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. [29], [11]). 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 [28]. 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. [14]). 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 [13] 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 [13], 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.

$$\theta \coloneqq \varphi \mid s \mid (\theta \oplus \theta) \mid (\theta \odot \theta) \mid \Sigma x.\theta \mid \Pi x.\theta \mid \Sigma X.\theta$$

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 [24].

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 subclass 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 [13], [4], [15], [14]. 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 [15]. 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 [10], [9], [3], [7]. In [10], [9], *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], [18]. We consider these models in Section VI to show the connection between different fragments of QMSO. In [5] and [18], 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 $w = w_1 \ldots w_n \in \Gamma^*$ be a word such that $|w| = n$. We represent w as a structure $(\{1,\ldots,n\}, \leq, (P_a)_{a\in\Gamma})$ where $P_a = \{i \mid w_i = a\}$. Further, we denote by $\mathcal{D}om(w) = \{1,\ldots,n\}$ 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 $(w,\sigma) \models \varphi$ 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 [27] 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 [13]. 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,*

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

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 $\varphi \in$ MSO and Quantitative formula $\theta \in$ 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.*

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

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 $\operatorname{Max} x$ 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:*

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

where $\mathrm{int}_b(x, y) := x \le y \wedge \forall z.(x \le z \wedge z \le y) \rightarrow P_b(z)$ 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:

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

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 $\mathrm{QMSO}(\rightharpoonup, \oplus_b, \odot)$ 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 [13] 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 [13], 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 [30], [26]. 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) [26]. Recall that FP (resp. FPSPACE) is the class of functions computable in polynomial time (resp. space). FPSPACE(poly) [26] 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 [26].

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 [25], [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 [24]. 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 [24], it is shown that for $\mathbb{N}_\infty(\min,+)$ the containment between these classes is strict:

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

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 [24]). 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 $(\cdot)^{\rightharpoonup}$.

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* $\mathrm{QMSO}(^{\rightharpoonup}, \oplus_b, \odot)$*. That is,*

$$\mathrm{DWA} \;\equiv\; \mathrm{QMSO}(^{\rightharpoonup}, \oplus_b, \odot).$$

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

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* $\mathrm{QMSO}(^{\leftharpoonup}, \oplus_b, \odot)$*. That is,*

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

Recently, the determinization of WA has been extensively studied [24], [23], [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 $|\mathrm{Run}_{\mathcal{A}}(w)| \leq 1$ 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 $|\mathrm{Run}_{\mathcal{A}}(w)| \leq N$ 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 ([24]). 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* $\mathrm{QMSO}(\Pi^1_x, \oplus_b, \odot)$ *(resp.* $\mathrm{QMSO}(\Pi^1_x, \oplus, \odot)$*). That is,*

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

The previous result relies on a careful construction of the first-order Π-quantification and the *disambiguation theorem* proved in [24]. 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 [24], [23] and it was shown that they constitute another proper subclass in the family of WA. Further, in [23] 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 $\mathrm{QMSO}(\Sigma_x \Pi_x^1, \oplus, \odot)$. That is,*

$$\textit{poly-}\mathrm{WA} \;\equiv\; \mathrm{QMSO}(\Sigma_x \Pi_x^1, \oplus, \odot).$$

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 $\mathrm{QMSO}(\Sigma_x \Pi_x^1, \oplus, \odot)$ 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 [23]. 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], [20]. As a corollary we get the following result.

Corollary 5.5. *The following problems are decidable:*

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

VI. Towards a robust and expressive fragment of QIL

The previous results show that $\mathrm{QMSO}(\Pi_x^1, \oplus, \odot)$ 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 [18], [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* $\mathbb{S}$*:*

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

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*:*

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

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 [18], [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 ($0 \le i \le k$), 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 [18], [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 $Q \times (\Gamma \cup \{\triangleleft, \triangleright\}) \times \{0,1\}^k \times Q \times \{\to, \leftarrow, \downarrow, \uparrow\}$ to $\mathbb{S}$. Here, symbols $\{\triangleleft, \triangleright\}$ denote left and right markers of the beginning and end of a word, and $\{\to, \leftarrow, \downarrow, \uparrow\}$ 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] M. Chytil and V. Jákl. Serial composition of 2-way finite-state transducers and simple programs on strings. In *ICALP*, pages 135–147, 1977.

[9] T. Colcombet. Regular cost functions, part i: Logic and algebra over words. *CoRR*, abs/1212.6937, 2012.

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

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

[12] M. Davis. Hilbert's tenth problem is unsolvable. *The American Mathematical Monthly*, 80(3):233–269, 1973.

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

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

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

[16] J. Engelfriet and H. J. Hoogeboom. Mso definable string transductions and two-way finite-state transducers. *ACM Trans. Comput. Log.*, 2(2):216–254, 2001.

[17] J. Engelfriet and S. Maneth. Two-way finite state transducers with nested pebbles. In *MFCS*, pages 234–244, 2002.

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

[19] N. Globerman and D. Harel. Complexity results for two-way and multi-pebble automata and their logics. *Theor. Comput. Sci.*, 169(2):161–184, 1996.

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

[21] J. E. Hopcroft and J. D. Ullman. An approach to a unified theory of automata. *The Bell System Technical Journal*, 46:1793–1829, 1967.

[22] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, 1979.

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

[24] 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.

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

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

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

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

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

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

[31] A. Weber. Finite-valued distance automata. *Theor. Comput. Sci.*, 134(1):225–251, 1994.

Appendix

A. Proofs Omitted in Section IV

Theorem 4.1. *Let $\mathbb{S}$ be a commutative semiring and Γ be a finite alphabet. A function $f : \Gamma^* \to \mathbb{S}$ is definable by a weighted automaton over $\mathbb{S}$ and Γ if, and only if, f is definable by a formula in $\mathrm{QIL}[\mathbb{S}, \Gamma]$. In other words,*

$$\mathrm{WA} = \mathrm{QIL}\,.$$

Proof: ($\Rightarrow$) Let $\mathcal{A} = (\Gamma, \mathbb{S}, E, I, F)$ be a weighted automaton that defines the function $f : \Gamma^* \to \mathbb{S}$. For this direction, we construct a formula $\theta_{\mathcal{A}} \in \mathrm{QIL}$ such that $[\![\mathcal{A}]\!](w) = [\![\theta_{\mathcal{A}}]\!](w)$ for all $w \in \Gamma^*$. The idea is similar to the classical reduction from finite automata to MSO-logic [6]: we guess a run using second-order sum quantification, check that this is a correct run using an MSO-formula, and then aggregate the cost of this run using the Π-operator. For the sake of simplicity, in this proof we consider standard abbreviations of formulas that can be defined in MSO-logic (actually, FO-logic) like $\mathrm{first}(x) := \forall y.\, x \leq y$ and $\mathrm{last}(x) := \forall y.\, y \leq x$ to denote the first and last element of the linear order $\leq$, respectively, and $\mathrm{succ}(x, y) := x \leq y \wedge y \nleq x \wedge \forall z.\, (z \leq x \vee y \leq z)$ to denote the successor relation.

First of all, we define a formula $\mathrm{partition}(X_1, \ldots, X_n)$ which defines a partition of the domain into n-subsets $X_1, \ldots, X_n$:

$$\mathrm{partition}(X_1, \ldots, X_n) := \forall x.\, \bigvee_{i=1}^{n} \left(x \in X_i \;\wedge\; \bigwedge_{j \neq i} x \notin X_j \right).$$

Let $T = \{(p, a, q) \in Q \times \Gamma \times Q \mid E(p, a, q) \neq \mathbb{0}\}$ be the set of non-zero transitions in $\mathcal{A}$. Furthermore, let $T_I = \{(p, a, q) \in T \mid I(p) \neq \mathbb{0}\}$ and $T_F = \{(p, a, q) \in T \mid F(q) \neq \mathbb{0}\}$ be the set of *initial* and *final* transitions, respectively. Finally, let $\bar{X} = (X_1, \ldots, X_n)$ be an enumeration of variables in set $\{X_t \mid t \in T\}$. Next we define the formula $\mathrm{run}_{\mathcal{A}}(\bar{X})$ that checks if $\bar{X}$ encodes a valid run of $\mathcal{A}$:

$$\begin{aligned}
\mathrm{run}_{\mathcal{A}}(\bar{X}) \;:=\;\; & \mathrm{partition}(\bar{X}) \;\wedge \\
& \bigwedge_{(p,a,q) \in T} \forall x.\, (x \in X_{(p,a,q)} \;\to\; x \in P_a) \;\wedge && (1)\\
& \forall x.\, \forall y.\, \left(\mathrm{succ}(x, y) \;\to \bigvee_{(p,a,q),(q,b,r) \in T} (x \in X_{(p,a,q)} \wedge y \in X_{(q,b,r)}) \right) \;\wedge && (2)\\
& \exists x.\, \left(\mathrm{first}(x) \;\wedge \bigvee_{(p,a,q) \in T_I} x \in X_{(p,a,q)} \right) \;\wedge\; \exists x.\, \left(\mathrm{last}(x) \;\wedge \bigvee_{(p,a,q) \in T_F} x \in X_{(p,a,q)} \right) && (3)
\end{aligned}$$

Part (1) verifies that the labeling of the run is consistent with the letters in the word and Part (2) verifies that the sequence of transitions is *well matched.* Finally, Part (3) checks that $\bar{X}$ encodes an accepting run.

Next, we define the formulas that define the cost of the initial, final, and transition functions.

$$\begin{aligned}
\mathrm{initial}(\bar{X}) &:= \sum_{(p,a,q) \in T_I} \big(\exists x.\ \mathrm{first}(x) \;\wedge\; x \in X_{(p,a,q)}\big) \odot I(p)\\
\mathrm{final}(\bar{X}) &:= \sum_{(p,a,q) \in T_F} \big(\exists x.\ \mathrm{last}(x) \;\wedge\; x \in X_{(p,a,q)}\big) \odot F(q)\\
\mathrm{transition}(x, \bar{X}) &:= \sum_{(p,a,q) \in T} \big(x \in X_{(p,a,q)}\big) \odot E(p, a, q)
\end{aligned}$$

By encoding a run with variables for each transition in $\mathcal{A}$, we are not coovering the case of an empty word. However, this can be easily encoded with the following formula:

$$\mathrm{empty} \;:=\; (\forall x.\, x \nleq x) \odot \left(\sum_{p \in Q} I(p) \odot F(p) \right)$$

To conclude the construction, we can define our formula $\theta_{\mathcal{A}}$ as follow:

$$\theta_{\mathcal{A}} \;:=\; \Sigma \bar{X} \left(\mathrm{run}_{\mathcal{A}}(\bar{X}) \;\odot\; \mathrm{initial}(\bar{X}) \;\odot\; \big(\Pi x.\ \mathrm{transition}(x, \bar{X})\big) \;\odot\; \mathrm{final}(\bar{X}) \right) \;\oplus\; \mathrm{empty}$$

It is easily checked that $\mathcal{A}$ and $\theta_{\mathcal{A}}$ define the same function.

($\Leftarrow$) For this direction, we show by structural induction how to construct for each $\theta \in \mathrm{QIL}$ a weighted automata $\mathcal{A}_\theta$ such that $[\![\theta]\!](w) = [\![\mathcal{A}]\!](w)$ for every $w \in \Gamma^*$. This construction follows the same principles as the standard Büchi construction [6] combined with the main ideas used in [13].

First of all, given a set V of first- and second-order order variables we consider the standard encoding of a word w and an (V,w)-assignment σ over the alphabet $\Gamma_V = \Gamma \times \{0,1\}^{|V|}$. We say that a word $(w,\sigma) \in \Gamma_V$ *encodes* an (V,w)-assignment σ if w is the projection of (w,σ) over Γ and for every variable $v \in V$ we have $\sigma(v) = \{i \in \{1,\dots,|w|\} \mid (w,\sigma)[v]_i = 1\}$ where $(w,\sigma)[v]_i$ denotes the i-letter of the projection of (w,σ) over variable v.

Now, we show how to construct the weighted automata $\mathcal{A}_\theta$. For the Boolean level, the task is simple by using Büchi's theorem for MSO-logic over Γ.

Lemma A.1 ([6]). *For every* MSO*-formula φ over Γ with free variables V there exists a deterministic finite automaton $\mathcal{A}_\varphi$ over Γ_V such that for every word $w \in \Gamma^*$ and (V,w)-assignment σ we have:*

$$(w,\sigma) \models \varphi \quad \textit{iff} \quad (w,\sigma) \in \mathcal{L}(\mathcal{A}_\varphi)$$

Recall that a weighted automata $\mathcal{A} = (\Gamma, \mathbb{S}, Q, E, I, F)$ 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 $I(q_0) \neq \mathbb{0}$. For the sake of simplification, when $\mathcal{A}$ is deterministic we write $\mathcal{A} = (\Gamma, \mathbb{S}, Q, \delta, q_0, F)$ such that $\delta : Q \times \Gamma \to Q \times \mathbb{S}$ and $q_0 \in I$, and the semantic of $\mathcal{A}$ follows as usual.

From Lemma A.1 one can easily obtain a deterministic weighted automata $\mathcal{A}'_\varphi$ such that $[\![\varphi]\!](w,\sigma) = [\![\mathcal{A}'_\varphi]\!](w,\sigma)$ for every word $w \in \Gamma^*$ and (V,w)-assignment σ. More specifically, let $\mathcal{A}_\varphi = (\Gamma_V, Q, \delta, q_0, F)$ be the deterministic automata that recognizes φ. Then consider $\mathcal{A}'_\varphi = (\Gamma_V, \mathbb{S}, Q, \delta', q_0, F')$ such that:

- $\delta'(p,a) = (q, \mathbb{1})$ if $\delta(p,a) = q$ and $\delta'(p,a) = (q,\mathbb{0})$ otherwise, and
- $F'(q) = \mathbb{1}$ if, and only if, $q \in F$.

Thus, one can easily show that $\mathcal{A}'_\varphi$ define the same characteristic function as φ.

In the next step we consider QIL-formulas over the semiring level. We will first show how to translate formulas without any semi-ring quantification. Interestingly, it turns out that any such formula can always be defined by a deterministic weighted automata. This will be needed later, in the proof of Theorem 5.1.

Lemma A.2. *For every formula θ of the semiring level of* QIL *without sum or product quantification and with free variables V there exists a deterministic weighted automaton $\mathcal{A}_\theta$ over Γ_V such that for every word $w \in \Gamma^*$ and (V,w)-assignment σ we have:*

$$[\![\theta]\!](w,\sigma) = [\![\mathcal{A}_\theta]\!](w,\sigma)$$

Proof: Let θ be a formula on the semiring level with free variables V such that θ contains no sum or product quantification. First, note that we can rewrite θ by distributing multiplication over addition and by using the commutativity of $\odot$ such that:

$$\theta = \sum_{i=1}^{n} \left(s_i \odot \prod_{j=1}^{n_i} \theta_i^j \right)$$

where $s_i \in \mathbb{S}$ and $\theta_i^j \in \mathrm{MSO}$. Without lost of generality, we can assume here that each MSO-formula θ_i^j has the same set of free variables V.

By Lemma A.1, we can find a deterministic automaton $\mathcal{A}_{\theta_i^j}$ for each formula $\theta_i^j \in \mathrm{MSO}$. Furthermore, we can define a deterministic automaton $\mathcal{A}_i$, for $i \leq n$, such that:

$$(w,\sigma) \in \mathcal{L}(\mathcal{A}_i) \quad \text{iff} \quad (w,\sigma) \in \mathcal{L}(\bigcap_{j=1}^{n_i} \mathcal{A}_i)$$

Let $\mathcal{A}_i = (\Gamma_V, Q_i, \delta_i, q_0^i, F_i)$. Consider a deterministic weighted automaton $\mathcal{A}_\theta = (\Gamma_V, \mathbb{S}, Q, \delta, q_0, F)$ such that:

$$Q = \bigtimes_{i=1}^{n} (Q_i \cup \{\varnothing\}), \quad q_0 = \bigtimes_{i=1}^{n} \{q_0^i\}, \quad \delta(q,a) = \left(\bigtimes_{i=1}^{n} \{\delta_i(q(i),a)\}, \mathbb{1} \right), and \quad F(q) = \sum_{q(i) \in F_i} s_i.$$

where $F(q) = \mathbb{0}$ if $q(i) \notin F_i$ for all $i \leq n$. Notice that $\mathcal{A}_\theta$ is a deterministic weighted automata, runs all automata $\mathcal{A}_i$ in parallel and outputs the sum of the intersection. Finally, it is easy to check that $\mathcal{A}_\theta$ defines the same weighted function θ. ■

Interestingly, the construction of Lemma A.2 shows that a weighted function defined by a formula θ without semiring quantification divides Γ_V into a finite number of equivalence classes such that all words inside an equivalence class have the same value in $\mathbb{S}$. In other words, this means that θ defines a *recognizable step function* as defined in [13].

Next, we show how to translate general formulas of the semi-ring level into a weighted automata. We start with the Π-operator.

Lemma A.3. *Let $\theta(x)$ be a formula without any sum or product quantification. Let $x \notin V$ be a free first-order variable and let V be the set of the other free variables. For $\tau = \Pi x. \theta$ there exists a weighted automaton $\mathcal{A}_\tau$ over $\mathbb{S}$ and Γ_V such that for every word $w \in \Gamma^*$ and (V, w)-assignment σ we have:*

$$[\![\tau]\!](w, \sigma) = [\![\mathcal{A}_\tau]\!](w, \sigma)$$

Proof: Let $\mathcal{A} = (\Gamma_{V \cup \{x\}}, \mathbb{S}, Q, \delta, q_0, F)$ be the deterministic weighted automata of Lemma A.2 that defines the function $[\![\theta]\!]$. For simplicity, we assume that $V = \varnothing$ and, therefore, $\Gamma_{\{x\}} = \Gamma \times \{0, 1\}$. Furthermore, without loss of generality we assume that δ is a complete function and - given that all weights in δ are equal to $\mathbb{1}$ - sometimes we will abuse notation and use δ as a function from $Q \times \Gamma_{\{x\}}$ to Q.

The main idea of the automaton $\mathcal{A}^*$ that defines the function $[\![\Pi x. \theta]\!]$ is to run $\mathcal{A}$ over a word in Γ by considering two parallel runs: the first run is over the alphabet $\Gamma \times \{0\}$ and the second run non-deterministically guesses which is the final state that a run that moves by reading $\Gamma \times \{1\}$ could reach. For each non-deterministic step we count the value of this final state and at the end of the run we check if all guesses were correct.

More specifically, let $\mathcal{A}' = (\Gamma, \mathbb{S}, Q', E', I', F')$ be a weighted automata over $\mathbb{S}$ and Γ such that $Q' = Q \times 2^{Q \times Q}$, $I'(q, \varnothing) = \mathbb{1}$ and $\mathbb{0}$ otherwise, and $F'(q, R) = \mathbb{1}$ if $R \subseteq \{(q, q) \mid q \in Q\}$ and $\mathbb{0}$ otherwise. For the transition relation E', we have that $E'((q_1, R_1), a, (q_2, R_2)) = s$ if there exists $q_f \in Q$ such that:

- $F(q_f) = s$,
- $\delta(q_1, (a, 0)) = q_2$, and
- $R_2 = \{(p_2, q_f') \in Q^2 \mid \exists (p_1, q_f') \in R_1. \, \delta(p_1, (a, 0)) = p_2\} \cup \{(\delta(p, (a, 1)), q_f)\}$,

and $\mathbb{0}$ in any other case. Finally, it could happen that $E'((q_1, R_1), a, (q_2, R_2)) = \mathbb{0}$ but the triple $((q_1, R_1), a, (q_2, R_2))$ satisfies the above list of properties. In this case we say that $((q_1, R_1), a, (q_2, R_2))$ is a *valid* transition in $\mathcal{A}'$.

As explained previously, notice how the first component in Q' maintains a run over $\Gamma \times \{0\}$, and how q_f is the guessed state that will be reached at the end of the run. The second component in Q' records this fact which is checked at the final state where all the elements in the second component must be of the form (q, q) for some $q \in Q$.

First, we claim that for every word there exists a unique accepting run with valid transitions in $\mathcal{A}'$.

Claim A.4. *For every word $w = a_1 \dots a_n \in \Gamma^*$ there exists a unique run ρ of w:*

$$\rho = (q_0, \varnothing) \xrightarrow{a_1/s_1} (q_1, T_1) \xrightarrow{a_2/s_2} \cdots \xrightarrow{a_n/s_n} (q_n, T_n),$$

such that each transition is valid, and $I'(q_0, T_0) \neq \mathbb{0} \neq F'(q_n, T_n)$.

To prove this claim, first notice that there always exists at least one run for every word $w \in \Gamma^*$. In fact, $\mathcal{A}$ is a deterministic and complete automata and by following the construction of $\mathcal{A}'$ one can easily check that an accepting and valid run always exists.

Now, suppose that for a word $w = a_1 \dots a_n$ there exists two runs ρ_1 and ρ_2 of w:

$$\rho_i = (q_0, T_0^i) \xrightarrow{a_1/s_1^i} (q_1, T_1^i) \xrightarrow{a_2/s_2^i} \cdots \xrightarrow{a_n/s_n^i} (q_n, T_n^i)$$

for every $i \in \{1, 2\}$ such that each transition is valid, and $I'(q_0, T_0^i) \neq \mathbb{0} \neq F'(q_n, T_n^i)$. We show that $\rho_1 = \rho_2$. For the sake of contradiction, assume that $\rho_1 \neq \rho_2$ and let $j \leq n$ be the smallest position where both runs are different, that is, $(q_j, T_j^1) \neq (q_j, T_j^2)$. First, we know that $j \neq 0$ since $T_0^1 = T_0^2 = \varnothing$. Furthermore, we know by construction of $\mathcal{A}'$ that there exists $q_f^1, q_f^2 \in Q$ such that $q_f^1 \neq q_f^2$ and $(q, q_f^i) \in T_j^i$ for $i \in \{1, 2\}$ where $q = \delta(q_{j-1}, (a_j, 1))$. This means that $(\delta(q, w'), q_f^1) \in T_n^1$ and $(\delta(q, w'), q_f^2) \in T_n^2$ for $w' = (a_{j+1}, 0) \dots (a_n, 0)$. Given that $q_f^1 \neq q_f^2$, this leads us to a contradiction because we have that $F'(q_n, T_n^1) = \mathbb{0}$ or $F'(q_n, T_n^2) = \mathbb{0}$. Therefore, we conclude that $\rho_1 = \rho_2$.

With the previous claim, the proof is almost complete. Indeed, given the unique run $\rho = (q_0, \varnothing) \xrightarrow{a_1/s_1} \cdots \xrightarrow{a_n/s_n} (q_n, T_n)$ of $w = a_1 \dots a_n$, it is straightforward to prove that actually $[\![\theta]\!](w, \sigma[x \to i]) = s_i$ by the construction of $\mathcal{A}'$ and, thus:

$$[\![\Pi x.\, \theta]\!](w, \sigma) \quad = \quad \prod_{i=1}^{n} s_i \quad = \quad [\![\mathcal{A}']\!](w, \sigma)$$

■

We turn now to the proof of operators $\oplus$ and $\odot$ in the semiring level. It is well known that the addition and product of two weighted automata over a commutative semiring is also definable by weighted automata [14], [13]. These results basically follow the standard construction of the union and intersection in finite automata theory. For the sake of completeness and for future use, we sketch the proof of both operators in the next lemma.

Lemma A.5 ([13], [14])**.** *Let τ_1 and τ_2 be formulas in the semiring level such that V is the set of free variables and let $\mathcal{A}_1, \mathcal{A}_2$ be weighted automata defining the formulas. For $\tau = (\tau_1 \star \tau_2)$ with $\star \in \{\oplus, \odot\}$ there exists a weighted automaton $\mathcal{A}_\tau$ over $\mathbb{S}$ and Γ_V such that for every word $w \in \Gamma^*$ and (V, w)-assignment σ we have:*

$$[\![\tau]\!](w, \sigma) = [\![\mathcal{A}_\tau]\!](w, \sigma)$$

Proof: Let $\mathcal{A}_{\tau_1} = (\Gamma_V, \mathbb{S}, Q_1, E_1, I_1, F_1)$ and $\mathcal{A}_{\tau_2} = (\Gamma_V, \mathbb{S}, Q_2, E_2, I_2, F_2)$ with $Q_1 \cap Q_2 = \varnothing$ be the weighted automata that define the functions $[\![\tau_1]\!]$ and $[\![\tau_2]\!]$, respectively. For the function $\tau_1 \oplus \tau_2$, we define the weighted automata $\mathcal{A}_\oplus = (\Gamma_V, \mathbb{S}, Q_1 \cup Q_2, E_1 \cup E_2, I_1 \cup I_2, F_1 \cup F_2)$ where $\cup$ is used as the disjoint union of two functions. Thus, one can easily proof that $[\![\tau_1 \oplus \tau_2]\!](w, \sigma) = [\![\mathcal{A}_\oplus]\!](w, \sigma)$ for every $w \in \Gamma^*$ and (V, w)-assignment σ.

For the function $\tau_1 \odot \tau_2$, we define the weighted automata $\mathcal{A}_\odot = (\Gamma_V, \mathbb{S}, Q_1 \times Q_2, E_1 \times E_2, I_1 \times I_2, F_1 \times F_2)$ such that:

- $I_1 \times I_2(q_1, q_2) = I_1(q_1) \odot I_2(q_2)$ and $F_1 \times F_2(q_1, q_2) = F_1(q_1) \odot F_2(q_2)$ for all $(q_1, q_2) \in Q_1 \times Q_2$, and
- $E_1 \times E_2((p_1, p_2), a, (q_1, q_2)) = E(p_1, a, q_1) \odot E(p_2, a, q_2)$ for all $(p_1, p_2), (q_1, q_2) \in Q_1 \times Q_2$ and $a \in \Gamma_V$.

Similarly, one can easily proof that $[\![\tau_1 \odot \tau_2]\!](w, \sigma) = [\![\mathcal{A}_\odot]\!](w, \sigma)$ for every $w \in \Gamma^*$ and (V, w)-assignment σ. Finally, it is worth remarking that for this last construction it is crucial the fact that $\mathbb{S}$ is commutative. ■

The only operations left to prove is the construction of the automata for the sum first- and second-order quantifiers.

We first show how to translate an sum second-order quantifier over the semiring level. Let τ be a QIL-formula and V' the set of free set variables of τ. Let $X \in V'$ and let $V := V \setminus \{X\}$. Let $\mathcal{A} = (\Gamma_V \times \{0, 1\}, \mathbb{S}, Q, E, I, F)$ be the weighted automaton over $\mathbb{S}$ and $\Gamma_{V \cup \{X\}}$ that defines τ. We can define a new weighted automaton $\mathcal{A}' = (\Gamma_V, \mathbb{S}, Q, E', I, F)$ as the projection of $\Gamma_{V'}$ onto Γ_V such that $E'(p, a, q) = E(p, (a, 0), q) \oplus E(p, (a, 1), q)$ for every $p, q \in Q$ and $a \in \Gamma$.

Next, we show that for all $w \in \Gamma^*$ it holds that $[\![\Sigma X \tau]\!](w) = [\![\mathcal{A}']\!](w)$. Let $w = a_1 \dots a_n \in \Gamma$ and ρ be a run of $\mathcal{A}'$ over w. Then we have by definition that:

$$\begin{aligned}
|\rho| \quad &= \quad I(\rho_1^s) \odot \left(\prod_{i=1}^{n} E'(\rho_i^s, a_i, \rho_i^e) \right) \odot F(\rho_n^e) \\
&= \quad I(\rho_1^s) \odot \left(\prod_{i=1}^{n} E(\rho_i^s, (a_i, 0), \rho_i^e) \oplus E(\rho_i^s, (a_i, 1), \rho_i^e) \right) \odot F(\rho_n^e) \\
&= \quad \sum_{\sigma \in \{0,1\}^n} I(\rho_1^s) \odot \left(\prod_{i=1}^{n} E(\rho_i^s, (a_i, \sigma_i), \rho_i^e) \right) \odot F(\rho_n^e)
\end{aligned}$$

where the last equality holds by the distributivity law of $\mathbb{S}$. By the semantics of weighted automata we get that

$$[\![\mathcal{A}']\!](w) = \sum_{\rho \in \mathrm{Run}_{\mathcal{A}'}(w)} |\rho| = \sum_{\sigma \in \{0,1\}^n} \sum_{\rho \in \mathrm{Run}_{\mathcal{A}'}(w)} I(\rho_1^s) \odot \left(\prod_{i=1}^{n} E(\rho_i^s, (a_i, \sigma_i), \rho_i^e) \right) \odot F(\rho_n^e)$$

From the last equality we can see that the first sum ranges over all the $(\{X\}, V, w)$-assignments σ over $X \cup V$ and the second sum considers all the runs of $\mathcal{A}$ given σ, that is, the second sum is equal to $[\![\mathcal{A}]\!](w, \sigma)$. Therefore, we can conclude that $[\![\tau]\!](w) = [\![\mathcal{A}']\!](w)$ for all $w \in \Gamma^*$. This was to be shown.

The last step is to translate existential first-order quantification over the semiring. Let τ be a QIL-formula, x be a free first-order variable over the semiring and let V the set of the other free variables of τ. Let $\mathcal{A} = (\Gamma_V \times \{0,1\}, \mathbb{S}, Q, E, I, F)$ be the weighted automaton over $\mathbb{S}$ and $\Gamma_{V\cup\{x\}}$ that defines τ. The construction is essentially the same as for the second-order case. The only difference is that once the automaton has taken a transition corresponding to the position in x, it cannot guess a second position for x. Furthermore, the automaton has to at least guess one position for x. To achieve this, we double the number of states where the first copy is used before guessing the value of x and the second copy is used once x has been guessed.

Formally, let $Q' := \{q' \mid q \in Q\}$ be a disjoint copy of Q. We define a new weighted automaton $\mathcal{A}' = (\Gamma_V, \mathbb{S}, Q \cup Q', E', I', F')$ as follows. $I'(q) = I(q)$ if $q \in Q$ and $I(q) = \mathbb{0}$ if $q \in Q'$. $F'(q) = F(q)$ if $q \in Q'$ and otherwise $F(q) = \mathbb{0}$. Finally,

- $E'(p, a, q) = E(p, (a, 0), q)$ if $p, q \in Q$ and $a \in \Gamma$,
- $E'(p', a, q') = E(p, (a, 0), q)$ if $p', q' \in Q'$ and $a \in \Gamma$,
- $E'(p, a, q') = E(p, (a, 1), q)$ for every $p \in Q$, $q' \in Q'$ and $a \in \Gamma$ and
- $E'(p', a, q) = \mathbb{0}$ for all $p' \in Q'$ and $q \in Q$.

It is easily seen that $[\![\Sigma x.\, \tau]\!](w) = [\![\mathcal{A}']\!](w)$. ■

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.*

Proof: (1) This result follows easily from the effective procedure given in Theorem 4.1 to translate any formula $\theta \in \mathrm{QIL}[\mathbb{N}(+,\cdot), \Gamma]$ into a weighted automaton $\mathcal{A} = (\Gamma, \mathbb{N}(+,\cdot), Q, E, I, F)$. Clearly, if θ is of constant size, then $\mathcal{A}$ is also of constant size. Then for any input word $w \in \Gamma^*$ one can compute $[\![\mathcal{A}]\!](w)$ efficiently by running $\mathcal{A}$ over w [27]. This can be done in linear time with respect to $|w|$ by converting $\mathcal{A}$ into its matrix representation $(\{M_a\}_{a\in\Gamma}, i, f)$ such that $M_a \in \mathbb{N}^{Q\times Q}$ and $i, f \in \mathbb{N}^Q$. Then it can be shown that for every word $w = a_1 \dots a_n \in \Sigma^*$ it holds that $[\![\mathcal{A}]\!](w) = i \cdot M_{a_1} \cdot \ldots \cdot M_{a_n} \cdot f$. Given that $\mathcal{A}$ is of constant size then $(\{M_a\}_{a\in\Gamma}, i, f)$ is of constant size and the product of $i \cdot M_{a_1} \cdot \ldots \cdot M_{a_n} \cdot f$ can be done in time $O(n)$.

(2) We start showing that QIL-EVALUATION is in FPSPACE(poly) by computing the output over the parsing tree of the formula. Furthermore, we show that for every $\theta \in$ QIL and every $w \in \Gamma^*$ the size of the output $|[\![\theta]\!](w)|$ (i.e. the size of its binary codification) is in $O(|\theta| \cdot |w|)$. First, let φ be a formula in MSO and $w \in \Gamma^*$. Further, let V be the finite set of first-order and second-order variables and σ an (V, w)-assignment. Then we have $[\![\varphi]\!](w, \sigma) = 1$ iff $(w, \sigma) \models \varphi$. It is a folklore result that this problem is in PSPACE. Therefore, it can be computed with a polynomial space machine and the output is of size 1.

Consider θ_1, θ_2 be formulas in QIL. In addition, let V bet the set of free variables of θ_1, θ_2, and σ a (V, w)-assignment. Assume that $[\![\theta_1]\!](w, \sigma)$ and $[\![\theta_2]\!](w, \sigma)$ can be computed in polynomial space and the output is of size $c_1 \cdot |\theta_1| \cdot |w|$ and $c_2 \cdot |\theta_2| \cdot |w|$, respectively, for some constants c_1 and c_2. Furthermore, assume that if θ_1, θ_2 are in QMSO$(\oplus, \odot)$ then the output is of size $c_1 \cdot |\theta_1|$ and $c_2 \cdot |\theta_2|$, respectively. Next, we show by case analysis that QIL-EVALUATION(θ, w) is in FPSPACE(poly).

- If $\theta = \theta_1 \star \theta_2$ with $\star \in \{\oplus, \odot\}$, we can compute the output $[\![\theta]\!](w, \sigma)$ in polynomial space by first running the polynomial space machine for θ_1 over w and σ, and storing its polynomial size output o_1 in memory. Then we can run the polynomial space machine for θ_2 with polynomial size output o_2, and operate $o_1 \star o_2$. The complete process uses polynomial space and the output is polynomial size bounded:
$$|o_1 \star o_2| \;\leq\; c_1 \cdot |\theta_1| \cdot |w| + c_2 \cdot |\theta_2| \cdot |w| \;\leq\; \max\{c_1, c_2\} \cdot (|\theta_1| + |\theta_2|) \cdot |w|$$
For the specific case where θ_1 and θ_2 are in QMSO$(\oplus, \odot)$, it is straightforward to show that $|o_1 \star o_2| \leq \max\{c_1, c_2\} \cdot (|\theta_1| + |\theta_2|)$.
- If $\theta = \Pi x.\, \theta_1$ with $\theta_1 \in$ QMSO$(\oplus, \odot)$, we can compute $[\![\theta]\!](w, \sigma)$ in polynomial space by iterating over all positions of w. The output o is always stored in memory and for each $i \in \{1, \dots, |w|\}$, we run the polynomial space machine for $[\![\theta_1]\!](w, \sigma[x \to i])$ and updates the output to $o \cdot [\![\theta_1]\!](w, \sigma[x \to i])$. Given that o is of polynomial space this can always be stored in memory. Moreover, we know that $\theta_1 \in$ QMSO$(\oplus, \odot)$ and, thus,

we have:

$$|[\![\Pi x.\, \theta_1]\!](w,\sigma)| \;\leq\; \sum_{i=1}^{|w|} c_1 \cdot |\theta_1| \;\leq\; c_1 \cdot |\theta_1| \cdot |w|.$$

- If $\theta = \Sigma X.\, \theta_1$ with $\theta_1 \in$ QIL we compute the value $[\![\theta]\!](w,\sigma)$ in polynomial space by iterating over all subsets of $\{1,\ldots,|w|\}$ and maintaining the polynomial size output o in memory. For each $I \subseteq \{1,\ldots,|w|\}$, the polynomial space machine computes $[\![\theta_1]\!](w,\sigma[X \to I])$ and updates the output to $o + [\![\theta_1]\!](w,\sigma[X \to I])$. Notice that $[\![\Sigma X.\, \theta_1]\!](w,\sigma) \leq 2^{|w|} \cdot v$ for $v = \max_{I \subseteq \{1,\ldots,|w|\}}([\![\theta_1]\!](w,\sigma[X \to I]))$. Therefore, the complete process runs in polynomial space and we have:

$$|[\![\Sigma X.\, \theta_1]\!](w,\sigma)| \;\leq\; \log(2^{|w|} \cdot v) \;\leq\; \log(2) \cdot |w| + c_1 \cdot |\theta_1| \cdot |w| \;\leq\; c_1 \cdot (|\theta_1| + 1) \cdot |w|.$$

We conclude that QIL-EVALUATION is in FPSPACE(poly).

In the rest of the proof, we show that the problem is FPSPACE(poly)-hard. Towards this goal, we consider the parsimonious reductions used in [26]: a function f is reducible to g in polynomial time ($f \leq^P g$), if there is a function h that is computable in polynomial time such that $f(x) = g(h(x))$ for all input x. We say that a function g is *complete* for a class of functions C if $g \in C$ and $f \leq^P g$ for all $f \in C$.

A quantified boolean formula (QBF) is a formula of the form:

$$\alpha(y_1,\ldots,y_n) = \exists \vec{x}_1.\, \forall \vec{x}_2.\, \ldots \exists \vec{x}_{n-1}.\, \forall \vec{x}_n.\, \beta(\vec{x}_1,\ldots,\vec{x}_n,y_1,\ldots,y_m)$$

where y_i is a free boolean variable for $i \in \{1,\ldots,n\}$, $\vec{x}_j$ is a vector of boolean variables for $j \in \{1,\ldots,n\}$, and β is a boolean formula. Given a boolean assignment $\sigma : \{y_1,\ldots,y_n\} \to \{\text{true},\text{false}\}$, we say that σ satisfies α (i.e. $\sigma \models \alpha$) if $\alpha(\sigma(y_1),\ldots,\sigma(y_n))$ is true. We denote by $\text{Assign}(y_1,\ldots,y_n)$ the set of all assignments of $y_1,\ldots,y_n$ into $\{\text{true},\text{false}\}$. Consider the function $\natural QBF$ such that:

$$\natural QBF(\alpha) \;=\; |\{\sigma \in \text{Assign}(y_1,\ldots,y_n) \mid \sigma \models \alpha\}|$$

for every quantified boolean formula α as above. In [26], it was shown that $\natural QBF$ is complete for FPSPACE(poly).

Next, we show how to reduce $\natural QBF$ to QIL-EVALUATION. Let α be a formula as above and $\Gamma = \{a,b\}$. Take α and build a formula $\varphi(y_1,\ldots,y_n) \in$ MSO similar to α by changing x to $P_a(x)$ each time that x appears positively, and $\neg x$ to $P_b(x)$ each time that x appears negatively for all variables x in β. Then define the formula $\theta \in$ QIL as follows:

$$\theta \;=\; \Sigma y_1.\, \ldots \Sigma y_n.\, \varphi(y_1,\ldots,y_n)$$

One can easily show that for $w = ab$ it holds that $\natural QBF(\alpha) = [\![\theta]\!](w)$ which proves that QIL-EVALUATION is FPSPACE(poly)-hard. Thus, we conclude that QIL-EVALUATION is FPSPACE(poly)-complete.

■

Proposition 4.3. *The following problems are undecidable:*

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

Proof: (1) We show this result by an easy reduction from a variant of the Diophantine equation problem [12]. More specific, let $p(x_1,\ldots,x_k)$ be a polynomial in k variables with integer coefficients. The equation $p(x_1,\ldots,x_k) = 0$ is called a *Diophantine equation* when only integer solutions are allowed. Then the problem of determining whether there exists a non-negative integer solution to such an equation is well-known to be undecidable [12]. Furthermore, for any two polynomials $p_1(x_1,\ldots,x_k)$ and $p_2(x_1,\ldots,x_k)$ in k variables with non-negative coefficients one can easily derived that there exists no procedure to decide if for all $n_1,\ldots,n_k \in \mathbb{N}$ it holds that:

$$p_1(n_1,\ldots,n_k) \;\leq\; p_2(n_1,\ldots,n_k)$$

We use this last problem to show that containment of formulas in $\text{QMSO}(\Sigma_x\Pi_x^1,\oplus,\odot)$ over $\mathbb{N}(+,\cdot)$ is undecidable. Towards this goal, let $p(x_1,\ldots,x_k)$ be a polynomial in k variables of the form:

$$p(x_1,\ldots,x_k) \;=\; \sum_{i=1}^{m} v_i \cdot x_1^{c(i,1)} \cdot \ldots \cdot x_k^{c(i,k)}$$

where $v_i \in \mathbb{N}$ and $c_{(i,j)} \in \mathbb{N}$ for all $i \leq m$ and $j \leq k$. Further, let $\Gamma^* = \{a_1, \ldots, a_k\}$ be a finite alphabet of k different letters. We show next how to construct a formula $\theta_p \in \mathrm{QMSO}(\Sigma_x \Pi_x^1, \oplus, \odot)$ over $\mathbb{N}(+,\cdot)$ such that for every word $w \in \Gamma^*$ it holds that:

$$p(|w|_{a_1}, \ldots, |w|_{a_1}) \;=\; [\![\theta_p]\!](w) \tag{4}$$

where $|w|_{a_i}$ denotes the number of a_i's in w. Then from the construction of θ_p (below) and Equation (4), the undecidability of the containment of formulas in $\mathrm{QMSO}(\Sigma_x \Pi_x^1, \oplus, \odot)$ over $\mathbb{N}(+,\cdot)$ easily follows and we left this to the reader.

Then it only left to show the construction of θ_p. For any variable $x_i \in \{x_1, \ldots, x_k\}$ consider the formula in $\mathrm{QMSO}(\Sigma_x \Pi_x^1, \oplus, \odot)$ over $\mathbb{N}(+,\cdot)$:

$$\tau_{x_i} := \Sigma z.\, P_{a_i}(z)$$

Basically, this formula counts the number of a_i-letters in a word over Γ (i.e $[\![\tau_{x_i}]\!](w) = |w|_{a_i}$). Further, for any $c \in \mathbb{N}$ consider the formula $\tau_{x_i}^c = \prod_{i=1}^c \tau_{x_i}$. Clearly, this formula is also in $\mathrm{QMSO}(\Sigma_x \Pi_x^1, \oplus, \odot)$. With all these ingredients we can easily define the formula θ_p as follows:

$$\theta_p \;=\; \sum_{i=1}^{m} v_i \cdot \tau_{x_1}^{c_{(i,1)}} \cdot \ldots \cdot \tau_{x_k}^{c_{(i,k)}}$$

From the definition of $p(x_1, \ldots, x_k)$ and θ_p it is straightforward to show that Equation (4) holds and the undecidability of (1) is shown.

(2) These results follows easily from Theorem 5.4 and the undecidability of the equivalence problem between polynomial ambiguous distance automata [1] (i.e. weighted automata over $\mathbb{N}_\infty(\min,+)$). ■

B. *Proofs Omitted in Section V*

Depending on restrictions imposed on the amount of "ambiguity" allowed in the definition of weighted automata, one can capture different classes of functions over words [24]. 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 well known that there exists a weighted automaton that cannot be defined as a deterministic weighted automaton (DWA). Furthermore, *unambiguous weighted automata* (*unamb*- WA), *finitely ambiguous weighted automata* (fin- WA), and *polynomially ambiguous weighted automata* (*poly*- WA) are different classes of weighted automata which define different classes of functions over words. In [24], it is shown that these classes are strictly contained in the following way:

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

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 the class with a subset of QIL.

C. *Deterministic weighted automata*

It is well known that deterministic weighted automata are less expressive than weighted automata. As an example, let $\Gamma = \{a\}$ and let $f_1 : \Gamma^* \to \mathbb{N}_\infty$ be the function such that $f_1(w) = |w|$ whenever w is of even length and $f_1(w) = 0$ otherwise. Figure 1 shows the distance (i.e. $\mathbb{N}_\infty(\min,+)$) automaton that defines f_1.

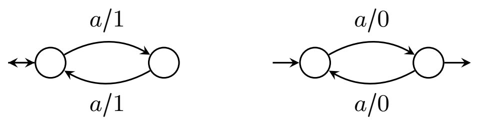


Figure 1: An unambiguous automaton for f_1.

In [24], it is shown that f_1 cannot be defined by any deterministic weighted automata over the min-plus semiring. Therefore, DWA forms a proper subclass inside weighted automata. Surprisingly, this class can be characterized by a subclass of QIL.

Theorem 5.1. *Let $\mathbb{S}$ be a commutative semi-ring and Γ a finite alphabet. A function $f : \Gamma^* \to \mathbb{S}$ is definable by a deterministic weighted automaton over $\mathbb{S}$ and Γ if, and only if, f is definable by a formula in* $\mathrm{QMSO}_{\mathbb{S},\Gamma}(^{\rightarrow}, \oplus_b, \odot)$. *That is,*

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

Proof: ($\Rightarrow$) Let $\mathcal{A} = (\Gamma, \mathbb{S}, Q, \delta, q_0, F)$ be a deterministic weighted automaton that defines the function $f : \Gamma^* \to \mathbb{S}$. This direction works similar to the proof of Theorem 4.1, but here we use the fact that $\mathcal{A}$ is deterministic: each time that we guess a run inside an MSO-formula, we know that this run will be unique by the determinism of $\mathcal{A}$. For this reason, we do not need to use external set quantification in our formula, or the Π-quantification. Furthermore, in order to aggregate the weights of the transitions it is enough to use the forward operator which tell us which is the last transition in order to add its weight. Next, we construct a formula $\tau_{\mathcal{A}} \in \mathrm{QMSO}(^{\rightarrow}, \oplus_b, \odot)$ such that $[\![\mathcal{A}]\!](w) = [\![\tau_{\mathcal{A}}]\!](w)$ for every $w \in \Gamma^*$.

Similar to proof of Theorem 4.1, we define the set of non-zero transitions T of $\mathcal{A}$ and the enumeration $\bar{X} = (X_1, \dots, X_n)$ of variables in set $\{X_t \mid t \in T\}$. Furthermore, in this proof we reuse formula $\mathrm{run}_{\mathcal{A}}(\bar{X})$ of the proof of Theorem 4.1 that checks if variables $\bar{X}$ encode an accepting run of $\mathcal{A}$ over a word $w \in \Gamma$. In particular, given that $\mathcal{A}$ is deterministic, we know that whenever $\bar{X}$ encodes a run, this must be unique.

In the following equations, we define the formulas for the weights of the transition and final function:

$$\begin{aligned}
\text{det-transition} \;&:=\; \sum_{(p,a,q)\in T} \big(\exists \bar{X}.\ \mathrm{run}_{\mathcal{A}}(\bar{X}) \ \wedge\ \big(\exists x.\ \mathrm{last}(x) \ \wedge\ x \in X_{(p,a,q)}\big)\big) \odot \delta(p,a,q) \\
\text{det-final} \;&:=\; ((\forall x.\, x \notin x) \odot F(q_0)) \oplus \\
&\quad \sum_{p\in Q} \left(\exists \bar{X}.\ \mathrm{run}_{\mathcal{A}}(\bar{X}) \ \wedge\ \left(\exists x.\ \mathrm{last}(x) \ \wedge \bigvee_{q\in Q, a\in\Gamma} x \in X_{(q,a,p)} \right)\right) \odot F(p)
\end{aligned}$$

Notice that at the beginning of det-final we are considering the case when the input word is equal to ϵ. To conclude, we define the formula $\tau_{\mathcal{A}}$ in $\mathrm{QMSO}(^{\rightarrow}, \oplus_b, \odot)$ as follows:

$$\tau_{\mathcal{A}} \;\; := \;\; (\text{det-transition})^{\rightarrow} \;\odot\; \text{det-final}$$

One can easily check that $\mathcal{A}$ and $\tau_{\mathcal{A}}$ define the same function.

($\Leftarrow$) Similar to the proof of Theorem 4.1, we show inductively on the syntax of a formula $\tau \in \mathrm{QMSO}(^{\rightarrow}, \oplus_b, \odot)$ how to construct a deterministic weighted automata $\mathcal{A}_\tau$. Actually, we can reuse some of the work of Theorem 4.1 by noticing that the construction of a weighed automaton for formulas without semi-ring quantification produces a deterministic automaton (Lemma A.2) and the construction of the product $\odot$ (Lemma A.5) preserves this determinism. Then it is only left to show how to construct a deterministic weighted automaton from a formula $(\theta)^{\rightarrow}$. The following lemma shows how to do this last step.

Lemma A.6. *Let θ be a formula without any sum or product quantification. For $\tau = (\theta)^{\rightarrow}$ there exists a deterministic weighted automaton $\mathcal{A}_\tau$ over $\mathbb{S}$ and Γ such that for every word $w \in \Gamma^*$ we have:*

$$[\![\tau]\!](w) = [\![\mathcal{A}_\tau]\!](w)$$

Let $\mathcal{A} = (\Gamma, \mathbb{S}, Q, \delta, q_0, F)$ be the deterministic weighted automata of Lemma A.2 that defines the characteristic function $[\![\theta]\!]$. By the construction of $\mathcal{A}$, we know that δ is a function from $Q \times \Gamma$ into $Q \times \{\mathbb{1}\}$. As a consequence, we use δ as a function from $Q \times \Gamma$ into Q in order to simplify the notation.

We define the deterministic weighted automaton $\mathcal{A}^{\rightarrow} = (\Gamma, \mathbb{S}, Q, \delta', q_0, F')$ such that $\delta'(q,a) = (\delta(q,a), F(\delta(q,a)))$ and $F'(q) = \mathbb{1}$ for every $q \in Q$ and $a \in \Gamma$. Clearly, $\mathcal{A}^{\rightarrow}$ is deterministic and computes the function $(\theta)^{\rightarrow}$. In fact, given the unique run $\rho = q_0 \xrightarrow{a_1/s_1} q_1 \xrightarrow{a_2/s_2} \cdots \xrightarrow{a_n/s_n} q_n$ of $\mathcal{A}^{\rightarrow}$ over a word $w = a_1 \dots a_n$, we have that $s_i = F(q_i)$. This means that each transition $\xrightarrow{a_i/s_i}$ computes the weight in $\mathcal{A}$ of a prefix $w[1..i]$, that is, $[\![\theta]\!](w[1..i]) = s_i$. Then we have that:

$$[\![\theta^{\rightarrow}]\!](w) \;\; = \;\; \prod_{i=1}^{n} [\![\theta]\!](w[1..i]) \;\; = \;\; \prod_{i=1}^{n} s_i \;\; = \;\; |\rho|$$

This was to be shown. ■

Compared to the Π-operator, the construction of the automaton for the $(\cdot)^{\rightarrow}$-operator is very simple. In particular, it is linear with respect to the size of the weighted automaton for a characteristic function, in contrast to the exponential blow-up in the construction for Π.

Another interesting class of weighted automata are *co-deterministic* weighted automata. 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. That is, 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 at most one state $q_f \in Q$ such that $F(q_f) \neq \mathbb{0}$. Co-deterministic weighted automata (*co*-DWA) form a disjunct subclass with respect to deterministic weighted automata. For example, consider the function $f_2 : \Gamma^* \to \mathbb{N}_\infty$ with $\Gamma = \{a, b, c\}$ such that $f_2(c^n \cdot a) = n$, $f_2(c^n \cdot b) = 0$, and $f_2(w) = \infty$ otherwise. One can prove that f_2 is definable by a co-deterministic automata like the one of Figure 2. However, there does not exist a deterministic weighted automata that can specify f_2.

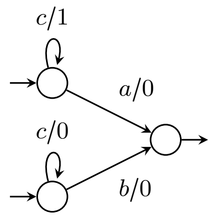


Figure 2: Co-deterministic weighted automata for function f_2.

The following theorem shows that co-deterministic weighted automata can also be characterized by a subset of the operators of QMSO.

Theorem 5.2. *Let $\mathbb{S}$ be a commutative semi-ring and Γ a finite alphabet. A function $f : \Gamma^* \to \mathbb{S}$ is definable by a co-deterministic weighted automaton over $\mathbb{S}$ and Γ if, and only if, f is definable by a formula in $\mathrm{QMSO}_{\mathbb{S},\Gamma}(^{\leftarrow}, \oplus_b, \odot)$. That is,*

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

Proof: The proof is similar to the proof of Theorem 5.1. The only difference is that we have to apply the same ideas in a backward manner and use the co-determinism of the weighted automaton. We leave this proof to the reader. ∎

D. Unambiguous and finitely ambiguous weighted automata

We say that a weighted automaton $\mathcal{A} = (\Gamma, \mathbb{S}, Q, E, I, F)$ is *unambiguous* if $|\mathrm{Run}_{\mathcal{A}}(w)| \leq 1$ for every word $w \in \Gamma^*$, 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 $|\mathrm{Run}_{\mathcal{A}}(w)| \leq N$ for every word $w \in \Gamma^*$. 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 other proper subclasses of weighted automata. For example, the weighted automaton of Figure 1 is unambiguous. Furthermore, in [24] it is shown that unambiguous automata form a proper subclass of finitely ambiguous automata and, also, that finitely ambiguous automata form a proper subclass inside weighted automata. As before, the expressiveness of both classes can be captured if we consider a subset of the operators of QMSO.

Theorem 5.3. *Let $\mathbb{S}$ be a commutative semiring and Γ a finite alphabet. A function $f : \Gamma^* \to \mathbb{S}$ is definable by an unambiguous (resp. finitely ambiguous) weighted automaton over $\mathbb{S}$ and Γ if, and only if, f is definable by a formula in $\mathrm{QMSO}_{\mathbb{S},\Gamma}(\Pi^1_x, \oplus_b, \odot)$ (resp. $\mathrm{QMSO}_{\mathbb{S},\Gamma}(\Pi^1_x, \oplus, \odot)$). That is,*

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

First we prove the unambiguous case to then follow to finitely ambiguous case, which is a corollary of the former and the results in [31], [24].

Proof: ($\Rightarrow$) Let $\mathcal{A} = (\Gamma, \mathbb{S}, Q, E, I, F)$ be an unambiguous weighted automaton that defines the function $f : \Gamma^* \to \mathbb{S}$. As before, this proof follows the same ideas of Theorem 4.1 and Theorem 5.1. Similar to the deterministic case, there exists at most one accepting run of $\mathcal{A}$ for each word in Γ^*. Therefore, we can use the same technique in Theorem 5.1 of guessing for each word a unique run and then use the Π-operator in order to collect all the weights of this run. Next, we show how to define a formula $\tau_{\mathcal{A}} \in \text{QMSO}(\Pi^1_x, \oplus_b, \odot)$ such that $[\![\mathcal{A}]\!](w) = [\![\tau_{\mathcal{A}}]\!](w)$ for every $w \in \Gamma^*$.

As before (Theorem 4.1), we define the set of non-zero transitions T of $\mathcal{A}$, the set of initial and final transitions T_I and T_F, and the enumeration $\bar{X} = (X_1, \dots, X_n)$ of variables in set $\{X_t \mid t \in T\}$. Further, we reuse formula $\text{run}_{\mathcal{A}}(\bar{X})$ of the proof of Theorem 4.1 that checks if variables $\bar{X}$ encode an accepting run of $\mathcal{A}$ over a word $w \in \Gamma$. In particular, given that $\mathcal{A}$ is unambiguous, we know that whenever $\bar{X}$ encodes a run, this must be unique.

Next, we define formulas to collect the transition, initial and final function weights:

$$\begin{aligned}
\text{unamb-transition}(x) &:= \sum_{(p,a,q)\in T} \big(\exists \bar{X}.\ \text{run}_{\mathcal{A}}(\bar{X}) \wedge x \in X_{(p,a,q)}\big) \odot E(p,a,q) \\
\text{unamb-initial} &:= \left((\forall x.\, x \nleq x) \odot \sum_{q\in Q} I(q)\right) \oplus \\
&\quad\ \sum_{(p,a,q)\in T_I} \big(\exists \bar{X}.\ \text{run}_{\mathcal{A}}(\bar{X}) \wedge \big(\exists x.\ \text{first}(x) \wedge x \in X_{(p,a,q)}\big)\big) \odot I(p) \\
\text{unamb-final} &:= \left((\forall x.\, x \nleq x) \odot \sum_{q\in Q} F(q)\right) \oplus \\
&\quad\ \sum_{(p,a,q)\in T_F} \big(\exists \bar{X}.\ \text{run}_{\mathcal{A}}(\bar{X}) \wedge \big(\exists x.\ \text{last}(x) \wedge x \in X_{(p,a,q)}\big)\big) \odot F(q)
\end{aligned}$$

In formula $\text{unamb-transition}(x)$, x is used to iterate over the transitions of the unique run encoded in $\bar{X}$ and collected the weight of each transition by using the Π-operator. Notice also that we are considering the case when the input word is equal to ϵ at the beginning of unamb-initial and unamb-final. To conclude, we define the formula $\tau_{\mathcal{A}}$ in $\text{QMSO}(\Pi^1_x, \oplus_b, \odot)$ as follows:

$$\tau_{\mathcal{A}} \ := \ \text{unamb-initial} \odot (\Pi x.\ \text{unamb-transition}(x)) \odot \text{unamb-final}$$

Similar to previous proofs, it is easily to check that $\mathcal{A}$ and $\tau_{\mathcal{A}}$ define the same function.

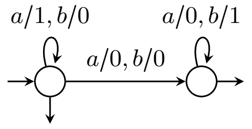
($\Leftarrow$) For this direction, all the work was done in the proof of Theorem 4.1 given that all constructions of operators in $\text{QMSO}(\Pi^1_x, \oplus_b, \odot)$ preserve unambiguity of weighted automata. First, a formula θ on the semiring level can be defined by a deterministic weighted automaton as was shown in Lemma A.2. Second, given two unambiguous weighted automata $\mathcal{A}_1$ and $\mathcal{A}_2$ one can easily see that the product construction $\mathcal{A}_1 \odot \mathcal{A}_2$ of Lemma A.5 is also unambiguous. Finally, for the Π-operator we prove in Claim A.4 that for every word $w \in \Gamma$ there exists at most one valid run in the automata $\mathcal{A}'$ where $\mathcal{A}'$ define $\Pi x.\ \theta(x)$. This means that $\mathcal{A}'$ is unambiguous and, therefore, the construction of Lemma A.3 produces only unambiguous weighted automata. Therefore, we conclude from the proof of Theorem 4.1 that for every formula $\tau \in \text{QMSO}(\Pi^1_x, \oplus_b, \odot)$ there exists an unambiguous weighted automata $\mathcal{A}_\tau$ such that $[\![\varphi]\!](w) = [\![\mathcal{A}]\!](w)$ for every $w \in \Gamma^*$.

Now, we consider the finitely ambiguous case. The following lemma is crucial for this proof.

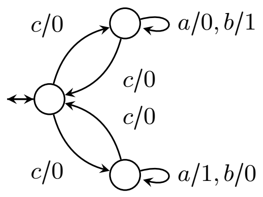
Lemma A.7. *[24], [31] Given a semiring $\mathbb{S}$, every finitely ambiguous weighted automata can be defined by the disjoint union of a finite number of unambiguous weighted automata. That is, for every finitely ambiguous weighted automata $\mathcal{A}$ over $\mathbb{S}$ and Γ, there exists a finite number of unambiguous automata $\mathcal{A}_1, \dots, \mathcal{A}_n$ such that for every $w \in \Gamma^*$:*

$$[\![\mathcal{A}]\!](w) = [\![\mathcal{A}_1 \oplus \cdots \oplus \mathcal{A}_n]\!](w)$$

By using this lemma and the equivalence between unambiguous weighted automata and $\text{QMSO}(\Pi^1_x, \oplus_b, \odot)$, we can easily show that fin-WA $= \text{QMSO}(\Pi^1_x, \oplus, \odot)$. To prove the direction from fin-WA to $\text{QMSO}(\Pi^1_x, \oplus, \odot)$, one has to translate a finitely ambiguous weighted automaton into the finite set of unambiguous weighted automata, translate each automata into a formula in $\text{QMSO}(\Pi^1_x, \oplus_b, \odot)$ and then take the addition $\oplus$ of all formulas. One can easily prove that the resulting formula is equal to the initial finitely ambiguous weighted automaton. For the other



(a) Polynomially ambiguous



(b) Non-polynomially ambiguous

Figure 3: Example of polynomially and non-polynomially ambiguous weighted automata

direction, one simply has to notice that the construction of the $\oplus$-operator (Lemma A.5) between finitely ambiguous weighted automata preserves finite ambiguity. Similarly, Π-operator and $\odot$-operator also preserves ambiguity and this shows that $\mathrm{QMSO}(\Pi^1_x, \oplus, \odot) \subseteq fin\text{-}\mathrm{WA}$. ■

E. Polynomially ambiguous weighted automata

We now introduce another operator, the $/$-operator, which does not change the expressive power of QMSO but is often very useful in this section. Formally, if $\varphi, \psi \in \mathrm{QMSO}$ over a semi-ring $\mathbb{S}$ then we can build the formula $(\varphi \,/\, \psi)$. The semantics is defined as

$$[\![(\varphi \,/\, \psi)]\!](w, \sigma) \coloneqq \bigoplus_{w = w_1 w_2} \big([\![\varphi]\!](w_1, \sigma) \odot [\![\psi]\!](w_2, \sigma)\big)$$

that is, we sum over all ways in which the word w can be split into two parts $w_1 w_2$ and then evaluate the formulas φ, ψ on the two parts respectively, taking the product of the results.

Example A.8. *Let $\mathbb{N}_\infty(\min, +)$ be the min-plus semiring and $\Gamma = \{a, b\}$. We want to define the length of the minimum sequence of a's between two b's. Using the $/$-operator, we can express this as follows.*

$$\mathrm{last}_b \;/\; \Sigma x.\,(P_a(x) + 1) \;/\; \mathrm{first}_b$$

where $\mathrm{first}_b, \mathrm{last}_b$ are first-order formulas defining that the first and last position of the word is equal to b.

Obviously, the $/$-operator can easily be expressed using the first order sum operator. Somewhat less obviously, the converse is often true as well, in particular the logic QIL below capturing weighted automata can equally be defined using the $/$-operator instead of Σ. However, a direct translation from Σ-formulas into $/$-formulas is difficult and we refrain from giving an explicit construction at this point.

A weighted automata $\mathcal{A} = (\Gamma, \mathbb{S}, Q, E, I, F)$ is called *polynomially ambiguous* if there exists a polynomial $p(x)$ such that $|\mathrm{Run}_{\mathcal{A}}(w)| \leq p(|w|)$ for every word $w \in \Gamma^*$. Polynomially ambiguous weighted automata ($poly\text{-}\mathrm{WA}$) were studied in [24], [23] and it was shown that they constitute another proper subclass in the family of weighted automata. Further, in [23] an algorithm for deciding whether a polynomially ambiguous weighted automata is determinizable was given. For example, Figure 3a shows a weighted automata over the min-plus semiring that is polynomially ambiguous but it cannot be defined with finite ambiguity. Interestingly, this function can easily be defined by the formula:

$$(\min\{\mathrm{end}_a + 1, \mathrm{end}_b\})^{\rightarrow} \,/\, (\min\{\mathrm{end}_a, \mathrm{end}_b + 1\})^{\rightarrow}.$$

where $\mathrm{end}_d \coloneqq \exists x.\ \mathrm{last}(x) \wedge P_d(x)$, that is, the word ends with letter d for $d \in \{a, b\}$. Furthermore, Figure 3b shows a weighted automaton over the min-plus semiring that cannot be specified with a weighted automaton by using only polynomially ambiguity. This shows that $poly\text{-}\mathrm{WA}$ forms another proper subclass inside the family of weighted automata.

Polynomially weighted automata can also be captured by a subset of QMSO as follow.

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

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

Proof: ($\Rightarrow$) We start by introducing some notation in order to simplify the presentation of this proof. Let $\mathcal{A} = (\Gamma, \mathbb{S}, Q, E, I, F)$ be a polynomial ambiguous automaton and q a state in $\mathcal{A}$. We denote by:

- $e_q : Q \to \mathbb{S}$ the $\mathbb{1}$-function on q (i.e. $e_q(q) = \mathbb{1}$ and $\mathbb{0}$ otherwise),
- $E_q^{\to} : Q \times \Gamma \times Q \to \mathbb{S}$ the restriction of E without transitions starting from q (i.e. $E_q^{\to}(p, a, r) = E(p, a, r)$ whenever $p \neq q$ and $\mathbb{0}$ otherwise),
- $E_q^{\leftarrow} : Q \times \Gamma \times Q \to \mathbb{S}$ the restriction of E without transitions going into q (i.e. $E_q^{\leftarrow}(p, a, r) = E(p, a, r)$ whenever $r \neq q$ and $\mathbb{0}$ otherwise), and
- for every function $f : Q \to \mathbb{S}$ (or $f : Q \times \Gamma \times Q \to \mathbb{S}$) we denote by f_{-q} the restriction of f on the set $Q - \{q\}$ ($(Q - \{q\}) \times \Gamma \times (Q - \{q\})$ resp.).

Furthermore, we say that a state $q \in Q$ has a *cycle* in $\mathcal{A}$ if there exists a word $w \neq \epsilon$ over Γ and a run of $\mathcal{A}$ over w that starting from q reaches q again. In other words, q has a cycle in Q if there exists a cycle on q when $\mathcal{A}$ is seen as a graph.

With this notation, we define the following different restrictions of $\mathcal{A}$:

- $\mathcal{A}_{-q} = (\Gamma, \mathbb{S}, Q - \{q\}, E_{-q}, I_{-q}, F_{-q})$ is the weighted automata $\mathcal{A}$ restricted to the set $Q - \{q\}$,
- $\mathcal{A}_q = (\Gamma, \mathbb{S}, Q, E, e_q, e_q)$ is the weighted automata $\mathcal{A}$ where the initial and final states are restricted to q,
- $\mathcal{A}_q^I = (\Gamma, \mathbb{S}, Q, E_q^{\to}, I, e_q)$ is the weighted automata $\mathcal{A}$ where final states are restricted to q and $\mathcal{A}_q^I$ does not have transitions starting from q, and
- $\mathcal{A}_q^F = (\Gamma, \mathbb{S}, Q, E_q^{\leftarrow}, e_q, F)$ is the weighted automata $\mathcal{A}_q^F$ where initial states are restricted to q and $\mathcal{A}$ does not have transitions reaching q.

Notice that if $\mathcal{A}$ is polynomially ambiguous, then the weighted automata $\mathcal{A}_{-q}$, $\mathcal{A}_q$, $\mathcal{A}_q^I$, and $\mathcal{A}_q^F$ are also polynomially ambiguous. Moreover, for every $q \in Q$ it holds that q does not have cycles neither in $\mathcal{A}_q^I$ nor in $\mathcal{A}_q^F$.

The following lemma is a useful characterization of polynomial ambiguity.

Lemma A.9. *[23] A weighted automata $\mathcal{A} = (\Gamma, \mathbb{S}, Q, E, I, F)$ is polynomially ambiguous if, and only if, for every state q and every $w \in \Gamma^*$ there is at most one path for w from q to q, that is, if $\mathcal{A}_q$ is unambiguous for every state $q \in Q$.*

Now, we present how to construct a formula $\tau \in \mathrm{QMSO}(\Sigma_x \Pi_x^1, \oplus, \odot)$ recursively over $\mathcal{A}$ by using Lemma A.9, Theorem 5.3, and $\,/\,$-operator. Given an unambiguous automata $\mathcal{A}$, we denote by $\mu(\mathcal{A})$ the $\mathrm{QMSO}(\Pi_x^1, \oplus_b, \odot)$-formula constructed in Theorem 5.3 that defines the behaviour of $\mathcal{A}$. Consider a recursively procedure $\pi(\cdot)$ over a polynomially ambiguous weighted automata $\mathcal{A}$ defined as follow. First, pick any state $q \in Q$ that has a cycle in $\mathcal{A}$. If q does not exist, then $\mathcal{A}$ is *acyclic* and, thus, definable by an unambiguous weighted automata $\mathcal{A}'$ such that $\pi(\mathcal{A}) = \mu(\mathcal{A}')$. Otherwise, do the following procedure:

- If $Q = \{q\}$, then $\pi(\mathcal{A}) = \mu(\mathcal{A})$. This is correct because a weighted automata with only one state is always unambiguous.
- If $Q \neq \{q\}$, then:
$$\pi(\mathcal{A}) = \pi(\mathcal{A}_{-q}) \;\oplus\; \big(\, \pi(\mathcal{A}_q^I) \;/\; \mu(\mathcal{A}_q) \;/\; \pi(\mathcal{A}_q^F) \,\big)$$

First of all, it is straightforward to check that the recursive definition of π over a polynomially ambiguous weighted automata $\mathcal{A}$ is correct given that $\mathcal{A}_q$ is always unambiguous (Lemma A.9) and $\mathcal{A}_{-q}$, $\mathcal{A}_q^I$, and $\mathcal{A}_q^F$ are also polynomial ambiguous. Second, $\mathcal{A}_{-q}$ has less state than $\mathcal{A}$. Moreover, q has no cycles in $\mathcal{A}_q^I$ and $\mathcal{A}_q^F$ and, thus, the induction holds. Third, the formula defined by π is in $\mathrm{QMSO}(\Sigma_x \Pi_x^1, \oplus, \odot)$ given that the definition only uses the $\,/\,$-operator and $\oplus$-operator, and μ always return a formula in $\mathrm{QMSO}(\Pi_x^1, \oplus_b, \odot)$. Finally, it is easy to check by induction over the set of states that $[\![\pi(\mathcal{A})]\!](w) = [\![\mathcal{A}]\!](w)$ for any word $w \in \Gamma^*$.

($\Leftarrow$) This direction is trivial by the proof of Theorem 4.1. One only need to notice that the construction of the first-order sum operator preserves polynomial ambiguity. ■

F. Proofs Omitted in Section VI-A

Proposition 6.1. *Let Γ be a finite alphabet and $\mathbb{S}$ be a commutative semiring. The following classes of* PNWA *and sub-fragments of* QMSO *are equally expressive over Γ and $\mathbb{S}$:*

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

Proof: In this proof we show that PNWA $\equiv$ QMSO$(\Sigma_X, \oplus, \odot_b)$. This proof is a simplified version of the proof of Theorem 4.1. First, consider the direction from a pure-nondeterministic weighted automaton $\mathcal{A} = (\Gamma, \mathbb{S}, \delta, I, F)$ to a formula in QMSO$(\Sigma_X, \oplus, \odot_b)$. Recall from the proof of Theorem 4.1 the formulas $\mathrm{run}_{\mathcal{A}}(\bar{X})$, $\mathrm{final}(\bar{X})$, and empty. From these three formulas, we have to only redefine the formula empty into:

$$\mathrm{empty} \;:=\; (\forall x.\, x \notin x) \odot \left(\sum_{q \in I} F(q) \right)$$

Then we can define a formula $\theta_{\mathcal{A}} \in$ QMSO$(\Sigma_X, \oplus, \odot_b)$ as follow:

$$\theta_{\mathcal{A}} \;:=\; \Sigma \bar{X}.\, \big(\mathrm{run}_{\mathcal{A}}(\bar{X}) \,\odot\, \mathrm{final}(\bar{X})\big) \,\oplus\, \mathrm{empty}$$

It is easily checked that $\mathcal{A}$ and $\theta_{\mathcal{A}}$ define the same function.

For the other direction, the proof also follows the same lines than Theorem 4.1. First, notice that for a formula without semiring quantification in QMSO$(\Sigma_X, \oplus, \odot_b)$, Lemma A.2 produces a pure non-deterministic weighted automata. Specifically, all the transitions of the output automata of Lemma A.2 has cost equal to $\mathbb{1}$ and it is straightforward to show how to convert this automata into a pure non-deterministic weighted automata. Finally, for the formula $\theta_1 \oplus \theta_2$ or $\Sigma X.\, \theta(X)$ one can easily obtain a pure non-deterministic weighted automaton from the proof of Theorem 4.1.

For the proof of *poly*-PNWA $\equiv$ QMSO$(\Sigma_x, \oplus, \odot_b)$, we detour the reader to the proof of Theorem 6.2. ■

Theorem 6.2. *Let Γ be a finite alphabet and $\mathbb{S}$ be a commutative semiring. For all $k \in \mathbb{N}$, the following class of* PNWA *and sub-fragments of* QMSO *are equally expressive over Γ and $\mathbb{S}$:*

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

Proof: ($\Leftarrow$) Let $\mathcal{A} = (\Gamma, \mathbb{S}, Q, \delta, I, F)$ be a pure-nondeterministic weighted automata over Γ and $\mathbb{S}$ such that $\mathcal{A}$ is polynomial ambiguous with degree k (i.e. $\mathcal{A} \in \textit{poly}^k$-PNWA). We show next how to decompose $\mathcal{A}$ inductively and construct a formula $\varphi_{\mathcal{A}} \in$ QMSO$(\Sigma_x^k, \oplus, \odot_b)$ that is equivalent to $\mathcal{A}$. Consider for each pair $p, q \in Q$ the automata $\mathcal{A}_{p,q} = (\Gamma, \mathbb{S}, Q, \delta, \{p\}, F_q)$ such that $F_q(q) = \mathbb{1}$ and $\mathbb{0}$ otherwise. That is, the automata $\mathcal{A}_{p,q}$ is equivalent to $\mathcal{A}$ restricted to p and q to be the initial and final state, respectively. We define the construction of $\varphi_{\mathcal{A}}$ inductively by using a function π that takes as parameters an automaton of the form $\mathcal{A}_{p,q}$ with $p, q \in Q$ and two first order variable x and y, and build a formula $\pi(\mathcal{A}_{p,q}, x, y)$ in QMSO$(\Sigma_x^k, \oplus, \odot_b)$ with $k = \mathrm{degree}(\mathcal{A}_{p,q})$. Intuitively, the function $[\![\pi(\mathcal{A}_{p,q}, x, y)]\!]$ over $w \in \Gamma^*$ defines the $\oplus$-aggregation of all runs from p to q of $\mathcal{A}$ over w restricted to the interval (x, y). Before going into the details of π, notice that by using π we can define $\varphi_{\mathcal{A}}$ as follows:

$$\varphi_{\mathcal{A}} \;=\; \sum_{(p,q) \in I \times Q} \pi(\mathcal{A}_{p,q}, \mathrm{first}, \mathrm{last}) \odot F(q).$$

Note that the above sum is not a quantification in the logic, and first and last are constant in the logic that defines the first and last position over the input string. Furthermore, the product $\pi(\mathcal{A}_{p,q}, \mathrm{first}, \mathrm{last}) \odot F(q)$ is still in the logic QMSO$(\Sigma_x^k, \oplus, \odot_b)$ given that one can always "push" the value $F(q)$ (by distributivity of $\mathbb{S}$) to the non-quantified level of the formula and rewrite $\varphi_{\mathcal{A}}$ into a formula in QMSO$(\Sigma_x^k, \oplus, \odot_b)$.

In the sequel, we define the output of $\pi(\mathcal{A}_{p,q}, x, y)$ by induction over the degree of $\mathcal{A}_{p,q}$. For the base case ($\mathrm{degree}(\mathcal{A}_{p,q}) = 0$), we have that $r_{\mathcal{A}_{p,q}} \in O(x^0)$, that is, there exists a constant N such that $|\mathrm{Run}_{\mathcal{A}_{p,q}}(w)| < N$ for all $w \in \Gamma$ (recall that $r_{\mathcal{A}_{p,q}}(n) = \max_{v \in \Gamma^n} |\mathrm{Run}_{\mathcal{A}_{p,q}}(v)|$ for $n = |w|$). In other words, $\mathcal{A}_{p,q}$ is finitely ambiguous. By Lemma A.7, we know that $\mathcal{A}_{p,q}$ can be decomposed in the disjoint union of unambiguous weighted automata $\mathcal{A}_1, \ldots, \mathcal{A}_N$ such that:

$$[\![\mathcal{A}_{p,q}]\!](w) = [\![\mathcal{A}_1 \oplus \cdots \oplus \mathcal{A}_N]\!](w)$$

for every $w \in \Gamma^*$. Since $\mathcal{A}_i$ is unambiguous for each $i \leq N$, we can find a formula $\varphi_i(x, y)$ in MSO restricted to the interval (x, y) such that $\mathcal{A}_i$ is equivalent to $\varphi_i(x, y)$. Indeed, $\mathcal{A}_i$ outputs $\mathbb{0}$ or $\mathbb{1}$ whenever it rejects or accepts a words which is equivalent to a property defined in MSO. Finally, we have that:

$$\pi(\mathcal{A}_{p,q}, x, y) = \varphi_1(x, y) \oplus \ldots \oplus \varphi_N(x, y)$$

For the inductive case, suppose that $\pi(\mathcal{A}_{p',q'}, x, y) \in$ QMSO$(\Sigma_x^k, \oplus, \odot_b)$ is defined for every $p', q' \in Q$ such that $\mathrm{degree}(\mathcal{A}_{p',q'}) = k$. Then we show how to construct the output of $\pi(\mathcal{A}_{p,q}, x, y)$ for any pair $p, q \in Q$ such that

$\text{degree}(\mathcal{A}_{p,q}) = k+1$. First, we need to introduce some notation related with the graph-structure of $\mathcal{A}$. Let $G_{\mathcal{A}}$ be $\mathcal{A}$ seen as a graph, that is, $G_{\mathcal{A}} = (Q, E_\delta)$ where Q is the set of vertex and E_δ is the set of edges such that $(p,q) \in E_\delta$ if, and only if, $(p,a,q) \in \delta$ for some $a \in \Gamma$. Using the graph representation of an automaton $\mathcal{A}$, we can derive the notion of *strongly connected component* (or simply a *component*) of $\mathcal{A}$: this is a maximal set A of nodes of $G_{\mathcal{A}}$ such that for all $p, q \in A$, there is a directed path from p to q visiting only nodes in A and traversing edges in E_δ. We denote by $\text{SCC}(\mathcal{A})$ the set of all components of $\mathcal{A}$. Notice that for each $p \in Q$ there exists only one $A \in \text{SCC}(\mathcal{A})$ such that $p \in A$. Thus, this allows us to talk about the component of p and we denote it by $\text{SCC}(p)$.

Coming back to the proof, the main idea of the inductive step is to sum over all positions where a particular transition between two components is being used and, then, separate the formula between the prefix and the suffix of this transition. We argue that (1) the sub-automata over the prefix has degree at most k and we can apply our inductive hypothesis, and (2) the sub-automata of the suffix has degree 0 which is trivial to define (using the ideas of the base case). In order to develop this idea, the following claim is crucial.

Claim A.10. *There exists $p', q' \in Q$ satisfying the following conditions:*

1) $(p', q') \in E_\delta$,
2) *p' and q' are in different components (*$\text{SCC}(p') \neq \text{SCC}(q')$*),*
3) $\text{degree}(\mathcal{A}_{p,p'}) < k+1$, *and*
4) $\text{degree}(\mathcal{A}_{q,q'}) = 0$.

Proof: First of all, notice that $\text{degree}(\mathcal{A}_{p,q}) = 0$ whenever p and q are in the same component (Lemma A.9). Furthermore, one can easily show that $\text{degree}(\mathcal{A}_{p,q}) = \text{degree}(\mathcal{A}_{p',q'})$ for every states p', q' such that p and p' (q and q') are in the same component. Indeed, for every $w \in \Gamma^*$ we can always find $u, v \in \Gamma$ such that $|\text{Run}_{\mathcal{A}_{p,q}}(w)| \leq |\text{Run}_{\mathcal{A}_{p',q'}}(u \cdot w \cdot v)|$ and from here we can easily get that $\text{degree}(\mathcal{A}_{p,q}) = \text{degree}(\mathcal{A}_{p',q'})$. These facts allow us to define the *degree* of two components $A, B \in \text{SCC}(\mathcal{A})$ in $\mathcal{A}$ by:

$$\text{degree}_{\mathcal{A}}(A, B) \;=\; \text{degree}(\mathcal{A}_{p,q})$$

where p and q are any state of A and B, respectively. Notice that by the previous fact, p and q are not important in the definition and, furthermore, it always holds that $\text{degree}_{\mathcal{A}}(A, A) = 0$ for all $A \in \text{SCC}(\mathcal{A})$.

Without lost of generality, we assume that all the component in $\text{SCC}(\mathcal{A})$ are reachable from p and q, that is, for each component in $A \in \text{SCC}(\mathcal{A})$ there exists a path from p to q that passes through A. Otherwise, we can always trim $\mathcal{A}$ by removing the components that are not reachable and co-reachable from p and q. For each component $A \in \text{SCC}(\mathcal{A})$, define the set $\text{Reach}_{\mathcal{A}}(A)$ of all components directly reachable from A, i.e. $\text{Reach}_{\mathcal{A}}(A) = \{B \in \text{SCC}(\mathcal{A}) - \{A\} \mid (A \times \Gamma \times B) \cap \delta \neq \varnothing\}$. Now, take any component $A \in \text{SCC}(\mathcal{A})$ such that $\text{degree}_{\mathcal{A}}(A, \text{SCC}(q)) \neq 0$ and $\text{degree}_{\mathcal{A}}(B, \text{SCC}(q)) = 0$ for all $B \in \text{Reach}_{\mathcal{A}}(A)$. Note first that A always exists. This can be shown constructively as follows. Let $\mathcal{C} = \{\text{SCC}(q)\}$ and take $A^\star \in \text{SCC}(\mathcal{A})$ such that $\text{Reach}_{\mathcal{A}}(A^\star) \subseteq \mathcal{C}$. If $\text{degree}_{\mathcal{A}}(A^\star, \text{SCC}(q)) \neq 0$, then we are done. Otherwise, redefine $\mathcal{C} = \mathcal{C} \cup \{A^\star\}$ and repeat with procedure. At some point, we know that either $\mathcal{C} = \text{SCC}(\mathcal{A})$ in which case $\text{degree}(\mathcal{A}_{p,q}) = 0$ (a contradiction), or we find our component $A = A^\star$ such that $\text{degree}_{\mathcal{A}}(A^\star, \text{SCC}(q)) \neq 0$ and $\text{degree}_{\mathcal{A}}(B, \text{SCC}(q)) = 0$ for all $B \in \text{Reach}_{\mathcal{A}}(A^\star)$. Therefore, we conclude that A exists.

Take now $p' \in A$ and $q' \in B$ for some $B \in \text{Reach}_{\mathcal{A}}(A)$ such that $(p', q') \in E_\delta$. By the definition of A, we know that p' and q' exists. Furthermore, we have that $\text{degree}(\mathcal{A}_{q',q}) = 0$ given that $\text{degree}_{\mathcal{A}}(B, \text{SCC}(q)) = 0$. Thus, it only left to show that $\text{degree}(\mathcal{A}_{p,p'}) < k+1$. By contradiction, suppose that $\text{degree}(\mathcal{A}_{p,p'}) \geq k+1$. Then for every $u \cdot v \in \Gamma^*$, we have:

$$r_{\mathcal{A}_{p,p'}}(|u|) \cdot r_{\mathcal{A}_{p',q}}(|v|) \;\leq\; r_{\mathcal{A}_{p,q}}(|u \cdot v|).$$

Since $r_{\mathcal{A}_{p,p'}} \in \Omega(n^{k+1})$ and $r_{\mathcal{A}_{p',q}} \in \omega(1)$, then we have that $r_{\mathcal{A}_{p,q}} \in \omega(n^{k+1})$. Notice that we can assume the strong fact $r_{\mathcal{A}_{p,p'}} \in \Omega(n^{k+1})$ (and not just $r_{\mathcal{A}_{p,p'}} \in \omega(n^k)$) since it is a folklore result that the number of runs of any automata structure is either constant, polynomial, or exponential in the length of the word. This leads us to a contradiction given that $r_{\mathcal{A}_{p,q}} \in O(n^{k+1})$. ∎

The previous claim is all what we need to define the formula $\pi(\mathcal{A}_{p,q}, x, y)$. Consider the pair of states $p', q' \in Q$ that satisfies the above conditions of Claim A.10. Consider the set of transitions $T = \{(p', a, q') \mid a \in \Gamma\}$. Then we define the formula $\pi(\mathcal{A}_{p,q}, x, y)$ as follows:

$$\pi(\mathcal{A}_{p,q}, x, y) \;=\; \left(\Sigma z.\, \pi(\mathcal{A}_{p,p'}, x, z-1) \odot \Big(\sum_{(p',a,q')\in T} P_a(z)\Big) \odot \pi(\mathcal{A}_{q',q}, z+1, y)\right) \oplus \pi(\mathcal{A}^{-T}_{p,q}, x, y)$$

where $\mathcal{A}^{-T}_{p,q}$ is equal to $\mathcal{A}_{p,q}$ without all transitions that connects p' with q', that is, $\mathcal{A}^{-T}_{p,q} = (\Gamma, \mathbb{S}, Q, \delta - T, \{p\}, F_q)$. First, notice that, by some abuse of notation, we are using $(z-1)$ and $(z+1)$ to denote the predecessor and successor of z. Without lost of generality, these two elements can be referenced into a formula by using the formula succ. Second, the formula $\phi_1 = \sum_{(p',a,q')\in T} P_a(z)$ is free of Σ-quantifiers and, thus, it does not increase the nesting of quantifiers. The same happens to the output $\phi_2 = \pi(\mathcal{A}_{q',q}, z+1, y)$ where the degree of $\mathcal{A}_{q',q}$ is 0 (by Lemma A.10) and, thus, the output formula is also free of Σ-quantifiers. Since both ϕ_1 and ϕ_2 are free of Σ-quantifiers, we can always "push" the product of $\phi_1 \odot \phi_2$ inside $\pi(\mathcal{A}_{p,p'}, x, z-1)$ (by distributivity of $\mathbb{S}$) to the non-quantified level of the formula and rewrite $\pi(\mathcal{A}_{p,q}, x, y)$ into a formula in $\mathrm{QMSO}(\Sigma^{k+1}_x, \oplus, \odot_b)$. Further, the automata $\mathcal{A}_{p,p'}$ has degree strictly less than $k+1$ (by Lemma A.10), it satisfies our inductive hypothesis and, thus, it can be defined in a formula in $\mathrm{QMSO}(\Sigma^{k}_x, \oplus, \odot_b)$. Finally, the last part of the formula $\pi(\mathcal{A}_{p,q}, x, y)$ continues the procedure π with the sub-automata $\mathcal{A}^{-T}_{p,q}$ that has strictly less transitions than $\mathcal{A}_{p,q}$.

For the correctness of the formula, it is straightforward to check that $\pi(\mathcal{A}_{p,q}, x, y)$ is the sum over all the runs that (a) use a transition from p' to q' and (b) do not use a transition from p' to q' at all. Clearly, the sum of runs of both types are defined by the first and second formulas of $\pi(\mathcal{A}_{p,q}, x, y)$, respectively. Notice here that each run passes at most one time through a transition from p' to q' given that p' and q' are in different components. Therefore, we conclude that the formula $\pi(\mathcal{A}_{p,q}, x, y)$ computes the sum of all the runs from p to q.

($\Rightarrow$) The proof of this direction is shown by analysing the ambiguity of the weighted automata constructed in the proof of Theorem 4.1. First of all, consider a formula in $\mathrm{QMSO}(\Sigma^{k}_x, \oplus, \odot_b)$ without any first-order Σ-quantification. By using Lemma A.2, we can extract a pure-deterministic automata defining the same function. Notice that this automata is deterministic and, therefore, its ambiguity degree is equal to 0.

The translation of the formula $\theta_1 \oplus \theta_2$ can also be derived easily from Theorem 4.1. Consider $\mathcal{A}_1$ and $\mathcal{A}_2$ in $poly^k$-PNWA equivalent to θ_1 and θ_2 in $\mathrm{QMSO}(\Sigma^{k}_x, \oplus, \odot_b)$, respectively. Then $\mathcal{A}_1 \oplus \mathcal{A}_2$ is equivalent to the formula $\theta_1 \oplus \theta_2$ where $\mathcal{A}_1 \oplus \mathcal{A}_2$ is the disjoint union of $\mathcal{A}_1$ and $\mathcal{A}_2$. It is important to remark here that $\mathrm{degree}(\mathcal{A}_1 \oplus \mathcal{A}_2) \le k$ whenever $\mathrm{degree}(\mathcal{A}_1) \le k$ and $\mathrm{degree}(\mathcal{A}_2) \le k$.

The last step is to translate the first-order Σ-quantification into a PNWA and show that this translation only increases the ambiguity of the output at most by one. For proving this, the translation of the first-order Σ-quantification of Theorem 4.1 suffices. Let $\mathcal{A} = (\Gamma \times \{0,1\}, \mathbb{S}, Q, \delta, I, F)$ be the pure non-deterministic weighted automaton over $\Gamma_{\{x\}}$ and $\mathbb{S}$ equivalent to $\theta \in \mathrm{QMSO}(\Sigma^{k}_x, \oplus, \odot_b)$ where x is a free first-order variable and $\mathrm{degree}(\mathcal{A}) = k$ (for the sake of simplicity we omit the set V of the other free variables of θ). Furthermore, let $\mathcal{A}' = (\Gamma, \mathbb{S}, Q \cup Q', \delta', I', F')$ be the resulting pure non-deterministic weighted automaton equivalent to the formula $\Sigma x.\theta$ constructed in Theorem 4.1. From this construction, one can easily check that every $w \in \Gamma^*$ satisfies:

$$|\mathrm{Run}_{\mathcal{A}'}(w)| = \sum_{i=1}^{|w|} |\mathrm{Run}_{\mathcal{A}}(w_i)| \tag{5}$$

where $w_i \in (\Gamma \times \{0,1\})^*$ is equal to w marked only at position i. Indeed, each accepting run $\rho \in \mathrm{Run}_{\mathcal{A}'}(w)$ passes exactly once through a transition $\rho(i) = (p, a, q') \in \delta'$ where $p \in Q$ and $q' \in Q'$ and $i \le |w|$. That is, ρ is of the form:

$$\rho \;=\; q_0 \xrightarrow{a_1} \cdots \xrightarrow{a_{i-1}} q_{i-1} \xrightarrow{a_i} q'_i \xrightarrow{a_{i+1}} \cdots \xrightarrow{a_n} q'_n$$

where $w = a_1 \cdots a_n$, $q_0, \ldots, q_{i-1} \in Q$, and $q'_i, \ldots, q'_n \in Q'$. Then we can define the function $f : \mathrm{Run}_{\mathcal{A}'}(w) \to \bigcup_{i=1}^{n} \mathrm{Run}_{\mathcal{A}}(w_i)$ such that:

$$f(\rho) = q_0 \xrightarrow{a_1} \cdots \xrightarrow{a_{i-1}} q_{i-1} \xrightarrow{a_i} q_i \xrightarrow{a_{i+1}} \cdots \xrightarrow{a_n} q_n,$$

i.e. f maps ρ to a run where each state q' after i-position is mapped to his original version q. Clearly, f is a bijection between $\mathrm{Run}_{\mathcal{A}'}(w)$ and $\bigcup_{i=1}^{n} \mathrm{Run}_{\mathcal{A}}(w_i)$, and then Equation 5 holds. Finally, one can easily check the following inequality by using Equation 5:

$$r_{\mathcal{A}'}(n) = \max_{w\in\Gamma^n} |\mathrm{Run}_{\mathcal{A}'}(w)| \le \max_{w\in\Gamma^n} \sum_{i=1}^{n} |\mathrm{Run}_{\mathcal{A}}(w_i)| \le \sum_{i=1}^{n} \max_{w_j\in\Gamma^n} |\mathrm{Run}_{\mathcal{A}}(w_j)| = n \cdot r_{\mathcal{A}}(n).$$

Thus, we have $r_{\mathcal{A}'} \in O(n^{k+1})$ and then $\text{degree}(\mathcal{A}') \leq k+1$. ■

G. Proofs Omitted in Section VI-B

In this section we give the proofs omitted in Section VI-B. We first give a formal definition of two-way weighted automata with k-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. In order to recognize only regular languages, pebbles are dropped in a nested discipline, that is, 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. In other words, the k-pebbles follow a stack discipline over the numbers $1, \ldots, k$. Formally, a two-way weighted automata with k-nested pebbles (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). If $E(p, a, \vec{b}, q, \rightsquigarrow) \neq \mathbb{0}$, then we say that tuple $t = (p, a, \vec{b}, q, \rightsquigarrow)$ is a transition of $\mathcal{A}$ with cost $E(t)$. Intuitively, this means that if $\mathcal{A}$ is in state p, reading a, and pebbles $\{i \mid b(i) = 1\}$ are placed in the current position, then $\mathcal{A}$ changes its state into q and execute the instruction $\rightsquigarrow$ with cost $E(t)$, i.e. moves the reading head one position to the right ($\rightarrow$) or left ($\leftarrow$), drops next pebble ($\downarrow$), or lifts last pebble ($\uparrow$). Given a word $w \in \Gamma^*$, a *configuration* of $\mathcal{A}$ over w is a triple $(q, l, \vec{l})$ where q is the current state of $\mathcal{A}$, $l \in \{0, \ldots, |w|+2\}$ is the current position of the reading head over $\triangleleft\, w\, \triangleright$, and $\vec{l} \in \{1, \ldots, |w|\}^{\leq k}$ are the positions of the $|\vec{l}|$-pebbles in $\triangleleft\, w\, \triangleright$. Notice that if $l = 0$ or $l = |w|+1$, this means that the reading is over the left ($\triangleleft$) or right ($\triangleright$) markers respectively, and if $l = |w|+2$, then the reading head "falls" out of the input word. Further, if $\vec{l} = l_1 \ldots l_j$ with $j \leq k$, this means that the i-pebble is placed in the l_i-position of w for $i \leq j$. We say that $(q, l, \vec{l})$ is an initial (final) configuration of $\mathcal{A}$ over w if $l = 0$ ($l = |w|+2$) and $\vec{l} = \epsilon$. We extend a transition t from states to configurations in the expected way and we write this graphically as $(q, l, \vec{l}) \xrightarrow{t} (q', l', \vec{l}')$. Notice that $\mathcal{A}$ is in a final configuration if the reading head "falls" from the input and, thus, it cannot move any more (i.e. $\mathcal{A}$ is in position $|w|+2$).

Similar to one-way weighted automata, we define a *run* ρ of $\mathcal{A}$ over w as a sequence of configurations and transitions:

$$\rho = (q_0, l_0, \vec{l}_0) \xrightarrow{t_1} (q_1, l_1, \vec{l}_1) \cdots \xrightarrow{t_m} (q_m, l_m, \vec{l}_m).$$

A run ρ like above is *accepting* if $(q_0, l_0, \vec{l}_0)$ is an initial configuration, $(q_m, l_m, \vec{l}_m)$ is a final configuration, $I(q_0) \neq \mathbb{0}$, $F(q_m) \neq \mathbb{0}$, and $E(t_i) \neq \mathbb{0}$ for every $i \leq m$. In this case, the *weight* of an accepting run ρ of $\mathcal{A}$ over w is defined by:

$$|\rho| = I(q_0) \odot \prod_{i=1}^{m} E(t_i) \odot F(q_m).$$

We define by $\text{Run}_{\mathcal{A}}(w)$ the set of all accepting runs of $\mathcal{A}$ over w. Furthermore, we assume that for all the two-way weighted automata consider in this paper, the set $\text{Run}_{\mathcal{A}}(w)$ is finite for each $w \in \Gamma^*$. Finally, the weight of $\mathcal{A}$ over a word w is defined by:

$$[\![\mathcal{A}]\!](w) = \sum_{\rho \in \text{Run}_{\mathcal{A}}(w)} |\rho|$$

where the sum is equal to $\mathbb{0}$ if $\text{Run}_{\mathcal{A}}(w)$ is empty.

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. Interestingly, in the next result we show that 2DWA-0 and unamb-2WA-0 coincide with the class of unamb-WA. By Proposition 5.3, this implies that all this subclasses are equally expressive than $\text{QMSO}(\Pi_x^1, \oplus_b, \odot)$ which further shows the robustness of this class of functions for any commutative semiring.

Theorem 6.3. *Let Γ be a finite alphabet and $\mathbb{S}$ be a commutative semi-ring. 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) $\text{QMSO}(\Pi_x^1, \oplus_b, \odot)$.

Proof: From the above claim, clearly $unamb$- WA and 2DWA-0 are special cases of $unamb$- 2WA-0. Furthermore, in Theorem 5.3 we already prove that $unamb$- WA and $\text{QMSO}(\Pi_x^1, \oplus_b, \odot)$ are equivalent. Then it only left to show that $unamb$- 2WA-0 $\subseteq$ $unamb$- WA and $unamb$- 2WA-0 $\subseteq$ 2DWA-0. For the second implication, it was shown in [17] (Theorem 3) that *two-way deterministic transducers* are equal expressive than *two-way unambiguous transducers*. In simple terms, a two-way transducer is a two-way weighted automata over the free semiring. This means that $unamb$- 2WA-0 $\subseteq$ 2DWA-0 is a special case of the results in [17]. Thus, we dedicate the rest of the proof to show that $unamb$- 2WA-0 $\subseteq$ $unamb$- WA.

Let $\mathcal{A} = (\Gamma, \mathbb{S}, Q, E, I, F)$ be a two-way unambiguous weighted automata. The main idea of the proof is to define an unambiguous weighted automata $\mathcal{A}'$ equal to $\mathcal{A}$ by considering the *crossing sequences* of $\mathcal{A}$ over a word w [22]. Intuitively, the strategy of $\mathcal{A}'$ is to guess the crossing sequences of $\mathcal{A}$ over an input word w and checks at the end whether the guessing was correct. Given that $\mathcal{A}$ is unambiguous, there exists only one correct guessing of the crossing sequences used in a run of $\mathcal{A}$. By adding the *directional information* of the crossing sequences, $\mathcal{A}'$ can unambiguously guess this run and compute the same function defined by $\mathcal{A}$ over words in Γ.

Formally, fix an (unique) accepting run $\rho = (q_0, l_0) \xrightarrow{t_1} (q_1, l_1) \ldots \xrightarrow{t_n} (q_n, l_n)$ of $\mathcal{A}$ over w where $t_i = (q_{i-1}, a_i, q_i, d_i)$ for each $i \leq n$. For each $j \leq |w| + 1$, we say that a configuration (q_i, l_i) *crosses* the j-boundary of w if $l_{i-1} = j - 1$ and $l_i = j$, or $l_{i-1} = j$ and $l_i = j - 1$. Furthermore, we call a subsequence $(q_0', l_0'), \ldots, (q_m', l_m')$ of configurations of ρ a *j-crossing sequence of configurations* if (q_i', l_i') crosses the j-boundary of w for each $i \leq n$. In other words, a j-crossing sequence of configurations is the subsequence of configurations of ρ which previous transition crosses the j-boundary of w. In [22], the sequence of states $q_0', \ldots, q_m'$ of a crossing sequence of configurations is called a crossing sequence of $\mathcal{A}$ when $\mathcal{A}$ is seen as a two-way finite automaton. For finite state automata, crossing sequences are enough information to convert a two-way finite automaton into a one-way finite automaton [22]. However, for two-way weighted automata this information is not enough to unambiguously determine the cost of a transition between two crossing sequences. To solve this issue, we extend crossing sequences with directions $\rightarrow$, $\leftarrow$, $\hookleftarrow$, and $\hookrightarrow$. Formally, we consider the *direction* $\text{dir}_\rho(q_i, l_i)$ of a configuration (q_i, l_i) in ρ as follows:

$$\text{dir}_\rho(q_i, l_i) \quad = \quad \begin{cases} \rightarrow & \text{if } d_i = d_{i+1} = \rightarrow \\ \leftarrow & \text{if } d_i = d_{i+1} = \leftarrow \\ \hookleftarrow & \text{if } d_i = \rightarrow \text{ and } d_{i+1} = \leftarrow \\ \hookrightarrow & \text{if } d_i = \leftarrow \text{ and } d_{i+1} = \rightarrow \end{cases}$$

In other words, the direction $\text{dir}_\rho(q_i, l_i)$ is equal to $\rightarrow$ ($\hookleftarrow$) if (q_i, l_i) crosses its boundary from left to right and continues to the right (left), and it is equal to $\leftarrow$ ($\hookrightarrow$) if it crosses its boundary from right to left and continues to the left (right). By using $\text{dir}_\rho(q_i, l_i)$ we define a *directional crossing sequence* of ρ to be a sequence $(q_0', d_0'), \ldots, (q_m', d_m') \in (Q \times \{\rightarrow, \leftarrow, \hookrightarrow, \hookleftarrow\})^*$ such that $(q_0', l_0'), \ldots, (q_m', l_m')$ is a crossing sequence of configurations and $d_i' = \text{dir}_\rho(q_i', l_i')$ for every $i \leq m$. We define $C_{\mathcal{A}} = (Q \times \{\rightarrow, \leftarrow, \hookrightarrow, \hookleftarrow\})^*$ to be the set of all directional crossing sequences of $\mathcal{A}$. Since $\mathcal{A}$ cannot cross a boundary with the same state twice (otherwise $\mathcal{A}$ will get stack in a loop), we can consider only the directional crossing sequences that are of length at most $2|Q|$. Thus, in the sequel we use the set of directional crossing sequences of length at most $2|Q|$ to run $\mathcal{A}$ with only one pass over the input.

Given two directional crossing sequences of $\mathcal{A}$ we can determine the *cost* between them unambiguously by using the directions in both sequences. Specifically, we define the cost functions $E^{\rightarrow} : C_{\mathcal{A}} \times \Gamma \times C_{\mathcal{A}} \rightarrow \mathbb{S}$ and

$E^{\leftarrow}: C_{\mathcal{A}} \times \Gamma \times C_{\mathcal{A}} \to \mathbb{S}$ recursively for any $p, q \in Q$, $d, e \in \{\rightarrow, \leftarrow, \hookrightarrow, \hookleftarrow\}$, and $u, v \in C_{\mathcal{A}}$ as follows:

$$\begin{array}{rcll} E^{\rightarrow}(\epsilon, a, \epsilon) & = & \mathbb{1} & \\ E^{\leftarrow}(\epsilon, a, \epsilon) & = & \mathbb{1} & \\ & & & \\ E^{\rightarrow}((p, \rightarrow) \cdot u, a, (q, d) \cdot v) & = & E(p, a, q, \rightarrow) \odot E^{\leftarrow}(u, a, v) & \text{where } d \in \{\rightarrow, \hookleftarrow\} \\ E^{\rightarrow}((p, \hookleftarrow) \cdot (q, d) \cdot u, a, v) & = & E(p, a, q, \leftarrow) \odot E^{\rightarrow}(u, a, v) & \text{where } d \in \{\leftarrow, \hookrightarrow\} \\ & & & \\ E^{\leftarrow}((p, d) \cdot u, a, (q, \leftarrow) \cdot v) & = & E(q, a, p, \leftarrow) \odot E^{\rightarrow}(u, a, v) & \text{where } d \in \{\leftarrow, \hookrightarrow\} \\ E^{\leftarrow}(u, a, (q, \hookrightarrow) \cdot (p, d) \cdot v) & = & E(q, a, p, \rightarrow) \odot E^{\leftarrow}(u, a, v) & \text{where } d \in \{\rightarrow, \hookleftarrow\} \end{array}$$

Otherwise, we define $E^d(u, a, v) = \mathbb{0}$ in all other cases. From the previous definition it is important to notice that pairs of directional crossing sequences are incompatible iff the cost assigned by $E^{\rightarrow}$ is equal to $\mathbb{0}$. Moreover, the cost over compatibles sequences follows the directions of both sequences unambiguously. Intuitively, the meaning of function $E^{\rightarrow}$ ($E^{\leftarrow}$) is to consider that the reading head is moving forward (backward) and the first state of the first (second) sequence is the current control state of $\mathcal{A}$.

We have all the necessary ingredients to define the unambiguous weighted automata $\mathcal{A}'$. First of all, without lost of generality we assume that (1) $\mathcal{A}$ never falls out from the left side of the input (i.e. $E(p, \triangleleft, q, \leftarrow) = \mathbb{0}$ for all $p, q \in Q$), (2) there exists only one initial state $q_0 \in Q$ such that $I(q_0) \neq \mathbb{0}$, and (3) for every $p \in Q$ there exists at most one $q \in Q$ such that $E(p, \triangleright, q, \rightarrow) \odot F(q) \neq \mathbb{0}$. The first two conditions are trivial to impose over any two-way unambiguous weighted automata. The last condition stems from the fact that $\mathcal{A}$ is unambiguous and, therefore, they cannot exist two states q_1 and q_2 such that $E(p, \triangleright, q_i, \rightarrow) \odot F(q_i) \neq \mathbb{0}$ for $i \in \{1, 2\}$ whenever p is reachable at the end of the input from the initial configuration. This implies that for every crossing sequence $u \in C_{\mathcal{A}}$ of the form $u = u' \cdot (p, d)$ we can assign a unique state q_u such that $E(p, \triangleright, q_u, \rightarrow) \odot F(q_u) \neq \mathbb{0}$. Let $C_{\mathcal{A}}^{2|Q|}$ be the crossing sequence of length at most $2|Q|$. Then we define the weighted automata $\mathcal{A}' = (\Gamma, \mathbb{S}, C_{\mathcal{A}}^{2|Q|}, E', I', F')$ such that for every $u, v \in C_{\mathcal{A}}^{2|Q|}$ and $a \in \Gamma$ we have:

$$\begin{array}{rcl} E'(u, a, v) & = & E^{\rightarrow}(u, a, v) \\ I'(u) & = & I(q_0) \odot E^{\rightarrow}((q_0, \rightarrow), \triangleleft, u) \\ F'(u) & = & E^{\rightarrow}(u, \triangleright, (q_u, \rightarrow)) \odot F(q_u) \end{array}$$

For the last part of the proof, we show that $[\![\mathcal{A}]\!](w) = [\![\mathcal{A}']\!](w)$ for every $w \in \Gamma^*$. Fix a word $w = a_1 \cdots a_n \in \Gamma^*$ and let ρ be an accepting run of $\mathcal{A}$ over w. From the construction of $\mathcal{A}'$, it is easy to check that from ρ we can extract a compatible sequence of directional crossing sequences $u_0, \ldots, u_n \in C_{\mathcal{A}}^{2|Q|}$ such that:

$$|\rho| = I'(u_0) \odot \prod_{i=1}^{n} E'(u_{i-1}, a_i, u_i) \odot F'(u_n)$$

Then we have that $u_0 \xrightarrow{a_1/s_1} u_1 \xrightarrow{a_2/s_2} \cdots \xrightarrow{a_n/s_n} u_n$ is an accepting run of $\mathcal{A}'$ over w.

Now, we show by induction that for any accepting run ρ' of $\mathcal{A}'$ over w, there exists an accepting run ρ of $\mathcal{A}$ over w such that ρ' consists on the directional crossing sequence of ρ over w. Given that $\mathcal{A}$ is unambiguous, this will imply that ρ' is unique. For the rest of the proof, we call a run $\rho = (q_0, l_0) \xrightarrow{t_1} \cdots \xrightarrow{t_m} (q_m, l_m)$ of $\mathcal{A}$ *partial accepting* iff $E(t_i) \neq \mathbb{0}$ for all $i \leq m$. Furthermore, we define the partial cost of ρ by:

$$\lceil \rho \rceil = \prod_{i=1}^{m} E(t_i)$$

Fix an accepting run $u_0 \xrightarrow{a_1/s_1} \cdots \xrightarrow{a_n/s_n} u_i$ of $\mathcal{A}'$ over $a_1 \ldots a_n$ and suppose that $u_i = (q_1^i, d_1^i) \ldots (q_{k_i}^i, d_{k_i}^i)$. We make the following claim for all $i \leq n$.

Claim A.11. *1) There exists a partial accepting run ρ_0 of $\mathcal{A}$ over $a_1 \ldots a_i$ that starting from $(q_0, 0)$ reaches $(q_1^i, i+1)$.*

2) For each $j = 2, 4, \ldots, k_i - 1$ there exists a partial accepting run ρ_j of $\mathcal{A}$ over $a_1 \ldots a_i$ that starting from (q_j^i, i)

reaches $(q^i_{j+1}, i+1)$.

3) *The following equivalence holds for runs* $\rho_0, \rho_2, \dots, \rho_{k_i-1}$*:*

$$I'(u_0) \odot \prod_{j=1}^{i} E'(u_{i-1}, a_i, u_i) = I(q_0) \odot \prod_{j=0,2,\dots k_i-1} \lceil \rho_j \rceil.$$

The proof of the above claim is straightforward (see [22]) and we leave this proof to the reader. Instead, we show how to use this claim to prove the last part of this proof which, in fact, are similar to the arguments need to prove the above claim.

Assume that Claim A.11 holds for the accepting run $\rho' = u_0 \xrightarrow{a_1/s_1} \cdots \xrightarrow{a_n/s_n} u_n$ of $\mathcal{A}'$ over w where $u_n = (q_1, d_1) \dots (q_k, d_k)$ and $\rho_0, \rho_2, \dots, \rho_{k-1}$ are the partial accepting runs mention above. Given that ρ' is accepting we know that $F'(u_n) = E^{\rightarrow}(u_n, \triangleright, (q_{u_n}, \rightarrow)) \odot F(q_{u_n}) \neq \mathbb{0}$. This implies that $E^{\rightarrow}(u_n, \triangleright, (q_{u_n}, \rightarrow)) \neq \mathbb{0}$. Furthermore, from the definition of $E^{\rightarrow}$ it is straightforward to prove that k is odd and:

$$E^{\rightarrow}(u_n, \triangleright, (q_{u_n}, \rightarrow)) = \left(\prod_{j=1,3,\dots k-2} E(q_j, \triangleright, q_{j+1}, \leftarrow) \right) \odot E(q_k, \triangleright, q_{u_n}, \rightarrow)$$

For the sake of presentation, let t_j be the transition $(q_j, \triangleright, q_{j+1}, \leftarrow)$ for $j = 1, 3, \dots k-2$ and let t_k be the transition $(q_k, \triangleright, q_{u_n}, \rightarrow)$. By Claim A.11, we have that ρ_0 is a partial accepting run from $(q_0, 0)$ to $(q_1, n+1)$ and ρ_j is a partial accepting run from (q_j, n) to $(q_{j+1}, n+1)$ for $j = 2, 4, \dots, k-1$. Then plugging all together we get that:

$$\begin{aligned}
|\rho'| &= I'(u_0) \odot \prod_{j=1}^{i} E'(u_{i-1}, a_i, u_i) \odot F'(u_n) \\
&= I(q_0) \odot \prod_{j=0,2,\dots k-1} \lceil \rho_j \rceil \odot \prod_{j=1,3,\dots k-2} E(t_j) \odot E(t_k) \odot F(u_{q_n}) \\
&= I(q_0) \odot \prod_{j=0,2,\dots,k-1} (\lceil \rho_j \rceil \odot E(t_{j+1})) \odot F(u_{q_n}) \\
&= I(q_0) \odot \lceil \rho \rceil \odot F(u_{q_n}) \\
&= |\rho|
\end{aligned}$$

where ρ is the resulting accepting run of $\mathcal{A}$ by sequentially connecting $\rho_0, \rho_2, \dots, \rho_{k-1}$ with the transitions $t_1, t_3, \dots, t_{k-2}$ and t_k. We conclude that ρ is an accepting run of $\mathcal{A}$ over w and ρ' consists on the directional crossing sequences of ρ over w. ■

Theorem 6.4. *Let* Γ *be a finite alphabet and* $\mathbb{S}$ *a commutative semi-ring. For every* $k \in \mathbb{N}$*, there exists an effective translation between the following classes of weighted automata and subfragments of* QMSO *over* Γ *and* $\mathbb{S}$*:*

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

Proof: We show this proof by induction over $k \in \mathbb{N}$. Given that the base case ($k = 0$) was proven in Proposition 6.3, we assume that Theorem 6.4 holds for $k \in \mathbb{N}$ and we prove this result for $k+1$.

Let θ_1, θ_2 be a formula in QMSO$(\Pi_x^{k+1}, \oplus_b, \odot)$ and let $\mathcal{A}_1, \mathcal{A}_2$ be the respectively 2DWA-k equivalent to θ_1, θ_2 by the inductive hypothesis. For the product formula $\theta = \theta_1 \odot \theta_2$, one can construct a 2DWA-k $\mathcal{A}$ equivalent to θ by sequentially running $\mathcal{A}_2$ after $\mathcal{A}_1$ over the input word. More specific, $\mathcal{A}$ runs $\mathcal{A}_1$ over $w \in \Gamma^*$ and just after "falling" out of the input ($\triangleright$) it change to a transient state in order to move the reading head backward until the beginning of the input ($\triangleleft$) and runs $\mathcal{A}_2$ over w from its initial state. One can easily check that $[\![\mathcal{A}]\!](w) = [\![\mathcal{A}_1]\!](w) \odot [\![\mathcal{A}_2]\!](w)$ and, thus, $[\![\mathcal{A}]\!](w) = [\![\theta]\!](w)$ for all $w \in \Gamma^*$. Now, suppose that $\theta = \Pi x.\, \theta_1(x)$ and $\mathcal{A}_1$ is a 2DWA-k over the alphabet $\Gamma \times \{0, 1\}$ such that $\theta_1(x) \equiv \mathcal{A}_1$. We can construct a 2DWA-$(k+1)$ $\mathcal{A}$ from $\mathcal{A}_1$ by moving forward the first pebble, position by position, starting from the beginning of the input and running $\mathcal{A}_1$ from scratch each time that this pebble is moved. Clearly, we will have that $[\![\theta]\!](w) = [\![\mathcal{A}]\!](w)$ for all $w \in \Gamma^*$ and θ is definable by a 2DWA-$(k+1)$. We conclude that any QMSO$(\Pi_x^{k+2}, \oplus_b, \odot)$-formula is definable by a two-way deterministic weighted automata with $(k+1)$-pebbles and the direction from 3 to 1 is shown.

Fix now a $unamb$- 2WA-(k + 1) $\mathcal{A} = (\Gamma, \mathbb{S}, Q, E, I, F)$ over the finite alphabet Γ and the commutative semiring $\mathbb{S}$. We show how to construct a formula $\theta \in \text{QMSO}(\Pi_x^{k+2}, \oplus_b, \odot)$ such that $\theta \equiv \mathcal{A}$. First of all, without lost of generality we assume that I and F are functions from Q into $\{\mathbb{0}, \mathbb{1}\}$. Indeed, one can always modify an unambiguous weighted automata to satisfy this restriction by moving the cost of I and F into its transitions. Now, let $w \in \Gamma^*$ and $\rho = (q_0, l_0, \vec{l}_0) \xrightarrow{t_1} (q_1, l_1, \vec{l}_1) \cdots \xrightarrow{t_m} (q_m, l_m, \vec{l}_m)$ a run of $\mathcal{A}$ over w. We classify the configurations in ρ into two types: *outer* and *inner* configurations. We say that $(q_i, l_i, \vec{l}_i)$ is an *outer* configuration if $\vec{l}_i = \epsilon$ and is *inner* otherwise (i.e. $\vec{l}_i \neq \epsilon$). That is, inner configurations have at least one pebble placed in the input where outer configuration have not. We can also divide the transitions of ρ into outer and inner transitions. We say that a transition $(q_{i-1}, l_{i-1}, \vec{l}_{i-1}) \xrightarrow{t_i} (q_i, l_i, \vec{l}_i)$ is an inner transition iff $(q_{i-1}, l_{i-1}, \vec{l}_{i-1})$ and $(q_i, l_i, \vec{l}_i)$ are inner configurations. Otherwise, we say that $(q_{i-1}, l_{i-1}, \vec{l}_{i-1}) \xrightarrow{t_i} (q_i, l_i, \vec{l}_i)$ is an outer transition. A run ρ can be seen as a sequence of outer and inner transitions starting always with an outer transition (i.e. no pebbles are in the input). This division between outer and inner transition allows us to define what we call an outer and inner $unamb$- 2WA-k. We say that $\mathcal{A}$ is an outer (inner) $unamb$- 2WA-k if a transition $(q, l, \vec{l}) \xrightarrow{t} (q', l', \vec{l}')$ of ρ is an inner (outer) transition, then $E(t) = \mathbb{1}$ for every w and every accepting run ρ of $\mathcal{A}$ over w. In other words, $\mathcal{A}$ is an outer (inner) $unamb$- 2WA-k if $\mathcal{A}$ only uses transitions different from $\{\mathbb{0}, \mathbb{1}\}$ over outer (inner) transitions. By considering the previous distinction between outer and inner automata, the following lemma shows that any $unamb$- 2WA-k can be decomposed into the product of an outer and inner $unamb$- 2WA-k.

Lemma A.12. *There exists an outer weighted automata $\mathcal{A}_1 \in unamb$- 2WA-(k + 1) and an inner weighted automata $\mathcal{A}_2 \in unamb$- 2WA-(k + 1) over Γ and $\mathbb{S}$ such that for every $w \in \Gamma^*$:*

$$[\![\mathcal{A}]\!](w) = [\![\mathcal{A}_1]\!](w) \odot [\![\mathcal{A}_2]\!](w)$$

Proof: The idea of the proof is to extend the states of $\mathcal{A}$ with one bit that keeps track if $\mathcal{A}$ is in *outer* or *inner* mode. Then we use this bit to decide whether the current transition t is an outer or inner transition in order to multiply its cost by $E(t)$ or by $\mathbb{1}$ depending whether we are running the outer or inner version of $\mathcal{A}$. More specific, let q_{in}, q_{out} be two different states not in Q. Define the automata $\mathcal{A}_1 = (\Gamma, \mathbb{S}, Q \times \{q_{in}, q_{out}\}, E_1, I_1, F_1)$ such that for all $q \in Q$, $I_1(q, q_{out}) = I(q)$ and $F_1(q, q_{out}) = F(q)$, and $\mathbb{0}$ otherwise. Further, we define the transition function E_1 as follows:

$$\begin{array}{rcll}
E_1((p, q_{out}), a, \vec{b}_0, (q, q_{out}), d) & = & E(p, a, \vec{b}_0, q, d) & \text{if } d \in \{\rightarrow, \leftarrow\} \\
E_1((p, q_{out}), a, \vec{b}_0, (q, q_{in}), \downarrow) & = & E(p, a, \vec{b}_0, q, \downarrow) & \\
E_1((p, q_{in}), a, \vec{b}_1, (q, q_{out}), \uparrow) & = & E(p, a, \vec{b}_1, q, \uparrow) & \\
E_1((p, q_{in}), a, \vec{b}, (q, q_{in}), d) & = & \mathbb{1} & \text{if } E(p, a, \vec{b}, q, d) \neq \mathbb{0} \wedge d \in \{\rightarrow, \leftarrow, \downarrow\} \\
E_1((p, q_{in}), a, \vec{b}, (q, q_{in}), \uparrow) & = & \mathbb{1} & \text{if } E(p, a, \vec{b}, q, d) \neq \mathbb{0} \wedge \vec{b} \neq \vec{b}_1
\end{array}$$

where $p, q \in Q$, $a \in \Gamma$, $\vec{b} \in \{0,1\}^{k+1}$, $\vec{b}_0 = (0, \ldots, 0)$ and $\vec{b}_1 = (1, 0, \ldots, 0)$. For all other cases we define $E_1(t) = \mathbb{0}$. Notice that $\mathcal{A}_1$ only maintains the value of E when it is either in outer mode or it is changing from outer to inner mode (and vice versa). By analogy, we define the weighted automata $\mathcal{A}_2 = (\Gamma, \mathbb{S}, Q \times \{q_{in}, q_{out}\}, E_2, I_1 F_1)$ such that:

$$\begin{array}{rcll}
E_2((p, q_{out}), a, \vec{b}_0, (q, q_{out}), d) & = & \mathbb{1} & \text{if } E(p, a, \vec{b}_0, q, d) \neq \mathbb{0} \wedge d \in \{\rightarrow, \leftarrow\} \\
E_2((p, q_{out}), a, \vec{b}_0, (q, q_{in}), \downarrow) & = & \mathbb{1} & \text{if } E(p, a, \vec{b}_0, q, \downarrow) \neq \mathbb{0} \\
E_2((p, q_{in}), a, \vec{b}_1, (q, q_{out}), \uparrow) & = & \mathbb{1} & \text{if } E(p, a, \vec{b}_1, q, \uparrow) \neq \mathbb{0} \\
E_2((p, q_{in}), a, \vec{b}, (q, q_{in}), d) & = & E_2(p, a, \vec{b}, q, d) & \text{if } d \in \{\rightarrow, \leftarrow, \downarrow\} \\
E_2((p, q_{in}), a, \vec{b}, (q, q_{in}), \uparrow) & = & E_2(p, a, \vec{b}, q, \uparrow) & \text{if } \vec{b} \neq \vec{b}_1
\end{array}$$

where $p, q \in Q$, $a \in \Gamma$, $\vec{b} \in \{0,1\}^{k+1}$, $\vec{b}_0 = (0, \ldots, 0)$ and $\vec{b}_1 = (1, 0, \ldots, 0)$. For all other cases we define $E_2(t) = \mathbb{0}$.

From the definition, it is clear that $\mathcal{A}_1, \mathcal{A}_2 \in unamb$-2WA-(k + 1). Furthermore, for every $w \in \Gamma^*$ and every accepting run ρ of $\mathcal{A}$, we can find an accepting run ρ_1 (ρ_2) of $\mathcal{A}_1$ ($\mathcal{A}_2$) over w such that ρ_1 (ρ_2) contains the cost of the outer (inner) transitions of ρ. This implies that we have the desired result:

$$[\![\mathcal{A}]\!](w) \ = \ [\![\mathcal{A}_1]\!](w) \odot [\![\mathcal{A}_2]\!](w)$$

for all $w \in \Gamma^*$. This was to be shown. ■

The previous lemma allows us to reduce our proof to find formulas θ_1 and θ_2 that defines the outer and inner part of $\mathcal{A}$, respectively. First, we focus how to define the inner $unamb$-2WA-(k + 1) $\mathcal{A}_2$ to later show how to define the outer $unamb$-2WA-(k + 1) $\mathcal{A}_1$. For the sake of simplification, in the rest of the proof we use $\vec{b}_0 = (0, \overset{k+1}{\ldots}, 0)$ and $\vec{b}_1 = (1, \overset{k+1}{\ldots}, 0)$ to denote the vector that represents no pebbles in the current position, or just the first pebble, respectively.

Fix an inner $unamb$-2WA-(k + 1) $\mathcal{A}_2 = (\Gamma, \mathbb{S}, Q_2, E_2, I_2, F_2)$ and let $\rho = (q_0, l_0, \vec{l}_0) \xrightarrow{t_1} \cdots \xrightarrow{t_m} (q_m, l_m, \vec{l}_m)$ be an accepting run of $\mathcal{A}_2$ over some $w \in \Gamma^*$. From the definition of an inner $unamb$-2WA-(k + 1), we know that the only transitions t_i that are different from $\mathbb{1}$ are the inner transitions in ρ. These transitions are grouped in disjoint intervals between 1 and m. Each interval starts when $\mathcal{A}_2$ drops the first pebble in moment $i \le m$, and ends when $\mathcal{A}_2$ removes this pebble again for the first time after i. Let $H_\rho = \{(i_\downarrow, i_\uparrow, i) \in [1, m] \times [1, m] \times [1, |w|]\}$ be the set of disjoint intervals and position in w such that $t_{i_\downarrow}$ drops the first pebble in ρ at position i and $t_{i_\uparrow}$ is the first transition after t_i that removes the first pebble for each $(i_\downarrow, i_\uparrow, i) \in H_\rho$. For each interval $h = (i_\downarrow, i_\uparrow, i)$, let $(p_h, q_h) \in Q^2$ be the pair of states such that p_h (q_h) is the first (last) state of the interval h, i.e. $t_{i_\downarrow} = (p, a, p_h, \vec{b}_0, \downarrow)$ and $t_{i_\uparrow} = (q_h, a, q, \vec{b}_1, \uparrow)$ for some $p, q \in Q$ and $a \in \Gamma$. Since $\mathcal{A}_2$ is unambiguous, we know that for each interval $h = (i_\downarrow, i_\uparrow, i)$ there exists a unique pair of states (p_h, q_h) and, thus, this pair of states is determined and well defined for each h. Furthermore, for each pair of states (p, q) and position i in w we can define the function:

$$f_{p,q}(i) \ = \ \prod_{j=i_\downarrow+1}^{i_\uparrow-1} E(t_j)$$

if there exists an accepting run ρ and an interval $(i_\downarrow, i_\uparrow, i) \in H_\rho$ such that $p = p_h$ and $q = q_h$, and $f_{p,q}(i) = \mathbb{1}$ otherwise. In other words, $f_{p,q}(i)$ outputs the cost of the inner part of $\mathcal{A}_2$ when it drops the first pebble in position i starting in state p and ending in state q. Clearly, if we show that the function $f_{p,q}(x)$ can be defined with a formula $\theta_{p,q}(x) \in \mathrm{QMSO}(\Pi_x^{k+1}, \oplus_b, \odot)$, then we can define $\mathcal{A}_2$ in $\mathrm{QMSO}(\Pi_x^{k+2}, \oplus_b, \odot)$ by:

$$\theta_2 \ = \ \Pi x. \left(\prod_{p,q \in Q} \theta_{p,q}(x) \right)$$

where the inner product is a finite product for all combinations of $p, q \in Q$. To proof that $f_{p,q}(x)$ can be define with a formula $\theta_{p,q}(x) \in \mathrm{QMSO}(\Pi_x^{k+1}, \oplus_b, \odot)$, we argue that $\theta_{p,q}(x)$ can be computed with a $unamb$-2WA-k $\mathcal{A}_{p,q}$ and used the inductive hypothesis to get a formula in $\mathrm{QMSO}(\Pi_x^{k+1}, \oplus_b, \odot)$. The task of $\mathcal{A}_{p,q}$ to compute $f_{p,q}(x)$ is to check that there exists an accepting run such that $\mathcal{A}_2$ (1) drops the first pebble in position x and enters into state p, (2) removes the first pebble in position x from state q, and (3) multiplies the cost of the inner transitions between the two states p and q. Property (1) and (2) are regular properties over w (marked with position x) and can be checked with a two-way automata without pebbles [19]. If we know that property (1) and (2) holds over w (marked with position x), Property (3) can be defined with a $unamb$-2WA-k that just run the transitions of $\mathcal{A}_2$ starting from p in position x and ending when it reaches q in position x. Then the final automata is the sequential running of these three automata that computes (1), (2), and (3). By the previous lines, it is straightforward to give the explicit construction of $\mathcal{A}_{p,q}$ and, therefore, omit its definition. We conclude that $\theta_{p,q}(x)$ can be defined in $\mathrm{QMSO}(\Pi_x^{k+1}, \oplus_b, \odot)$ and, thus, we have that $\mathcal{A}_2$ is definable in $\mathrm{QMSO}(\Pi_x^{k+2}, \oplus_b, \odot)$.

We turn now on the definition of $\mathcal{A}_1$ in $\mathrm{QMSO}(\Pi_x^{k+2}, \oplus_b, \odot)$. First, notice that in this case the cost is computed in the outer intervals of an accepting run of $\mathcal{A}_1$. Moreover, the inner intervals and pebbles are just used to check a regular property based on the position where the first pebble is dropped and removed. To characterize this intuition, we use two-way automata with MSO-transitions proposed in [16] (which is a generalization of Lemma 3 in [21]). A two-way weighted automata with MSO-transitions is a tuple $\bar{\mathcal{A}} = (\Gamma, \mathbb{S}, \bar{Q}, \bar{E}, \bar{I}, \bar{F})$ such that $\bar{Q}$, $\bar{I}$, and $\bar{F}$ are defined as before, and $\bar{E}$ is a finite subset of $\bar{Q} \times \mathrm{MSO}[\Gamma, \le] \times \bar{Q} \times \{\leftarrow, \rightarrow\} \times \mathbb{S}$ where $\mathrm{MSO}[\Gamma, \le]$ are formulas in MSO over words in Γ with one free variable. Intuitively, transitions in $\bar{\mathcal{A}}$ are of the form $(p, \varphi(x), q, d, s)$ which means that if $\bar{\mathcal{A}}$ is in state p and $\varphi(i)$ is true where i is the current position of the reading head, then $\bar{\mathcal{A}}$ changes the control state to q and moves the reading head in the direction d with cost s. Configurations and accepting runs of $\bar{\mathcal{A}}$ over a word $w \in \Gamma^*$ are defined as usual. Also, we say that $\bar{\mathcal{A}}$ is a two-way *unambiguous* weighted automaton with MSO-transitions ($unamb$-$2\mathrm{WA}^{\mathrm{MSO}}$) if for every $w \in \Gamma$ there exists at most one accepting run of $\bar{\mathcal{A}}$ over w. The following result can be easily obtained from [16] (Theorem 7).

Theorem A.13. *[16] Let $\bar{\mathcal{A}}$ be a $unamb$-$2\mathrm{WA}^{\mathrm{MSO}}$ over a finite alphabet Γ and a semiring $\mathbb{S}$. Then there exists a $unamb$-$2\mathrm{WA}$ $\mathcal{A}'$ such that for all $w \in \Gamma^*$:*

$$[\![\bar{\mathcal{A}}]\!](w) = [\![\mathcal{A}']\!](w)$$

To be precise, the previous result is a rewriting of the original result which can be easily derived from [16]. Indeed, the original result is shown for two-way deterministic transducers which are the same as two-way deterministic weighted automata over the free semiring. To extend the original result from deterministic to unambiguous, the same techniques in [16] can be used to prove this result (see also [21], [8]).

For the last part of this proof, we show that $\mathcal{A}_1$ can be defined by a $unamb$-$2\mathrm{WA}^{\mathrm{MSO}}$ which proves that $\mathcal{A}_1$ can be defined in $\mathrm{QMSO}(\Pi_x^{k+2}, \oplus_b, \odot)$. Indeed, by Theorem A.13 this will imply that $\mathcal{A}_1$ is equivalent to a $unamb$-$2\mathrm{WA}$-0 and this is equivalent to a formula in $\mathrm{QMSO}(\Pi_x^{k+2}, \oplus_b, \odot)$ (actually, in $\mathrm{QMSO}(\Pi_x^{1}, \oplus_b, \odot)$) by Proposition 6.3. We proceed similar to the construction of $\mathcal{A}_2$. Let $\mathcal{A}_1 = (\Gamma, \mathbb{S}, Q_1, E_1, I_1, F_1)$ and $w \in \Gamma^*$. For each $p, q \in Q_1$, let $f_{p,q}(x)$ be a Boolean function over the positions of w such that for all $i \in \{1, \ldots, |w|\}$, $f_{p,q}(i)$ is true iff there exists a run $(p,i,i) \xrightarrow{t_1} (q_1, l_1, \vec{l}_1) \cdots (q_m, l_m, \vec{l}_m) \ldots \xrightarrow{t_{m+1}} (q,i,i)$ with $\vec{l}_j \neq \epsilon$ for $j \leq m$, that is, there exists a run of $\mathcal{A}_1$ starting in p with the first pebble and reading head placed in position i that reaches q with the reading head in position i without removing the first pebble. It is easy to prove that $f_{p,q}(x)$ is a regular property and can be defined by an MSO-formula over Γ. Indeed, we can define a two-way boolean automata with pebbles that simulates the non-zero part of $\mathcal{A}_1$ and checks that starting from (p,i,i) one can reach the configuration (q,i,i). For each $p, q \in Q_1$, let $\varphi_{p,q}(x)$ be the MSO-formula over Γ equivalent to $f_{p,q}(x)$.

Now, we define an $unamb$-$2\mathrm{WA}^{\mathrm{MSO}}$ $\bar{\mathcal{A}}$ equivalent to $\mathcal{A}_1$. The main idea of $\bar{\mathcal{A}}$ is to use the MSO-transitions to simulate the inner transitions of $\mathcal{A}_1$ that have cost $\mathbb{1}$. For the sake of simplification, we extend $unamb$-$2\mathrm{WA}^{\mathrm{MSO}}$ with a "stay" direction $\circlearrowleft$ that means that the reading head stay in the same position. Let $\bar{\mathcal{A}} = (\Gamma, \mathbb{S}, Q, \bar{E}, I, F)$ be a $unamb$-$2\mathrm{WA}^{\mathrm{MSO}}$ such that $\bar{E}$ is defined as follows:

- if $E_1(p, a, \vec{b}_0, q, d) = s$ with $d \in \{\leftarrow, \rightarrow\}$ then $(p, \{x \in P_a\}, q, d, s) \in \bar{E}$,
- if $E_1(p_1, a, \vec{b}_0, p, \downarrow) = s_1$ and $E_1(q, a, \vec{b}_1, p_2, \uparrow) = s_2$ then $(p_1, \{\varphi_{p,q}(x) \wedge x \in P_a\}, p_2, \circlearrowleft, s_1 \odot s_2) \in \bar{E}$.

From the previous definition one can easily check that $\mathcal{A}_1 \equiv \bar{\mathcal{A}}$. For any accepting run ρ of $\mathcal{A}_1$ over $w \in \Gamma^*$, one can get an accepting run of $\bar{\mathcal{A}}$ over w that *skips* the inner transitions of $\mathcal{A}_1$ by using the MSO-transitions and has the same total cost as ρ. For the other direction, for any accepting run of $\bar{\mathcal{A}}$ over w, one can derive an accepting run of $\mathcal{A}_1$ by extending the formulas $\varphi_{p,q}(x)$ into a sequence of inner transitions of $\mathcal{A}_1$ over w. Therefore, we conclude that $\mathcal{A}_1 \equiv \bar{\mathcal{A}}$. ■