Quantitative Monadic Second-Order Logic

Stephan Kreutzer
Technical Universtiy Berlin
School of Elect. Eng. and Computer Science
stephan.kreutzer@tu-berlin.de

Cristian Riveros
The University of Oxford
Department of Computer Science
cristian.riveros@cs.ox.ac.uk

***Abstract*—While monadic second-order logic is a prominent logic for specifying languages of finite words, it lacks the power to compute quantitative properties, e.g. to count. An automata model capable of computing such properties are weighted automata, but logics equivalent to these automata have only recently emerged.**

We propose a new framework for adding quantitative properties to logics specifying Boolean properties of words. We use this to define *Quantitative Monadic Second-Order Logic* (QMSO). In this way we obtain a simple logic which is equally expressive to weighted automata. We analyse its evaluation complexity, both data and combined complexity, and show completeness results for combined complexity.

We further refine the analysis of this logic and obtain fragments that characterise exactly subclasses of weighted automata defined by the level of ambiguity allowed in the automata. In this way, we define a quantitative logic which has good decidability properties while being resonably expressive and enjoying a simple syntactical definition.

I. Introduction

Using logics as specification languages for properties of finite and infinite words or trees has a long history in computer science. Of particular importance in this context is *Monadic Second-Order Logic (MSO)*, the extension of first-order logic by quantification over sets of positions in the input word (see e.g. [23], [10]). The prominence of MSO as logic over words has many causes: It is a very expressive and yet simple logic in which many properties can be expressed very naturally. In fact, Büchi's classical theorem [6] states that a language is recognisable by a finite state automaton if, and only if, it is definable in MSO. Hence, MSO can define precisely the regular languages and provides an elegant specification mechanism for regular properties. Furthermore, the proof of the theorem is algorithmic which implies that MSO formulas can effectively be compiled into finite automata which can then be run in linear time on any input word. Finally, MSO has very good decidability properties and standard problems such as satisfiability and therefore equivalence and containment of formulas are decidable over finite words.

While MSO is an elegant and highly successful mechanism for specifying word languages, there are many applications where we are not only interested in accepting or rejecting a word but in computing quantitative properties of words. Consider, for instance, an application where a server provides a number of different services which can be requested by clients. Let Γ be the set of services provided. A word over Γ is then a sequence of client requests. In this context, different services may incur different costs and we may be interested in computing the total cost of a word $w \in \Gamma^*$. Quantitative properties of this form cannot be expressed in MSO or by finite state automata.

Several authors have studied automata models that allow to compute quantitative properties of words or trees. An automata model that is capable of computing such properties is the model of weighted automata (WA), essentially going back to Schützenberger [21]. Weighted automata are an extension of classical finite state automata by weights. The main idea is that every transition of the automaton is mapped to an element of a fixed semiring such as the semiring of natural numbers. A run of the automaton on a word is then mapped to the product of its transitions and the value of an automaton on a word is the sum over all runs. By choosing a suitable semiring this allows to compute, for example, the minimal costs of a run over a word or similar quantitative properties of word languages as in the example above.

Weighted automata have found numerous applications in computer science such as in text and speech processing, model-checking, image processing and many other areas (see e.g. [12]). However, a logic corresponding to weighted automata in the same way as MSO corresponds to finite state automata has only recently been defined by Droste and Gastin [11] with the introduction of *weighted monadic second-order logic*. In full generality, this logic is more expressive than weighted automata and, therefore, they define a restricted version of it by disallowing universal second-order quantification and restricting universal first-order quantification to formulas which define "recognisable step functions", a semantic property of formal power series.

While this logic provides a specification language for all properties computable by weighted automata, it suffers from its rather complicated definition, where valid formulas are not defined by a grammar but checking whether a formula belongs to this logic requires some analysis of its behaviour. Even though it is decidable whether a formula defines a recognisable step function, this makes parsing and writing formulas much more complicated than for plain MSO.

Another problem of weighted MSO and, also, of weighted automata is its lack of *decidability* properties. While weighted automata provide a powerful and expressive model for computing quantitative properties of words, they are equally well-known for undecidability of many elementary problems. For instance, equivalence or containment of weighted automata is undecidable for many semirings. Hence, any logic effectively equivalent to weighted automata immediately inherits these properties. As equivalence and containment tests are important problems (arising for instance in query optimisation) it is interesting to define a logic for which these problems become decidable.

In this paper, we propose a different form of weighted or quantitative logic. The motivation for doing so is two-fold: The first goal is to define a logic equivalent to weighted automata which has a simple and purely syntactical definition. The second goal is to refine this logic so that a) the above problems become decidable but in which b) we can still express interesting quantitative properties in a natural way.

Our contributions. To achieve our goals we propose a generic framework for adding quantitative properties to any logic capable of expressing Boolean properties of words. A crucial feature of our approach, distinguishing it fundamentally from the weighted logics in [11], is that we maintain a clear syntactical distinction between quantifiers and operators defining Boolean properties of words and those which evaluate into elements of the semiring.

We use this framework to define *Quantitative Monadic Second-Order Logic*, a highly expressive logic for the specification of quantitative properties of words. More precisely, let $(\mathbb{S}, \oplus, \odot, \mathbb{0}, \mathbb{1})$ be a semiring and let Γ be an alphabet. The formulas of *Quantitative MSO over* $\mathbb{S}$ *and* Γ ($\mathrm{QMSO}[\mathbb{S}, \Gamma]$) are defined by the following grammar.

$$\theta := \varphi \mid s \mid (\theta \oplus \theta) \mid (\theta \odot \theta) \mid \Sigma x.\, \theta \mid \Pi x.\, \theta \mid \Sigma X.\, \theta$$

where $\varphi \in \mathrm{MSO}[\leq, (P_a)_{a \in \Gamma}]$, $s \in \mathbb{S}$, x and y are first-order variables and X is a set variable. We refer to Π, Σ as semiring quantification to distinguish it from the Boolean quantification inside the MSO-formulas φ. Hence, in QMSO the semiring quantification is added explicitly on top of MSO which will be useful in our analysis below. In Section III, we define the semantics of $\mathrm{QMSO}[\mathbb{S}, \Gamma]$ formally.

In the same way that unrestricted weighted MSO is too expressive, QMSO is also more expressive than weighted automata. It turns out that the nesting of product-quantification is a problem if we want to remain within weighted automata. In Section IV, we therefore study the fragment of QMSO, which we call *Quantitative Iteration Logic* (QIL), where universal semiring quantification can only be applied to formulas without any semiring quantifiers. We show that QIL captures exactly weighted automata. Here the fact that we distinguish between semiring and Boolean quantification allows us to keep full MSO as part of the logic and we can therefore still express naturally all regular properties of words. We claim that with QIL we satisfy our first goal of defining a logic for weighted automata that has a simple syntactical definition and in which we can naturally specify quantitative properties of words.

Having defined QIL, we study its evaluation complexity in relation to counting complexity classes and we show that QIL has FP data complexity whereas its combined complexity is FPSPACE(poly)-complete. As FPSPACE(poly) is strictly contained in FPSPACE, QIL-evaluation is strictly below FPSPACE (compared to MSO-evaluation which is in PSPACE-complete).

As explained above, any logic effectively equivalent to weighted automata must have analogous undecidable problems. To find a logic for which these problems become decidable, we study the fragments of QIL obtained by excluding some operators of the semiring level and we relate them with classes of weighted automata.

The *ambiguity* of a non-deterministic weighted automaton is the maximal number of different accepting runs an automaton can take on any input word (with respect to the length of the word). Restricting the level of ambiguity yields a natural hierarchy of subclasses of weighted automata: *(co-)deterministic* WA*, unambiguous* WA*, finitely ambiguous* WA and *polynomially ambiguous* WA. These classes have all been studied in the literature and, for some semirings, it has been shown to form a strict hierarchy [17].

It turns out that every class in this hierarchy corresponds exactly to a natural fragment of QIL obtained by excluding some of the operators $\odot, \oplus, \Pi, \Sigma$ of the semiring level (see Section V for details). We take the fact that the obvious and natural fragments of QIL correspond exactly to natural fragments of WA as further evidence supporting our concept of quantitative logics.

Finitely ambiguous weighted automata form a sub-

class of WA with very good closure properties. As a consequence of this classification of fragments of QIL by classes of weighted automata we obtain a simple fragment of QIL which corresponds to finitely ambiguous WA and for which the containment and equivalence problem are therefore decidable. This fragment is obtained by disallowing nested products and sums and is a very good candidate for achieving our second goal, a quantitative logic with good decidability properties.

Having found such a logic, we aim at increasing its expressive power while preserving decidability. In Section VI, we study the fragments where we allow a bounded number of nested products and sums. We again obtain a characterisation of these fragments by weighted automata but this time we have to use non-standard models. We believe that these fragments are promising logics combining good decidability properties with reasonable expressive power which deserve further investigation.

A. Related work

As mentioned before, QMSO follows the line of *Weighted Logic (WL)* over words that has extensively been studied in [11], [4], [13], [12]. Our logic has a different syntax with respect to WL and we believe that it is simpler as a specification language. In particular, we make an explicit syntactical distinction between semiring and Boolean quantification. This allows us to establish a very strong connection between QMSO and weighted automata as outlined above (see Section V for details). The importance of the division between the boolean and semiring level was mentioned implicitly in [4] and highlighted as useful in [13]. However, both papers use this division as a technical tool and do not make this distinction explicit in the syntax as proposed in this paper. As far as we are aware, this is the first paper to make full use of this distinction.

Logics defining quantitative properties over structures have also been studied in different contexts before, for instance in [9], [8], [3], [7]. In [9], [8], *cost monadic logic* has been introduced to define quantitative functions over structures. This logic is restricted to functions over natural numbers and its semantics is designed to study "boundedness properties". Extensions of Linear Temporal Logic have been studied in [3], [7]. The logic proposed in [3] is to reason about accumulative values over *Quantitative Kripke structures*. In particular, the logic is not allowed to (naturally) calculate quantitative properties of the structure itself.

Non-standard models of weighted automata to capture unrestricted WL were studied in [5], [14]. We consider these models in Section VI to show the connection between different fragments of QMSO. In [5] and [14], no connection between the logic and deterministic automata was established. Furthermore, no connection between the features of the model and the operators in the logic was studied.

II. Preliminaries

In this section, we summarise the notation and definitions used for MSO-logic and weighted automata.

MSO. Let Γ be a finite alphabet. The syntax of MSO over Γ is given by:

$$\varphi := P_a(x) \mid x \leq y \mid x \in X \mid (\varphi \vee \varphi) \mid \neg\varphi \mid \exists x.\varphi \mid \exists X.\varphi$$

where $a \in \Gamma$, x and y are first-order variables and X is a set variable. As usual, we also allow universal quantification $\forall X, \forall x$ that can be obtained from $\exists X, \exists x$ and $\neg$ as well as the propositional operators $\wedge$, $\rightarrow$, and $\leftrightarrow$ that can be obtained from $\vee$ and $\neg$.

Let $w = w_1 \dots w_n \in \Gamma^*$ be a word such that $|w| = n$. We represent w as a structure $(\{1, \dots, n\}, \leq, (P_a)_{a\in\Gamma})$ where $P_a = \{i \mid w_i = a\}$. Further, we denote by $\mathcal{D}om(w) = \{1, \dots, n\}$ the domain of w as a structure. Given a finite set V of first-order and second-order variables, a (V, w)-assignment σ is a function that maps every first order variable in V to $\mathcal{D}om(w)$ and every second order variable in V to $2^{\mathcal{D}om(w)}$. Furthermore, we denote by $\sigma[x \to i]$ the extension of the (V, w)-assignment σ such that $\sigma[x \to i](x) = i$ and $\sigma[x \to i](y) = \sigma(y)$ for all variables $y \neq x$. The assignment $\sigma[X \to I]$, where X is a second-order variable and $I \subseteq \mathcal{D}om(w)$, is defined analogously. Consider an MSO-formula φ and a (V, w)-assignment σ where V is the set of free variables of φ. We write $(w, \sigma) \vDash \varphi$ if (w, σ) satisfies φ using the standard MSO-semantics.

Semirings and weighted automata. A semiring signature $\xi := (\oplus, \odot, \mathbb{0}, \mathbb{1})$ is a tuple containing two binary function symbols $\oplus, \odot$, where $\oplus$ is called the addition and $\odot$ the multiplication, and two constant symbols $\mathbb{0}$ and $\mathbb{1}$. A semiring over the signature ξ is a ξ-structure $\mathbb{S} = (S, \oplus, \odot, \mathbb{0}, \mathbb{1})$ where $(S, \oplus, \mathbb{0})$ is a commutative monoid, $(S, \odot, \mathbb{1})$ is a monoid, multiplication distributes over addition, and $\mathbb{0} \odot s = s \odot \mathbb{0} = \mathbb{0}$ for each $s \in S$. If the multiplication is commutative, then we say that $\mathbb{S}$ is commutative. In this paper, we always assume that $\mathbb{S}$ is commutative. Note that a semiring signature is a tuple rather than a set to make it clear which symbol serves as addition and which as multiplication. For simplicity, we usually denote the set of elements S by the name of the semiring $\mathbb{S}$. As standard examples of semirings we will consider the *semiring of natural numbers* $\mathbb{N}(+, \cdot) = (\mathbb{N}, +, \cdot, 0, 1)$, the *min-plus semiring*

$\mathbb{N}_\infty(\min,+) = (\mathbb{N}_\infty, \min, +, \infty, 0)$, and the *max-plus semiring* $\mathbb{N}_{-\infty}(\max,+) = (\mathbb{N}_{-\infty}, \max, +, -\infty, 0)$ which are standard semirings in the field of weighted automata.

Fix a semiring $\mathbb{S}$ and a finite alphabet Γ. A *weighted automata* over $\mathbb{S}$ and Γ [20] is a tuple $\mathcal{A} = (\Gamma, \mathbb{S}, Q, E, I, F)$ where Q is a finite set of states, $E : Q \times \Gamma \times Q \to \mathbb{S}$ is the transition relation, and $I, F : Q \to \mathbb{S}$ is the initial and final function, respectively. Usually, if $E(p,a,q) = s$, we denote this transition graphically by $p \xrightarrow{a/s} q$. Given a word $w = w_1 \dots w_n$ over Γ, a *run* ρ of $\mathcal{A}$ over w is a sequence of states and transitions:

$$\rho \;=\; q_0 \xrightarrow{w_1/s_1} q_1 \xrightarrow{w_2/s_2} \cdots \xrightarrow{w_n/s_n} q_n.$$

A run ρ like above is *accepting* if $I(q_0) \neq \mathbb{0}$, $F(q_n) \neq \mathbb{0}$, and $s_i \neq \mathbb{0}$ for every $i \in [1,n]$. In this case, the *weight* $|\rho|$ of an accepting run ρ of $\mathcal{A}$ over w is defined by $|\rho| = I(q_0) \odot \prod_{i=1}^{n} s_i \odot F(q_n)$. We define $\mathrm{Run}_{\mathcal{A}}(w)$ as the set of all accepting runs of $\mathcal{A}$ over w. Finally, the weight of $\mathcal{A}$ over w is defined by $[\![\mathcal{A}]\!](w) = \sum_{\rho \in \mathrm{Run}_{\mathcal{A}}(w)} |\rho|$ where the sum is equal to $\mathbb{0}$ if $\mathrm{Run}_{\mathcal{A}}(w) = \varnothing$.

We say that a function $f : \Gamma^* \to \mathbb{S}$ is definable by a weighted automaton over $\mathbb{S}$ and Γ if there exists a weighted automaton $\mathcal{A}$ such that $f(w) = [\![\mathcal{A}]\!](w)$ for every $w \in \Gamma^*$. We define the set of all functions definable by a weighted automaton over $\mathbb{S}$ and Γ by WA where $\mathbb{S}$ and Γ are understood from the context.

Proviso. In this paper, we always assume that Γ is a finite alphabet and $\mathbb{S}$ is a generic commutative semiring over the semiring signature $\xi := (\oplus, \odot, \mathbb{0}, \mathbb{1})$.

III. Quantitative logics

In this section, we propose a new kind of logic in the direction of Weighted Logic [11]. Our logic calls for a different spirit in the syntax by making explicit the division between the Boolean and the semiring world.

A. Quantitative Monadic Second-Order Logic

The main idea of this logic is to explicitly separate in the syntax the quantitative properties from the qualitative ones. Following this idea, we divide the syntax into two levels. The first level of the logic consists of full MSO formulas and we call it the *Boolean level*. We choose MSO-logic in order to have the full expressibility of word automata, but any Boolean logic over words (like FO or LTL) can be used. Basically, with an MSO-formula we can define a characteristic function over a word, that is, a function that returns $\mathbb{1}$ or $\mathbb{0}$ depending on whether the formula is true or false. In the second level the semiring comes into play and one can define functions using the characteristic functions and constants, and operate them by using addition, multiplication, or first- or second-order quantification over the semiring. This level is called the *semiring level*.

Definition 3.1 (Syntax of $\mathbf{QMSO[\mathbb{S}, \Gamma]}$)**.** *The formulas of* Quantitative MSO over $\mathbb{S}$ and Γ ($\mathrm{QMSO}[\mathbb{S}, \Gamma]$) *are defined by the following grammar.*

$$\theta := \varphi \mid s \mid (\theta \oplus \theta) \mid (\theta \odot \theta) \mid \Sigma x.\, \theta \mid \Pi x.\, \theta \mid \Sigma X.\, \theta$$

where $\varphi \in \mathrm{MSO}[\leq, (P_a)_{a \in \Gamma}]$, $s \in \mathbb{S}$, x *and* y *are first-order variables and* X *is a set variable.*

Note that the syntax of $\mathrm{QMSO}[\mathbb{S}, \Gamma]$ depends on the signature ξ of the semiring, i.e. the operators $\oplus$, $\odot$, Σ, and Π depend on ξ. However, as this is uniquely determined by $\mathbb{S}$ we refrain from listing ξ explicitly and simply write $\mathrm{QMSO}[\mathbb{S}, \Gamma]$ instead of $\mathrm{QMSO}[\mathbb{S}, \xi, \Gamma]$. To distinguish the quantification in the Boolean level from the quantification in the semiring level, we will refer to the operators $\Sigma X, \Sigma x$ as (second- and first-order) sum quantification and to Πx as product quantification. Note that we could also have introduced a universal second-order quantifier $\Pi X.\varphi$ at the semiring level. While such an operator might be interesting to study and adds expressiveness, we refrain from doing so as it has no influence on any of our results.

Definition 3.2 (Semantics of $\mathbf{QMSO[\mathbb{S}, \Gamma]}$)**.** *Let* $w = w_1 \dots w_n \in \Gamma^*$ *where* $n = |w|$. *For the first level, the Boolean level* φ*, the semantics is the usual semantics of MSO, i.e. for any assignment* σ*,*

$$[\![\varphi]\!](w,\sigma) := \begin{cases} \mathbb{1} & \text{if } (w,\sigma) \vDash \varphi \\ \mathbb{0} & \text{otherwise.} \end{cases}$$

The semantics of the semiring level is defined as follows.

$$\begin{aligned}
[\![s]\!](w,\sigma) &:= s \\
[\![(\theta_1 \oplus \theta_2)]\!](w,\sigma) &:= [\![\theta_1]\!](w,\sigma) \oplus [\![\theta_2]\!](w,\sigma) \\
[\![(\theta_1 \odot \theta_2)]\!](w,\sigma) &:= [\![\theta_1]\!](w,\sigma) \odot [\![\theta_2]\!](w,\sigma) \\
[\![\Sigma x.\, \theta]\!](w,\sigma) &:= \textstyle\bigoplus_{i=1}^{n} [\![\theta]\!](w, \sigma[x \to i]) \\
[\![\Pi x.\, \theta]\!](w,\sigma) &:= \textstyle\bigodot_{i=1}^{n} [\![\theta]\!](w, \sigma[x \to i]) \\
[\![\Sigma X.\, \theta]\!](w,\sigma) &:= \textstyle\bigoplus_{I \subseteq [1,n]} [\![\theta]\!](w, \sigma[X \to I])
\end{aligned}$$

For the special case $w := \epsilon$ *we have that* $[\![\Pi x.\, \theta]\!](w,\sigma) := \mathbb{1}$ *and* $[\![\Sigma x.\, \theta]\!](w,\sigma) := \mathbb{0}$.

Example 3.3. *Let* $\Gamma = \{a, b\}$ *and consider the* semiring of natural numbers $\mathbb{N}(+,\cdot)$. *Suppose the symbols* a *and* b *are services provided by a server where service* a *costs* 3 *and service* b *costs* 4. *A word over* Γ *corresponds to a sequence of services performed and we want to compute the total cost of* w*, i.e.* $3 \cdot |w|_a + 4 \cdot |w|_b$. *This can naturally be specified by the formula* $\Sigma x.\, \big(3 \cdot P_a(x) + 4 \cdot P_b(x)\big)$ *over* $\mathbb{N}(+,\cdot)$.

In QMSO it is useful to have *Boolean filtering* that gives some *syntactic sugar* to the logic. For a Boolean formula $\varphi \in \mathrm{MSO}$ and Quantitative formula $\theta \in \mathrm{QMSO}$ we define: $\varphi \mapsto \theta := (\varphi \odot \theta) \oplus (\neg\varphi)$. In words, $[\![\varphi \mapsto \theta]\!](w)$ outputs $[\![\theta]\!](w)$ whenever φ holds on w and $\mathbb{1}$ otherwise. In the following examples we give some intuition of the expressive power of QMSO.

Example 3.4. *Let $\Gamma = \{a, b, c\}$ and consider the min-plus semiring $\mathbb{N}_\infty(\min, +)$. The following formula τ_1 over $\mathbb{N}_\infty(\min, +)$ defines the minimum of the number of a's and the number of b's in a word.*

$$\tau_1 := \min\left\{ \Sigma x.\, (P_a(x) \mapsto 1)\ ,\ \Sigma x.\, (P_b(x) \mapsto 1) \right\}$$

Note that here Σx plays the rôle of product-quantification and $\min$ the rôle of sum-quantification. Recall that in the min-plus semiring the formula $P_a(x)$ evaluates to $\mathbb{1} := 0$ in case x points to a position labeled by a and $\mathbb{0} := \infty$ otherwise. This implies that the Boolean filtering $(P_a(x) \mapsto 1)$ is equal to 1 in an a-position and 0 otherwise. Therefore, formula $\Sigma x.\, P_a(x) \mapsto 1$ sums over all a-positions and, then, τ_1 takes the minimum between the number of a's and the number of b's.

Example 3.5. *Let $\Gamma = \{a, b\}$ and $\mathbb{N}_{-\infty}(\max, +)$ be the max-plus semiring. In the following example the operators $\mathrm{Max}\, x$ and Σx play the rôle of the sum and product quantification of the general setting, as the signature of $\mathbb{N}_{-\infty}(\max, +)$ is $(\max, +, -\infty, 0)$.*

We want to specify the maximum length of all infix sequences of b's. We can easily define this quantitative property in QMSO *as follows:*

$$\mathrm{Max}\, x.\ \mathrm{Max}\, y.\ \mathrm{int}_b(x, y) \mapsto \big(\Sigma z.\, (x \le z \wedge z \le y) \mapsto 1\big).$$

where $\mathrm{int}_b(x, y) := x \le y \wedge \forall z.(x \le z \wedge z \le y) \to P_b(z)$ is an FO formula defining that (x, y) is an interval of b's.

B. Fragments and Variants of QMSO

We introduce two variants of the Π-operator, the *forward-iterator* $(\cdot)^{\rightarrow}$ and the *backward-iterator* $(\cdot)^{\leftarrow}$. While both operators can already be expressed in QMSO, they will be useful to characterize deterministic and co-deterministic weighted automata (see Section V). The difference between these two operators and the operator Π is that it does not need a free variable in θ. The semantics of these operators are:

$$[\![\theta^{\rightarrow}]\!](w, \sigma) := \prod_{i=1}^{n} [\![\theta]\!](w[1..i], \sigma)$$
$$[\![\theta^{\leftarrow}]\!](w, \sigma) := \prod_{i=1}^{n} [\![\theta]\!](w[i..n], \sigma)$$

where $w[1..i]$ ($w[i..n]$) denotes the prefix (suffix) of w at position i.

Note that all previous examples can also be defined using the forward iterator $(\cdot)^{\rightarrow}$. We illustrate a general use of this operator with the next example.

Example 3.6. *Let φ be a Boolean* MSO*-formula and we want to determine how many prefixes of w satisfy φ. The following formula τ_2 over the semiring $\mathbb{N}_\infty(\min, +)$ defines this function: $\tau_2 := (\varphi \mapsto 1)^{\rightarrow}$.*

Fragments of QMSO. As usual in logic, we will consider various fragments of QMSO obtained by restricting the type and the nesting of operations allowed in the logic. For any subset $\mathrm{Op} \subseteq \{\oplus, \odot, \Sigma_x, \Pi_x, \Sigma_X, ^{\rightarrow}, ^{\leftarrow}\}$ of operators in the semiring level we denote by $\mathrm{QMSO}(\mathrm{Op})$ the restriction of QMSO to the operators in Op. For example, we write $\mathrm{QMSO}(\Sigma_X, \Sigma_x, \Pi_x, \oplus, \odot)$ (or just QMSO) for the full logic, where we write Σ_X for second-order and Σ_x for first-order sum quantification.

Another type of fragment we consider is obtained by restricting the alternation and nesting of operators in the semiring level. For this, we specify the semiring quantifier alternation of formulas in the obvious way by a *quantifier pattern*, that is, by a word over $\{\Sigma_X^n, \Sigma_x^n, \Pi_x^n \mid n \in \mathbb{N}_\infty\}$. Here, the index $(\cdot)^n$ specifies the number of nested quantifiers in a block (or any number if $n = \infty$). For instance, the fragment $\mathrm{QMSO}(\Sigma_X^\infty \Sigma_x^\infty \Pi_x^1, \oplus, \odot)$ contains all QMSO-formulas with any number of second-order sum quantifiers followed by any number of first-order sum quantifiers followed by non-nested product quantification. Note that we do not require our formulas to be in prefix normal form, so formulas in $\mathrm{QMSO}(\Sigma_X^\infty \Sigma_x^\infty \Pi_x^1, \oplus, \odot)$ can contain more than one product quantifier, but no two nested inside each other. For example, formula τ_1 in Example 3.4 can be defined in $\mathrm{QMSO}(\Pi_x^1, \oplus, \odot)$ (where Σx plays the rôle of a product quantification). Often we do not distinguish between first- and second-order sum quantification and use $\Sigma_{X,x}^n$ in the specification of the quantifier pattern, meaning that we are allowed to use n nested sum quantifiers of any type. Therefore, the fragment $\mathrm{QMSO}(\Sigma_X^\infty \Sigma_x^\infty \Pi_x^1, \oplus, \odot)$ can more concisely be denoted by $\mathrm{QMSO}(\Sigma_{X,x}^\infty \Pi_x^1, \oplus, \odot)$. We often drop the superscript ∞ and, e.g., just write Σ_x for Σ_x^∞.

We also restrict the use of the $(\cdot)^{\rightarrow}$ or $(\cdot)^{\leftarrow}$ operators in the fragments studied in this paper. Specifically, if $(\cdot)^{\rightarrow}$ or $(\cdot)^{\leftarrow}$ are considered in a fragment of QMSO (e.g. $\mathrm{QMSO}(^{\rightarrow}, \oplus, \odot)$) we suppose that these operators cannot be nested.

Finally, some classes of weighted automata are characterized by a restricted use of the $\oplus$ or $\odot$ operators (see Section V). Given an operator $\star \in \{\oplus, \odot\}$ and any subset

Op of operators in the semiring level, we define the fragment $\mathrm{QMSO}(\mathrm{Op}, \star_b)$ such that $\theta \in \mathrm{QMSO}(\mathrm{Op}, \star_b)$ whenever $\theta \in \mathrm{QMSO}(\mathrm{Op}, \star)$, and for all subformulas $\theta_1 \star \theta_2$ of θ we have that $\theta_1, \theta_2 \in \mathrm{QMSO}(\oplus, \odot)$. Informally, the $\star$-operator in $\mathrm{QMSO}(\mathrm{Op}, \star_b)$ is restricted to a "base level" between characteristic formulas (without sum or product quantification). For example, τ_2 in Example 3.6 is in $\mathrm{QMSO}(^{\rightarrow}, \oplus_b, \odot)$ but τ_1 in Example 3.4 is not in $\mathrm{QMSO}(\Pi_x, \oplus_b, \odot)$ (as $\min$ is used outside Σx).

As usual we say that a function $f : \Gamma^* \to \mathbb{S}$ is definable by a $\mathrm{QMSO}(\mathrm{Op})$-formula over $\mathbb{S}$ and Γ if there exists a formula θ in $\mathrm{QMSO}(\mathrm{Op})$ such that $f(w) = [\![\theta]\!](w)$ for every $w \in \Gamma^*$. We define the set of all functions definable in $\mathrm{QMSO}(\mathrm{Op})$ over $\mathbb{S}$ and Γ by $\mathrm{QMSO}(\mathrm{Op})$ where $\mathbb{S}$ and Γ are understood from the context.

IV. Quantitative Iteration Logic

The most important fragment of QMSO we study in this paper is the $\mathrm{QMSO}(\Sigma^{\infty}_{X,x}\Pi^1_x, \oplus, \odot)$-fragment, which we call *Quantitative Iteration Logic* (QIL). We will show next that QIL captures exactly the expressiveness of weighted automata provided that the semiring $\mathbb{S}$ is commutative.

Theorem 4.1. *A function $f : \Gamma^* \to \mathbb{S}$ is definable by a weighted automaton over $\mathbb{S}$ and Γ iff f is definable by a formula in* QIL*, and this translation is effective. In other words,*

$$\mathrm{WA} \;\equiv\; \mathrm{QIL}\,.$$

The proof of Theorem 4.1 (postponed to the full version due to space restrictions) resembles in part the proof in [11] where the equivalence of weighted automata and *Weighted Logic* is established. However, our proof is somewhat different and the translation of the product quantification has better complexity. In particular, only one exponentiation is needed to construct the weighted automaton that defines Πx. In [11], the construction is more complicated and it induces a weighted automaton of at least double exponential size.

Having defined QIL as a specification language for weighted automata, we turn our attention to its complexity, namely, its evaluation complexity as well as the decidability of standard formula construction problems such as *equivalence* and *containment* of formulas.

We start by studying the evaluation of QIL-formulas over the natural numbers with respect to counting complexity classes [25], [19]. Given a formula $\theta \in \mathrm{QIL}[\mathbb{N}(+,\cdot), \Gamma]$ and a word $w \in \Gamma^*$, we study the data complexity and combined complexity of the functions QIL-Evaluation$_\theta$ and QIL-Evaluation which receive as parameter a word $w \in \Gamma^*$ or a tuple (θ, w), respectively, and output $[\![\theta]\!](w)$.

To study the complexity of these functions, we consider two counting complexity classes: FP and FPSPACE(poly) [19]. Recall that FP (resp. FPSPACE) is the class of functions computable in polynomial time (resp. space). FPSPACE(poly) [19] is defined as the class of functions in FPSPACE such that the output is of polynomial size with respect to the input. Note that the counting classes FP and FPSPACE are the analogs of the classes PTIME and PSPACE of decision problems.

It is well-known that the data-complexity of MSO over words is in PTIME and that its corresponding combined complexity is PSPACE-complete. In the next result, we show that its quantitative counterpart $\mathrm{QIL}[\mathbb{N}(+,\cdot), \Gamma]$ inherits similar complexity bounds but this time in the counting world.

Theorem 4.2. *For any formula $\theta \in \mathrm{QIL}[\mathbb{N}(+,\cdot), \Gamma]$:*

1) QIL-Evaluation$_\theta$ *is in FP.*
2) QIL-Evaluation *is FPSPACE(poly)-complete.*

Interestingly, the combined complexity of QIL is strictly below FPSPACE given that it is known that FPSPACE(poly) $\subsetneq$ FPSPACE [19].

We now turn to problems such as containment and equivalence of formulas. For this purpose, we restrict our analysis to the semirings $\mathbb{N}(+,\cdot)$ and $\mathbb{N}_\infty(\min, +)$. Equivalence and containment of formulas in QMSO are the quantitative generalizations of the classical decision problems in logics [7]. Formally, given two sentences $\theta_1, \theta_2 \in \mathrm{QMSO}$ over $\mathbb{S}$ and Γ with a total order $\leq$, we want to decide:

- *Equivalence*: $[\![\theta_1]\!](w) = [\![\theta_2]\!](w)$ for all $w \in \Gamma^*$,
- *Containment*: $[\![\theta_1]\!](w) \leq [\![\theta_2]\!](w)$ for all $w \in \Gamma^*$.

The formalism of WA is well-known for its lack of good decidability properties. Many interesting questions, such as equivalence, containment, or even boundedness, quickly become undecidable over useful semirings like $\mathbb{N}(+,\cdot)$ and $\mathbb{N}_\infty(\min, +)$. More precisely, it is a folklore result that containment of WA over $\mathbb{N}(+,\cdot)$ is undecidable. Furthermore, in [18], [1] it was shown that the equivalence and containment problem are undecidable for WA over $\mathbb{N}_\infty(\min, +)$. Combined with Theorem 4.1 this implies that containment of QIL is also undecidable over both semirings. These results can be made stronger by showing that containment of the fragment $\mathrm{QMSO}(\Sigma_x\Pi^1_x, \oplus, \odot)$ is undecidable in both cases.

Proposition 4.3. *The following problems are undecidable:*

1) *Containment of formulas in* $\mathrm{QMSO}(\Sigma_x\Pi^1_x, \oplus, \odot)$ *over* $\mathbb{N}(+,\cdot)$.
2) *Equivalence and containment of formulas in* $\mathrm{QMSO}(\Sigma_x\Pi^1_x, \oplus, \odot)$ *over* $\mathbb{N}_\infty(\min, +)$.

As these results show, if we are interested in a logic for quantitative properties with good decidability properties for equivalence and containment, we will need to further restrict the logic QIL. In the following sections we will show the richness of our setting by analysing different fragments of QIL and relate them to corresponding classes of weighted automata.

V. Characterization of different classes of weighted automata

Depending on restrictions imposed on the amount of "ambiguity" allowed in the definition of weighted automata (WA), one can capture different classes of functions over words [17]. Here, by ambiguity we mean the maximum number of different accepting runs an automaton can take on any input word. As an example, it is known that there exists a weighted automaton that cannot be defined as a deterministic weighted automaton (DWA). Furthermore, *unambiguous* WA ($unamb$-WA), *finitely ambiguous* WA (fin-WA), and *polynomially ambiguous* WA ($poly$-WA) are different classes of WA which define different classes of functions over words. In [17], it is shown that for $\mathbb{N}_\infty(\min,+)$ the containment between these classes is strict:

$$\text{DWA} \subsetneq \textit{unamb}\text{-WA} \subsetneq \textit{fin}\text{-WA} \subsetneq \textit{poly}\text{-WA} \subsetneq \text{WA}$$

Interestingly, all these classes can be characterized by restricting QMSO, or rather QIL, to different sets of operators. In the following subsections we explain each class in detail and show how to capture it with a subset of QIL.

A. Deterministic weighted automata

A weighted automaton $\mathcal{A} = (\Gamma, \mathbb{S}, Q, E, I, F)$ is called *deterministic* if (1) for every $p \in Q$ and $a \in \Gamma$ there exists only one $q \in Q$ such that $E(p,a,q) \neq \mathbb{0}$ and (2) there exists at most one state $q_0 \in Q$ such that $I(q_0) \neq \mathbb{0}$. It is known that deterministic WA are less expressive than WA (see [17]). Therefore, DWA forms a proper subclass inside WA. We show next that this class can be characterized by a subclass of QIL and the forward iterator $(\cdot)^{\rightarrow}$.

Theorem 5.1. *A function $f : \Gamma^* \to \mathbb{S}$ is definable by a deterministic weighted automaton over $\mathbb{S}$ and Γ iff f is definable by a formula in* $\text{QMSO}(^{\rightarrow}, \oplus_b, \odot)$. *That is,*

$$\text{DWA} \equiv \text{QMSO}(^{\rightarrow}, \oplus_b, \odot).$$

Compared to the Π-operator, the construction of the automaton for a formula of the form $\theta^{\rightarrow}$ (see the full paper) is very simple. In particular, it is linear with respect to the size of the weighted automaton for $\theta \in \text{QMSO}(\oplus, \odot)$, in contrast to the exponential blow-up in the construction for Π.

Another interesting class of WA are *co-deterministic* WA (co-DWA). We say that a weighted automaton $\mathcal{A} = (\Gamma, \mathbb{S}, Q, E, I, F)$ is co-deterministic if the *reverse* automata of $\mathcal{A}$ is deterministic. Formally, if (1) for every $q \in Q$ and $a \in \Gamma$ there exists at most one $p \in Q$ such that $E(p,a,q) \neq \mathbb{0}$ and (2) there exists only one state $q_f \in Q$ such that $F(q_f) \neq \mathbb{0}$. Co-deterministic weighted automata form a subclass incomparable to deterministic weighted automata. The following theorem shows that co-DWA can also be characterized but this time using the backward iterator $(\cdot)^{\leftarrow}$.

Theorem 5.2. *A function $f : \Gamma^* \to \mathbb{S}$ is definable by a co-deterministic weighted automaton over $\mathbb{S}$ and Γ iff f is definable by a formula in* $\text{QMSO}(^{\leftarrow}, \oplus_b, \odot)$. *That is,*

$$co\text{-DWA} \equiv \text{QMSO}(^{\leftarrow}, \oplus_b, \odot).$$

Recently, the determinization of WA has been extensively studied [17], [16], [2], being still a main open problem in this area. We think that the understanding of the expressiveness of DWA (and the other classes) by our logic could give clues towards a final solution for this problem.

B. Unambiguous and finitely ambiguous WA

A weighted automaton $\mathcal{A} = (\Gamma, \mathbb{S}, Q, E, I, F)$ is *unambiguous* if $|\text{Run}_{\mathcal{A}}(w)| \leq 1$ for every $w \in \Gamma^*$, i.e. if there exists at most one accepting run of $\mathcal{A}$ on w. We call $\mathcal{A}$ *finitely ambiguous* if there is a constant $N \in \mathbb{N}$ such that $|\text{Run}_{\mathcal{A}}(w)| \leq N$ for every $w \in \Gamma^*$. Clearly, if $\mathcal{A}$ is unambiguous then it is finitely ambiguous with $N = 1$.

Unambiguous and finitely ambiguous weighted automata ($unamb$-WA and fin-WA, respectively) are another proper subclasses of weighted automata ([17]). As before, the expressiveness of both classes can be captured if we consider a subset of the operators of QMSO.

Theorem 5.3. *A function $f : \Gamma^* \to \mathbb{S}$ is definable by an unambiguous (resp. finitely ambiguous) weighted automaton over $\mathbb{S}$ and Γ iff f is definable by a formula in* $\text{QMSO}(\Pi^1_x, \oplus_b, \odot)$ *(resp.* $\text{QMSO}(\Pi^1_x, \oplus, \odot)$*). That is,*

$$\begin{aligned} \textit{unamb}\text{-WA} &\equiv \text{QMSO}(\Pi^1_x, \oplus_b, \odot) \\ \textit{fin}\text{-WA} &\equiv \text{QMSO}(\Pi^1_x, \oplus, \odot). \end{aligned}$$

The previous result relies on a careful construction of the first-order Π-quantification and the *disambiguation theorem* proved in [17]. As an example, this result shows that the formula τ_1 in Example 3.4 can be computed by a finitely ambiguous weighted automaton over $\mathbb{N}_\infty(\min,+)$.

C. Polynomially ambiguous weighted automata

A weighted automaton $\mathcal{A} = (\Gamma, \mathbb{S}, Q, E, I, F)$ is called *polynomially ambiguous* if there exists a polynomial $p(x)$ such that $|\mathrm{Run}_{\mathcal{A}}(w)| \leq p(|w|)$ for every $w \in \Gamma^*$. Polynomially ambiguous WA ($poly$-WA) were studied in [17], [16] and it was shown that they constitute another proper subclass in the family of WA. Further, in [16] an algorithm for deciding whether a polynomially ambiguous WA is determinizable was given. Polynomially ambiguous WA can also be captured by a subset of QMSO as follow.

Theorem 5.4. *A function $f : \Gamma^* \to \mathbb{S}$ is definable by a polynomially ambiguous weighted automaton over $\mathbb{S}$ and Γ iff f is definable by a formula in $\mathrm{QMSO}(\Sigma_x \Pi^1_x, \oplus, \odot)$. That is,*

$$poly\text{-WA} \equiv \mathrm{QMSO}(\Sigma_x \Pi^1_x, \oplus, \odot).$$

As in the previous characterizations, the translation above is effective and both directions are interesting. In particular, to go from a polynomial ambiguous WA to a formula in $\mathrm{QMSO}(\Sigma_x \Pi^1_x, \oplus, \odot)$ we use a refinement of the first order sum operator, which we call the "split-operator", and a recursive decomposition of the automaton by using a characteristic property already exploited in [16]. As an example, the formula in Example 3.5 can be computed by a polynomial ambiguous weighted automaton over $\mathbb{N}_{-\infty}(\max, +)$.

D. A robust and decidable fragment of QIL

Recall that one motivation for studying fragments of QIL in relation to classes of WA was the quest for a logic for quantitative properties that has decidable problems such as equivalence and containment. In the context of the previous section, we can rephrase this question in terms of WA: which subclass of WA has good decidability properties with respect to the containment and equivalence problem? A subclass that gives a positive answer to this question is the class of $unamb$-WA [7], [15]. As a corollary we get the following result.

Corollary 5.5. *The following problems are decidable:*

1) *Equivalence and containment problem of formulas in $\mathrm{QMSO}(\Pi^1_x, \oplus_b, \odot)$ over $\mathbb{N}(+, \cdot)$.*
2) *Equivalence and containment problem of formulas in $\mathrm{QMSO}(\Pi^1_x, \oplus, \odot)$ over $\mathbb{N}_{\infty}(\min, +)$.*

We are not aware that the containment problem over finitely ambiguous WA over $\mathbb{N}(+, \cdot)$ has been studied before but we conjecture that it is decidable. A positive answer over this problem would give a positive answer for the logic $\mathrm{QMSO}(\Pi^1_x, \oplus, \odot)$ over $\mathbb{N}(+, \cdot)$.

VI. Towards a robust and expressive fragment of QIL

The previous results show that $\mathrm{QMSO}(\Pi^1_x, \oplus, \odot)$ can make a reasonable claim towards being a quantitative logic with good decidability properties. Therefore, we take this logic as a starting point and try to expand its expressive power by studying the expressiveness of QMSO when we allow nesting of sum-quantification ($\mathrm{QMSO}(\Sigma_x, \oplus, \odot)$) or product-quantification ($\mathrm{QMSO}(\Pi_x, \oplus_b, \odot)$). The problem of these fragments is that either there exists no restriction in the ambiguity of WA that captures these fragments or they go easily beyond the expressiveness of WA. However, they can still be captured by non-standard weighted automata models. For the additive fragment, we show that it is equally expressive as *pure-nondeterministic* WA. On the other hand, we use the model of two-way WA with nested pebbles [14], [5] to capture the expressiveness of the multiplicative fragment. Furthermore, in both cases we show that the number of nested quantifiers is directly related to a specific feature of the corresponding weighted model.

A. Additive fragment

We start with the fragments $\mathrm{QMSO}(\Sigma_X, \oplus, \odot)$ and $\mathrm{QMSO}(\Sigma_x, \oplus, \odot)$ where only sum-quantification is allowed. First of all, note that these fragments are equally expressive as $\mathrm{QMSO}(\Sigma_X, \oplus, \odot_b)$ and $\mathrm{QMSO}(\Sigma_x, \oplus, \odot_b)$, respectively. In fact, for any formula in $\mathrm{QMSO}(\Sigma_X, \oplus, \odot)$ we can always "push" single products to the "base level" of the formula (by distributivity) and get an equivalent formula in $\mathrm{QMSO}(\Sigma_X, \oplus, \odot_b)$. Thus, during this section we can restrict our analysis, without loss of generality, to the fragments $\mathrm{QMSO}(\Sigma_X, \oplus, \odot_b)$ and $\mathrm{QMSO}(\Sigma_x, \oplus, \odot_b)$.

Any fragment of WA studied in the previous section can easily go beyond $\mathrm{QMSO}(\Sigma_x, \oplus, \odot_b)$ by using multiple products. For example, over $\mathbb{N}(+, \cdot)$ the function $2^{|w|}$ can be computed by a deterministic WA (actually with a single state) but it is not difficult to show that $[\![\theta]\!] \in o(2^n)$ for any $\theta \in \mathrm{QMSO}(\Sigma_x, \oplus, \odot_b)$.

Given the above reason, we consider in this section a WA model that does not use products on its edges. We call such automata pure-nondeterministic WA (also known as *multiplicity automata*). Specifically, a pure-nondeterministic weighted automaton over Γ and $\mathbb{S}$ is a tuple $\mathcal{A} = (\Gamma, \mathbb{S}, Q, \delta, I, F)$ such that Q is the finite set of control states, $\delta \subseteq Q \times \Gamma \times Q$ is the transition relation, $I \subseteq Q$ is the set of initial states, and $F : Q \to \mathbb{S}$ is the final function. Note that neither the transitions nor the initial states are weighted in this model. Given

a word $w \in \Gamma^*$, the set of accepting runs $\mathrm{Run}_{\mathcal{A}}(w)$ is defined as usual except that the cost of a run $\rho \in \mathrm{Run}_{\mathcal{A}}(w)$ is defined by $|\rho| = F(q)$ where q is the last state in ρ. Finally, we define $[\![\mathcal{A}]\!](w)$ (i.e. the weight of $\mathcal{A}$ over w) as usual. We denote by PNWA the set of all pure-nondeterministic WA and by $poly$-PNWA the polynomial ambiguous subclass of these weighted automata.

The following result shows the connection between this pure-nondeterministic model and the fragments $\mathrm{QMSO}(\Sigma_X, \oplus, \odot_b)$ and $\mathrm{QMSO}(\Sigma_x, \oplus, \odot_b)$.

Proposition 6.1. *The following classes of* PNWA *and subfragments of* QMSO *are equally expressive over* Γ *and* $\mathbb{S}$*:*

$$\begin{aligned} \text{PNWA} &\equiv \mathrm{QMSO}(\Sigma_X, \oplus, \odot_b) \\ poly\text{-PNWA} &\equiv \mathrm{QMSO}(\Sigma_x, \oplus, \odot_b) \end{aligned}$$

The previous result is not surprising given the results presented in the previous section. The interesting part emerges when we focus on the fragment $\mathrm{QMSO}(\Sigma_x, \oplus, \odot_b)$ and compare each formula with the corresponding weighted automata in $poly$-PNWA. We show that the number of nested Σ-quantifications coincides with the "degree" of ambiguity of an equivalent weighted automata in $poly$-PNWA. We formalize this claim as follows. Given an automaton $\mathcal{A} \in poly$-PNWA, we can define the function $r_{\mathcal{A}} : \mathbb{N} \to \mathbb{N}$ that counts the maximum number of accepting runs of $\mathcal{A}$ over all words of a given size (i.e. $r_{\mathcal{A}}(n) = \max_{w \in \Gamma^n} |\mathrm{Run}_{\mathcal{A}}(w)|$). Recall that $r_{\mathcal{A}} \in O(n^k)$ for some $k \in \mathbb{N}$ given that $\mathcal{A}$ is polynomial ambiguous. Then we define the *degree of ambiguity* of $\mathcal{A}$ by:

$$\mathrm{degree}(\mathcal{A}) \;:=\; \min_{k \in \mathbb{N}} \{\; k \;\mid\; r_{\mathcal{A}} \in O(n^k)\}.$$

In line with the previous definition, we denote by $poly^k$-PNWA the set of all pure-nondeterministic WA with degree of ambiguity at most $k \in \mathbb{N}$.

The next result shows the close connection between the nesting of first-order Σ-quantification and the degree of ambiguity of $poly$-PNWA.

Theorem 6.2. *For all* $k \in \mathbb{N}$*, the following classes of* PNWA *and fragments of* QMSO *are equally expressive over* $\mathbb{S}$ *and* Γ*:*

$$poly^k\text{-PNWA} \;\equiv\; \mathrm{QMSO}(\Sigma_x^k, \oplus, \odot_b).$$

In the proof of the above result we make a finer analysis of the graph structure of $poly^k$-PNWA to decompose the weighted automata and construct inductively an equivalent formula in $\mathrm{QMSO}(\Sigma_x^k, \oplus, \odot_b)$. We are not aware of any previous work that has made this connection before.

The previous result sheds light on a new (infinite) hierarchy of sub-classes of polynomial ambiguous WA that are definable by reasonable fragments of QMSO and could have good decidability properties.

B. Multiplicative fragment

In this subsection, we study the expressiveness of QMSO when only products and Π-quantification are used (i.e. $\odot$ and Πx). Clearly, this fragment of QMSO is no longer contained in WA. For example, the formula $\Pi x.\, \Pi y.\, 2$ is not definable by any weighted automata over $\mathbb{N}(+, \cdot)$. Indeed, this formula defines the function $f(w) = 2^{|w|^2}$ for every $w \in \Gamma^*$, but $[\![\mathcal{A}]\!] \in O(2^n)$ for every weighted automata $\mathcal{A}$.

In order to capture the expressive power of $\mathrm{QMSO}(\Pi_x, \oplus_b, \odot)$ we consider *two-way weighted automata with k-nested pebbles (*2WA-k*)* studied in [14], [5]. We show that $\mathrm{QMSO}(\Pi_x, \oplus_b, \odot)$ is equally expressive than 2WA-k. Similar to the previous subsection, we show that there is a close relation between the nesting depth of Π-quantification and the number k of nested pebbles.

A *two-way weighted automaton with k-nested pebbles* is a finite state weighted machine that can move its reading head in any of the two directions (left or right) and can drop or lift pebbles over the input word for marking. Pebbles are dropped in a nested discipline: at any moment of a run if pebbles 1 to i are placed over the word ($0 \leq i \leq k$), then the only pebble that can be dropped is pebble $(i+1)$ and the only pebble that can be lifted is pebble i. We briefly recall the definition of 2WA-k here but we refer to [14], [5] for details. Formally, a 2WA-k is a tuple $\mathcal{A} = (\Gamma, \mathbb{S}, Q, E, I, F)$ where $\Gamma, \mathbb{S}, Q, I, F$ are defined as usual and E is the transition function from $Q \times (\Gamma \cup \{\triangleleft, \triangleright\}) \times \{0,1\}^k \times Q \times \{\rightarrow, \leftarrow, \downarrow, \uparrow\}$ to $\mathbb{S}$. Here, symbols $\{\triangleleft, \triangleright\}$ denote left and right markers of the beginning and end of a word, and $\{\rightarrow, \leftarrow, \downarrow, \uparrow\}$ are abbreviations to denote the possible actions of $\mathcal{A}$ after reading a letter (move right, move left, drop pebble, or lift pebble). The concepts of a *configuration*, a *run* and the *weight* of a run of the automaton on a word $w \in \Gamma^*$ are defined in the usual way.

Similar to the previous sections, we also consider the deterministic and unambiguous restrictions of 2WA-k and we denote them by 2DWA-k and $unamb$-2WA-k, respectively. The special case when $k = 0$ are just two-way weighted automata that do not use pebbles at all.

In the next result, we show that 2DWA-0 and $unamb$-2WA-0 coincide with the one-way class of $unamb$-WA. By Proposition 5.3, this implies that all these classes are equally expressive as

$\mathrm{QMSO}(\Pi^1_x, \oplus_b, \odot)$ which further shows the robustness of this class of functions for any commutative semiring.

Theorem 6.3. *The following classes of* WA *and subfragments of* QMSO *are equally expressive over* Γ *and* $\mathbb{S}$*:*

1) 2DWA-0*,*
2) $unamb$-2WA-0*,*
3) $unamb$-WA*, and*
4) $\mathrm{QMSO}(\Pi^1_x, \oplus_b, \odot)$.

As far as we are aware, we are the first to point out this equivalence between both models for any commutative semiring. It is important to note, though, that the above result does only hold for commutative semirings. For non-commutative semirings, it is well-known that 2DWA-0 and $unamb$-WA are not equivalent.

Given the above characterization of 2DWA-0, a natural question for 2DWA-k arises: does there exist a natural fragment of QMSO that captures two-way deterministic weighted automata with k-pebbles? The following theorem gives a positive answer to this question. Furthermore, it shows a direct connection between the number of nested pebbles used by 2DWA and the nesting of first-order Π-quantification of its equivalent formula in $\mathrm{QMSO}(\Pi_x, \oplus_b, \odot)$.

Theorem 6.4. *For every* $k \in \mathbb{N}$*, there exists an effective translation between the following classes of weighted automata and subfragments of* QMSO *over* $\mathbb{S}$ *and* Γ*:*

1) 2DWA-k*,*
2) $unamb$-2WA-k*, and*
3) $\mathrm{QMSO}(\Pi^{k+1}_x, \oplus_b, \odot)$.

VII. Conclusion

The aim of this paper was to define a quantitative logic with a simple and intuitive syntax that is equivalent to weighted automata (WA). Moreover, as a second goal, we aimed at defining a quantitative logic within the framework of WA for which important computational problems such as equivalence are decidable.

Towards these goals, we introduced a generic framework for adding quantitative properties to any Boolean logic over words. This allowed us to meet both aims. With QIL we have introduced a simple logic equivalent to WA. The explicit distinction between Boolean and semiring quantification in our logic allowed us to give the first logical characterisations of natural classes of WA (defined in terms of the ambiguity of automata) that have been studied in the literature before. As a consequence, we obtained an interesting logic with very good decidability properties defined by disallowing nested sums. Motivated by the good properties this logic enjoys, we started to relax the restriction to non-nested operators by studying fragments of QMSO with bounded nesting of product or sum. We believe that there is much more potential in analysing these fragments further to obtain more expressive quantitative logics with still good decidability results. We leave this for future research.

References

[1] S. Almagor, U. Boker, and O. Kupferman. What's decidable about weighted automata? In *ATVA*, pages 482–491, 2011.

[2] B. Aminof, O. Kupferman, and R. Lampert. Rigorous approximated determinization of weighted automata. In *LICS*, pages 345–354, 2011.

[3] U. Boker, K. Chatterjee, T. A. Henzinger, and O. Kupferman. Temporal specifications with accumulative values. In *LICS*, pages 43–52, 2011.

[4] B. Bollig and P. Gastin. Weighted versus probabilistic logics. In *Developments in Language Theory*, pages 18–38, 2009.

[5] B. Bollig, P. Gastin, B. Monmege, and M. Zeitoun. Pebble weighted automata and transitive closure logics. In *ICALP (2)*, pages 587–598, 2010.

[6] J. R. Buchi. Weak second-order arithmetic and finite automata. *Mathematical Logic Quarterly*, 6(1-6):66–92, 1960.

[7] K. Chatterjee, L. Doyen, and T. A. Henzinger. Quantitative languages. *ACM Trans. Comput. Log.*, 11(4), 2010.

[8] T. Colcombet. Regular cost functions, Part I: Logic and algebra over words. *CoRR*, abs/1212.6937, 2012.

[9] T. Colcombet and C. Löding. Regular cost functions over finite trees. In *LICS*, pages 70–79, 2010.

[10] B. Courcelle and J. Engelfriet. *Graph Structure and Monadic Second-Order Logic*. Cambridge University Press, 2012.

[11] M. Droste and P. Gastin. Weighted automata and weighted logics. *Theor. Comput. Sci.*, 380(1-2):69–86, 2007.

[12] M. Droste, W. Kuich, and H. Vogler. *Handbook of Weighted Automata*. Springer, 1st edition, 2009.

[13] M. Droste and I. Meinecke. Weighted automata and weighted MSO logics for average and long-time behaviors. *Inf. Comput.*, 220:44–59, 2012.

[14] P. Gastin and B. Monmege. Adding pebbles to weighted automata. In *CIAA*, pages 28–51, 2012.

[15] K. Hashiguchi, K. Ishiguro, and S. Jimbo. Decidability of the equivalence problem for finitely ambiguous finance automata. *IJAC*, 12(3):445, 2002.

[16] D. Kirsten and S. Lombardy. Deciding unambiguity and sequentiality of polynomially ambiguous min-plus automata. In *STACS*, pages 589–600, 2009.

[17] I. Klimann, S. Lombardy, J. Mairesse, and C. Prieur. Deciding unambiguity and sequentiality from a finitely ambiguous max-plus automaton. *Theor. Comput. Sci.*, 327(3):349–373, 2004.

[18] D. Krob. The equality problem for rational series with multiplicities in the tropical semiring is undecidable. In *ICALP*, pages 101–112, 1992.

[19] R. E. Ladner. Polynomial space counting problems. *SIAM J. Comput.*, 18(6):1087–1097, 1989.

[20] J. Sakarovitch. *Elements of Automata Theory*. Cambridge University Press, 2009.

[21] M. P. Schützenberger. On the definition of a family of automata. *Information and Control*, 4:245–270, 1961.

[22] H. Seidl. Deciding equivalence of finite tree automata. *SIAM J. Comput.*, 19(3):424–437, 1990.

[23] W. Thomas. Languages, automata, and logic. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume III, pages 389–455. Springer, 1997.

[24] W. G. Tzeng. A polynomial-time algorithm for the equivalence of probabilistic automata. *SIAM J. Comput.*, 21(2):216–227, 1992.

[25] L. G. Valiant. The complexity of enumeration and reliability problems. *SIAM J. Comput.*, 8(3):410–421, 1979.