $\phi'_1 \wedge \phi'_2$ and $\mathcal{R}_{\phi_1}$ collapses $\neg\phi_1$ to $\neg\phi'_1$. □

We have shown above that all formulas in $\mathcal{L}_{\mathcal{S}}[<,+]$ can be collapsed to active domain formulas. Now using a Ramsey type argument we show that addition is useless, giving us a formula in $\mathcal{L}_{\mathcal{S}}[<]$. This corresponds to the second step in our three step proof strategy.

Let R be any set of relations on $\mathbb{N}$ and let $\phi(x_1, \ldots, x_t)$ be an active domain formula in $\mathcal{L}_{\mathcal{S}}[R]$. We say that ϕ has the *Ramsey property* if for all infinite subsets X of $\mathbb{N}$, there exists an infinite set $Y \subseteq X$ and an active domain formula $\psi \in \mathcal{L}_{\mathcal{S}}[<]$ that satisfies the following conditions. If $w \in \Sigma^*\lambda^\omega$ and $\text{nnp}(w) \subseteq Y$, then for all $a_1, \ldots, a_t \in Y$,

$$w \vDash \phi(a_1, \ldots, a_t) \Leftrightarrow w \vDash \psi(a_1, \ldots, a_t)$$

The Ramsey property for first order logic has been considered by Libkin [17]. We show that these results can be extended to our logic.

Theorem 15. *Let R be a set of relations on $\mathbb{N}$. Every active domain formula in $\mathcal{L}_{\mathcal{S}}[R]$ satisfies the Ramsey property.*

Proof. Let $\phi \in \mathcal{L}_{\mathcal{S}}[R]$ be a formula. We now prove by induction on the structure of the formula. Let $P(x_1, \ldots, x_k)$ be a term in ϕ. We assume without loss of generality that for all $i \neq j, x_i \neq x_j$. Now consider the infinite complete hypergraph, whose vertices are labelled by numbers from X and whose edges are k tuple of vertices. Let $i_1, \ldots, i_k$ be some permutation of numbers from 1 to k. Consider the edge formed by the vertices $v_1 < v_2 < \cdots < v_k$. We color this edge by the formula $x_{i_1} < x_{i_2} < \cdots < x_{i_k}$ if $P(v_{i_1}, \ldots, v_{i_k})$ is true. Observe that each edge can have multiple colors and therefore the total number of different colorings possible is $k!$. Ramsey theory gives us that there exists an infinite set $Y \subseteq X$, such that the induced subgraph on the vertices in Y will have a monochromatic color, ie. all the edges will be colored using the same color. Let us assume that the edges in Y are colored $x_1 < x_2 < \cdots < x_k$. Then for all $a_1, \ldots, a_t \in Y$

$$a_1, \ldots, a_t \models R(x_1, \ldots, x_k) \Leftrightarrow a_1, \ldots, a_t \models x_1 < x_2 < \cdots < x_k$$

This shows that $P(x_1, \ldots, x_k)$ satisfies the Ramsey property and thus all atomic formulas satisfy the Ramsey property. We now show that Ramsey property is preserved while taking Boolean combination of formulas. Consider the formula $\phi_1(x_1, \ldots, x_k) \wedge \phi_2(x_1, \ldots, x_k)$. We know that by induction hypothesis there exists a formula ψ_1 and an infinite set X such that for all $a_1, \ldots, a_t \in X$, $w \models \phi_1(a_1, \ldots, a_t) \Leftrightarrow w \models \psi(a_1, \ldots, a_t)$. We can now find an infinite set $Y \subseteq X$ and a formula ψ_2 such that the Ramsey property holds for the formula ϕ_2. Therefore for all $a_1, \ldots, a_t \in Y$

$$w, a_1, \ldots, a_k \vDash \phi_1 \wedge \phi_2 \Leftrightarrow w, a_1, \ldots, a_k \vDash \psi_1 \wedge \psi_2$$

Similarly we can show that the Ramsey property holds for disjunctions and negations. We need to now show that active domain quantification also preserves Ramsey property. So let X be an infinite subset of $\mathbb{N}$ and let

$$\phi(\boldsymbol{x}) = Q_M^m z \ \neg\lambda(z) \ \langle \phi_1(z,\boldsymbol{x}),\dots,\phi_K(z,\boldsymbol{x})\rangle$$

be a formula in $\mathcal{L}_{\mathcal{S}}[R]$. By induction hypothesis we know that there exists an infinite set $Y_1 \subseteq X$ and an active domain formula $\psi_1 \in \mathcal{L}[<]$ such that for all $\boldsymbol{a} \in Y_1^t$ the Ramsey property is satisfied. That is $w \models \phi_1(\boldsymbol{a}) \Leftrightarrow w \models \psi_1(\boldsymbol{a})$. Now for ϕ_2, using the infinite set Y_1 we can find an infinite set $Y_2 \subseteq Y_1$ and a formula ψ_2 satisfying the Ramsey property. Continuing like this will give us a set Y_K and formulas $\psi_1,\dots,\psi_K$ such that $\forall j \leq K$ and for all $w \in \Sigma^*\lambda^\omega$ with $\mathrm{nnp}(w) \subseteq Y_K$, we have that $\forall b \in Y_K, \boldsymbol{a} \in Y_K^t, \ w \vDash \phi_j(b,\boldsymbol{a}) \Leftrightarrow w \vDash \psi_j(b,\boldsymbol{a})$. Hence we also have that $\forall j \leq K$

$$\{b \in Y_K \mid w \vDash \phi_j(b,\boldsymbol{a})\} = \{b \in Y_K \mid w \vDash \psi_j(b,\boldsymbol{a})\}$$

Therefore for the formula $\psi = Q_M^m z \ \neg\lambda(z) \ \langle \psi_1,\dots,\psi_K\rangle$, we have $\forall w$ where $\mathrm{nnp}(w) \subseteq Y_K$ and $a_1,\dots,a_t \in Y_K$ that

$$w \vDash \phi(a_1,\dots,a_t) \Leftrightarrow w \vDash \psi(a_1,\dots,a_t)$$

Observe that ψ is an active domain formula in $\mathcal{L}_{\mathcal{S}}[<]$. □

Now we show the third step of our three step proof strategy.

Lemma 16. *Every active domain sentence in $\mathcal{L}_{\mathcal{S}}[<]$ define a language with a neutral letter.*

Proof. Let $\phi \in \mathcal{L}_{\mathcal{S}}[<]$ be an active domain formula over letter $\lambda \in \Sigma$. Let $w \in \Sigma^\omega$. Let $w' \in \Sigma^\omega$ got by inserting letter λ in w at some positions. Let $n_1 < n_2 < \dots$ belong to $\mathrm{nnp}(w)$ and $m_1 < m_2 < \dots$ be in $\mathrm{nnp}(w')$. Let $\rho : \mathrm{nnp}(w) \to \mathrm{nnp}(w')$ be the bijective map $\rho(n_i) = m_i$. We show that for any subformula ψ of ϕ and any $\boldsymbol{t} \in \mathrm{nnp}(w)^s$, we have that $w, \boldsymbol{t} \vDash \psi \Leftrightarrow w', \rho(\boldsymbol{t}) \vDash \psi$. The claim holds for the atomic formula $x > y$, because $n_i > n_j$ iff $\rho(n_i) > \rho(n_j)$ for an i, j. Similarly the claim also hold for all other atomic formulas $x < y, x = y$ and $a(x)$ for an $a \in \Sigma$. The claim remains to hold under conjunctions, negations and active domain quantifications. Hence $w \models \phi \Leftrightarrow w' \models \phi$. This shows that λ is a neutral letter for $L(\phi)$. □

Now we can prove our main theorem.

Proof (Proof of Theorem 2). Let $\phi \in \mathcal{L}_{\mathcal{S}}[<,+]$, such that $L(\phi)$ is a language with a neutral letter, λ. By Theorem 14 there exists an active domain sentence $\phi' \in \mathcal{L}_{\mathcal{S}}[<,+]$ over λ and a set $\mathcal{D}_{\mathcal{R}}$ such that $\mathcal{R}$ collapses ϕ to ϕ'. Theorem 15 now gives an active domain formula $\psi \in \mathcal{L}_{\mathcal{S}}[<]$ and an infinite set $Y \subseteq \mathcal{D}_{\mathcal{R}}$. We now show that $L(\phi) = L(\psi)$. Let $w \in \Sigma^*\lambda^\omega$. Consider the word $w' \in \Sigma^*\lambda^\omega$ got by inserting the neutral letter λ in w in such a way that $\mathrm{nnp}(w') \subseteq Y$. Since $L(\phi)$ is a language with a neutral letter we have that $w \models \phi \Leftrightarrow w' \vDash \phi$. From Theorem 14 and Theorem 15 we get $w' \vDash \phi \Leftrightarrow w' \vDash \phi' \Leftrightarrow w' \vDash \psi$. Finally as shown in Lemma 16, ψ defines a language with a neutral letter and hence $w' \models \psi \Leftrightarrow w \models \psi$. □

5 Proof of Lemma 12

In this section we show how to replace a group quantifier by an active domain formula. Here we make use of the fact that we can a priory restrict our domain as shown in the previous section.

Recall that $\phi = Q_G^m z \langle \phi_1,\dots,\phi_K\rangle$ and $G = \{m_1,\dots,m_K,1\}$. We know that for all $i \leq K$, there exists $\mathcal{R}_{\phi_i}$ and a formula ϕ_i' such that $\mathcal{R}_{\phi_i}$ collapse ϕ_i to ϕ_i'. Then clearly $max\{\mathcal{R}_{\phi_i}\}$ collapse ϕ_i to ϕ_i' for all $i \leq K$. So without loss of generality we assume ϕ_is are active domain formulas.

Before we go in the details we will give a rough overview of the proof idea. The group quantifier will evaluate a product $\prod_j u(j)$ where $u(j)$ is a group element that depends on the set of i such that $w, j \models \phi_i$. So we start and analyze the sets $J_i = \{j \mid w, j \models \phi_i\}$. Since the formulas ϕ_i are active domain formulas, we will see that there exists a set of intervals such that inside an interval the set J_i is periodic. Boundary points for these intervals are either points in the domain, or linear combinations of these. In the construction of the active domain formula for ϕ we will show how to iterate over all these boundary points in a strictly increasing order. An active domain quantifier can only iterate over active domain positions, hence we will need nested active domain quantifiers, and a way how to "encode" the boundary points by tuples of active domain positions in a unique and order preserving way. Additionally we need to deal with the periodic positions inside the intervals, without being able to compute the length of such an interval, or even check if the length is zero. Here will make use of the inverse elements that always exist in groups.

We start by analyzing the intervals which occur. We will pick an $\mathcal{R}_\phi \geq max\{\mathcal{R}_{\phi_i}\}$ to collapse the formula ϕ. During the course of the proof we will require $\mathcal{R}_\phi$ to be greater than a few others constants, which will be specified then. But always observe that $\mathcal{R}_\phi$ will depend only on ϕ.

Since we consider a fixed set $\mathcal{S}$ for the rest of the paper, we will write $\mathcal{L}[<,+]$ for the logic $\mathcal{L}_\mathcal{S}[<,+,0,\{\equiv_q : q>1\}]$.

5.1 Intervals and Linear Functions

We first show that every formula ψ with at least one free variable has a normal form.

Lemma 17. *Let $\psi(z) \in \mathcal{L}[<,+]$. Then there exists a formula $\hat{\psi}(z) \in \mathcal{L}[<,+]$ such that ψ is equivalent to $\hat{\psi}$, where all atomic formulas in $\hat{\psi}$ with z are of the form $z > \rho, z = \rho, z < \rho, z \equiv_n \rho$, where ρ is a linear function on variables other than z.*

Proof. Terms in our logic are expressions of the form

$$\alpha_0 + \alpha_1 x_1 + \cdots + \alpha_s x_s \qquad \text{, where } \alpha_i \in \mathbb{N}$$

and atomic formulas are of the form

$$\sigma = \gamma, \sigma < \gamma, \sigma > \gamma, \sigma \equiv_m \gamma, c(\sigma)$$

whee σ, γ are linear functions, $c \in \Sigma$ and $m > 1$.
Now using any $M \in \mathcal{S}$, where $m_1 \in M$ is not the neutral element, we can rewrite $c(\sigma)$ as

$$Q_M^{m_1} x \ \neg\lambda(x) \langle (x = \sigma) \wedge c(x), false, \ldots, false \rangle$$

Now consider the atomic formulas containing the free variable z in $\psi(z)$. By multiplying with appropriate numbers, we can re-write these atomic formulas as $nz = \rho, nz < \rho, nz > \rho, nz \equiv_l \rho$ for one particular n, which is the least common multiple (lcm) of all the coefficients in ψ. Here ρ does not contain z and also it might contain subtraction. That is $nz = \rho$ might stand for $nz + \rho_1 = \rho_2$. Now we replace nz by z and conjunct the formula with $z \equiv_n 0$. □

For any formula $\psi(z)$, the notation $\hat{\psi}(z)$ denotes the normal form as in Lemma 17. Let $x_1, \ldots, x_s$ be the bounded variables occurring in $\hat{\phi}_i(z)$ and $y_1, \ldots, y_r$ be the free variables other than z in $\hat{\phi}_i(z)$. Hence the terms ρ that appear in the formula $\hat{\phi}_i(z)$ can be identified as functions, : $\mathbb{N}^{s+r} \to \mathbb{N}$.

We collect all functions $\rho(\boldsymbol{x}, \boldsymbol{y})$ that occur in the formulas $\hat{\phi}_i(z)$ for an $i \leq K$:

$$R = \{\rho \mid \text{where } \rho \text{ is a linear term occurring in } \hat{\phi}_i(z), i \leq K\}$$

We define the set T of offsets as a set of terms which are functions using the variables $y_1, \ldots, y_r$ as parameters:

$$T = \{\rho(0, \ldots, 0, y_1, \ldots, y_r) \mid \rho \in R\} \cup \{0\}$$

Consider the set of absolute values of all the coefficients appearing in one of the functions in R. Let $\alpha' \in \mathbb{N}$ be the maximum value among these. That is $\alpha' = max\{|\gamma| \mid f \in R, \gamma \text{ is a coefficient in } f\}$. Let $\varDelta = s \cdot \alpha'$.

Now we can define our set of extended functions. For a $t \in T$ we define a set of terms which are functions using the variables $x_1, \dots, x_s, y_1, \dots, y_r$ as parameters:

$$F_t = \{ \sum_i^{s'} \alpha_i x_i + t \mid s' \leq s, -\Delta \leq \alpha_i \leq \Delta, \alpha_i \in \mathbb{N} \}.$$

We denote by $F = \cup_{t \in T} F_t$.

For a fixed word $w \in \Sigma^* \lambda^\omega$ and a fixed assignment of the free variables $\boldsymbol{y}$ to $\boldsymbol{a}$ we let $B^{w,\boldsymbol{a}} =$

$$\{ f(\boldsymbol{d}, \boldsymbol{a}) \mid t \in T, f \in F_t, \boldsymbol{d} \in \text{nnp}(w)^{s'}, d_1 > d_2 > \dots > d_{s'} \}$$

be the set of *boundary points*. Note that the assignments to the functions are of strictly decreasing order. Let $b_1 < b_2 < \dots < b_l$ be the boundary points in $B^{w,\boldsymbol{a}}$. Then the following sets are called *intervals*: $(-1, b_1), (b_1, b_2), \dots, (b_{l-1}, b_l), (b_l, \infty)$. Here $(a,b) = \{x \in \mathbb{N} \mid a < x < b\}$. We also split the set of points in $B^{w,\boldsymbol{a}}$ depending on the offset $B_t^{w,\boldsymbol{a}} =$

$$\{ f(\boldsymbol{d}, \boldsymbol{a}) \mid f \in F_t, \boldsymbol{d} \in \text{nnp}(w)^{s'}, d_1 > d_2 > \dots > d_{s'} \}.$$

Let q be the lcm of all q' where $\equiv_{q'}$ occurs in one of the ϕ_i. In the following we fix a word $w \in \Sigma^* \lambda^\omega$ and an $\boldsymbol{a} \in \mathbb{N}^r$.

Lemma 18. $\{\rho(d_1, \dots, d_s, \boldsymbol{a}) \mid \rho \in R, d_i \in nnp(w)\} \cup \text{nnp}(w) \subseteq B^{w,\boldsymbol{a}}$

Proof. Let $S = \{\rho(d_1, \dots, d_s, \boldsymbol{a}) \mid \rho \in R, d_i \in nnp(w)\} \cup \text{nnp}(w)$. Since $\rho(x_1) = x_1$ is in F_t, for some $t \in T$, we have $nnp(w) \subseteq B^{w,\boldsymbol{a}}$. Let $b \in S$. Then there is a function $\rho = \sum_i^{s'} \alpha_i x_i + t(\boldsymbol{y})$ in F_t and values $p_1, \dots, p_{s'} \in nnp(w)$ such that $b = \rho(p_1, \dots, p_{s'}, \boldsymbol{a})$. Let $p'_1 > p'_2 > \dots > p'_l$ be the ordered set of all p_is in the above assignment. We let $\rho'(x_1, \dots, x_l) = \sum_i \beta_i x_i + t$, where $\beta_i = \sum_{j : p_j = p'_i} \alpha_j$. Therefore $b = \rho'(p'_1, \dots, p'_l)$. Since $|\beta_i| \leq \Delta \cdot s$ we have $\rho' \in F_t$ and hence $b \in B_t^{w,\boldsymbol{a}}$. □

We need the following lemma, that inside an interval with only neutral letters, the congruence relations decide the truth of an active domain formula.

Lemma 19. *Let $a_1, \dots, a_r \in \mathbb{N}$ and let $c, d \in \mathbb{N}$ belong to the same interval in $B^{w,\boldsymbol{a}}$ such that $c \equiv_q d$. Then for all $i \leq K$: $w, c \vDash \phi_i(z, \boldsymbol{a}) \Leftrightarrow w, d \vDash \phi_i(z, \boldsymbol{a})$.*

Proof. Proof is by induction on the structure of the formula $\hat{\phi}_i$. We will now show that $\forall b_i \in \text{nnp}(w)$ and all subformulas $\psi(z, \boldsymbol{x}, \boldsymbol{y})$ of $\hat{\phi}_i$ that $w, c, \boldsymbol{b}, \boldsymbol{a} \vDash \psi \Leftrightarrow w, d, \boldsymbol{b}, \boldsymbol{a} \vDash \psi$. The atomic formulas of $\hat{\phi}_i(z, \boldsymbol{a})$ are of the following form: $z < \rho(\boldsymbol{x}, \boldsymbol{a}), z = \rho(\boldsymbol{x}, \boldsymbol{a}), z > \rho(\boldsymbol{x}, \boldsymbol{a}), z \equiv_{q'} \rho(\boldsymbol{x}, \boldsymbol{a}), a(z)$ and formulas which does not depend on z. It is clear that the truth of formulas which does not depend on z, $a(z)$ and $z \equiv_{q'} \rho$ does not change whether we assign c or d to z. Let $\boldsymbol{b} \in nnp(w)^s$. By Lemma 18 we know that $\rho(\boldsymbol{b}, \boldsymbol{a})$ is in $B^{w,\boldsymbol{a}}$ and since c, d lies in the same interval it follows that $c < \rho(\boldsymbol{b}, \boldsymbol{a}) \Leftrightarrow d < (\boldsymbol{b}, \boldsymbol{a})$. Similarly we can show that the truth of $z > \rho, z = \rho$ does not change on z being assigned c or d. Thus we have that the claim holds for atomic formulas. The claim clearly holds for conjunction and negation of formulas. Now let the claim hold for subformulas $\psi_1, \dots, \psi_K$. Therefore $\forall i \leq K$ we have that $\{\boldsymbol{b} \in \text{nnp}(w)^s \mid w, c, \boldsymbol{b}, \boldsymbol{a} \vDash \psi_i\} = \{\boldsymbol{b} \in \text{nnp}(w)^s \mid w, d, \boldsymbol{b}, \boldsymbol{a} \vDash \psi_i\}$. Therefore we have that

$$\begin{aligned} & w, c, b_2, \dots, b_s, \boldsymbol{a} \vDash Q_M^m x \ \neg\lambda(x) \langle \psi_1, \dots, \psi_K \rangle \\ \Leftrightarrow\ & w, d, b_2, \dots, b_s, \boldsymbol{a} \vDash Q_M^m x \ \neg\lambda(x) \langle \psi_1, \dots, \psi_K \rangle \end{aligned}$$

And hence it is closed under active domain quantification. □

The following Lemma shows how to deal with the infinite interval.

Lemma 20. *Let b belong to the infinite interval and $\boldsymbol{a} \in \mathbb{N}^r$. If $w, \boldsymbol{a} \vDash \phi$ then $w, b, \boldsymbol{a} \nvDash \phi_i$ for any $i \leq K$.*

Proof. Let $i \le K$ and b be in the infinite interval and $w, b, \boldsymbol{a} \vDash \phi_i$. From Lemma 19 we know that all points $c \equiv_q b$ and such that c is also in the infinite interval will be a witnesses for ϕ_i. This means the set of witnesses is infinite and hence $w, \boldsymbol{a} \nvDash \phi$. □

Lemma 19 shows that inside an interval, the congruence relations decide the satisfiability of the formulas ϕ_is. This shows that it is enough to know the truth values of ϕ_i at a distance of $\ge q$ from the boundary points, since the truth values inside an interval are going to repeat after every q positions. The rest of the proof shows

1. How we can treat each B_t differently.
2. There is an active domain formula which goes through the points in B_t in an increasing order

We fix the word $w \in \Sigma^* \lambda^\omega$ and assignment $\boldsymbol{a}$. Therefore we drop the superscripts in $B^{w,\boldsymbol{a}}$ ($B_t^{w,\boldsymbol{a}}$) and call them B (B_t).

5.2 Treating each B_t differently

Let $p = q|G|$, where q was defined in the previous section and depends on the $\equiv_{q'}$ predicates. For an element $g \in G$, we have $g^{|G|} = 1_G$, so $g^x = g^{x+|G|}$. Recall the definitions of T, B from Section 5.

Recall from the Preliminaries (Section 2) that we denoted by $u(i)$ the group element at position i. That is $u(i) = m_j$ iff $w, \boldsymbol{a} \models \phi_j \wedge \bigwedge_{l<j} \neg\phi_l$. Our aim is to give an active domain formula such that the formula evaluates to true iff the group element $\prod_{i=0} u(i)$ is equal to m. The rest of this subsection will be devoted to computing this product in a way which helps in building an active domain formula.

Let $b < b'$ be boundary points in B. Below we compute $\prod_{i=b+1}^{b'-1} u(i)$ in a different way:

$$\prod_{i=b+1}^{b'-1} u(i) = \prod_{i>b} u(i) \left(\prod_{i \ge b'} u(i) \right)^{-1}.$$

Observe that we can compute the product of the interval using two terms that both need to know only one boundary of the interval. It becomes simpler if we note that the two products do not really need to multiply all the elements $u(i)$, for $i \ge b'$ but simply agree on a common set of elements to multiply.

For a $b \in B$, we define the function $\mathrm{IL}(b)$ to be the length of the interval to the left of b. That is if (b', b) form an interval then $\mathrm{IL}(b) = b - b' - 1$. Similarly we define $\mathrm{IR}(b)$ to be the length of the interval to the right of b. For all $k \le |T|$, we define functions $N_k(b)$ and $\hat{N}_k(b)$, which maps points $b \in B$ to a group element.

$$N_0(b) = \begin{cases} u(b+1)u(b+2)\dots u(b+\mathrm{IR}(b)) & \text{if } \mathrm{IR}(b) < p \\ u(b+1)u(b+2)\dots u(b+r) & \\ & \text{if } \mathrm{IR}(b) \ge p \text{ and } r < p, b+r \equiv_p 0 \end{cases}$$

$$\hat{N}_0(p) = \begin{cases} 1_G & \text{if } \mathrm{IL}(b) < p \\ u(b-p)\dots u(b-p+r) & \\ & \text{if } \mathrm{IL}(b) \ge p \text{ and } r < p, b+r \equiv_p 0 \end{cases}$$

Inductively we define

$$N_k(b) = N_{k-1}(b) \prod_{\substack{b' \in B_{t_k} \\ b' > b}} \left(\hat{N}_{k-1}(b')\right)^{-1} u(b') N_{k-1}(b'),$$

$$\hat{N}_k(b) = \hat{N}_{k-1}(b) \prod_{\substack{b' \in B_{t_k} \\ b' > b}} \left(\hat{N}_{k-1}(b')\right)^{-1} u(b') N_{k-1}(b').$$

We first show how $N_k(b)$ and $N_k(b')$ are related for $b, b' \in B$.

Lemma 21. *Let $0 \leq k \leq |T|$. Let $b < b' \in B$ such that there are no points $b'' \in \bigcup_{i>k} B_{t_i}$, where $b < b'' < b'$. Then $N_k(b)(\hat{N}_k(b'))^{-1} = \prod_{i=b+1}^{b'-1} u(i)$.*

Proof. We prove this by induction over k. Let $k = 0$ and let (b, b') form an interval in B. If $b' - b \leq p$ then $(N_0(b))(\hat{N}_0(b'))^{-1} =$

$$\big(u(b+1)u(b+2)\ldots u(b+\mathrm{IR}(b))\big)(1_G)^{-1} = \prod_{i=b+1}^{b'-1} u(i)$$

If the interval is large, i.e. $b' - b > p$, then let $s, t \in \mathbb{N}$, be the smallest, resp. the largest numbers such that $b \leq s \leq t \leq b'$ and $s \equiv_p t \equiv_p 0$. Lemma 19 shows that inside an interval all positions congruent modulo q satisfy the same formulas. Therefore $u(b'-p)u(b'-p+1)\ldots u(b'-1) = 1_G$, and hence $(u(b'-p)u(b'-p+1)\ldots u(t))^{-1} = (u(t+1)\ldots u(b'-1))$. So $N_0(b)(\hat{N}_0(b'))^{-1} =$

$$\big(u(b+1)u(b+2)\ldots u(s)\big)\big(u(t+1)\ldots u(b'-1)\big) = \prod_{i=b+1}^{b'-1} u(i)$$

The last equality being true since $u(s+1)\ldots u(t) = 1_G$

As induction hypothesis assume that the lemma is true for all $k' < k$. Since for all $b'' > b'$ the terms $\big(\hat{N}_{k-1}(b'')\big)^{-1} u(b'')N_{k-1}(b'')$ appear in both $N_k(b)$ and $\hat{N}_k(b')$ they cancel out (whatever they compute to). Thus $N_k(b)(\hat{N}_k(b'))^{-1} =$

$$\left(N_{k-1}(b) \prod_{\substack{b'' \in B_{t_k} \\ b < b'' < b'}} \big(\hat{N}_{k-1}(b'')\big)^{-1} u(b'')N_{k-1}(b'')\right)(\hat{N}_{k-1}(b'))^{-1}$$

Let $b = b_0 < b_1 < \cdots < b_{x-1} < b_x = b'$ be all positions in B_{t_k} between b and b'. By the requirements of the lemma the only positions of B between b_i and b_{i+1} are in $\bigcup_{i<k} B_{t_i}$. Writing out the product we get $N_k(b)(\hat{N}_k(b'))^{-1}$ is equal to

$$N_{k-1}(b_0)\left(\hat{N}_{k-1}(b_1)\right)^{-1} \prod_{i=1}^{x-1} u(b_i)\ N_{k-1}(b_i)\left(\hat{N}_{k-1}(b_{i+1})\right)^{-1}$$

By I.H. $N_{k-1}(b_i)\left(\hat{N}_{k-1}(b_{i+1})\right)^{-1} = \prod_{i=b_i+1}^{b_{i+1}-1} u(i)$. Hence $N_k(b)(\hat{N}_k(b')) = \prod_{i=b+1}^{b'-1} u(i)$. □

The following Lemma shows that $u(0)N_{|T|}(0)$ gives the product of the group elements.

Lemma 22. *We have that $u(0)N_{|T|}(0) = \prod_i u(i)$.*

Proof. Using appropriate induction hypothesis one can show that $N_{|T|}(0) = \prod_{i=1}^{l} u(i)$, where $l > \max(B)$. The lemma now follows from Lemma 20 which shows that $u(i) = 1_G$ for every i in the infinite interval. □

We now give active domain formulas γ^m, $m \in G$, such that γ^m is true iff $N_{|T|}(0) = m$. For this we make use of the inductive definition of N_k and show that there exists active domain formulas γ^m ($\hat{\gamma}^m$) such that $w \models \gamma^m(b) \Leftrightarrow N_k(b) = m$ ($w \models \hat{\gamma}^m(b) \Leftrightarrow \hat{N}_k(b) = m$). Observe that $N_k(b)$ is got by computing the product of $\left(\hat{N}_{k-1}(b')\right)^{-1} u(b')N_{k-1}(b')$, over b', where b' strictly increases. This requires us to traverse the elements in $B_{t_{k-1}}$ in an increasing order. The following section builds a Sorting tree to sort the elements of $B_{t_{k-1}}$ in an increasing order.

5.3 Sorting Tree

For a $t \in T$, we define a tree called *sorting tree*, $\mathcal{T}_t$ which corresponds to B_t. The tree satisfies the following property. If the leaves of the tree are enumerated from left to right, then we get the set B_t in ascending order. A node in $\mathcal{T}_t$ is labeled by a tuple (f, A), where $f(x_1, \ldots, x_l)$ is a function in F_t, A an assignment for the variables in f such that $A(x_1) > A(x_2) > \cdots > A(x_l)$ and $\forall i \le l : A(x_i) \in nnp(w)$.

We show how to inductively built the tree. The root is labeled by the tuple $(t, \{\})$, where t is the function which depends only on $\boldsymbol{y}$ (and hence constant on $\boldsymbol{x}$) and $\{\}$ is the empty assignment. The root is not marked a leaf node.

Consider the internal node $(f(x_1, \ldots, x_l), A)$. It will have three kinds of children ordered from left to right as follows.

1. Left children: These are labeled by tuples of the form (f'_α, A'_j) where $f'_\alpha(x_1, \ldots, x_{l+1}) = f(x_1, \ldots, x_l) + \alpha x_{l+1}$ and $-\Delta \le \alpha < 0$, $-\alpha \in \mathbb{N}$, $A'_j = A \cup [x_{l+1} \mapsto j]$, where $j < A(x_l)$ and $j \in \mathrm{nnp}(w)$.
   The tuples $(f'_{\alpha_1}, A'_{j_1})$ is on the left of $(f'_{\alpha_2}, A'_{j_2})$ if $j_1 > j_2$ or if $j_1 = j_2$ and $\alpha_1 < \alpha_2$.
2. Middle child: It is labeled by the tuple (f'', A) where $f''(x_1, \ldots, x_l) = f(x_1, \ldots, x_l)$. It is marked a **leaf** node.
3. Right children: These are labeled by tuples of the form (f'_α, A'_j) where $f'_\alpha(x_1, \ldots, x_{l+1}) = f(x_1, \ldots, x_l) + \alpha x_{l+1}$ and $0 < \alpha \le \Delta$, $\alpha \in \mathbb{N}$, $A'_j = A \cup [x_{l+1} \mapsto j]$, where $j < A(x_l)$ and $j \in \mathrm{nnp}(w)$.
   The tuple $(f'_{\alpha_1}, A'_{j_1})$ is on the left of $(f'_{\alpha_2}, A'_{j_2})$ if $j_1 < j_2$ or $j_1 = j_2$ and $\alpha_1 < \alpha_2$.

Observe that if there is no j such that $j < A(x_l)$ and $j \in \mathrm{nnp}(w)$, then (f, A) will only have the child (f'', A).

Note that in our tree construction the values of the children of a node increase from left to right. The tree is built until all functions with s variables appear in leaves and hence the depth of the tree is $s + 2$. Figure 1 shows part of a tree, where $\Delta = 2$, $t = 0$, $\mathcal{R} = 5$ and $\mathrm{nnp}(w) = \{5, 25, 625\} \subseteq \mathcal{D}_\mathcal{R}$.

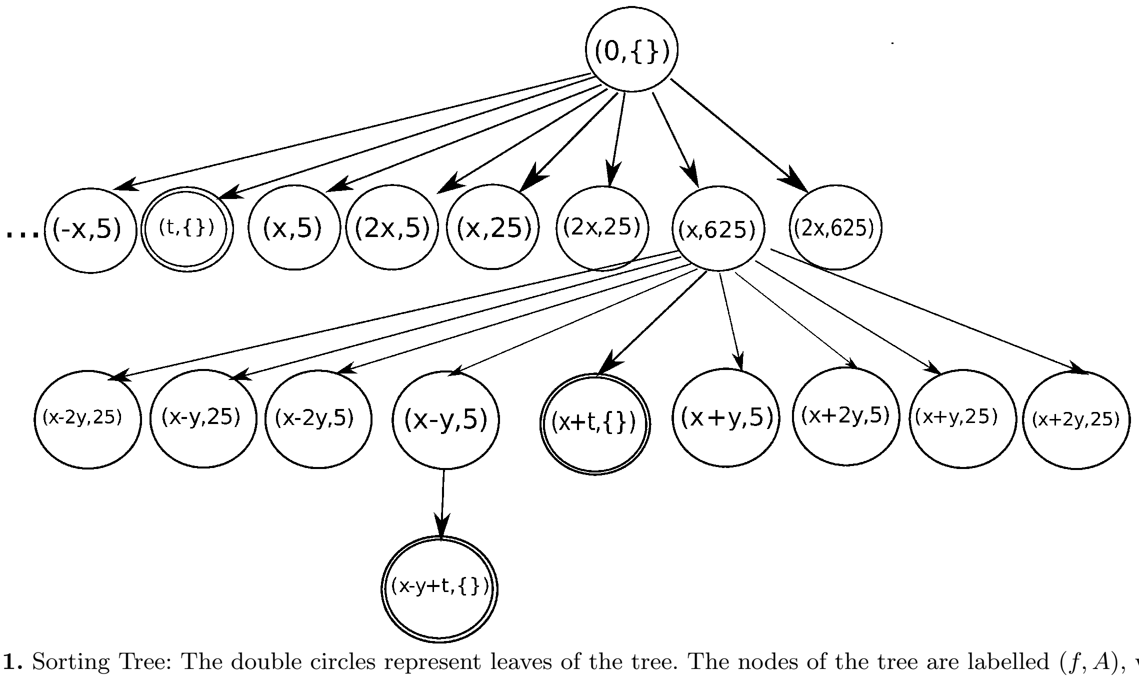

Fig. 1. Sorting Tree: The double circles represent leaves of the tree. The nodes of the tree are labelled (f, A), where A is an assignment for the function f and $t = 0$. For better presentation we only show the assignment to the newly introduced variable in a node. For example, the tuple $(x - 2y, 25)$ assigns $x = 625$ and $y = 25$. The assignment to x is given in the node's parent.

The following lemma holds if $\mathcal{R} > 3s\Delta$. We also assume that $nnp(w) \subseteq \mathcal{D}_\mathcal{R}$. Given a node (f, A), we say the value of the node is the function f evaluated under the assignment of A (denoted by $f(A)$).

Lemma 23. *Let N be an internal node labeled by a function $f(x_1, \ldots, x_l)$ with $l < s$ and an assignment A. If $A(x_l) = \mathcal{R}^c$ for some $c \geq 1$, then the children of this node have values in the range $[f(A) - \Delta\mathcal{R}^{c-1}, f(A) + \Delta\mathcal{R}^{c-1}]$. Moreover the values of the children increases from left to right.*

Proof. By construction. □

Next we show that for any two neighboring nodes in the tree, the values in the leaves of the subtree rooted at the left node is less than the values in the leaves of the subtree rooted at the right node. Let $V_{(f,A)}$ denote the set of values in the leaves of the subtree rooted at (f, A).

Lemma 24. *Let (f, A) and (f', A') be neighboring nodes of the same parent such that (f, A) is to the left of (f', A'). Then $u < v$ for every $u \in V_{(f,A)}$ and $v \in V_{(f',A')}$.*

Proof. Let $f = \sum_{i=1}^{l-1} \alpha_i x_i + \alpha_l x_l + t$ and $f' = \sum_{i=1}^{l-1} \alpha_i x_i + \alpha'_l x_l + t$. We show that the rightmost element, u in $V_{(f,A)}$ is less than the left most element, v in $V_{(f',A')}$. From Lemma 23 and applying induction on the depth of the tree, one can show that $u \leq f(A) + (s-l)\Delta\mathcal{R}^{c-1}$ and $v \geq f'(A') - (s-l)\Delta\mathcal{R}^{c'-1}$. Here $\mathcal{R}^c, \mathcal{R}^{c'}$ are the minimum assignments in A and A' respectively. Let us assume that both coefficients $\alpha'_l, \alpha_l > 0$. A similar analysis can be given for other combinations of α'_l and α_l. Now since (f, A) is the left neighbor of (f', A') we have $\mathcal{R}^c < \mathcal{R}^{c'}$. Then $v - u \geq \alpha'_l\mathcal{R}^{c'} - (s-l)\Delta\mathcal{R}^{c'-1} - \alpha_l\mathcal{R}^c - (s-l)\Delta\mathcal{R}^{c-1} \geq \mathcal{R}^{c'} - 3s\Delta\mathcal{R}^{c'-1} > 0$. The claim follows, since $\mathcal{R} > 3s\Delta$. □

The next lemma says that the values of the leaves of the tree increases as we traverse from left to right.

Lemma 25. *Let (f, A) and (f', A') be two distinct nodes such that $f(A) < f'(A')$. Then (f, A) appear to the left of (f', A').*

Proof. This follows from Lemma 24. □

Lemma 26 (Tree Lemma). *Fix $t \in T$. Assume that for every $b \in B_t$ we have an element $g_b \in G$. For all $f \in F_t$, $m \in G$ let $\gamma_f^m(\boldsymbol{x})$ be active domain formulas such that $w, \boldsymbol{d} \models \gamma_f^m(\boldsymbol{x})$ iff $g_{f(\boldsymbol{d})} = m$. Then there are active domain formulas $\Gamma^{m'}$ such that $w \models \Gamma^{m'}$ iff $\prod_{b \in B_t} g_b = m'$.*

Proof. We will use the sorting tree, $\mathcal{T}_t$ corresponding to B_t for the construction of our formula. Recall that the nodes are labeled by tuples (f, A), where f is a function and A is the assignment of the parameters of f. Let $V_{(f,A)} \subseteq B_t$ be the set of values at the leaves of the subtree rooted at the node labeled by (f, A), and $g_{(f,A)} = \prod_{b \in V_{(f,A)}} g_b$. We will do induction on the depth D of the tree. Let $\tau_f^{m,D}(\boldsymbol{x})$ be a formula such that $w, \boldsymbol{d} \models \tau_f^{m,D}(\boldsymbol{x})$ iff $\prod_{b \in V_{(f,\boldsymbol{d})}} g_b = m$ where $(f, \boldsymbol{d})$ is the label of a node that has a subtree of depth at most D. Hence we multiply all group elements g_b for which b is in $V_{(f,\boldsymbol{d})}$.

Base Case (leaves): We define $\tau_f^{m,0}(\boldsymbol{x}) = \gamma_f^m(\boldsymbol{x})$.

Induction Step: Let us assume that the claim is true for all nodes with a subtree of depth at most D. Let the node labeled by (f, A) have a subtree of depth $D+1$. We will need to specify the formula $\tau_f^{m,D+1}(\boldsymbol{x})$, where $\boldsymbol{x}$ agrees with the assignment A. For every child (f', A') of (f, A) the depth of the corresponding subtree is less than or equal to D. Hence we know we have already formulas by induction.

Recall what the children of (f, A) are: They are of form (f'_α, A'_j) and (f'', A_j). Moreover all nodes (f'_α, A'_j), where α is negative, come to the left of (f'', A_j) and all nodes (f'_α, A'_j), where α is positive, come to its right.

We start by grouping some of the children and computing their product. We let $T^-(A'_j)$ be the product of all subtrees labeled by (f'_α, A'_j) for $\alpha = -\Delta, -\Delta+1, \ldots, -1$. This is a finite product so we can compute this by a Boolean combination of the formulas $\tau_{f'_\alpha}^{m,D}(\boldsymbol{x}, x_{l+1})$.

$$\pi_f^{-,m,D}(\boldsymbol{x}, x_{l+1}) ::= \bigvee_{m_{-\Delta} \cdots m_{-1} = m} \left(\bigwedge_{\alpha=-\Delta}^{-1} \tau_{f'_\alpha}^{m_\alpha,D}(\boldsymbol{x}, x_{l+1}) \right)$$

Now we want to compute the product $\left(\prod_{j \in \mathrm{nnp}(w)} (T^{-}(A'_j))^{-1}\right)^{-1}$ which is the product of the $T^{-}(A'_j)$ where $j \in \mathrm{nnp}(w)$ is decreasing. But this can be computed using an active domain group quantifier, $\tau_f^{-,m,D}(\boldsymbol{x})$ as follows:

$$\tau_f^{-,m,D}(\boldsymbol{x}) = Q_G^{m^{-1}} x_{l+1} \left(\neg\lambda(x_{l+1}) \wedge (x_l > x_{l+1})\right)$$
$$\langle \pi_f^{-,m_1^{-1},D}(\boldsymbol{x}, x_{l+1}), \ldots, \pi_f^{-,m_K^{-1},D}(\boldsymbol{x}, x_{l+1})\rangle$$

Recall that the elements of group G are ordered $m_1, \ldots, m_K$. For the single node (f'', A) we already have the formulas $\tau_{f''}^{m,D}(\boldsymbol{x})$ by induction (here we have $\boldsymbol{x}$ since the assignment A is the same for (f, A) and (f'', A)).

Similarly we define formulas $\pi_f^{+,m,D}(\boldsymbol{x}, x_{l+1})$ for the positive coefficients, and compute their product $\prod_{j \in \mathrm{nnp}(w)} T^{+}(A'_j)$ in an increasing order.

$$\pi_f^{+,m,D}(\boldsymbol{x}, x_{l+1}) ::= \bigvee_{m_1 \ldots m_\Delta = m} \left(\bigwedge_{\alpha=1}^{\Delta} \tau_{f'_\alpha}^{m_\alpha,D}(\boldsymbol{x}, x_{l+1}) \right)$$

$$\tau_f^{+,m,D}(\boldsymbol{x}) ::= Q_G^{m} x_{l+1} \left(\neg\lambda(x_{l+1}) \wedge (x_l > x_{l+1})\right)$$
$$\langle \pi_f^{+,m_1,D}(\boldsymbol{x}, x_{l+1}), \ldots, \pi_f^{+,m_K,D}(\boldsymbol{x}, x_{l+1})\rangle$$

We have now computed the product of the group elements for the three different groups of children. So by a Boolean combination over these formulas we get $\tau_f^{m,D+1}(\boldsymbol{x})$:

$$\bigvee_{m'm''m'''=m} \left(\tau_f^{-,m',D}(\boldsymbol{x}) \wedge \tau_{f''}^{m'',D}(\boldsymbol{x}) \wedge \tau_f^{+,m''',D}(\boldsymbol{x})\right)$$

So finally we get $\Gamma^{m'}$ which is same as the formula $\tau_{(0,\{\})}^{m',s+2}$, which is valid at the root of the tree. □

Since the above lemma holds only for $\mathcal{R} > 3s\Delta$, our $\mathcal{R}_\phi$ should be greater than $3s\Delta$.

5.4 Construction of the formula

We know that for every $b \in B$ there is a function $f \in F$, $d_1, \ldots, d_{s'} \in \mathrm{nnp}(w)$, such that $b = f(\boldsymbol{d}, \boldsymbol{a})$, where $\boldsymbol{a}$ is the fixed assignment to the variables $\boldsymbol{y}$. We will use this encoding of a position and define a formula $\nu_{k,f}^m$ such that $w, \boldsymbol{d}, \boldsymbol{a} \models \nu_{k,f}^m(\boldsymbol{x}, \boldsymbol{y})$ iff $N_k(f(\boldsymbol{d}, \boldsymbol{a})) = m$. Similarly we define formulas $\hat{\nu}_{k,f}^m$ such that $w, \boldsymbol{d}, \boldsymbol{a} \models \hat{\nu}_{k,f}^m(\boldsymbol{x}, \boldsymbol{y})$ iff $\hat{N}_k(f(\boldsymbol{d}, \boldsymbol{a})) = m$.

We show this by induction over $k \leq |T|$. Starting with the base case $k = 0$.

Lemma 27. *Let $\boldsymbol{a} \in \mathbb{N}^r$. For each $m \in G$, there is an active domain formula $\nu_{0,f}^m(\boldsymbol{x}, \boldsymbol{y})$ in $\mathcal{L}[<,+]$, such that if $w \models \nu_{0,f}^m(\boldsymbol{d}, \boldsymbol{a})$ then $N_0(f(\boldsymbol{d}, \boldsymbol{a})) = m$.*
Similarly there is an active domain formula $\hat{\nu}_{0,f}^m(\boldsymbol{x}, \boldsymbol{y})$ in $\mathcal{L}[<,+]$ such that if $w, \boldsymbol{d} \models \hat{\nu}_{0,f}^m(\boldsymbol{x}, \boldsymbol{a})$ then $\hat{N}_0(f(\boldsymbol{d}, \boldsymbol{a})) = m$.

Proof. For an $i \leq K$, we denote by $\tilde{\phi}_{m_i}$ the formula $\bigwedge_{j<i} \neg\phi_j(x, \boldsymbol{y}) \wedge \phi_i(x, \boldsymbol{y})$. For a $l \in \mathbb{N}$, the following formula checks if there is a point b' in B such that $b + l = b'$. Since in each B there is at most one such element, we can use the group quantifier to simulate the existential quantifier.

$$\delta_f^l ::= \bigvee_{f' \in F \setminus f} Q_G^{m_1} \boldsymbol{x}' \langle f'(\boldsymbol{x}', \boldsymbol{y}) = f(\boldsymbol{x}, \boldsymbol{y}) + l, \mathit{false}, \ldots, \mathit{false}\rangle$$

So we have that $\mathrm{IR}(b) = l$ iff $\delta_f^{l+1} \wedge \bigwedge_{l'<l} \neg\delta_f^l$ is true. We define $\pi_f^{m,l}$ to be true if the product of the first l group elements is m.

$$\pi_f^{m,l} ::= \bigvee_{g_0 \cdots g_l = m} \left(\bigwedge_{i=0}^{l} \tilde{\phi}_{g_i}(f(\boldsymbol{x}, \boldsymbol{y}) + i) \right)$$

Now we have two cases to consider.
Case IR$(b) < p$**:** For each of the case $b < b'$ such that $l = b' - b \leq p$, the formula $\pi_f^{m,l}$ compute the product of the group elements. Hence $\nu_{0,f}^m$ in this case can be given as:

$$\bigwedge_{l=0}^{p-1} \Big(\big(\delta_f^l(\boldsymbol{x},\boldsymbol{y}) \wedge \bigwedge_{l'=0}^{l-1} \neg\delta_f^{l'}(\boldsymbol{x},\boldsymbol{y})\big) \rightarrow \pi_f^{m,l}(\boldsymbol{x},\boldsymbol{y}) \Big)$$

Case IR$(b) \geq p$**:** When $b' - b > p$ we have to compute the product for the first r group elements, where $b + r \equiv_p 0$ and $r < p$. Therefore $\nu_{0,f}^m$ in this case is

$$\bigwedge_{l=0}^{p-1} \big(f(\boldsymbol{x},\boldsymbol{y}) + r \equiv_p 0\big) \rightarrow \pi_f^{m,r}(\boldsymbol{x},\boldsymbol{y})$$

A Boolean combination over δ_f^l can differentiate the two cases. Similarly we can give active domain formulas $\hat{\nu}_{0,f}^m(\boldsymbol{x},\boldsymbol{y})$. □

The induction step follows.

Lemma 28. *Let $\boldsymbol{a} \in \mathbb{N}^r$. For each $m \in G$, there is an active domain formula $\nu_{k,f}^m$ in $\mathcal{L}[<,+]$, such that $w, \boldsymbol{d}, \boldsymbol{a} \models \nu_{k,f}^m(\boldsymbol{x},\boldsymbol{y})$ then $N_k(f(\boldsymbol{d},\boldsymbol{a})) = m$.*
Similarly there is an active domain formula $\hat{\nu}_{k,f}^m(\boldsymbol{x},\boldsymbol{y})$ in $\mathcal{L}[<,+]$, such that $w, \boldsymbol{d}, \boldsymbol{a} \models \hat{\nu}_{k,f}^m(\boldsymbol{x},\boldsymbol{y})$ then $\hat{N}_k(f(\boldsymbol{d},\boldsymbol{a})) = m$.

Proof. For all $m \in G$ and $f' \in F_{t_k}$ we give formulas $\gamma_{f'}^m$ such that for all $\boldsymbol{d}' \in nnp(w)^s$ the following holds. Let $f'(\boldsymbol{d}',\boldsymbol{a}) = b'$ and $f(\boldsymbol{d},\boldsymbol{a}) = b$. Then $w, \boldsymbol{d}, \boldsymbol{d}', \boldsymbol{a} \models \gamma_{f'}^m(\boldsymbol{x},\boldsymbol{z},\boldsymbol{y}) \Leftrightarrow b' \leq b$ and $m = 1_G$ or $b' > b$ and $\left(\hat{N}_{k-1}(b')\right)^{-1} u(b') N_{k-1}(b') = m$. By induction hypothesis there exists formulas $\nu_{k-1,f'}^m$ and $\hat{\nu}_{k-1,f'}^m$ which corresponds to $N_{k-1}(f'(\boldsymbol{x},\boldsymbol{y}))$ and $\hat{N}_{k-1}(f'(\boldsymbol{x},\boldsymbol{y}))$ respectively. Taking a Boolean combination over these formulas we get the required formula $\gamma_{f'}^m$. We now apply our Tree Lemma 26 which gives us formulas Γ^m, for all $m \in G$, such that $w, \boldsymbol{d}, \boldsymbol{a} \models \Gamma^m(\boldsymbol{x},\boldsymbol{y})$ iff

$$w \models \prod_{\substack{b' \in B_{t_k} \\ b' > b}} \left(\hat{N}_{k-1}(b')\right)^{-1} u(b') N_{k-1}(b') = m$$

Taking Boolean combination over Γ^m and $\hat{\nu}_{k-1,f}^m$ will give us the formula $\nu_{k,f}^m$. Similarly we can build active domain formulas $\hat{\nu}_{k,f}^m(\boldsymbol{x},\boldsymbol{y})$, for all $m \in G$. □

Proof (Proof of Lemma 12). By Lemma 22 we know that it suffices to compute $N_{|T|}(0)$ and by Lemma 28 we know that there are active domain formulas $\Gamma^m \in \mathcal{L}[<,+]$ such that $N_{|T|}(0) = m$ iff $w, \boldsymbol{0}, \boldsymbol{a} \models \Gamma^m$

We need to do one last thing. Check that the infinite interval evaluates to 1_G. Replace all formulas $z > \rho$, $z < \rho$, $c(z)$ for a $c \neq \lambda$ and $\lambda(z)$ by $true$, $false$, $false$, $true$ respectively in the formulas $\hat{\phi}_i$ and call these formulas $\hat{\psi}_i$. There exists a witness in the infinite interval for the formula $\hat{\phi}_i$ iff $\hat{\psi}_i$ evaluates to true. By Theorem 20 there should not be any witness in the infinite interval. Hence there exists a $\hat{\psi}_i$ which evaluates to true iff the infinite interval does not evaluate to 1_G. □

6 Lower bounds on Graphs

7 Almost Neutral Letter languages

8 Discussion

We have shown that in the presence of a neutral letter the addition relation collapse to linear ordering no matter what monoid quantifier is been used. All languages definable using monoid quantifiers and an order predicate, on the other hand, are regular [3]. Now using semigroup theoretic methods we can separate these classes [26]. This enabled us to show separation between various logics which uses addition and order predicates.

Unfortunately if both addition and multiplication are present, then the collapse does not happen. It is also interesting to note that non-solvable groups do not show any surprising property if only addition is present, but as we know from Barrington's theorem non-solvable groups behave quite differently when both addition and multiplication are present.

The ultimate objective is to show non-expressibility results for arbitrary predicates or at least when both addition and multiplication are present. As a first step one can look at extending these results for other kinds of predicates.

Another way to look at separating the "natural uniform" versions of the complexity classes will be to ask whether one can come up with other suitable restrictions on the set of languages. Inside this restricted set of languages can one show addition and multiplication collapse to order relation? This seems to be the idea Straubing considers in [27]. Straubing [26] proposes word problems over Regular language as a suitable restriction, while McKenzie, Thomas, Vollmer [20] consider context free languages as a restriction.

Another interesting question which our result fails to answer is whether word problems over non-solvable groups can be defined in MAJ$[<,+]$ [13]?

Acknowledgement

We like to thank Baskar Anguraj, Christoph Behle, Michaël Cadilhac, Klaus-Jörn Lange, Nutan Limaye, T. Mubeena, and Ramanujam for a lot of useful comments on the draft of this paper.

References

1. David A. Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1. *Journal of Computer and System Sciences*, 38(1):150–164, February 1989.
2. David A. Mix Barrington, Neil Immerman, Clemens Lautemann, Nicole Schweikardt, and Denis Thérien. First-order expressibility of languages with neutral letters or: The Crane Beach conjecture. *J. Comput. Syst. Sci.*, 70(2):101–127, 2005.
3. David A. Mix Barrington, Neil Immerman, and Howard Straubing. On uniformity within NC^1. *Journal of Computer and System Sciences*, 41(3):274–306, December 1990.
4. David A. Mix Barrington and Howard Straubing. Superlinear lower bounds for bounded-width branching programs. In *Structure in Complexity Theory Conference*, pages 305–313, 1991.
5. Christoph Behle, Andreas Krebs, and Stephanie Reifferscheid. Non-solvable groups are not in FO+MOD+M$\hat{\text{A}}$J2[REG]. In *LATA*, pages 129–140, 2009.
6. Christoph Behle, Andreas Krebs, and Stephanie Reifferscheid. Regular languages definable by majority quantifiers with two variables. In *Developments in Language Theory*, pages 91–102, 2009.
7. Christoph Behle and Klaus-Jörn Lange. FO$[<]$-uniformity. In *IEEE Conference on Computational Complexity*, pages 183–189, 2006.
8. Michael Benedikt and Leonid Libkin. Relational queries over interpreted structures. *J. ACM*, 47(4):644–680, 2000.
9. Arkadev Chattopadhyay, Andreas Krebs, Michal Koucký, Mario Szegedy, Pascal Tesson, and Denis Thérien. Languages with bounded multiparty communication complexity. In *STACS*, pages 500–511, 2007.
10. Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984.
11. Neil Immerman. *Descriptive complexity.* Graduate texts in computer science. Springer, 1999.
12. Michal Koucký, Pavel Pudlák, and Denis Thérien. Bounded-depth circuits: separating wires from gates. In *STOC*, pages 257–265, 2005.
13. Andreas Krebs, Klaus-Jörn Lange, and Stephanie Reifferscheid. Characterizing TC^0 in terms of infinite groups. *Theory Comput. Syst.*, 40(4):303–325, 2007.
14. Klaus-Jörn Lange. Some results on majority quantifiers over words. In *IEEE Conference on Computational Complexity*, pages 123–129. IEEE Computer Society, 2004.

15. Clemens Lautemann, Pierre McKenzie, Thomas Schwentick, and Heribert Vollmer. The descriptive complexity approach to LOGCFL. *J. Comput. Syst. Sci*, 62(4):629–652, 2001.
16. Clemens Lautemann, Pascal Tesson, and Denis Thérien. An algebraic point of view on the crane beach property. In *CSL*, pages 426–440, 2006.
17. Leonid Libkin. *Elements of Finite Model Theory.* Springer-Verlag, Berlin, 2004.
18. P. Lindström. First order predicate logic with generalized quantifiers. *Theoria*, 32:186–195, 1966.
19. James F. Lynch. On sets of relations definable by addition. *J. Symb. Log.*, 47(3):659–668, 1982.
20. Pierre McKenzie, Michael Thomas, and Heribert Vollmer. Extensional uniformity for boolean circuits. *SIAM Journal on Computing*, 39(7):3186–3206, 2010.
21. Juha Nurmonen. Counting modulo quantifiers on finite structures. *Inf. Comput.*, 160(1-2):62–87, 2000.
22. Amitabha Roy and Howard Straubing. Definability of languages by generalized first-order formulas over $(N, +)$. *SIAM J. Comput*, 37(2):502–521, 2007.
23. M. Ruhl. Counting and addition cannot express deterministic transitive closure. In *14th Symposium on Logic in Computer Science (LICS'99)*, pages 326–335, Washington - Brussels - Tokyo, July 1999. IEEE.
24. Nicole Schweikardt. Arithmetic, first-order logic, and counting quantifiers. *ACM Trans. Comput. Log*, 6(3):634–671, 2005.
25. Alexei P. Stolboushkin and Damian Niwinski. y = 2x vs. y = 3x. *J. Symb. Log.*, 62(2):661–672, 1997.
26. Howard Straubing. *Finite automata, formal logic, and circuit complexity.* Birkhauser Verlag, Basel, Switzerland, 1994.
27. Howard Straubing. Inexpressibility results for regular languages in nonregular settings. In *Developments in Language Theory*, pages 69–77, 2005.
28. Howard Straubing, Denis Thérien, and Wolfgang Thomas. Regular languages defined with generalized quanifiers. *Inf. Comput*, 118(2):289–301, May 1995.
29. Heribert Vollmer. *Introduction to circuit complexity.* Springer-Verlag, Berlin-Heidelberg-New York-Barcelona-Hong Kong-London-Milan-Paris-Singapur-Tokyo, 1999.