ON SECOND-ORDER MONADIC MONOIDAL AND GROUPOIDAL QUANTIFIERS [*]

JUHA KONTINEN [a] AND HERIBERT VOLLMER [b]

[a] Department of Mathematics and Statistics, University of Helsinki, P.O. Box 68, FI-00014 University of Helsinki, Finland
e-mail address: juha.kontinen@helsinki.fi

[b] Institut für Theoretische Informatik, Universität Hannover, Appelstraße 4, 30167 Hannover, Germany
e-mail address: vollmer@thi.uni-hannover.de

ABSTRACT. We study logics defined in terms of second-order monadic monoidal and groupoidal quantifiers. These are generalized quantifiers defined by monoid and groupoid word-problems, equivalently, by regular and context-free languages. We give a computational classification of the expressive power of these logics over strings with varying built-in predicates. In particular, we show that ATIME(n) can be logically characterized in terms of second-order monadic monoidal quantifiers.

1. INTRODUCTION

We study logics defined in terms of so-called second-order monadic monoidal and groupoidal quantifiers. These are generalized quantifiers defined by monoid and groupoid word-problems, equivalently, by regular and context-free languages. A *groupoid* is a finite multiplication table with an identity element. For a fixed groupoid G, each $S \subseteq G$ defines a G-word-problem, i.e., a language $\mathcal{W}(S, G)$ composed of all words w, over the alphabet G, that can be bracketed in such a way that w multiplies out to an element of S. The word-problem of a *monoid*, i.e., an associative groupoid, is defined analogously. Groupoid word-problems relate to context-free languages in the same way as monoid word-problems relate to regular languages: every such word-problem is context-free, and every context-free language is a homomorphic pre-image of a groupoid word-problem (this result is credited to Valiant in [5]).

In descriptive complexity, (first-order) monoidal quantifiers have been studied extensively in connection to the complexity class NC^1 and its sub-classes (see [4, 3, 24, 25]). However, in order to define non-regular languages in terms of monoidal quantifiers, some built-in relations, in addition to $<$, need to be assumed. It was shown already in [4] that first-order logic with unnested unary monoidal quantifiers characterizes the class of regular languages, REG, over strings without auxiliary built-in relations. This characterization of REG was generalized in [16] to allow also non-unary monoidal quantifiers, even with arbitrary nestings. In [11], the same was shown to hold for second-order monadic monoidal quantifiers:

$$\text{mon-}Q^1_{\mathrm{Mon}}\mathrm{FO} \equiv \mathrm{SOM}(\text{mon-}Q^1_{\mathrm{Mon}}) \equiv \mathrm{REG} \equiv \exists\mathrm{SOM}. \tag{1.1}$$

In (1.1), $\exists\mathrm{SOM}$ denotes existential second-order monadic logic and the logic $\text{mon-}Q^1_{\mathrm{Mon}}\mathrm{FO}$ consists of all formulas in which a monadic second-order monoidal quantifier Q^1_L is applied to an appropriate tuple of FO-formulas without further occurrences of second-order quantifiers. On the other hand, in $\mathrm{SOM}(\text{mon-}Q^1_{\mathrm{Mon}})$ arbitrary nestings of monoidal quantifiers are allowed. Here a crucial assumption is that there are no auxiliary built-in relations, besides the order, since already $\mathrm{SOM}(+)$, i.e., second-order monadic logic with built-in addition, defines exactly the languages in the linear fragment of the polynomial hierarchy [20].

We see that with monoidal quantifiers the situation is clear-cut, i.e., formulas with monadic second-order monoidal quantifiers cannot define non-regular languages. On the other hand, over strings with built-in arithmetic (i.e., built-in $+$ and $\times$) the classes in (1.1) are presumably not equal, e.g., $\exists\mathrm{SOM} \subseteq \mathrm{NP}$ and already in $\text{mon-}Q^1_{\mathrm{Mon}}\mathrm{FO}(+,\times)$ PSPACE-complete languages can be defined as we show below in Corollary 5.2.

In [5], the elaborate theory connecting monoids to the fine structure of NC^1 was generalized to groupoids and LOGCFL. It was shown in [5] that there exists a single groupoid whose word-problem is complete for LOGCFL under DLOGTIME-reductions, implying also a logical characterization for LOGCFL in terms of first-order groupoidal quantifiers. Building on this result, a systematic investigation of first-order groupoidal quantifiers was initiated in [16].

In [11] it was asked what is the relationship of the corresponding (second-order) logics if monoidal quantifiers are replaced by groupoidal quantifiers in (1.1). Here we address this question and show the following (see Corollary 3.4):

$$\text{mon-}Q^1_{\mathrm{Grp}}\mathrm{FO}(+,\times) \equiv \mathrm{SOM}(\text{mon-}Q^1_{\mathrm{Grp}}). \tag{1.2}$$

It is interesting to note that for groupoidal quantifiers we have a similar collapse result as for monoidal quantifiers, but this time assuming built-in arithmetic on the left. Note that, over ordered structures, the relations $+$ and $\times$ are definable in the logic $\mathrm{SOM}(\text{mon-}Q^1_{\mathrm{Grp}})$ (see [4] and [16]). It is an open question whether the built-in relations $+$ and $\times$ are really needed for the equivalence in (1.2) to hold.

In the literature, second-order monadic quantifiers have been studied under two slightly different semantics (for each L, quantifiers Q^1_L and $Q^\star_L$). We will show that the analogue of (1.2) for the alternative semantics $Q^\star_L$ remains valid even if we drop the built-in predicates $+$ and $\times$ from $\text{mon-}Q^\star_{\mathrm{Grp}}\mathrm{FO}(+,\times)$, i.e.,

$$\text{mon-}Q^\star_{\mathrm{Grp}}\mathrm{FO} \equiv \mathrm{SOM}(\text{mon-}Q^\star_{\mathrm{Grp}}). \tag{1.3}$$

Since the logics in (1.2) and (1.3) are all equivalent (see Corollary 3.4), it follows that the only remaining open question regarding the equivalences between logics with groupoidal

quantifiers is whether

$$\text{mon-}Q^1_{\text{Grp}}\text{FO} \equiv \text{mon-}Q^\star_{\text{Grp}}\text{FO}?$$

This question is directly concerned with the problem of pinning down the exact expressive power of the so-called finite leaf automata with context-free leaf languages (see Theorem 2.13 and Corollary 2.14).

In this paper we aim for a concise classification of the expressive power of the logics with second-order monadic monoidal and groupoidal quantifiers. We first note that the difference between the two semantics, i.e., Q^1_L and $Q^\star_L$, disappears assuming built-in arithmetic. This already simplifies the picture considerably. However, especially in the monoidal case, the expressive power of the quantifiers $Q^\star_L$ without built-in arithmetic remains open. For groupoidal quantifiers, we find that

$$\text{mon-}Q^\star_{\text{Grp}}\text{FO} \equiv \text{SOM}(\text{mon-}Q^\star_{\text{Grp}}) \equiv 2^{\text{LOGCFL}},$$

where 2^{LOGCFL} equals the class of languages whose tally version resides in LOGCFL. For monoidal quantifiers, we show that

$$\text{SOM}(\text{mon-}Q^\star_{\text{Mon}}, +, \times) \equiv \text{ATIME}(n).$$

Table 1 below contains a summary of our complexity results.

2. Preliminaries

We follow standard notation for second-order monadic logic with linear order, see, e.g., [24]. We mainly restrict our attention to *string structures*, i.e., structures of *string signatures* $\tau = \langle P_{a_1}, \ldots, P_{a_s} \rangle$, where all the predicates P_{a_i} are unary. We assume that the universe $\text{dom}(\mathcal{A})$ of each structure $\mathcal{A}$ is of the form $\{0, \ldots, n-1\}$ and that the logic's linear order symbol refers to the numerical order on $\{0, \ldots, n-1\}$. We restrict attention to structures $\mathcal{A}$ in which the interpretations $P^{\mathcal{A}}_{a_i}$ of the predicates P_{a_i} satisfy the following: $P^{\mathcal{A}}_{a_i} \cap P^{\mathcal{A}}_{a_j} = \emptyset$, for $i \neq j$, and $\cup_{1 \leq i \leq s} P^{\mathcal{A}}_{a_i} = \text{dom}(\mathcal{A})$. Such τ-structures correspond to strings over the alphabet $\{a_1, \ldots, a_s\}$ in the usual way.

An alphabet Σ is a finite set of symbols. For technical reasons to be motivated shortly, we assume that every alphabet has a built-in linear order, and, to indicate that order, we write alphabets as sequences of symbols, e.g., in the above case we write $(a_1, \ldots, a_s)$. The set of all finite Σ-strings is denoted by Σ^* and $\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$, where ϵ is the empty string. For a string w, $|w|$ denotes the length of w and $|w|_a$ the number of occurrences of the letter a in w. The concatenation of the strings w and w' is denoted by $w^\frown w'$, and a^k denotes the string $b_1 \cdots b_k$, where $b_i = a$ for $1 \leq i \leq k$. For $L \subseteq \Sigma^*$ and $e \in \Sigma$, the letter e is a *neutral letter* of L if for all $u, v \in \Sigma^*$, we have $uv \in L \iff uev \in L$. The class of languages that have a neutral letter is denoted by $\mathcal{N}$.

For a signature $\tau = \langle P_{a_1}, \ldots, P_{a_s} \rangle$, the first-order τ-formulas, FO$[\tau]$, are built from first-order variables in the usual way, using the Boolean connectives $\{\wedge, \vee, \neg\}$, the predicates P_{a_i} together with $\{=, <\}$, the constants min and max, the first-order quantifiers $\{\exists, \forall\}$, and parentheses. SOM$[\tau]$ extends FO$[\tau]$ in terms of unary second-order variables and second-order quantifiers $\{\exists, \forall\}$. (The letters SOM stand for second order monadic logic; in the literature, this logic is sometimes denoted by MSO.)

For a complexity class $\mathcal{C}$ and logics $\mathcal{L}$ and $\mathcal{L}'$, we write $\mathcal{L} \leq \mathcal{L}'$ if for every string signature τ (unless otherwise specified), and every sentence $\varphi \in \mathcal{L}[\tau]$ there is an equivalent sentence $\psi \in \mathcal{L}'[\tau]$. Analogously, we write $\mathcal{L} \leq \mathcal{C}$ if the class of languages, over any

alphabet, which can be defined in $\mathcal{L}$ is contained in $\mathcal{C}$. We write $\mathcal{L} \equiv \mathcal{L}'$ ($\mathcal{L} \equiv \mathcal{C}$) if $\mathcal{L} \leq \mathcal{L}'$ and $\mathcal{L}' \leq \mathcal{L}$ ($\mathcal{L} \leq \mathcal{C}$ and $\mathcal{C} \leq \mathcal{L}$). It is known [19] that FO is equal to the class of star-free regular languages and that SOM $\equiv$ REG, where REG is the class of regular languages (see [8, 7, 26]).

Sometimes we assume that our structures (and logics) are equipped with auxiliary built-in predicates in addition to $<$, e.g., the ternary predicates $+$ and $\times$. The predicates $+$ and $\times$ are defined as

$$\begin{aligned} +(i,j,k) &\Leftrightarrow i + j = k, \\ \times(i,j,k) &\Leftrightarrow i \times j = k. \end{aligned}$$

The predicate BIT is a further important predicate which is defined by: $\mathrm{BIT}(a,j)$ holds iff the bit with weight 2^j is 1 in the binary representation of a. The presence of built-in predicates is signalled, e.g., by the notation $\mathrm{FO}(+,\times)$ and $\mathrm{FO}(Q_L,+,\times)$. It is well known that $\mathrm{FO}(+,\times) \equiv \mathrm{FO}(\mathrm{BIT})$ (see [14]). In fact, it was shown in [10] that BIT alone can define the corresponding canonical ordering, i.e., the symbol $<$ can dropped from $\mathrm{FO}(\mathrm{BIT})$ without a loss in expressive power.

2.1. **Generalized quantifiers.** Next, we extend logics in terms of generalized quantifiers. The Lindström quantifiers of Definition 2.1 are precisely what has been referred to as "Lindström quantifiers on strings" [9]. The original more general definition [17] uses transformations to arbitrary structures, not necessarily of string signature.

Definition 2.1. Consider a language L over an alphabet $\Sigma = (a_1, a_2, \ldots, a_s)$. Such a language gives rise to a Lindström quantifier Q_L, that may be applied to any sequence of $s - 1$ formulas as follows:

Let $\overline{x}$ be a k-tuple of pairwise distinct variables. Let $\mathcal{A}$ be a structure and $\mathrm{dom}(\mathcal{A}) = \{0, 1, \ldots, n-1\}$. We assume the lexicographic ordering on $\{0, 1, \ldots, n-1\}^k$, and we write $\overline{x}^{(0)} < \overline{x}^{(1)} < \cdots < \overline{x}^{(n^k-1)}$ for the sequence of potential values taken on by $\overline{x}$. The k-ary *Lindström quantifier* Q_L binding $\overline{x}$ takes a meaning if $s - 1$ formulas, each having as free variables the variables in $\overline{x}$ (and possibly others), are available. Let $\varphi_1(\overline{x})$, $\varphi_2(\overline{x})$, $\ldots$, $\varphi_{s-1}(\overline{x})$ be these $s - 1$ formulas. Then

$$\mathcal{A} \models Q_L \overline{x} \big[\varphi_1(\overline{x}), \varphi_2(\overline{x}), \ldots, \varphi_{s-1}(\overline{x}) \big]$$

iff the word of length n^k whose ith letter, $0 \leq i \leq n^k - 1$, is

$$\begin{cases} a_1 & \text{if } \mathcal{A} \models \varphi_1(\overline{x}^{(i)}), \\ a_2 & \text{if } \mathcal{A} \models \neg\varphi_1(\overline{x}^{(i)}) \wedge \varphi_2(\overline{x}^{(i)}), \\ \quad \vdots \\ a_s & \text{if } \mathcal{A} \models \neg\varphi_1(\overline{x}^{(i)}) \wedge \neg\varphi_2(\overline{x}^{(i)}) \wedge \cdots \wedge \neg\varphi_{s-1}(\overline{x}^{(i)}), \end{cases}$$

belongs to L.

As an example, take $s = 2$ and consider $L_\exists := 0^*1(0+1)^*$; then $Q_{L_\exists}$ is the usual first-order existential quantifier. Similarly, the universal quantifier can be expressed using the language $L_\forall := 1^*$. Finally, for $p > 1$ and $L_{\mathrm{mod}\ p} = \{w \in \{0,1\}^* \mid |w|_1 \equiv 0 \ (\mathrm{mod}\ p)\}$, the quantifiers $Q_{L_{\mathrm{mod}\ p}}$ are known as modular counting quantifiers [24].

Definition 2.2. Let τ be a signature, L a language over an alphabet $\Sigma = (a_1, a_2, \ldots, a_s)$, and $\mathcal{C}$ a class of languages.

- The set of τ-formulas, $Q_L\mathrm{FO}[\tau]$, of the logic $Q_L\mathrm{FO}$ consists of all formulas of the form
$$Q_L\overline{x}\big[\varphi_1(\overline{x}), \varphi_2(\overline{x}), \ldots, \varphi_{s-1}(\overline{x})\big],$$
  where, for some k, $\overline{x}$ is a k-tuple of pairwise distinct variables, and $\varphi_i(\overline{x})$ is a $\mathrm{FO}[\tau]$-formula for $1 \leq i \leq s-1$.
- The set of τ-formulas, $\mathrm{FO}(Q_L)[\tau]$, of the logic $\mathrm{FO}(Q_L)$ is defined by extending the formula formation rules of FO by the following clause: if, for some k, $\overline{x}$ is a k-tuple of pairwise distinct variables, and $\varphi_i(\overline{x})$ is a formula for $1 \leq i \leq s-1$, then
$$Q_L\overline{x}\big[\varphi_1(\overline{x}), \varphi_2(\overline{x}), \ldots, \varphi_{s-1}(\overline{x})\big]$$
  is a formula, too.
- Define the sets of τ-formulas of the logics $Q_\mathcal{C}\mathrm{FO}$ and $\mathrm{FO}(Q_\mathcal{C})$ by
$$Q_\mathcal{C}\mathrm{FO}[\tau] \ := \ \bigcup_{L\in\mathcal{C}} Q_L\mathrm{FO}[\tau],$$
$$\mathrm{FO}(Q_\mathcal{C})[\tau] \ := \ \bigcup_{L\in\mathcal{C}} \mathrm{FO}(Q_L)[\tau].$$

In this article we are especially interested in quantifiers defined by monoid and groupoid word-problems.

Definition 2.3. A *groupoidal quantifier* (a *monoidal quantifier*) is a Lindström quantifier Q_L where L is a word-problem of some finite groupoid (monoid). The usage of groupoidal quantifiers and monoidal quantifiers in our logical language is signalled by the subscripts Grp and Mon, respectively. We define

$$Q_{\mathrm{Grp}}\mathrm{FO} := Q_\mathcal{C}\mathrm{FO} \qquad \mathrm{FO}(Q_{\mathrm{Grp}}) := \mathrm{FO}(Q_\mathcal{C})$$
$$Q_{\mathrm{Mon}}\mathrm{FO} := Q_{\mathcal{C}'}\mathrm{FO} \qquad \mathrm{FO}(Q_{\mathrm{Mon}}) := \mathrm{FO}(Q_{\mathcal{C}'}),$$

where $\mathcal{C}$ ($\mathcal{C}'$) is the class of all word-problems of finite groupoids (monoids).

Second-order Lindström quantifiers on strings were introduced in [9]. Here, we are mainly interested in those binding only set variables (i.e., unary relations), so-called *monadic quantifiers*. For each language L, we define two monadic quantifiers Q_L^1 and $Q_L^\star$ with slightly different interpretations. It turns out that the interpretation Q_L^1, which was used in [11], is natural in the context of finite leaf automata. On the other hand, the quantifier $Q_L^\star$ is the exact second-order analogue of the corresponding first-order quantifier Q_L.

Definition 2.4. Consider a language L over an alphabet $\Sigma = (a_1, a_2, \ldots, a_s)$. Let $\overline{X} = (X_1, \ldots, X_k)$ be a k-tuple of pairwise distinct unary second-order variables and let $\mathcal{A}$ be a structure with $\mathrm{dom}(\mathcal{A}) = \{0, 1, \ldots, n-1\}$. There are 2^{nk} different instances (assignments) of $\overline{X}$ over $\mathcal{A}$. We assume the following ordering on those instances: Let each instance of a single X_i be encoded by the bit string $s_0^i \cdots s_{n-1}^i$ with the meaning $s_j^i = 1 \iff j \in X_i$. Then

(1) we encode an instance of $\overline{X}$ by the bit string
$$s_0^1 s_0^2 \cdots s_0^k s_1^1 s_1^2 \cdots s_1^k \cdots s_{n-1}^1 s_{n-1}^2 \cdots s_{n-1}^k$$
  and order the instances lexicographically by their codes.
(2) we encode an instance of $\overline{X}$ by the bit string
$$s_0^1 s_1^1 \cdots s_{n-1}^1 s_0^2 s_1^2 \cdots s_{n-1}^2 \cdots s_0^k s_1^k \cdots s_{n-1}^k$$
  and order the instances lexicographically by their codes.

The *monadic second-order Lindström quantifier* Q_L^1 (respectively $Q_L^\star$) binding $\overline{X}$ takes a meaning if $s - 1$ formulas, each having free variables $\overline{X}$, are available. Let $\varphi_1(\overline{X})$, $\varphi_2(\overline{X})$, $\ldots$, $\varphi_{s-1}(\overline{X})$ be these $s - 1$ formulas. Then

$$\mathcal{A} \models Q_L^1 \overline{X}[\varphi_1(\overline{X}), \varphi_2(\overline{X}), \ldots, \varphi_{s-1}(\overline{X})]$$

iff the word of length 2^{nk} whose ith letter, $0 \leq i \leq 2^{nk} - 1$, is

$$\begin{cases} a_1 & \text{if } \mathcal{A} \models \varphi_1(\overline{X}^{(i)}), \\ a_2 & \text{if } \mathcal{A} \models \neg\varphi_1(\overline{X}^{(i)}) \wedge \varphi_2(\overline{X}^{(i)}), \\ \quad \vdots \\ a_s & \text{if } \mathcal{A} \models \neg\varphi_1(\overline{X}^{(i)}) \wedge \neg\varphi_2(\overline{X}^{(i)}) \wedge \cdots \wedge \neg\varphi_{s-1}(\overline{X}^{(i)}), \end{cases}$$

belongs to L. Above, $\overline{X}^{(0)} < \overline{X}^{(2)} < \cdots < \overline{X}^{(2^{nk}-1)}$ denotes the sequence of all instances ordered as in (1). The notation $Q_L^\star$ is used when the instances are ordered according to (2).

Again, taking as examples the languages $L_\exists$ and $L_\forall$, we obtain the usual second-order monadic existential and universal quantifiers. Note that for $L \in \{L_\exists, L_\forall\}$ the quantifiers Q_L^1 and $Q_L^\star$ are "equivalent". This is due to the fact that, for the membership in L, the order of the letters in a word does not matter.

Definition 2.5. Let τ be a signature, L a language over an alphabet $\Sigma = (a_1, a_2, \ldots, a_s)$, and $\mathcal{C}$ a class of languages.

• The set of τ-formulas, mon-Q_L^1FO$[\tau]$, of the logic mon-Q_L^1FO consists of all formulas of the form

$$Q_L^1 \overline{X}[\varphi_1(\overline{X}), \varphi_2(\overline{X}), \ldots, \varphi_{s-1}(\overline{X})], \tag{2.1}$$

where, for some k, $\overline{X}$ is a k-tuple of pairwise distinct unary second-order variables, and $\varphi_i(\overline{X})$ is a FO$[\tau]$-formula with variables $\overline{X}$, for $1 \leq i \leq s - 1$.

• The set of τ-formulas, SOM(mon-Q_L^1)$[\tau]$, of the logic SOM(mon-Q_L^1) is defined by extending the formula formation rules of SOM$[\tau]$ by the following clause: if, for some k, $\overline{X}$ is a k-tuple of pairwise distinct unary second-order variables, and $\varphi_i(\overline{X})$ is a formula for $1 \leq i \leq s - 1$, then

$$Q_L^1 \overline{X}[\varphi_1(\overline{X}), \varphi_2(\overline{X}), \ldots, \varphi_{s-1}(\overline{X})]$$

is a formula, too.

• Define the sets of τ-formulas of the logics mon-$Q_\mathcal{C}^1$FO and SOM(mon-$Q_\mathcal{C}^1$)$[\tau]$ by

$$\text{mon-}Q_\mathcal{C}^1\text{FO}[\tau] \quad := \quad \bigcup_{L \in \mathcal{C}} \text{mon-}Q_L^1\text{FO}[\tau],$$

$$\text{SOM(mon-}Q_\mathcal{C}^1)[\tau] \quad := \quad \bigcup_{L \in \mathcal{C}} \text{SOM(mon-}Q_L^1)[\tau].$$

• The logics mon-$Q_L^\star$FO, SOM(mon-$Q_L^\star$), mon-$Q_\mathcal{C}^\star$FO, and SOM(mon-$Q_\mathcal{C}^\star$) are defined analogously by replacing Q_L^1 everywhere with $Q_L^\star$.

Analogously to the first-order case (see Definition 2.3), we use the subscripts Grp and Mon to indicate that all groupoidal quantifiers or monoidal quantifiers are available in the corresponding logic, e.g., SOM(mon-Q_{Grp}^1) := SOM(mon-$Q_\mathcal{C}^1$), where $\mathcal{C}$ is the class of all word-problems of finite groupoids.

The next proposition shows that the difference between the two semantics of second-order monadic quantifiers disappears in the presence of built-in arithmetic (or if the arithmetic predicates are definable). Below, we write $\psi^{\mathcal{A}}$ for the relation defined by the formula ψ in a structure $\mathcal{A}$, i.e., if ψ has k free variables, then

$$\psi^{\mathcal{A}} = \{\overline{a} \in \text{dom}(\mathcal{A})^k \mid \mathcal{A} \models \psi(\overline{a})\}.$$

Lemma 2.6. *Let $X_1, \ldots, X_k$ be unary second-order variables. There are $\text{FO}(+, \times)$-formulas $\phi_1(x, \overline{X}), \ldots, \phi_k(x, \overline{X})$ and $\psi_1(x, \overline{X}), \ldots, \psi_k(x, \overline{X})$ such that for all $\mathcal{A}$ and $A_1, \ldots, A_k \subseteq \text{dom}(\mathcal{A}) = \{0, 1, \ldots, n-1\}$, (where A_i is encoded by $s_0^i \cdots s_{n-1}^i$ as in Definition 2.4) it holds that the encoding of $(\phi_1^{(\mathcal{A}, \overline{A})}, \ldots, \phi_k^{(\mathcal{A}, \overline{A})})$ as a bit string as in clause 1 of Definition 2.4 results with*

$$s_0^1 s_1^1 \cdots s_{n-1}^1 s_0^2 s_1^2 \cdots s_{n-1}^2 \cdots s_0^k s_1^k \cdots s_{n-1}^k, \tag{2.2}$$

and the encoding of $(\psi_1^{(\mathcal{A}, \overline{A})}, \ldots, \psi_k^{(\mathcal{A}, \overline{A})})$ as a bit string as in clause 2 of Definition 2.4 results with

$$s_0^1 s_0^2 \cdots s_0^k s_1^1 s_1^2 \cdots s_1^k \cdots s_{n-1}^1 s_{n-1}^2 \cdots s_{n-1}^k. \tag{2.3}$$

Proof. Let us show how to construct the formulas $\phi_1(x, \overline{X}) \ldots, \phi_k(x, \overline{X})$. The idea simply is that

$$\mathcal{A} \models \phi_i(j, \overline{A})$$

should hold if the bit in position $jk + i$ from the left in (2.2) is 1 if and only if $jk + i = (c-1)n + r$ and $r - 1 \in A_c$, where $0 \leq j \leq n-1$, $1 \leq i \leq k$, $1 \leq c \leq k$, and $1 \leq r \leq n$. This condition can be easily expressed in $\text{FO}(+, \times)$. The formulas $\psi_1(x, \overline{X}) \ldots, \psi_k(x, \overline{X})$ can be constructed completely analogously.

Note also that the formulas $\phi_1(x, \overline{X}), \ldots, \phi_k(x, \overline{X})$ define a permutation of k-tuples of unary relations and that $\psi_1(x, \overline{X}), \ldots, \psi_k(x, \overline{X})$ define the inverse of this permutation. $\square$

Proposition 2.7. *For any L, $\text{mon-}Q_L^1\text{FO}(+, \times) \equiv \text{mon-}Q_L^{\star}\text{FO}(+, \times)$.*

Proof. by Lemma 2.6, $Q_L^{\star}\overline{X}\big[\varphi_1(\overline{X}), \ldots, \varphi_{s-1}(\overline{X})\big]$ can be expressed as

$$Q_L^1\overline{X}\big[\varphi_1(X_1/\psi_1(\overline{X}), \ldots, X_k/\psi_k(\overline{X})), \ldots, \varphi_{s-1}(X_1/\psi_1(\overline{X}), \ldots, X_k/\psi_k(\overline{X}))\big].$$

Analogously, $Q_L^1\overline{X}\big[\varphi_1(\overline{X}), \ldots, \varphi_{s-1}(\overline{X})\big]$ can be expressed as

$$Q_L^{\star}\overline{X}\big[\varphi_1(X_1/\phi_1(\overline{X}), \ldots, X_k/\phi_k(\overline{X})), \ldots, \varphi_{s-1}(X_1/\phi_1(\overline{X}), \ldots, X_k/\phi_k(\overline{X}))\big]. \qquad \square$$

By Proposition 2.7, the two semantics of second-order quantifiers coincide for all the logics (with built-in or definable arithmetic) considered in this article.

Remark 2.8. Let L and $\mathcal{A}$ be as in Definition 2.4. It is worth noting that, for $m > 1$, the m-ary second-order quantifiers Q_L^1 and $Q_L^{\star}$ can be defined by straightforward modifications to Definition 2.4. The m-ary quantifiers binds a k-tuple $\overline{X} = (X_1, \ldots, X_k)$ (for some k) of m-ary second-order variables in $s - 1$ many formulas. Each X_i is encoded by the bit string $s_0 \cdots s_{n^m - 1}$ with the meaning $s_j = 1$ if and only if the jth tuple in the lexicographic ordering of $\{0, 1, \ldots, n-1\}^m$ is in X_i. The semantics of the m-ary quantifiers Q_L^1 and $Q_L^{\star}$ can be now defined analogously to Definition 2.4. We use the notation $Q_L^1\text{FO}$ and $Q_L^{\star}\text{FO}$ for the analogues of $\text{mon-}Q_L^1\text{FO}$ and $\text{mon-}Q_L^{\star}\text{FO}$ in which the m-ary quantifiers Q_L^1 and $Q_L^{\star}$ are allowed for $m \geq 1$.

We end this section by showing that, in the non-monadic case, the analogue of Proposition 2.7 holds without built-in arithmetic if L has a neutral letter.

Proposition 2.9. *For any $L \in \mathcal{N}$, $Q_L^\star \mathrm{FO} \equiv Q_L^1 \mathrm{FO}$.*

Proof. We may assume that L has an alphabet $\Sigma = (a_1, a_2, \ldots, a_s)$, where a_s is a neutral letter.

We will first show that $Q_L^\star \mathrm{FO} \leq Q_L^1 \mathrm{FO}$. The idea of the proof is to show that a formula $\psi \in Q_L^\star \mathrm{FO}$ can be replaced by a formula $\psi' \in Q_L^\star \mathrm{FO}$ in which only one second-order variable with higher arity is quantified. Now, in ψ', the quantifier $Q_L^\star$ can be replaced by Q_L^1 since the difference of the two semantics only appears if more than one variable is quantified. The converse inclusion follows directly from the fact that $Q_L^1 \mathrm{FO} \leq \mathrm{Leaf}^\mathrm{P}(L) \equiv Q_L^\star \mathrm{FO}$ (see Theorem 2.11).

Let $\psi \in Q_L^\star \mathrm{FO}$ be of the

$$\psi := Q_L^\star \overline{X} \big[\varphi_1(\overline{X}), \varphi_2(\overline{X}), \ldots, \varphi_{s-1}(\overline{X}) \big],$$

where $\overline{X} = (X_1, \ldots, X_k)$ is a tuple of m-ary second-order variables. It is straightforward to construct a formula $\psi' \in Q_L^\star \mathrm{FO}$

$$\psi' := Q_L^\star R_1 \big[\varphi_1'(R_1), \varphi_2'(R_1), \ldots, \varphi_{s-1}'(R_1) \big],$$

where the arity of R_1 is $m + \lfloor \log(k) \rfloor + 1$, which is equivalent to ψ over structures $\mathcal{A}$ with $|\mathrm{dom}(\mathcal{A})| \geq 2$. Let $\mathcal{A}$ be a structure such that $|\mathrm{dom}(\mathcal{A})| \geq 2$ and $A_i \subseteq \mathrm{dom}(\mathcal{A})^m$. The idea is to encode the tuple $\overline{A} = (A_1, \ldots, A_k)$ by a unique $(m + \lfloor \log(k) \rfloor + 1)$-ary relation $B_{\overline{A}}$

$$B_{\overline{A}} = \bigcup_{1 \leq i \leq k} \{ (j_1^i, \ldots, j_{\lfloor \log(k) \rfloor + 1}^i) \} \times A_i,$$

where $j_1^i \cdots j_{\lfloor \log(k) \rfloor + 1}^i$ is the length $\lfloor \log(k) \rfloor + 1$ binary representation of i. This ensures that the ordering of the tuples $\overline{A}$ (see Definition 2.4 and Remark 2.8) coincides with the ordering of the corresponding codes $B_{\overline{A}}$. Therefore, it suffices to construct the formulas $\varphi_i'(R_1)$ in such a way that, for all $A_1, \ldots, A_k \subseteq \mathrm{dom}(\mathcal{A})^m$

$$\mathcal{A} \models \varphi_i'(B_{\overline{A}}) \iff \mathcal{A} \models \varphi_i(A_1, \ldots, A_k),$$

and, if $B \neq B_{\overline{A}}$ for all $\overline{A}$, then $\mathcal{A} \not\models \varphi_i'(B)$ implying that the formulas $\varphi_i'(R_1)$ output the neutral letter when R_1 is interpreted by the relation B.

In order to ensure that ψ and ψ' are equivalent also over structures $\mathcal{A}$ for which $|\mathrm{dom}(\mathcal{A})| = 1$, we may further replace the formulas $\varphi_i'(R_1)$ by formulas $\varphi_i^*(R_1, \ldots, R_k)$, where each R_i, for $2 \leq i \leq k$, is also $(m + \lfloor \log(k) \rfloor + 1)$-ary and $\varphi_i^*(R_1, \ldots, R_k)$ has the following form

$$(|\mathrm{dom}(\mathcal{A})| = 1 \wedge \chi_i(R_1, \ldots, R_k)) \vee (|\mathrm{dom}(\mathcal{A})| > 1 \wedge \bigwedge_{2 \leq i \leq k} R_i = \emptyset \wedge \varphi_i'(R_1))),$$

where χ_i simulates the behavior of φ_i on structures with cardinality 1 (on structures with cardinality 1 the quantifiers Q_L^1 and $Q_L^\star$ are equivalent). Note that, for $\mathcal{A}$ with $|\mathrm{dom}(\mathcal{A})| \geq 2$, the formulas $\varphi_i^*(R_1, \ldots, R_k)$ output the neutral letter if $R_i^\mathcal{A} \neq \emptyset$ for some $2 \leq i \leq k$. It follows that for all $\mathcal{A}$

$$\mathcal{A} \models \psi \Leftrightarrow \mathcal{A} \models Q_L^1 \overline{R} \big[\varphi_1^*(\overline{R}), \varphi_2^*(\overline{R}), \ldots, \varphi_{s-1}^*(\overline{R}) \big].$$

For the converse, it suffices to note that a polynomial-time non-deterministic Turing machine with the leaf language L can easily evaluate sentences of $Q_L^1 \mathrm{FO}$ implying that $Q_L^1 \mathrm{FO} \leq \mathrm{Leaf}^{\mathrm{P}}(L)$ (see [9]). Therefore, by Theorem 2.11, we get that $Q_L^1 \mathrm{FO} \leq Q_L^\star \mathrm{FO}$. □

2.2. **Leaf languages.** In this section we give a brief introduction to the leaf languages approach in computational complexity.

The leaf languages approach was introduced by Bovet, Crescenzi and Silvestri in [6] and independently by Vereshchagin in [28]. In this approach the acceptance of a word input to a nondeterministic Turing machine depends only on the values printed at the leaves of the computation tree.

Let M be a nondeterministic Turing machine which halts on every computation path with some order on the nondeterministic choices. The order of the nondeterministic choices induces a left-to-right ordering of all the leaves in the computation tree of M on input x. Define leafstring$^M(x)$ to be the concatenation of the symbols printed at the leaves of the computation tree in this order. Given now a language B, the class $\mathrm{Leaf}^{\mathrm{P}}(B)$ contains those languages L for which there is a polynomial-time non-deterministic Turing machine M such that for all inputs x: $x \in L$ iff leafstring$^M(x) \in B$.

Let us look at some examples. Define $\mathrm{Maj} := \{w \in \{0,1\}^+ \mid |w|_1 > |w|_0\}$.

Example 2.10. The following leaf language classes are well known:
- $\mathrm{NP} = \mathrm{Leaf}^{\mathrm{P}}(0^*1(0+1)^*)$,
- $\mathrm{PP} = \mathrm{Leaf}^{\mathrm{P}}(\mathrm{Maj})$,
- $\mathrm{Mod_qP} = \mathrm{Leaf}^{\mathrm{P}}(L_{\mathrm{mod}\ q})$.

In [9] complexity classes defined by leaf languages were logically characterized in terms of generalized second-order quantifiers. In particular, for every language B that has a neutral letter the following was shown to hold.

Theorem 2.11 ([9]). *For any $B \in \mathcal{N}$, $\mathrm{Leaf}^{\mathrm{P}}(B) \equiv Q_B^\star \mathrm{FO}$.*

Note that, for Theorem 2.11 to hold, the quantifier $Q_B^\star$ must be allowed to bind relation variables of arbitrary arity (see Remark 2.8). Although the m-ary second-order quantifiers Q_B^1 and $Q_B^\star$ differ, in Theorem 2.11 we can equivalently use the semantics Q_B^1 instead of $Q_B^\star$ by Proposition 2.9.

Since it is known that there are regular languages B, e.g., the word-problem for the group S_5, for which $\mathrm{Leaf}^{\mathrm{P}}(B) \equiv \mathrm{PSPACE}$ [13], we conclude that for such B,

$$Q_B^\star \mathrm{FO} \equiv \mathrm{PSPACE}.$$

2.3. **Finite leaf automata.** The automata theoretic analogue of a Turing machine with a leaf language is the so-called finite leaf automaton [22].

A *finite leaf automaton* is a tuple $M = (Q, \Sigma, \delta, s, \Gamma, \beta)$ where Q is a finite set of states, Σ is an alphabet, the input alphabet, $\delta \colon Q \times \Sigma \to Q^+$ is the transition function, $s \in Q$ is the initial state, Γ is an alphabet, the leaf alphabet, and $\beta \colon Q \to \Gamma$ is a function that associates a state q with its value $\beta(q)$. The sequence $\delta(q, a)$, for $q \in Q$ and $a \in \Sigma$, contains all possible successor states of M when reading letter a while in state q, and the order of

letters in that sequence defines a total order on these successor states. This definition allows the same state to appear more than once as a successor in $\delta(q, a)$.

Let M be as above. The computation tree $T_M(w)$ of M on input w is a labeled directed rooted tree defined as follows:

- The root of $T_M(w)$ is labeled (s, w).
- Let v be a node in $T_M(w)$ labeled by (q, x), where $x \neq \epsilon$ (the empty word), $x = ay$ for $a \in \Sigma$, $y \in \Sigma^*$. Let $\delta(q, a) = q_1 q_2 \cdots q_k$. Then v has k children in $T_M(w)$, and these are labeled by $(q_1, y), (q_2, y), \ldots, (q_k, y)$ in this order.

If we look at the tree $T_M(w)$ and attach the symbol $\beta(q)$ to a leaf in this tree with label (q, ε), then leafstring$^M(w)$ is defined to be the string of symbols attached to the leaves, read from left to right in the order induced by δ.

Definition 2.12. For $A \subseteq \Gamma^*$, the class Leaf$^{\text{FA}}(A)$ consists of all languages $B \subseteq \Sigma^*$, for which there is a leaf automaton M as just defined, with input alphabet Σ and leaf alphabet Γ such that for all $w \in \Sigma^*$, $w \in B$ iff leafstring$^M(w) \in A$. If C is a class of languages then Leaf$^{\text{FA}}(C) \equiv \cup_{A \in C}$Leaf$^{\text{FA}}(A)$.

In [22] the acceptance power of leaf automata with different kinds of leaf languages was examined. It was shown that, with respect to resource-bounded leaf language classes, there is not much difference, e.g., between automata and Turing machines. On the other hand, if the leaf language class is a formal language class then the differences can be huge. In particular, it was shown that

$$\text{Leaf}^{\text{FA}}(\text{REG}) \equiv \text{REG},$$

while it is known that

$$\text{Leaf}^{\text{P}}(\text{REG}) \equiv \text{PSPACE}.$$

In [22] the power of Leaf$^{\text{FA}}(\text{CFL})$ was left as an open question. The only upper and lower bounds known at that time were CFL $\subsetneq$ Leaf$^{\text{FA}}(\text{CFL}) \subseteq \text{DSPACE}(n^2) \cap \text{DTIME}\big(2^{O(n)}\big)$. Recently it was shown by Lohrey [18] that indeed Leaf$^{\text{FA}}(\text{CFL})$ does contain a PSPACE-complete language.

In [11] the class Leaf$^{\text{FA}}(L)$ was logically characterized assuming that the language L has a neutral letter.

Theorem 2.13 ([11])**.** *For any* $L \in \mathcal{N}$, Leaf$^{\text{FA}}(L) \equiv \text{mon-}Q_L^1\text{FO}$.

Corollary 2.14. PSPACE-*complete languages can be defined in* $\text{mon-}Q_{\text{Grp}}^1\text{FO}$.

Proof. By the result of [18], there is a language $L \in \text{CFL}$ such that the class Leaf$^{\text{FA}}(L)$ contains a PSPACE-complete language. Since L reduces via a length-preserving homomorphism to some groupoid word-problem A [5], it follows that also the class Leaf$^{\text{FA}}(A) \equiv \text{mon-}Q_A^1\text{FO}$ contains a PSPACE-complete language. □

2.4. **Complexity theory.** We assume familiarity with the basic notions in formal languages and complexity theory, e.g., complexity classes such as NP, PP, PH, and PSPACE. REG and CFL refer to the regular and context-free languages. Also, LOGCFL denotes the closure of CFL under log-space reductions.

In this article AC^0, ACC^0, TC^0, NC^1, and SAC^1 refer to the classes of languages recognized by DLOGTIME-uniform families $(C_n)_{n \in \mathbb{N}}$ of polynomial-size circuits with the following kinds of gates:

AC^0: the circuit C_n may have NOT, unbounded fan-in AND and OR gates, and constant depth.

ACC^0: the circuit C_n may have NOT, unbounded fan-in AND, OR and MOD$_q$ gates, and constant depth.

TC^0: the circuit C_n may have NOT, unbounded fan-in AND, OR, and MAJORITY gates, and constant depth.

NC^1: the circuit C_n may have NOT, bounded fan-in AND and OR gates, and $O(\log(n))$ depth.

SAC^1: the circuit C_n may have input level NOT gates, bounded fan-in AND and unbounded fan-in OR gates, and $O(\log(n))$ depth.

The requirement of DLOGTIME-uniformity means that $(C_n)_{n\in\mathbb{N}}$, as a family of directed acyclic graphs, can be recognized by a deterministic Turing machine, with random access to its input, in time $O(\log(n))$ (see [29] for details). Note that, e.g., the classes AC^i and TC^i are defined analogously as above but allowing $O(\log^i(n))$ circuit-depth.

In this article we also discuss certain complexity classes defined in terms of alternating Turing machines. We denote by ATIME($t(n)$), the class of languages which can be recognized in time $t(n)$ by some alternating Turing machine.

Let M be an alternating Turing machine accepting x and denote by T the computation tree produced by M with input x. An *accepting computation subtree* S of M on input x is a subtree of T witnessing that M accepts x. The idea is that all the nodes in S must be accepting configurations, and, furthermore, S must contain the initial configuration, i.e., the root of T, all successors of universal configurations, and exactly one successor of each existential configuration.

We say that an alternating machine M is *tree-size bounded* by $t\colon \mathbb{N}\to\mathbb{N}$ if for every x accepted by M there is an accepting computation subtree of M on input x which has at most $t(|x|)$ nodes. Let now

$$\text{ASPACE-TREESIZE}(s(n), t(n))$$

denote the class of languages which can be recognized by an alternating Turing machine M which is space bounded by s and tree-size bounded by t.

The following (non-trivial) inclusions and equalities are known to hold among the classes defined above:

$$\begin{aligned}
\text{TC}^0 \subseteq \text{NC}^1 = \text{ATIME}(\log(n)) \subseteq \text{SAC}^1 \;\;&=\;\; \text{LOGCFL} \\
&=\;\; \text{ASPACE-TREESIZE}(\log(n), n^{O(1)}).
\end{aligned}$$

The last two equalities where shown by Venkateswaran [27] and Ruzzo [23], respectively.

3. Groupoidal quantifiers

In this section we consider second-order monadic groupoidal quantifiers. We show that the extension of SOM in terms of all second-order monadic groupoidal quantifiers collapses in expressive power to its fragment mon-$Q_{\text{Grp}}^\star$FO (respectively to mon-Q_{Grp}^1FO$(+, \times)$).

The following result on first-order groupoidal quantifiers will be central for our reasoning. Below, QF refers to the set of quantifier-free formulas (of suitable signature) in which the predicates $+$ and $\times$ do not appear.

Theorem 3.1 ([16]). $Q_{\text{Grp}}\text{QF} \equiv \text{FO}(Q_{\text{Grp}}) \equiv \text{FO}(Q_{\text{Grp}}, +, \times) \equiv \text{LOGCFL}$.

We shall use the following version of Theorem 3.1.

Lemma 3.2. *Let* $\tau = \{c_1, \ldots, c_s\}$, *where* $c_1, \ldots, c_s$ *are constant symbols. Then on* τ-*structures*

$$Q_{\mathrm{Grp}}\mathrm{QF} \equiv \mathrm{FO}(Q_{\mathrm{Grp}}) \equiv \mathrm{FO}(Q_{\mathrm{Grp}}, +, \times).$$

Proof. The idea of the proof is to translate $\varphi \in \mathrm{FO}(Q_{\mathrm{Grp}}, +, \times)[\tau]$ into $\varphi^* \in \mathrm{FO}(Q_{\mathrm{Grp}}, +, \times)$ of a suitable string signature using a simple encoding of τ-structures into strings. By Theorem 3.1, we may then replace φ^* by an equivalent formula $\theta \in Q_{\mathrm{Grp}}\mathrm{QF}$. Finally, we show that θ can be translated back to a formula $\theta^* \in Q_{\mathrm{Grp}}\mathrm{QF}[\tau]$ in such a way that θ^* and φ are equivalent.

Suppose that K is a class of τ-structures definable by $\varphi \in \mathrm{FO}(Q_{\mathrm{Grp}}, +, \times)$. We shall encode K as a class of strings over signature $\langle P_{S_1}, \ldots, P_{S_{2^s}} \rangle$, where $S_1, \ldots, S_{2^s}$ is some fixed enumeration of the subsets of $\{c_1, \ldots, c_s\}$. We associate every τ-structure $\mathcal{A}$ with a unique string $w_\mathcal{A}$ over the same universe in the following way. For $S \subseteq \{c_1, \ldots, c_s\}$, define

$$P_S^{w_\mathcal{A}} = \{b \mid c_i^\mathcal{A} = b \Leftrightarrow c_i \in S\}.$$

Note that the predicate $P_\emptyset$ is interpreted by the set $\{0, \ldots, n-1\} \setminus \{c_1^\mathcal{A}, \ldots, c_s^\mathcal{A}\}$ where $\{0, \ldots, n-1\}$ is the universe of $\mathcal{A}$.

Let φ^* be acquired from φ by replacing atomic subformulas of the form $c_i = t$ by $\vee_{c_i \in S} P_S(t)$ and $c_i = c_j$ by the formula $\exists y(\vee_{c_i, c_j \in S} P_S(y))$. It is now obvious how to translate atomic formulas using the predicates $+, \times$, and $<$, e.g., the formula $c_i < t$ is replaced by $\exists y(\vee_{c_i \in S} P_S(y)) \wedge y < t)$. It is easy to show using induction on the construction of $\varphi \in \mathrm{FO}(Q_{\mathrm{Grp}}, +, \times)$ that for all τ-structures $\mathcal{A}$,

$$\mathcal{A} \models \varphi \Leftrightarrow w_\mathcal{A} \models \varphi^*.$$

By Theorem 3.1 there is a sentence $\theta \in Q_{\mathrm{Grp}}\mathrm{QF}$ which is equivalent to φ^* over strings. Let θ^* be acquired from θ by replacing subformulas $P_S(t)$ by

$$(\bigwedge_{c_i \in S} c_i = t) \wedge (\bigwedge_{c_j \in \{c_1, \ldots, c_s\} \setminus S} c_j \neq t).$$

Again by induction on $\theta \in Q_{\mathrm{Grp}}\mathrm{QF}$ we get that for all τ-structures $\mathcal{A}$,

$$\mathcal{A} \models \theta^* \Leftrightarrow w_\mathcal{A} \models \theta.$$

It follows that $\theta^* \in Q_{\mathrm{Grp}}\mathrm{QF}$ defines K. $\square$

We are now ready for the main result of this section. Note that the built-in predicates $+$ and $\times$ are definable already in terms of (first-order) majority quantifiers (see [4] and [16]) and hence definable in the logics in which groupoidal quantifies are allowed to be nested.

Theorem 3.3. $\mathrm{SOM}(\text{mon-}Q_{\mathrm{Grp}}^\star) \equiv \text{mon-}Q_{\mathrm{Grp}}^\star\mathrm{FO}$.

Proof. Fix a signature $\tau = \langle P_{a_1}, \ldots, P_{a_s} \rangle$. Suppose that $\varphi \in \mathrm{SOM}(\text{mon-}Q_{\mathrm{Grp}}^\star)$ is a sentence. We will show how to construct a sentence of the logic $\text{mon-}Q_{\mathrm{Grp}}^\star\mathrm{FO}$ equivalent to φ. The idea of the proof is to represent $\varphi \in \mathrm{SOM}(\text{mon-}Q_{\mathrm{Grp}}^\star)$, and the language of signature τ defined by φ, in terms of $\varphi^* \in \mathrm{FO}(Q_{\mathrm{Grp}}, +, \times)$, and the class of σ-structures defined by φ^*, where $\sigma = \{c_1, \ldots, c_s\}$ and $c_1, \ldots, c_s$ are constant symbols. More precisely, by representing τ-structures of cardinality n by σ-structures of cardinality 2^n, we can replace second-order variables over the domain $\{0, \ldots, n-1\}$ by first-order variables ranging over $\{0, \ldots, 2^n - 1\}$ using the BIT-predicate. Then we apply Lemma 3.2 to get a formula $\theta \in Q_{\mathrm{Grp}}\mathrm{QF}$ equivalent

to φ^*. Finally, we show that θ can be translated back to a formula $\theta' \in \text{mon-}Q^\star_{\text{Grp}}\text{FO}[\tau]$ in such a way that the original formula φ and θ' are equivalent.

Denote by $\sigma = \{c_1, \ldots, c_s\}$ the signature where each c_i is a constant symbol. For a τ-structure $\mathcal{A} = \langle \{0, \ldots, n-1\}, <, P^{\mathcal{A}}_{a_1}, \ldots, P^{\mathcal{A}}_{a_s} \rangle$, let $\mathcal{A}^*$ be the following σ-structure

$$\mathcal{A}^* = \langle \{0, \ldots, 2^n - 1\}, <, +, \times, c_1^{\mathcal{A}^*}, \ldots, c_s^{\mathcal{A}^*} \rangle,$$

where $c_i^{\mathcal{A}^*}$ is the unique integer $(< 2^n)$ whose binary representation is given by $s_0 \cdots s_{n-1}$ where $s_j = 1 \iff j \in P^{\mathcal{A}}_{a_i}$.

We shall first show that there is a sentence $\varphi^* \in \text{FO}(Q_{\text{Grp}}, +, \times)[\sigma]$ such that for all τ-structures $\mathcal{A}$,

$$\mathcal{A} \models \varphi \Leftrightarrow \mathcal{A}^* \models \varphi^*. \tag{3.1}$$

The translation $\varphi \rightsquigarrow \varphi^*$ is defined inductively as follows. For φ of the form $x_i = x_j$ or $x_i < x_j$, $\varphi^* := \varphi$, and in the remaining cases (we may exclude the definable constants min, max, and the second-order existential quantifier from $\text{SOM}(\text{mon-}Q^\star_{\text{Grp}})$ since $Q^\star_{L_\exists}$ is available) the translation is defined in the following way:

$$
\begin{aligned}
P_{a_i}(x_i) &\rightsquigarrow \text{BIT}(c_i, n - (x_i + 1)) \\
Y_i(x_j) &\rightsquigarrow \text{BIT}(y_i, n - (x_j + 1)) \\
\psi \wedge \phi &\rightsquigarrow \psi^* \wedge \phi^* \\
\neg\psi &\rightsquigarrow \neg\psi^* \\
\exists x_i \psi &\rightsquigarrow \exists x_i (x_i < n \wedge \psi^*(x_i)) \\
Q^\star_L Y_1, \ldots, Y_k[\psi_1, \ldots, \psi_{s-1}] &\rightsquigarrow Q_L y_1, \ldots, y_k[\psi_1^*, \ldots, \psi_{s-1}^*]
\end{aligned}
$$

It is straightforward to show using induction on the construction of $\varphi \in \text{SOM}(\text{mon-}Q^\star_{\text{Grp}})[\tau]$, that for all τ-structures $\mathcal{A}$ and assignments s,

$$\mathcal{A} \models_s \varphi \Leftrightarrow \mathcal{A}^* \models_{s^*} \varphi^*,$$

where the assignment s^* over $\mathcal{A}^*$ is defined such that $s^*(x_i) = s(x_i)$, for all first-order variables x_i, and, for a second-order Y_i: if $s(Y_i) = A \subseteq \{0, \ldots, n-1\}$, then $s^*(y_i)$ is the unique $a < 2^n$ whose length n binary representation is given by $s_0 \cdots s_{n-1}$ where $s_j = 1 \iff j \in A$.

Above, we use the predicate BIT which is definable in $\text{FO}(+, \times)$ (see, e.g., [14]). Note also that, using the predicate BIT, the integer n can be easily defined over the structure $\mathcal{A}^*$.

By Lemma 3.2, there is a sentence

$$\theta = Q_L x_1, \ldots, x_l[\chi_1, \ldots, \chi_w],$$

where each χ_i is quantifier-free and does not contain the predicates $+$ and $\times$, equivalent to φ^*. The idea is now to translate θ into $\theta' \in \text{mon-}Q^\star_{\text{Grp}}\text{FO}$ by changing first-order variables to second-order variables. Denote by $X = Y$ the formula $\forall z (X(z) \leftrightarrow Y(z))$, and by $X < Y$ the first-order formula defining the ordering of subsets when treated as length n binary

strings. The translation $\theta \rightsquigarrow \theta'$ is now defined by

$$\begin{aligned}
t = \hat{t} &\rightsquigarrow X_t = X_{\hat{t}} \\
t < \hat{t} &\rightsquigarrow X_t < X_{\hat{t}} \\
\psi \wedge \phi &\rightsquigarrow \psi' \wedge \phi' \\
\neg\psi &\rightsquigarrow \neg\psi' \\
Q_L x_1, \ldots, x_v[\psi_1, \ldots, \psi_v] &\rightsquigarrow Q_L^\star X_1, \ldots, X_v[\psi_1', \ldots, \psi_v']
\end{aligned}$$

Above, t is either min, max, c_l, for $1 \leq l \leq s$, or a variable x, and, respectively, X_{t_i} is either $\bot$, $\top$, P_{a_l}, or X. A straightforward induction implies, in particular, that for all sentences $\psi \in Q_{\mathrm{Grp}}\mathrm{QF}[\sigma]$, and τ-structures $\mathcal{A}$

$$\mathcal{A} \models \psi' \Leftrightarrow \mathcal{A}^* \models \psi,$$

where $\mathcal{A}^*$ is defined as above. It is now immediate that θ' and the original sentence $\varphi \in \mathrm{SOM}(\mathrm{mon}\text{-}Q_{\mathrm{Grp}}^\star)[\tau]$ are equivalent. $\qquad\square$

Note that, by Proposition 2.7, we do not need to consider the semantics Q_L^1 separately. By combining Theorem 3.3 and Proposition 2.7, we get

Corollary 3.4.

$$\mathrm{SOM}(\mathrm{mon}\text{-}Q_{\mathrm{Grp}}^1) \equiv \mathrm{mon}\text{-}Q_{\mathrm{Grp}}^1\mathrm{FO}(+, \times)$$
$$\equiv \mathrm{mon}\text{-}Q_{\mathrm{Grp}}^\star\mathrm{FO} \equiv \mathrm{SOM}(\mathrm{mon}\text{-}Q_{\mathrm{Grp}}^\star).$$

We close this section by showing that a much stronger analogue of Corollary 3.4 holds. Recall that the so-called Greibach's hardest context-free language H is a nondeterministic version of the Dyck language D_2, the language of all syntactically correct sequences consisting of letters for two types of parentheses. It is known that every $L \in \mathrm{CFL}$ reduces to H under some homomorphism [12]. It was shown in [16] that the statement of Theorem 3.1 remains valid even if the logic $Q_{\mathrm{Grp}}\mathrm{QF}$ is replaced by the logic $Q_{\mathrm{pad}(H)}\mathrm{QF}$, where $\mathrm{pad}(H)$ is H extended by a neutral letter. This result directly implies the following strengthening of Corollary 3.4.

Theorem 3.5.

$$\mathrm{SOM}(\mathrm{mon}\text{-}Q_{\mathrm{Grp}}^1) \equiv \mathrm{mon}\text{-}Q_{\mathrm{pad}(H)}^1\mathrm{FO}(+, \times)$$
$$\equiv \mathrm{mon}\text{-}Q_{\mathrm{pad}(H)}^\star\mathrm{FO} \equiv \mathrm{SOM}(\mathrm{mon}\text{-}Q_{\mathrm{Grp}}^\star).$$

Proof. The proof is analogous to the proof of Theorem 3.3. It suffices to prove the last equality in the statement of the theorem. Suppose that $\varphi \in \mathrm{SOM}(\mathrm{mon}\text{-}Q_{\mathrm{Grp}}^\star)$ is a sentence. By an analogous argument as in the proof of Theorem 3.3, we first translate φ into a sentence $\varphi^* \in \mathrm{FO}(Q_{\mathrm{Grp}}, +, \times)$. Then we replace φ^* by an equivalent sentence $\theta \in Q_{\mathrm{pad}(H)}\mathrm{QF}$. Now, again by an analogous argument as in the proof of Theorem 3.3, θ can be translated back to the logic $\mathrm{mon}\text{-}Q_{\mathrm{pad}(H)}^\star\mathrm{FO}$. $\qquad\square$

4. Monoidal quantifiers

In this section we consider second-order monadic monoidal quantifiers.

As already mentioned, the following result completely characterizes the picture in the case of the semantics Q_L^1 without built-in arithmetic.

Theorem 4.1 ([11]). $\text{mon-}Q_{\text{Mon}}^1\text{FO} \equiv \text{SOM}(\text{mon-}Q_{\text{Mon}}^1) \equiv \text{REG} \equiv \exists\text{SOM}.$

Interestingly, the expressive power of monoidal quantifiers collapses to regular languages when built-in arithmetic is not present. Under reasonable complexity theoretic assumptions, the corresponding equalities between the logics in Theorem 4.1 do not hold with built-in arithmetic. Furthermore, it is an open question if the analogue of Theorem 4.1 holds with respect to the semantics $Q_L^\star$. Again, by Proposition 2.7, we however know that the semantics coincide assuming built-in arithmetic.

Theorem 4.2. *The following equivalences hold*
(1) $\text{mon-}Q_{\text{Mon}}^1\text{FO}(+, \times) \equiv \text{mon-}Q_{\text{Mon}}^\star\text{FO}(+, \times),$
(2) $\text{SOM}(\text{mon-}Q_{\text{Mon}}^1, +, \times) \equiv \text{SOM}(\text{mon-}Q_{\text{Mon}}^\star, +, \times).$

Note that also in equivalence 2 of Theorem 4.2 the arithmetic predicates (in fact + would also suffice) need to be assumed by Theorem 4.1, i.e., + and × are not definable in $\text{SOM}(\text{mon-}Q_{\text{Mon}}^1)$. It is an open question whether the equivalences of Theorem 4.2 hold without built-in arithmetic.

5. Complexity results

In this section we study the data complexity of the logics discussed in the previous sections.

We begin with a simple logical padding argument which allows us to utilize Theorem 2.11 in the context of second-order monadic quantifiers. Recall that, in the statement of Theorem 2.11, the quantifier $Q_B^\star$ is allowed to bind relation variables of arbitrary arity (see Remark 2.8). Below, we do not distinguish notationally between a string w of alphabet $(a_1, \ldots, a_s)$, and the string structure of signature $\langle P_{a_1}, \ldots, P_{a_s} \rangle$ corresponding to w.

Proposition 5.1. *Let L be a language and suppose that a language A of alphabet Σ is definable by a sentence $\varphi \in Q_L^\star\text{FO}$. Let k be the arity of the relations quantified in φ and $\sharp \notin \Sigma$. Then the language*

$$A^* = \{w^\frown\sharp^{|w|^k - |w|} \mid w \in A\}$$

is definable in $\text{FO}(\text{mon-}Q_L^\star, +, \times)$.

Proof. Let φ be of the form

$$Q_L^\star R_1, \ldots, R_t[\psi_1, \ldots, \psi_s],$$

where each of the relations R_i has arity k. Define a translation $\varphi \rightsquigarrow \varphi^*$ as follows. For φ of the form $x_i = x_j$, $x_i < x_j$, or $P_{a_i}(x_j)$, $\varphi^* := \varphi$, and in the remaining cases (we exclude the definable constants min and max) the translation is defined in the following way:

$$
\begin{aligned}
R_i(x_1, \ldots, x_k) &\rightsquigarrow \exists z(X_{R_i}(z) \wedge z = |w|^{k-1}x_1 + \cdots + |w|x_{k-1} + x_k) \\
\psi \wedge \phi &\rightsquigarrow \psi^* \wedge \phi^* \\
\neg\psi &\rightsquigarrow \neg\psi^* \\
\exists x\psi &\rightsquigarrow \exists x(x < |w| \wedge \psi^*(x)) \\
Q_L^\star R_1, \ldots, R_t[\psi_1, \ldots, \psi_{s-1}] &\rightsquigarrow Q_L^\star X_{R_1}, \ldots, X_{R_t}[\psi_1^*, \ldots, \psi_{s-1}^*]
\end{aligned}
$$

It is straightforward to show using induction on $\psi \in Q_L^\star \mathrm{FO}$ that for all w and assignments s

$$w \models_s \psi \Leftrightarrow w^\frown \sharp^{|w|^k - |w|} \models_{s^*} \psi^*,$$

where s^* agrees with s with respect to first-order variables, and

$$s^*(X_{R_i}) = \{a \mid a = |w|^{k-1} b_1 + \cdots + |w| b_{k-1} + b_k \text{ for some } (b_1, \ldots, b_k) \in s(R_i)\}.$$

Note also that, by our conventions, the universe of $w^\frown \sharp^{|w|^k - |w|}$ is $\{0, \ldots, n^k - 1\}$ hence there is 1-1 correspondence between the subsets of $\{0, \ldots, n^k - 1\}$ and the k-ary relations over the universe, $\{0, \ldots, n-1\}$, of w.

Finally, the language A^* is defined by $\varphi^* \wedge \chi$, where $\chi \in \mathrm{FO}(+, \times)$ and

$$w \models \chi \Leftrightarrow \exists \tilde{w}(w = \tilde{w}^\frown \sharp^{|\tilde{w}|^k - |\tilde{w}|}). \qquad \square$$

Proposition 5.1 shows that logics $\mathrm{FO}(\mathrm{mon}\text{-}Q_L^\star, +, \times)$ can be quite powerful. In fact, it is apparent from the proof that if, e.g., $\varphi \in Q_L^\star \mathrm{FO}$ in the proof of Proposition 5.1 defines a PSPACE-complete language, then the language defined by $\varphi^* \in \mathrm{mon}\text{-}Q_L^\star \mathrm{FO}(+, \times)$ is also PSPACE-complete.

Corollary 5.2. *In the logic* $\mathrm{mon}\text{-}Q_{\mathrm{Mon}}^1 \mathrm{FO}(+, \times)$, PSPACE-*complete languages can be defined.*

Proof. This follows, e.g., by the fact that

$$Q_B^\star \mathrm{FO} \equiv \mathrm{PSPACE},$$

where B is the word-problem for the group S_5 (see Section 2.2), and by Proposition 2.7. $\square$

Recall that in the case of groupoidal quantifiers, already in $\mathrm{mon}\text{-}Q_{\mathrm{Grp}}^1 \mathrm{FO}$ PSPACE-complete languages can be defined by Corollary 2.14.

Next we show that the logics $\mathrm{SOM}(\mathrm{mon}\text{-}Q_{\mathrm{Mon}}^\star, +, \times)$ and $\mathrm{SOM}(\mathrm{mon}\text{-}Q_{\mathrm{Grp}}^\star)$ capture the exponential versions of the language classes captured by the logics $\mathrm{FO}(Q_{\mathrm{Mon}}, +, \times)$ and $\mathrm{FO}(Q_{\mathrm{Grp}}, +, \times)$. As already noted in Theorem 3.1, the logic $\mathrm{FO}(Q_{\mathrm{Grp}}, +, \times)$ corresponds to LOGCFL [16]. On the other hand, in [4] it was show that

$$\mathrm{FO}(Q_{\mathrm{Mon}}, +, \times) \equiv \mathrm{NC}^1 = \mathrm{ATIME}(\log(n)). \tag{5.1}$$

Definition 5.3. For $n \in \mathbb{N}$, denote by $\mathrm{bin}(n)$ the binary representation of n without leading zeros. Let $L \subseteq \{0, 1\}^+$ and $1L = \{1w \mid w \in L\}$. Define now $\mathrm{tally}(L)$ as

$$\mathrm{tally}(L) = \{1^n \mid \mathrm{bin}(n) \in 1L\}.$$

Let us now define the classes of languages $2^{\mathrm{ATIME}(\log(n))}$ and 2^{LOGCFL} by

$$
\begin{aligned}
2^{\mathrm{ATIME}(\log(n))} &= \{L \subseteq \{0, 1\}^+ \mid \mathrm{tally}(L) \in \mathrm{ATIME}(\log(n))\}, \\
2^{\mathrm{LOGCFL}} &= \{L \subseteq \{0, 1\}^+ \mid \mathrm{tally}(L) \in \mathrm{LOGCFL}\}.
\end{aligned}
$$

The following is easily seen to hold:

Proposition 5.4. *The following equalities hold*
(1) $2^{\mathrm{ATIME}(\log(n))} = \mathrm{ATIME}(n)$,
(2) $2^{\mathrm{LOGCFL}} = \mathrm{ASPACE\text{-}TREESIZE}(n, 2^{O(n)})$.

Proof. The first equality is obvious and the second follows from Ruzzo's characterization of LOGCFL:

$$\text{LOGCFL} = \text{ASPACE-TREESIZE}(\log(n), n^{O(1)})$$

(see [23] and [29]).   □

Remark 5.5. By the above, we immediately get that

$$\text{NSPACE}(n) \subseteq 2^{\text{LOGCFL}}.$$

It is also straightforward to show that 2^{LOGCFL} includes the languages that can be recognized in linear time on a Threshold Turing machine (introduced in [21]).

The main result of this section can be now stated as follows:

Theorem 5.6. *The following equivalences hold*
(1) $\text{SOM}(\text{mon-}Q^\star_{\text{Mon}}, +, \times) \equiv \text{ATIME}(n)$,
(2) $\text{SOM}(\text{mon-}Q^\star_{\text{Grp}}) \equiv 2^{\text{LOGCFL}}$.

Proof. We will prove equivalence 2. Equivalence 1 is proved analogously using the fact that $\text{ATIME}(\log(n)) \equiv \text{FO}(Q_{\text{Mon}}, +, \times)$ (see (5.1)).

We will show that, for all $L \subseteq \{0, 1\}^+$, L is definable in $\text{SOM}(\text{mon-}Q^\star_{\text{Grp}})$ iff $\text{tally}(L) \in \text{LOGCFL}$. Since $\text{LOGCFL} \equiv \text{FO}(Q_{\text{Grp}}, +, \times)$, it suffices to show that for all $L \subseteq \{0, 1\}^+$, L is definable in $\text{SOM}(\text{mon-}Q^\star_{\text{Grp}})$ iff $\text{tally}(L)$ is definable in $\text{FO}(Q_{\text{Grp}}, +, \times)$.

We will first show that if L is definable in $\text{SOM}(\text{mon-}Q^\star_{\text{Grp}})$, then $\text{tally}(L)$ can be defined in $\text{FO}(Q_{\text{Grp}}, +, \times)$. The idea is now to translate formulas between string structures

$$w = \langle \{0, \ldots, m-1\}, <, P_1, P_0 \rangle, \tag{5.2}$$

and

$$1^n = \langle \{0, \ldots, n-1\}, P_1, <, +, \times \rangle, \tag{5.3}$$

where $w \in \{1, 0\}^+$, $\text{bin}(n) = 1w$, and $P_1 = \{0, \ldots, n-1\}$. Some technical difficulties arise here, which were not encountered in the proof of Theorem 3.3, due to the fact that the sizes of the universes of w and 1^n are not necessarily exactly of the form l and 2^l for some l.

We define a translation $\varphi \rightsquigarrow \varphi^*$ of $\varphi \in \text{SOM}(\text{mon-}Q^\star_{\text{Grp}})$ into $\varphi^* \in \text{FO}(Q_{\text{Grp}}, +, \times)$ below. Analogously to the proof of Theorem 3.3, it can be shown using induction on $\varphi \in \text{SOM}(\text{mon-}Q^\star_{\text{Grp}})$, that for all w and assignments s,

$$w \models_s \varphi \Leftrightarrow 1^n \models_{s^*} \varphi^*.$$

The assignment s^* is defined so that it agrees with s on first-order variables x_i, and, for a variable y_i, corresponding to a second-order variable Y_i, $s^*(y_i) = a < 2^m$, where $a = \Sigma_{i=0}^{m-1} s_i 2^{m-1-i}$ and $s_i = 1$ iff $i \in s(Y_i)$.

The translation $\varphi \rightsquigarrow \varphi^*$ is defined inductively as follows. For φ of the form $x_i = x_j$ or $x_i < x_j$, $\varphi^* := \varphi$, and in the remaining cases (again, we exclude the definable constants min, max, and the second-order existential quantifier from $\text{SOM}(\text{mon-}Q^\star_{\text{Grp}})$ since $Q^\star_{L_\exists}$ is available) the translation is defined in the following way (recall that by (5.2) and (5.3) we

have $\lfloor \log(n) \rfloor = m = |w|$):

$$
\begin{aligned}
P_1(x_i) &\rightsquigarrow \mathrm{BIT}(n, \lfloor \log(n) \rfloor - (x_i + 1)) \\
P_0(x_i) &\rightsquigarrow \neg \mathrm{BIT}(n, \lfloor \log(n) \rfloor - (x_i + 1)) \\
Y_i(x_j) &\rightsquigarrow \mathrm{BIT}(y_i, \lfloor \log(n) \rfloor - (x_j + 1)) \\
\psi \wedge \phi &\rightsquigarrow \psi^* \wedge \phi^* \\
\neg \psi &\rightsquigarrow \neg \psi^* \\
\exists x_i \psi &\rightsquigarrow \exists x_i(x_i < \lfloor \log(n) \rfloor \wedge \psi^*(x_i)) \\
Q_L^\star Y_1, \ldots, Y_k[\psi_1, \ldots, \psi_{s-1}] &\rightsquigarrow Q_L y_1, \ldots, y_k[\chi \wedge \psi_1^*, \ldots, \chi \wedge \psi_{s-1}^*]
\end{aligned}
$$

Note that, e.g., the formula $\mathrm{BIT}(n, \lfloor \log(n) \rfloor - (x_i + 1))$ above can be easily constructed even though the integer n is not in the universe of the structure 1^n. Without loss of generality, we may assume that the letter a_s in the alphabet $(a_1, a_2, \ldots, a_s)$ of L is a neutral letter. Now the formula χ, used to translate $Q_L^\star$, ensures that the interpretation $b_1, \ldots, b_k \in \{0, \ldots, n-1\}$ of the tuple $y_1, \ldots, y_k$ does correspond to some tuple of unary relations $B_1, \ldots, B_k \subseteq \{0, \ldots, m-1\}$. The problem is that there can be more tuples $\overline{b}$ than tuples $\overline{B}$. The formula χ is defined as

$$
\bigwedge_{1 \leq i \leq k} y_i < 2^{\lfloor \log(n) \rfloor} - 1.
$$

Now if χ is not satisfied by $\overline{b}$, then none of the formulas $\chi \wedge \psi_i^*$ will be satisfied and hence these formulas produce the neutral letter a_s when $\overline{y}$ is interpreted as $\overline{b}$.

We conclude that, by the above, if L is defined by a sentence $\varphi \in \mathrm{SOM}(\mathrm{mon}\text{-}Q_{\mathrm{Grp}}^\star)$, then the sentence

$$
(\varphi^* \wedge \forall x P_1(x)) \in \mathrm{FO}(Q_{\mathrm{Grp}}, +, \times),
$$

defines $\mathrm{tally}(L)$.

We will next define a formula translation $\varphi \rightsquigarrow \varphi'$ mapping $\varphi \in \mathrm{FO}(Q_{\mathrm{Grp}}, +, \times)$ into $\varphi' \in \mathrm{SOM}(\mathrm{mon}\text{-}Q_{\mathrm{Grp}}^\star)$. Again, an analogous induction on the construction of φ shows, in particular, that for all sentences φ and all $n \in \mathbb{N}$,

$$
\mathrm{bin}(n) \models \varphi' \Leftrightarrow 1^n \models \varphi.
$$

The translation $\varphi \rightsquigarrow \varphi'$ is defined by replacing first-order variables by unary second-order variables:

$$
\begin{aligned}
x = y &\rightsquigarrow X = Y \\
x < y &\rightsquigarrow X < Y \\
x + y = z &\rightsquigarrow X + Y = Z \\
x \times y = z &\rightsquigarrow X \times Y = Z \\
P_1(x) &\rightsquigarrow \top \\
\psi \wedge \phi &\rightsquigarrow \psi' \wedge \phi' \\
\neg \psi &\rightsquigarrow \neg \psi' \\
\exists x \psi &\rightsquigarrow \exists X(\delta \wedge \psi') \\
Q_L x_1, \ldots, x_v[\psi_1, \ldots, \psi_v] &\rightsquigarrow Q_L^\star X_1, \ldots, X_v[\delta \wedge \psi_1', \ldots, \delta \wedge \psi_v']
\end{aligned}
$$

Above, $X = Y$ denotes the formula $\forall z(X(z) \leftrightarrow Y(z))$. Also $X < Y$, $X + Y = Z$, and $X \times Y = Z$ are formulas defining the ordering, addition, and multiplication of unary

relations, when treated as binary strings. Finally, the formula δ is simply
$$\delta = X < P_1,$$
and it has an analogous role here as χ had above. Again, we may assume that a_s is a neutral letter of L when translating the quantifier Q_L.

By the above, it holds that for all $L \subseteq \{0,1\}^+$: if $\text{tally}(L) \in \text{LOGCFL}$, then $1L$ is definable in $\text{SOM}(\text{mon-}Q^\star_{\text{Grp}})$. In order to complete the proof, it suffices to show that L is definable in $\text{SOM}(\text{mon-}Q^\star_{\text{Grp}})$ iff $1L$ is definable in $\text{SOM}(\text{mon-}Q^\star_{\text{Grp}})$. Note that on the computational side, L and $1L$ are easily definable from each other. On the logical side, it follows from the fact that $\text{SOM}(\text{mon-}Q^\star_{\text{Grp}})$ is closed under logical reductions for which the target structure w^* has size linear in w. The idea is that if $|w^*| = k|w|$ then subsets of w^* can be encoded by k subsets over w and, hence, such a formula translation can be defined in terms of second-order monadic quantifiers. This implies[1], in particular, that for any sentence $\varphi \in \text{SOM}(\text{mon-}Q^\star_{\text{Grp}})$ we can construct a sentence $\varphi^* \in \text{SOM}(\text{mon-}Q^\star_{\text{Grp}})$ which holds over w iff $1w \models \varphi$. We do not give the proof here but refer to the proof of Corollary 8.6 in [15] in which an analogous result is proved for the extension of FO in terms of the second-order monadic majority quantifier. $\qquad \square$

Finally, we turn to the case of symmetric (commutative) languages. It is obvious that, for a symmetric language L, the quantifiers Q^1_L and Q^*_L are equivalent. Let CFL^s and REG^s denote the classes of symmetric context-free and regular languages, respectively.

Denote by Mod-LinH the linear analogue of the class Mod-PH. Recall that Mod-PH is the oracle hierarchy, analogous to the polynomial hierarchy PH, in which the building block of the hierarchy is $\text{NP} \cup (\text{Mod}_q\text{P})_{q>1}$ and the $(k+1)th$ level is defined by allowing access to oracles from the Boolean closure of the kth level. Similarly, we denote by Lin-CH the linear analogue of the counting hierarchy CH, which is the oracle hierarchy with PP as the building block.

The expressive power of second-order monadic quantifiers defined by symmetric regular and context-free languages can be characterized as follows:

Theorem 5.7. *The following equivalences hold*
(1) $\text{SOM}(\text{mon-}Q^\star_{\text{REG}^s}, +, \times) \equiv \text{Mod-LinH}$,
(2) $\text{SOM}(\text{mon-}Q^\star_{\text{CFL}^s}) \equiv \text{Lin-CH}$.

Proof. Let us first show equivalence 2. Note that by Parikh's theorem on context-free languages, every symmetric context-free language is already in TC^0 and, by [4], $\text{TC}^0 \equiv \text{FO}(Q_{\text{Maj}}, +, \times)$. Therefore, we get that
$$\text{SOM}(\text{mon-}Q^\star_{\text{CFL}^s}) \equiv \text{SOM}(\text{mon-}Q^\star_{\text{Maj}})$$
by an analogous argument as in Theorem 3.3. In [15] it was shown that
$$\text{SOM}(\text{mon-}Q^\star_{\text{Maj}}) \equiv \text{Lin-CH},$$
hence the claim follows.

For equivalence 1, note that $\text{FO}(Q_{\text{REG}^s}, +, \times) \equiv \text{ACC}^0$ (see [4]) and that, analogously to AC^0 and PH, ACC^0 is the logarithmic analogue of Mod-PH (see [2, 1]). Hence, by

---

[1] More generally, it also implies that $\text{SOM}(\text{mon-}Q^\star_{\text{Grp}})$ captures 2^{LOGCFL} over all string signatures, since a string w of any signature can be encoded in binary with length $O(|w|)$.

standard padding we get that $2^{\mathrm{ACC}^0} = \mathrm{Mod\text{-}LinH}$ and, mimicking the proof of Theorem 5.6, it follows that

$$\mathrm{SOM}(\mathrm{mon}\text{-}Q^\star_{\mathrm{REG}^s}, +, \times) \equiv \mathrm{Mod\text{-}LinH}. \qquad \square$$

We conclude this section by the following table summarising the results on the data-complexity of the logics studied in this paper.

Table 1: Summary of the results

Logic & built-ins	$\{\leq\}$	$\{+, \times\}$	Result
$\mathrm{mon}\text{-}Q^1_{\mathrm{Mon}}\mathrm{FO}$	REG	PSPACE-comp.	Thm 4.1 [11], Cor 5.2
$\mathrm{mon}\text{-}Q^\star_{\mathrm{Mon}}\mathrm{FO}$	$\geq$ REG	PSPACE-comp.	[7, 26], Cor 5.2
$\mathrm{SOM}(\mathrm{mon}\text{-}Q^1_{\mathrm{Mon}})$	REG	$\mathrm{ATIME}(n)$	Thm 4.1 [11], Thm 5.6
$\mathrm{SOM}(\mathrm{mon}\text{-}Q^\star_{\mathrm{Mon}})$	$\geq$ REG	$\mathrm{ATIME}(n)$	[7, 26], Thm 5.6
$\mathrm{mon}\text{-}Q^1_{\mathrm{Grp}}\mathrm{FO}$	PSPACE-comp.	2^{LOGCFL}	Cor 2.14 [18], Thm 5.6
$\mathrm{mon}\text{-}Q^\star_{\mathrm{Grp}}\mathrm{FO}$	2^{LOGCFL}	2^{LOGCFL}	Thm 5.6
$\mathrm{SOM}(\mathrm{mon}\text{-}Q^1_{\mathrm{Grp}})$	2^{LOGCFL}	2^{LOGCFL}	Thm 5.6
$\mathrm{SOM}(\mathrm{mon}\text{-}Q^\star_{\mathrm{Grp}})$	2^{LOGCFL}	2^{LOGCFL}	Thm 5.6
$\mathrm{SOM}(\mathrm{mon}\text{-}Q^\star_{\mathrm{REG}^s})$	REG	Mod-LinH	Thm 4.1 [11], Thm 5.7
$\mathrm{SOM}(\mathrm{mon}\text{-}Q^\star_{\mathrm{CFL}^s})$	Lin-CH	Lin-CH	Thm 5.7

6. CONCLUSION

We conclude with two questions for further study. The main open question regarding groupoidal quantifiers is to determine whether the two variants of semantics for second-order groupoidal quantifiers coincide in the most restricted case studied in this paper, i.e., is it the case that

$$\mathrm{mon}\text{-}Q^1_{\mathrm{Grp}}\mathrm{FO} \equiv \mathrm{mon}\text{-}Q^\star_{\mathrm{Grp}}\mathrm{FO}?$$

A positive answer would imply that

$$\mathrm{Leaf}^{\mathrm{FA}}(\mathrm{CFL}) = \mathrm{ASPACE\text{-}TREESIZE}(n, 2^{O(n)}).$$

This would strengthen the recent PSPACE-hardness result [18] considerably (showing that $\mathrm{Leaf}^{\mathrm{FA}}(\mathrm{CFL})$ contains PSPACE-complete problems, and answering the open question from [11]).

The second open question concerns the expressive power of the quantifiers $Q^\star_L$, for a regular L. It is an open question whether non-regular languages can be defined in $\mathrm{SOM}(\mathrm{mon}\text{-}Q^\star_{\mathrm{Mon}})$.

REFERENCES

[1] E. Allender. The permanent requires large uniform threshold circuits. *Chicago Journal of Theoretical Computer Science*, 1999.

[2] E. Allender and V. Gore. A uniform circuit lower bound for the permanent. *SIAM Journal on Computing*, 23:1026–49, 1994.

[3] D. A. M. Barrington, K. Compton, H. Straubing, and D. Thérien. Regular languages in NC^1. *Journal of Computer and System Sciences*, 44:478–499, 1992.

[4] D. A. M. Barrington, N. Immerman, and H. Straubing. On uniformity within NC^1. *Journal of Computer and System Sciences*, 41:274–306, 1990.

[5] F. Bédard, F. Lemieux, and P. McKenzie. Extensions to Barrington's M-program model. *Theoretical Computer Science*, 107:31–61, 1993.

[6] D. P. Bovet, P. Crescenzi, and R. Silvestri. A uniform approach to define complexity classes. *Theoretical Computer Science*, 104:263–283, 1992.

[7] J. R. Büchi. On a decision method in restricted second-order arithmetic. In *Proceedings Logic, Methodology and Philosophy of Sciences 1960*, Stanford, CA, 1962. Stanford University Press.

[8] J. R. Büchi and C. C. Elgot. Decision problems of weak second order arithmetics and finite automata, Part I. *Notices of the American Mathematical Society*, 5:834, 1958.

[9] H.-J. Burtschick and H. Vollmer. Lindström quantifiers and leaf language definability. *International Journal of Foundations of Computer Science*, 9:277–294, 1998.

[10] A. Dawar, K. Doets, S. Lindell, and S. Weinstein. Elementary properties of the finite ranks. *MLQ Math. Log. Q.*, 44(3):349–353, 1998.

[11] M. Galota and H. Vollmer. A generalization of the Büchi-Elgot-Trakhtenbrot theorem. In *Computer science logic (Paris, 2001)*, volume 2142 of *Lecture Notes in Comput. Sci.*, pages 355–368. Springer, Berlin, 2001.

[12] S. Greibach. The hardest context-free language. *SIAM Journal on Computing*, 2:304–310, 1973.

[13] U. Hertrampf, C. Lautemann, T. Schwentick, H. Vollmer, and K. W. Wagner. On the power of polynomial time bit-reductions. In *Proceedings 8th Structure in Complexity Theory*, pages 200–207, 1993.

[14] N. Immerman. *Descriptive Complexity*. Graduate Texts in Computer Science. Springer Verlag, New York, 1999.

[15] J. Kontinen and H. Niemistö. Extensions of MSO and the monadic counting hierarchy. *Information and Computation (to appear)*. Manuscript available at http://www.helsinki.fi/~jkontine/.

[16] C. Lautemann, P. McKenzie, T. Schwentick, and H. Vollmer. The descriptive complexity approach to LOGCFL. *Journal of Computer and Systems Sciences*, 62(4):629–652, 2001.

[17] P. Lindström. First order predicate logic with generalized quantifiers. *Theoria*, 32:186–195, 1966.

[18] M. Lohrey. Leaf languages and string compression. In R. Hariharan, M. Mukund, and V. Vinay, editors, *FSTTCS 2008*, volume 08004 of *Dagstuhl Seminar Proceedings*. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2008.

[19] R. McNaughton and S. Papert. *Counter-Free Automata*. MIT Press, 1971.

[20] M. More and F. Olive. Rudimentary languages and second-order logic. *Mathematical Logic Quarterly*, 43(3):419–426, 1997.

[21] I. Parberry and G. Schnitger. Parallel computation with threshold functions. *Journal of Computer and System Sciences*, 36:287–302, 1988.

[22] T. Peichl and H. Vollmer. Finite automata with generalized acceptance criteria. *Discrete Mathematics and Theoretical Computer Science*, 4:179–192, 2001.

[23] W. L. Ruzzo. Tree-size bounded alternation. *Journal of Computer and System Sciences*, 21:218–235, 1980.

[24] H. Straubing. *Finite Automata, Formal Logic, and Circuit Complexity*. Birkhäuser, Boston, 1994.

[25] H. Straubing, D. Thérien, and W. Thomas. Regular languages defined with generalized quantifiers. *Information and Computation*, 118:289–301, 1995.

[26] B. A. Trakhtenbrot. Finite automata and logic of monadic predicates. *Doklady Akademii Nauk SSSR*, 140:326–329, 1961. In Russian.

[27] H. Venkateswaran. Properties that characterize LOGCFL. *Journal of Computer and System Sciences*, 43:380–404, 1991.

[28] N. K. Vereshchagin. Relativizable and non-relativizable theorems in the polynomial theory of algorithms. *Izvestija Rossijskoj Akademii Nauk*, 57:51–90, 1993. In Russian.

[29] H. Vollmer. *Introduction to Circuit Complexity – A Uniform Approach*. Texts in Theoretical Computer Science. Springer Verlag, Berlin Heidelberg, 1999.