On Second-Order Monadic Groupoidal Quantifiers*

Juha Kontinen[1] and Heribert Vollmer[2]

[1] Department of Mathematics and Statistics, University of Helsinki, P.O. Box 68, FI-00014 University of Helsinki, Finland

[2] Institut für Theoretische Informatik, Universität Hannover, Appelstraße 4, 30167 Hannover, Germany

Abstract. We study logics defined in terms of so-called second-order monadic groupoidal quantifiers. These are generalized quantifiers defined by groupoid word-problems or equivalently by context-free languages. We show that, over strings with built-in arithmetic, the extension of monadic second-order logic by all second-order monadic groupoidal quantifiers collapses to its fragment mon-Q^1_{Grp}FO. We also show a variant of this collapse which holds without built-in arithmetic. Finally, we relate these results to an open question regarding the expressive power of finite leaf automata with context-free leaf languages.

1 Introduction

We study logics defined in terms of so-called second-order monadic groupoidal quantifiers. These are generalized quantifiers defined by groupoid word-problems or equivalently by 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. 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 [2]).

Monoidal quantifiers are generalized quantifiers defined by monoid word-problems or equivalently by regular languages. It was known [1] that first-order logic with unnested unary monoidal quantifiers characterizes the class of regular languages. In [6] this was extended to show the following

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

In (1), $\exists$SOM stands for existential second-order monadic logic and REG denotes the class of regular languages. The class mon-Q^1_{Mon}FO is the class of all languages

* The first author was partially supported by grant 106300 of the Academy of Finland and the Vilho, Yrjö and Kalle Väisälä Foundation. The second author was partially supported by DFG grant VO 630/6-1.

W. Hodges and R. de Queiroz (Eds.): WoLLIC 2008, LNAI 5110, pp. 238–248, 2008.

describable by applying a specific monadic second-order monoidal quantifier Q_L to an appropriate tuple of formulas without further occurrences of second-order quantifiers. On the other hand, in FO(mon-Q^1_{Mon}) arbitrary nestings of monoidal quantifiers is allowed, analogously to SOM(mon-Q^1_{Mon}) in which the base logic is second-order monadic logic.

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. Note that over strings with built-in arithmetic the classes in (1) are presumably not equal, e.g., $\exists$SOM $\subseteq$ NP and already in FO(mon-Q^1_{Mon}) PSPACE-complete languages can be defined by a similar argument as in Proposition 1. Similarly, the equivalences in (1) do not hold if non-monadic quantifiers are also allowed (under some reasonable complexity-theoretic assumptions).

In [6] it was asked what is the relationship of the corresponding logics if monoidal quantifiers are replaced by groupoidal quantifiers. In this paper we address this question and show the following:

$$\text{mon-}Q^1_{\text{Grp}}\text{FO}(+,\times) = \text{FO}(\text{mon-}Q^1_{\text{Grp}}) = \text{SOM}(\text{mon-}Q^1_{\text{Grp}},+,\times)\ . \qquad (2)$$

It is interesting to note that in the case of groupoidal quantifiers the collapse of the logics happens in the presence of built-in arithmetic.

In Sect. 4 we consider groupoidal quantifiers with a slight change in their semantics (notation $Q^\star_L$). We show that the analogue of (2) also holds in this case. It turns out that (2) remains valid even if we drop the built-in predicates $+$ and $\times$ from mon-$Q^\star_{\text{Grp}}$FO$(+,\times)$. Finally, we relate these results to an open question regarding the expressive power of finite leaf automata with context-free leaf languages.

2 Generalized Quantifiers

We follow standard notation for monadic second-order logic with linear order, see, e.g., [14]. We mainly restrict our attention to *string signatures*, i.e., signatures of the form $\langle P_{a_1},\ldots,P_{a_s}\rangle$, where all the predicates P_{a_i} are unary, and in every structure $\mathcal{A}$, $\mathcal{A} \models P_{a_i}(j)$ iff the jth symbol in the input is the letter a_i. Such structures are thus words over the alphabet $\{a_1,\ldots,a_s\}$. We assume that the universe of each structure $\mathcal{A}$ is of the form $\{0,\ldots,n-1\}$ and that the logic's linear order symbol refers to numerical order on $\{0,\ldots,n-1\}$. For technical reasons to be motivated shortly, we also assume that every alphabet has a built-in linear order, and we write alphabets as sequences of symbols to indicate that order, e.g., in the above case we write $(a_1,\ldots,a_s)$.

Our basic formulas are built from first- and second-order variables in the usual way, using the Boolean connectives $\{\wedge,\vee,\neg\}$, the relevant predicates P_{a_i} together with $\{=,<\}$, the constants min and max, the first- and second-order quantifiers $\{\exists,\forall\}$, and parentheses.

SOM is the class of all languages definable using formulas as just described. (The letters SOM stand for second order monadic logic; in the literature, this logic is sometimes denoted by MSO.) FO is the subclass of SOM restricted to languages definable by first-order formulas. It is known [12] that FO is equal to the class of star-free regular languages and that SOM equals the class REG of regular languages (see [4,3,15]). Sometimes we assume that our structures are also equipped with the built-in predicates $+$ and $\times$. This assumption is signalled, e.g., by the notation FO$(+, \times)$.

Next, we extend logics in terms of generalized quantifiers. The Lindström quantifiers of Def. 1 are precisely what has been referred to as "Lindström quantifiers on strings" [5]. The original more general definition [11] uses transformations to arbitrary structures, not necessarily of string signature.

Definition 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 variables. We assume the lexical ordering on $\{0, 1, \ldots, n-1\}^k$, and we write $\overline{x}^{(1)} < \overline{x}^{(2)} < \cdots < \overline{x}^{(n^k)}$ 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 $Q_L\overline{x}\big[\varphi_1(\overline{x}), \varphi_2(\overline{x}), \ldots, \varphi_{s-1}(\overline{x})\big]$ holds on a string $w = w_1 \cdots w_n$, iff the word of length n^k whose ith letter, $1 \leq i \leq n^k$, is*

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

belongs to L.

As an example, take $s = 2$ and consider $L_\exists =_{\text{def}} 0^*1(0+1)^*$; then $Q_{L_\exists}$ is the usual first-order existential quantifier. Similarly, the universal quantifier can be expressed using the language $L_\forall =_{\text{def}} 1^*$. The quantifiers $Q_{L_{\text{mod } p}}$ for $p > 1$ are known as modular counting quantifiers [14].

In this paper we are especially interested in quantifiers defined by groupoid word problems. The following definition is due to Bédard, Lemieux, and McKenzie [2]:

Definition 2. *A* groupoidal quantifier *is a Lindström quantifier Q_L where L is a word-problem of some finite groupoid.*

Usage of groupoidal quantifiers in our logical language is signalled by Q_{Grp}. The class Q_{Grp}FO is the class of all languages definable by applying a single groupoidal quantifier to an appropriate tuple of FO formulas. The class FO(Q_{Grp}) is defined analogously, but allowing groupoidal quantifiers to be used as any other quantifier would (i.e., allowing arbitrary nesting).

Second-order Lindström quantifiers on strings were introduced in [5]. Here, we are mainly interested in those binding only set variables, so called *monadic quantifiers*. For each language L we define two monadic quantifiers Q_L and $Q_L^\star$

with slightly different interpretations. It turns out that the interpretation Q_L, which was used in [6], is natural in the context of finite 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 3. *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 unary second-order variables, i.e., set variables. There are 2^{nk} different instances (assignments) of $\overline{X}$. We assume the following ordering on those instances: Let each instance of a single X_i be encoded by a bit string $s_0^i \cdots s_{n-1}^i$ with the meaning $s_j^i = 1 \iff j \in X_i$. Then*

i) 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.

ii) 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 *(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 $\varphi = Q_L\overline{X}\big[\varphi_1(\overline{X}), \varphi_2(\overline{X}), \ldots, \varphi_{s-1}(\overline{X})\big]$ holds on a string $w = w_1 \cdots w_n$, iff the word of length 2^{nk} whose ith letter, $1 \le i \le 2^{nk}$, is*

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

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

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

The class mon-Q_L^1FO is the class of all languages describable by applying a specific monadic second-order groupoidal quantifier Q_L to an appropriate tuple of formulas without further occurrences of second-order quantifiers. The class mon-Q_{Grp}^1FO is defined analogously using arbitrary monadic second-order groupoidal quantifiers. The class SOM(mon-Q_{Grp}^1) is defined analogously, but allowing groupoidal quantifiers to be used as any other quantifier would (i. e., allowing arbitrary nesting). Analogous notations are used for the quantifiers $Q_L^\star$.

3 Groupoidal Quantifiers Q_L

In this section we consider second-order monadic groupoidal quantifiers under the semantics Q_L. We show that the extension of SOM in terms of all second-order monadic groupoidal quantifiers collapses to its fragment mon-Q^1_{Grp}FO over strings with built-in arithmetic.

The following result on first-order groupoidal quantifiers will be central for our argumentation. Below, QFree denotes the set of quantifier-free formulas in which the predicates $+$ and $\times$ do not appear.

Theorem 1 ([10]). $Q_{\mathrm{Grp}}\mathrm{QFree} = \mathrm{FO}(Q_{\mathrm{Grp}}) = \mathrm{FO}(Q_{\mathrm{Grp}}+,\times) = \mathrm{LOGCFL}$ *over string signatures.*

We shall use the following version of Theorem 1.

Lemma 1. *Let $\tau = \{<, c_1, \ldots, c_s\}$, where $c_1, \ldots, c_s$ are constant symbols. Then on τ-structures*

$$Q_{\mathrm{Grp}}\mathrm{QFree} = \mathrm{FO}(Q_{\mathrm{Grp}}) = \mathrm{FO}(Q_{\mathrm{Grp}}+,\times) \ .$$

Proof. The idea is to encode τ-structures into strings and then apply Theorem 1. In order to encode the information about the identities among $c_1, \ldots, c_s$, we introduce a predicate symbol P_A for each non-empty $A \subseteq \{c_1, \ldots, c_s\}$. To simplify notation, let us assume that $\tau = \{<, c_1, c_2\}$. The general case is analogous.

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 $\langle P_{\{c_1\}}, P_{\{c_2\}}, P_{\{c_1,c_2\}}, P^* \rangle$. The predicate $P_{\{c_1,c_2\}}$ is used when the interpretations of c_1 and c_2 coincide and P^* is interpreted by all the elements different from c_1 and c_2. Denote by $\mathcal{A}'$ the string encoding a τ-structure $\mathcal{A}$. Let φ^* be acquired from φ by replacing atomic subformulas of the form $c_i = d$ by $P_{\{c_i\}}(d) \vee P_{\{c_1,c_2\}}(d)$ and $c_1 = c_2$ by the formula $\exists x P_{\{c_1,c_2\}}(x)$. It is now obvious how to translate atomic formulas using the predicates $+, \times$, and $<$, e.g., $c_i < x$ is replaced by $\exists y((P_{\{c_i\}}(y) \vee P_{\{c_1,c_2\}}(y)) \wedge y < x)$. It is easy to verify that for all $\mathcal{A}$, $\mathcal{A} \models \varphi \Leftrightarrow \mathcal{A}' \models \varphi^*$. By Theorem 1 there is a sentence $\theta \in Q_{\mathrm{Grp}}\mathrm{QFree}$ which is equivalent to φ^* over strings. Let θ^* be acquired from θ by the following substitutions: $P_{\{c_i\}}(d)$ is replaced by $c_i = d \wedge c_1 \neq c_2$, $P_{\{c_1,c_2\}}(d)$ by $c_1 = d \wedge c_1 = c_2$, and finally $P^*(d)$ by $c_1 \neq d \wedge c_2 \neq d$. Now $\theta^* \in Q_{\mathrm{Grp}}\mathrm{QFree}$ and θ^* defines K.

We are now ready for the main result of this section.

Theorem 2. mon-$Q^1_{\mathrm{Grp}}\mathrm{FO}(+,\times) = \mathrm{FO}(\text{mon-}Q^1_{\mathrm{Grp}}) = \mathrm{SOM}(\text{mon-}Q^1_{\mathrm{Grp}},+,\times)$ *over strings.*

Proof. Fix a signature $\tau = \langle P_{a_1}, \ldots, P_{a_s} \rangle$. Suppose that B is a language defined by some sentence $\varphi \in \mathrm{SOM}(\text{mon-}Q^1_{\mathrm{Grp}},+,\times)[\tau]$. We may assume that $\varphi \in \mathrm{FO}(\text{mon-}Q^1_{\mathrm{Grp}})[\tau]$ since the second-order existential quantifier is included in mon-Q^1_{Grp} and already the extension of FO by the quantifier corresponding to the (context-free) language *majority* can define the predicates $+$ and $\times$ on ordered structures [9].

Denote by $\sigma = \{<, +, \times, c_1, \ldots, c_s\}$ the signature where each c_i is a constant symbol. For a τ-structure $\mathcal{A} = \langle \{0, \ldots, n-1\}, <, P_{a_1}^{\mathcal{A}}, \ldots, P_{a_s}^{\mathcal{A}} \rangle$, 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 length n binary representation corresponds to $P_i^{\mathcal{A}}$.

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

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

We define φ^* via the following transformation:

$$\begin{aligned}
x_1 = x_2 &\rightsquigarrow x_1 = x_2 \\
x_1 < x_2 &\rightsquigarrow x_1 < x_2 \\
P_{a_i}(z) &\rightsquigarrow \mathrm{BIT}(c_i, z) \\
Y(x) &\rightsquigarrow \mathrm{BIT}(y, x) \\
\psi \wedge \phi &\rightsquigarrow \psi^* \wedge \phi^* \\
\neg\psi &\rightsquigarrow \neg\psi^* \\
\exists x \psi &\rightsquigarrow \exists x (x < n \wedge \psi^*(x)) \\
Q_L X_1, \ldots, X_k[\psi_1, \ldots, \psi_{s-1}] &\rightsquigarrow Q_{L'} x_1, \ldots, x_k[\psi_1^*, \ldots, \psi_{s-1}^*]
\end{aligned}$$

Each assignment f over $\mathcal{A}$ is associated with the assignment f^* over $\mathcal{A}^*$ such that if $f(X) = A \subseteq \{0, \ldots, n-1\}$ then $f^*(x)$ is the unique $a < 2^n$ whose binary representation is given by $s_0 \cdots s_{n-1}$ where $s_j = 1 \iff j \in A$. The predicate BIT, which is $\mathrm{FO}(+, \times)$-definable, allows us to recover the set A from the number a. In other words, $\mathrm{BIT}(a, j)$ holds if bit $n - j - 1$ in the binary representation of a is 1 iff $j \in A$. The language L' is defined by

$$L' = \{w \mid s(w) \in L\},$$

where s is defined as follows: s maps a word w to w if $|w| \neq 2^{km}$ for all $m \in \mathbb{N}^*$. Assuming $|w| = 2^{km}$, the position i of each letter in w is determined by a binary string of length km:

$$P_{bin}(i) = r_1^1 \cdots r_m^1 \cdots r_1^k \cdots r_m^k \ .$$

Now, s takes w to the unique string whose ith letter is identical with the letter in position $r_1^1 r_1^2 \cdots r_1^k r_2^1 r_2^2 \cdots r_2^k \cdots r_m^1 r_m^2 \cdots r_m^k$ in w. In other words, s corrects the asymmetry in the semantics of first-order and second-order quantifiers. It is easy to verify that the language L' is $\mathrm{FO}(+, \times)$ reducible to L and thus also definable in $\mathrm{FO}(Q_{\mathrm{Grp}}, +, \times)$. Therefore, the logic $\mathrm{FO}(Q_{\mathrm{Grp}}, +, \times)$ is also closed under the quantifier $Q_{L'}$.

By Lemma 1, 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 to the logic mon-Q^1_LFO$(+, \times)$ by changing first-order variables to second-order variables. We shall construct formulas $\delta_i(\overline{X})$ such that for all τ-structures $\mathcal{A}$

$$\mathcal{A} \models Q_L X_1, \ldots, X_l(\delta_1(\overline{X}), \ldots, \delta_w(\overline{X})) \Leftrightarrow \mathcal{A}^* \models \theta \ .$$

The formula $\delta_i(\overline{X})$ should be satisfied by $A_1, \ldots, A_l \subseteq \{0, \ldots, n-1\}$ iff χ_i is satisfied by the tuple $(a_1, \ldots, a_l) \in \{0, \ldots, 2^n - 1\}^l$ corresponding to $A_1, \ldots, A_l$. Again we need to correct the asymmetry caused by the difference in the semantics of first-order and second-order quantifiers. As in Definition 3, each A_i determines the string $s^i_0 \cdots s^i_{n-1}$ with the meaning $s^i_j = 1 \iff j \in A_i$. The tuple $\overline{A}$ is now encoded by the string

$$s^1_0 s^2_0 \cdots s^l_0 s^1_1 s^2_1 \cdots s^l_1 \cdots s^1_{n-1} s^2_{n-1} \cdots s^l_{n-1} \ . \tag{3}$$

Therefore, $\overline{A}$ should satisfy $\delta_i(\overline{X})$ iff the tuple $a^*_1, \ldots, a^*_l$ satisfies χ_i, where the concatenation of the length n binary representations of $a^*_1, \ldots, a^*_l$ correspond to the string in (3). In other words, the binary representation of a^*_i is given by $\mathrm{BIT}(a^*_i, j) = 1$ iff the $(n(i-1)+j)$th bit from the right is 1 in (3) iff $c \in A_r$ for the unique r and c for which $n(i-1)+j = cl+r-1$. Since χ_i is quantifier-free and contains only atomic formulas such as $x_1 < c_2$ or $x_l = x_k$, we can construct the formulas $\delta_i(\overline{X})$ using the fact that the binary representations of $a^*_1, \ldots, a^*_l$ can be recovered from $\overline{A}$ in a first-order way with the help of arithmetic. By the above, it is clear that the sentence $Q_L X_1, \ldots, X_l(\delta_1(\overline{X}), \ldots, \delta_w(\overline{X}))$ now defines B.

4 Groupoidal Quantifiers $Q^\star_L$

In [5] the expressive power of generalized second-order quantifiers was characterized in terms complexity classes given by so-called leaf languages. In particular, for every language B that has a neutral letter, i.e., a letter $\epsilon \in \Gamma$ such that, for all $u, v \in \Gamma^*$, we have $uv \in B \iff u\epsilon v \in B$, the following was shown to hold. Let $\mathcal{N}$ be the class of languages that have a neutral letter.

Theorem 3 ([5]). *For any $B \in \mathcal{N}$, $\mathrm{Leaf}^{\mathrm{P}}(B) = Q^\star_B\mathrm{FO}$.*

Above, $\mathrm{Leaf}^{\mathrm{P}}(B)$ denotes the class of languages defined in polynomial-time in terms of non-deterministic Turing machines using the leaf language B and $Q^\star_B$FO denotes the class of all languages describable by applying the quantifier $Q^\star_B$ to an appropriate tuple of first-order formulas (the quantifier $Q^\star_B$ is allowed to bind relation variables of arbitrary arity). Note that in this context we could equivalently use the semantics Q_L instead of $Q^\star_L$. This is due to the fact that the difference between Q_L and $Q^\star_L$ only appears if more than one second-order variable is quantified and this can be avoided by joining relations into a single relation of higher arity.

Since it is known that there are regular languages B, e.g., the word problem for the group S_5, for which $\text{Leaf}^{\text{P}}(B) = \text{PSPACE}$ [8], we conclude that for such B,

$$Q_B^{\star}\text{FO} = \text{PSPACE} \ . \tag{4}$$

By a simple padding argument, we see that already first-order logic with a monadic second-order quantifier $Q_B^{\star}$ is sufficient to define a PSPACE-complete language.

Proposition 1. *Let L be a language and suppose that a language A is definable by a sentence $\varphi \in Q_L^{\star}\text{FO}$. Let k be the maximum of the arities of the relations quantified in φ. Then the language*

$$A^* = \{w^\frown 0^{|w|^k - |w|} \mid w \in A\}$$

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

Proof. The proof using standard techniques will appear in the journal version of the paper.

Proposition 1 shows that logics $\text{FO}(\text{mon-}Q_L^{\star}, +, \times)$ can be quite powerful. In this section we show that a result analogous to Theorem 2 also holds with respect to the semantics $Q_L^{\star}$. We also show that in the most general case, i.e., when the logic in question is the extension of SOM by all second-order monadic groupoidal quantifiers, both semantics turn out to be equal in expressive power.

Theorem 4. $\text{mon-}Q_{\text{Grp}}^{\star}\text{FO} = \text{SOM}(\text{mon-}Q_{\text{Grp}}^{\star}, +, \times) = \text{SOM}(\text{mon-}Q_{\text{Grp}}^{1}, +, \times)$ *over strings.*

Proof. Let us first note that by an analogous argument as in the proof of Theorem 2 any sentence $\varphi \in \text{SOM}(\text{mon-}Q_{\text{Grp}}^{\star}, +, \times)$ can be first translated into $\text{FO}(Q_{\text{Grp}}, +, \times)$ and then to the logic $\text{mon-}Q_{\text{Grp}}^{1}\text{FO}(+, \times)$. In fact, the first translation can be even simplified since the quantifier $Q_{L'}$ is not needed. Therefore, it suffices to show the converse inclusion.

Let A be defined by a sentence $\varphi \in \text{SOM}(\text{mon-}Q_{\text{Grp}}^{1}, +, \times)$. We use the same argument as in the proof of Theorem 2. We only need to modify the last part of the proof and define the translation from a sentence θ of the form

$$Q_L x_1, \ldots, x_k(\chi_1, \ldots, \chi_v),$$

where each χ_i is quantifier-free and does not contain the predicates $+$ and $\times$. We do this in the following way. 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 transformation is now defined by

$$\begin{aligned} x = y &\rightsquigarrow X = Y \\ x < y &\rightsquigarrow X < Y \\ \psi \wedge \phi &\rightsquigarrow \psi' \wedge \phi' \\ \neg\psi &\rightsquigarrow \neg\psi' \\ Q_L x_1, \ldots, x_v[\psi_1, \ldots, \psi_v] &\rightsquigarrow Q_L^{\star} X_1, \ldots, X_v[\psi_1', \ldots, \psi_v'] \end{aligned}$$

The use of $Q_L^{\star}$ allows us to define the translation simply by changing first-order variables to second-order variables. It is easy to verify that θ' now defines the language A.

By combining Theorems 2 and 4, we get

Corollary 1. $\mathrm{SOM}(\text{mon-}Q^1_{\mathrm{Grp}}, +, \times) = \text{mon-}Q^1_{\mathrm{Grp}}\mathrm{FO}(+, \times) = \text{mon-}Q^{\star}_{\mathrm{Grp}}\mathrm{FO} = \mathrm{SOM}(\text{mon-}Q^{\star}_{\mathrm{Grp}}, +, \times)$.

5 Connection to Leaf Automata

A *finite leaf automaton* is a tuple $M = (Q, \Sigma, \delta, s, \Gamma, \beta)$ where Q is the 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 $\mathrm{leafstring}^M(w)$ is defined to be the string of symbols attached to the leaves, read from left to right in the order induced by δ.

Definition 4. *For $A \subseteq \Gamma^*$, the class* $\mathrm{Leaf}^{\mathrm{FA}}(A)$ *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* $\mathrm{leafstring}^M(w) \in A$. *If C is a class of languages then* $\mathrm{Leaf}^{\mathrm{FA}}(C) = \cup_{A \in C} \mathrm{Leaf}^{\mathrm{FA}}(A)$.

In [13] 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 in [13] that $\mathrm{Leaf}^{\mathrm{FA}}(\mathrm{REG}) = \mathrm{REG}$ while it is known that $\mathrm{Leaf}^{\mathrm{P}}(\mathrm{REG}) = \mathrm{PSPACE}$. In [13] the power of $\mathrm{Leaf}^{\mathrm{FA}}(\mathrm{CFL})$ was left as an open question. The only upper and lower bounds known are $\mathrm{CFL} \subsetneq \mathrm{Leaf}^{\mathrm{FA}}(\mathrm{CFL}) \subseteq \mathrm{DSPACE}(n^2) \cap \mathrm{DTIME}\big(2^{O(n)}\big)$.

In [6] the class $\mathrm{Leaf}^{\mathrm{FA}}(L)$ was logically characterized assuming that the language L has a neutral letter.

Theorem 5 ([6]). *For any $L \in \mathcal{N}$,* $\mathrm{Leaf}^{\mathrm{FA}}(L) = \text{mon-}Q^1_L\mathrm{FO}$.

We would like to use either Theorem 2 or Theorem 4 to show that the class $\text{Leaf}^{\text{FA}}(\text{CFL})$ contains PSPACE-complete languages. Unfortunately, Theorem 2 does not apply because it assumes built-in arithmetic which is not allowed in Theorem 5. On the other hand, due to the change in the interpretation of quantifiers in Theorem 4, it is not clear that Theorem 5 holds in this case.

Recall that Greibach's hardest context-free language H is a so-called 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 $L \in \text{CFL}$ reduces to H under some homomorphism [7]. It was shown in [10] that in Theorem 1 the logic $Q_{\text{Grp}}\text{QFree}$ can be even replaced by $Q_{\text{pad}(H)}\text{QFree}$, where $\text{pad}(H)$ is H extended by a neutral symbol. Therefore, we can similarly replace the logics mon-$Q^1_{\text{Grp}}\text{FO}$ and mon-$Q^\star_{\text{Grp}}\text{FO}$, in Theorems 2 and 4, by mon-$Q^1_{\text{pad}(H)}\text{FO}(+,\times)$ and mon-$Q^\star_{\text{pad}(H)}\text{FO}$, respectively.

We call a language symmetric if it is closed under permuting the letters of words. Note that if $\text{pad}(H)$ happened to be symmetric, then we could use the proof of Theorem 4 to show that mon-$Q^1_{\text{pad}(H)}\text{FO}$ = mon-$Q^1_{\text{pad}(H)}\text{FO}(+,\times)$. However, this assumption turns out not to be true, since a symmetric context-free language cannot be complete for all of CFL under homomorphims. It can be even shown that symmetric context-free languages are contained in TC^0.

6 Conclusion

In this paper we have studied several monadic second-order logics with groupoidal quantifiers. Our collapse results partially address an open question in [6]. However, the main open question of that paper remains: What is the power of finite leaf-automata with context-free leaf languages? If one could prove equality between the two variants of semantics for second-order quantifiers, i.e.,

$$\text{mon-}Q^1_{\text{Grp}}\text{FO} = \text{mon-}Q^\star_{\text{Grp}}\text{FO},$$

then it follows immediately from our results that such simple automata can even accept PSPACE-complete problems.

References

1. Barrington, D.A.M., Immerman, N., Straubing, H.: On uniformity within NC^1. Journal of Computer and System Sciences 41, 274–306 (1990)
2. Bédard, F., Lemieux, F., McKenzie, P.: Extensions to Barrington's M-program model. Theoretical Computer Science 107, 31–61 (1993)
3. Büchi, J.R.: On a decision method in restricted second-order arithmetic. In: Proceedings Logic, Methodology and Philosophy of Sciences 1960, Stanford University Press, Stanford (1962)
4. Büchi, J.R., Elgot, C.C.: Decision problems of weak second order arithmetics and finite automata, Part I. Notices of the American Mathematical Society 5, 834 (1958)
5. Burtschick, H.J., Vollmer, H.: Lindström quantifiers and leaf language definability. International Journal of Foundations of Computer Science 9, 277–294 (1998)

6. Galota, M., Vollmer, H.: A generalization of the Büchi-Elgot-Trakhtenbrot theorem. In: Fribourg, L. (ed.) CSL 2001 and EACSL 2001. LNCS, vol. 2142, pp. 355–368. Springer, Heidelberg (2001)
7. Greibach, S.: The hardest context-free language. SIAM Journal on Computing 2, 304–310 (1973)
8. Hertrampf, U., Lautemann, C., Schwentick, T., Vollmer, H., Wagner, K.W.: On the power of polynomial time bit-reductions. In: Proceedings 8th Structure in Complexity Theory, pp. 200–207 (1993)
9. Kontinen, J., Niemistö, H.: Extensions of MSO and the monadic counting hierarchy (2006), http://www.helsinki.fi/~jkontine/
10. Lautemann, C., McKenzie, P., Schwentick, T., Vollmer, H.: The descriptive complexity approach to LOGCFL. Journal of Computer and Systems Sciences 62, 629–652 (2001)
11. Lindström, P.: First order predicate logic with generalized quantifiers. Theoria 32, 186–195 (1966)
12. McNaughton, R., Papert, S.: Counter-Free Automata. MIT Press, Cambridge (1971)
13. Peichl, T., Vollmer, H.: Finite automata with generalized acceptance criteria. Discrete Mathematics and Theoretical Computer Science 4, 179–192 (2001)
14. Straubing, H.: Finite Automata, Formal Logic, and Circuit Complexity. Birkhäuser, Boston (1994)
15. Trakhtenbrot, B.A.: Finite automata and logic of monadic predicates (in Russian). Doklady Akademii Nauk SSSR 140, 326–329 (1961)