An Automata Theoretic Characterization of Weighted First-Order Logic

Dhruv Nevatia, Benjamin Monmege

An Automata Theoretic Characterization of Weighted First-Order Logic*

Dhruv Nevatia[1] [0009−0008−0845−6754]
and Benjamin Monmege[2] [0000−0002−4717−9955]

[1] ETH Zurich, Switzerland dhruv.nevatia@inf.ethz.ch
[2] Aix Marseille Univ, LIS, CNRS, Marseille, France benjamin.monmege@univ-amu.fr

Abstract. Since the 1970s with the work of McNaughton, Papert and Schützenberger [23,21], a regular language is known to be definable in the first-order logic if and only if its syntactic monoid is aperiodic. This algebraic characterisation of a fundamental logical fragment has been extended in the quantitative case by Droste and Gastin [10], dealing with polynomially ambiguous weighted automata and a restricted fragment of weighted first-order logic. In the quantitative setting, the full weighted first-order logic (without the restriction that Droste and Gastin use, about the quantifier alternation) is more powerful than weighted automata, and extensions of the automata with two-way navigation, and pebbles or nested capabilities have been introduced to deal with it [5,19]. In this work, we characterise the fragment of these extended weighted automata that recognise exactly the full weighted first-order logic, under the condition that automata are polynomially ambiguous.

Keywords: Weighted logics, weighted automata, aperiodic monoids

1 Introduction

Early works by McNaughton, Papert and Schützenberger [23,21] have enabled an automata-theoretic characterisation of first-order logic over finite words: a regular language is definable in the first-order logic if and only if its syntactic monoid is aperiodic. From the minimal automaton recognising the language, we can compute its syntactic monoid and check aperiodicity to conclude. Moreover, from the aperiodic minimal automaton, we can deduce a first-order formula equivalent to it.

More recently, Droste and Gastin [10] have extended this result to deal with quantitative extensions of the first-order logic and automata. These quantitative extensions find their origin in the works of Schützenberger [22] that investigated *weighted automata*, and their expressive power in terms of *(formal power) series* that are mappings from finite words to weights of a *semiring*. Weighted automata

* We thank the reviewers that helped greatly improving the readability of this article. The work was partially done during an internship of the first author at Aix-Marseille Université, partially funded by CNRS IRL 2000 ReLaX.

were originally thought as finite state automata where each transition (as well as initial and final states) are equipped with weights of a semiring. Along a run, weights are combined by the multiplication of the semiring, while non-determinism is resolved by considering the sum of the weights of all accepting runs over a word. By changing the semiring we consider, weights can model cost, rewards, energy or probabilities in a unified way: see [12]. Many extensions have then been considered, by allowing for more structures than words (infinite words [15], trees [16], nested words [14,3]) and more weights than semirings (valuation monoids [13], multioperator monoids [18]).

In order to describe the series describable by weighted automata in a more readable way, it might be useful to have more high-level ways of description, like weighted logics based on monadic second order (MSO) features, introduced by Droste and Gastin [9]. Based on the seminal result by Büchi, Elgot, and Trakhtenbrot [6,17,24], they have explored a weighted extension of MSO logic on finite words and semirings: the semantics of disjunction and existential quantification are based on the sum of the semiring, while the ones of conjunction and universal quantification are based on the product. The appropriate restriction on the logic was found in order to obtain the exact same expressivity as weighted automata: a restriction is needed for combinatorial reasons, certain operators of the logic being able to generate series growing too quickly with respect to the length of the input word. In particular, universal quantifications must be used only once over very basic formulas, and conjunction is not allowed. Once again, this seminal result relating weighted automata and weighted logics has been extended in many ways: on trees [16], on nested words [14,3], with weights valuation monoids [13], to cite only a very few.

In [20], the semantics of weighted automata and weighted MSO logic has been revisited in a uniform way allowing one to obtain many previous results in a simplified way. First, an *abstract semantics* is defined, mapping each word to a multiset of sequences of weights (one sequence per accepting run): this abstract semantics does not depend on the weight structure, since no actual computation is made. The abstract semantics can then be aggregated into a single output weight by an ad-hoc operator: we call this the *concrete semantics*. Methodologically speaking, showing that two models have equal abstract semantics is sufficient (but not necessary in general) to show that they have equivalent concrete semantics.

In [10], Droste and Gastin consider the first-order fragment WFO of the weighted MSO logic, with the same kind of restrictions as the one explored for weighted MSO logic to recover the same expressive power as weighted automata. Under this restriction, they show that the logic WFO is expressively equivalent to weighted automata that are *aperiodic* (defined similarly as in the unweighted setting) and *polynomially ambiguous*. Moreover, the proof is constructive and works for the abstract semantics (and thus for any concrete semantics).

In order to express more properties than the restricted logics (WFO and weighted MSO), weighted automata with two-way navigation and pebbles or nested capabilities have been introduced [5,19], with an equivalent logic based on

an extension of WFO with some limited transitive closure operators. As noted in [10] by Droste and Gastin, this is thus natural to ask what the models of two-way nested/pebble weighted automata are that recognise exactly the full WFO logic. In this work, we answer this question: the series recognised by WFO logic can be obtained from two-way nested/pebble weighted automata that are aperiodic and polynomially unambiguous. This generalises the results of [10] only working for a small fragment of WFO, and one-way (non-nested) weighted automata, but the condition has a similar flavour. The aperiodicity condition on two-way automata models has been explored in [7]. Our proof is constructive and goes through a special case of two-way automata that are called *sweeping* where every change of direction happens on the border of the input word (and not in the middle). This allows us to more easily reuse the work by Droste and Gastin, which only works for one-way models.

After defining the weighted first-order logic we study in Section 2, and the nested two-way weighted automata in Section 3, we prove the equivalence between the various formalisms in subsequent sections: the translation from the logic to sweeping nested weighted automata is done in Section 4; sweeping nested weighted automata are translated back in the logic in Section 5. The most difficult part of the proof is the translation from two-way nested weighted automata to sweeping nested weighted automata: this does not hold if we do not have nesting mechanisms, and this translation thus raises the number of nesting necessary in the model.

2 Weighted First-Order Logic

In this section, we introduce the weighted first-order logic whose power we will characterise in the following with respect to some automata model. The logic used in [10] is a fragment of this logic where nesting of operations is limited to be as expressive as weighted automata.

Definition 1. *For a set* K *of weights and an alphabet* A*, we let* $\mathsf{WFO}(\mathsf{K}, A)$ *be the logic defined by the following grammar:*

$$\varphi ::= \top \mid P_a(x) \mid x \leq y \mid \neg\varphi \mid \varphi \wedge \varphi \mid \forall x\, \varphi \qquad (\mathsf{FO})$$

$$\Phi ::= \mathbf{0} \mid \mathbf{1} \mid k \mid \varphi ? \Phi : \Phi \mid \Phi + \Phi \mid \Phi \cdot \Phi \mid \Sigma_x \Phi \mid \Pi_x \Phi \mid \Pi_x^{-1} \Phi \qquad (\mathsf{WFO})$$

where $a \in A$*,* $k \in \mathsf{K}$ *and* x, y *are first order variables.*

Formulas φ stand for the classical (Boolean) first-order logic over words on the alphabet A. Their semantics is defined classically over words $u = u_1 u_2 \cdots u_n \in A^*$ and valuations $\sigma \colon \mathcal{V} \to \{1, 2, \ldots, n\}$ of the free variables $\mathcal{V}$ of the formula, letting $u, \sigma \models \varphi$ when the formula is satisfied by the word and the valuation.

Formulas Φ are weighted formulas that intuitively associate a weight with each word and valuation of free variables. As described in [19], the semantics is defined in two steps: first we give an *abstract* semantics associating with each word and valuation a multiset of sequences of weights in K; then we may define

a *concrete* semantics by describing how to fuse the multiset of sequences into a single weight. This differs from the classical semantics that directly compute the concrete semantics, but for our later proofs the other equivalent definition is much easier to manipulate.

Let $u \in A^*$ be a word and $\sigma \colon \mathcal{V} \to \{1, 2, \dots, n\}$ be a valuation where $\mathcal{V}$ is a set of variables. The abstract semantics of a WFO formula ϕ, with $\mathcal{V}$ as free variables, is denoted by $\{\!|\phi|\!\}_{\mathcal{V}}(u, \sigma)$: it is a multiset of sequences of weights, i.e. a series of $\mathbb{N}\langle\mathsf{K}^*\rangle$ mapping each sequence to its multiplicity in the multiset. As usual, we denote multisets via the symbols $\{\!\{.\}\!\}$. The disjoint union of two multisets is obtained as the sum of the associated series, it is denoted by $S_1 \cup S_2$. The product of two multisets is obtained as the Cauchy product of the associated series, it is denoted by $S_1 \cdot S_2 = \{\!\{s_1 s_2 \mid s_1 \in S_1, s_2 \in S_2\}\!\}$. This defines a structure of semiring on multisets where neutral elements are the empty multiset (i.e. the series mapping all sequences to 0), denoted by $\emptyset$, and the singleton $\{\!\{\varepsilon\}\!\}$ that only contains the empty sequence. The constants $\mathbf{0}$ and $\mathbf{1}$ of the logic represent those constants.

The semantics of WFO is defined inductively as follows:

$$\{\!|\mathbf{0}|\!\}_{\mathcal{V}}(u,\sigma) = \emptyset \qquad \{\!|\mathbf{1}|\!\}_{\mathcal{V}}(u,\sigma) = \{\!\{\varepsilon\}\!\} \qquad \{\!|k|\!\}_{\mathcal{V}}(u,\sigma) = \{\!\{k\}\!\}$$

$$\{\!|\varphi?\Phi_1 : \Phi_2|\!\}_{\mathcal{V}}(u,\sigma) = \begin{cases} \{\!|\Phi_1|\!\}_{\mathcal{V}}(u,\sigma) & \text{if } u, \sigma \models \varphi \\ \{\!|\Phi_2|\!\}_{\mathcal{V}}(u,\sigma) & \text{otherwise} \end{cases}$$

$$\{\!|\Phi_1 + \Phi_2|\!\}_{\mathcal{V}}(u,\sigma) = \{\!|\Phi_1|\!\}_{\mathcal{V}}(u,\sigma) \cup \{\!|\Phi_2|\!\}_{\mathcal{V}}(u,\sigma)$$

$$\{\!|\Phi_1 \cdot \Phi_2|\!\}_{\mathcal{V}}(u,\sigma) = \{\!|\Phi_1|\!\}_{\mathcal{V}}(u,\sigma) \cdot \{\!|\Phi_2|\!\}_{\mathcal{V}}(u,\sigma)$$

$$\{\!|\Sigma_x \Phi|\!\}_{\mathcal{V}}(u,\sigma) = \bigcup_{i \in \{1,2,\dots,|u|\}} \{\!|\Phi|\!\}_{\mathcal{V} \cup \{x\}}(u, \sigma[x \mapsto i])$$

$$\{\!|\Pi_x \Phi|\!\}_{\mathcal{V}}(u,\sigma) = \{\!|\Phi|\!\}_{\mathcal{V} \cup \{x\}}(u, \sigma[x \mapsto 1]) \cdots \{\!|\Phi|\!\}_{\mathcal{V} \cup \{x\}}(u, \sigma[x \mapsto |u|])$$

$$\{\!|\Pi_x^{-1} \Phi|\!\}_{\mathcal{V}}(u,\sigma) = \{\!|\Phi|\!\}_{\mathcal{V} \cup \{x\}}(u, \sigma[x \mapsto |u|]) \cdots \{\!|\Phi|\!\}_{\mathcal{V} \cup \{x\}}(u, \sigma[x \mapsto 1])$$

For sentences (formulas without free variables), we remove the set $\mathcal{V}$ of variables as well as the valuation σ from the notation. Given a series $f \in (\mathbb{N}\langle\mathsf{K}^*\rangle)\langle A^*\rangle$ we say that f is WFO-*definable* if there exists a sentence Φ_f such that for all words $u \in A^*$, $f(u) = \{\!|\Phi_f|\!\}(u)$.

We also define the 1-way fragments $\mathsf{WFO}^{\to}$ and $\mathsf{WFO}^{\leftarrow}$ by discarding binary product $(\cdot)$, as well as Π_x^{-1} and Π_x, respectively.

The fragment $\mathsf{rWFO}^{\to}$ of logic studied in [10] is obtained by the following grammar:

$$\Psi ::= k \mid \varphi?\Psi : \Psi \qquad (\mathsf{step\text{-}wFO})$$

$$\Phi ::= \mathbf{0} \mid \varphi?\Phi : \Phi \mid \Phi + \Phi \mid \Sigma_x \Phi \mid \Pi_x \Psi \qquad (\mathsf{rWFO}^{\to})$$

where $k \in K$, φ is a formula of FO, and x is a first order variable.

Notice that the abstract semantics of a formula from $\mathsf{step\text{-}wFO}$ maps every word to a singleton multiset. Since $\mathbf{1}$ is removed, as well as the binary product, and Π_x is restricted to $\mathsf{step\text{-}wFO}$ formulas, it is easy to check inductively that the

abstract semantics of a formula from $\mathsf{rWFO}^{\to}$ maps every word u to a multiset of sequences of weights all of the length of u.

To interpret the abstract semantics in terms of a single quantity, we moreover provide an aggregation operator $\mathsf{aggr}\colon \mathbb{N}\langle\mathsf{K}^*\rangle \to S$ to a set S of weights. The concrete semantics of a formula Φ is then obtained by applying aggr over the multiset obtained via the abstract semantics. The set S can be equipped of various algebraic structures, like semirings or valuation monoids [11], as explained in [19]. In the case of a semiring, we for instance let $\mathsf{aggr}(f)$ be the sum over all sequences $k_1k_2\cdots k_n$ of $f(k_1k_2\cdots k_n) \times k_1 \times k_2 \times \cdots \times k_n$.

Example 1. As a first example, consider as a set of weights the languages over the alphabet A. It is naturally equipped with a structure of semiring, where the addition is the union of languages and the multiplication is the concatenation of languages. This semiring is non-commutative which validates our introduction of two product quantification operators, one from left to right and one from right to left. For instance, suppose we want to compute the mapping $f\colon A^* \to 2^{A^*}$ that associates to a word u all the words of the form $\tilde{w}\tilde{w}$ with w all factors of u (i.e. consecutive letters taken inside u), where $\tilde{w}$ denotes the mirror image of the word w. For instance, $f(abb) = \{aa, bb, baba, bbbb, bbabba\}$. We can describe this function via a formula of WFO as follows. We suppose that $A = \{a, b\}$ to simplify, and we let $\mathsf{K} = \{\mathbf{a}, \mathbf{b}\}$ be the weights that represent the singleton languages $\{a\}$ and $\{b\}$. Then, we describe a formula $\mathsf{mirror\text{-}factor}(x, y)$ that computes the mirror image of the factor in-between positions pointed by x and y:

$$\mathsf{mirror\text{-}factor}(x, y) = \Pi_z^{-1}\,(x \leq z \wedge z \leq y)?(P_a(z)?\mathbf{a} : \mathbf{b}) : \mathbf{1}$$

Then, the mapping f can be described with the formula Φ:

$$\Sigma_x\Sigma_y\,(x \leq y)?[\mathsf{mirror\text{-}factor}(x, y) \cdot \mathsf{mirror\text{-}factor}(x, y)] : \mathbf{0}$$

The abstract semantics of the formula associates a multiset of words $\tilde{w}\tilde{w}$ with w all factors of u. For instance, $\{\!|\Phi|\!\}(aa) = \{\!\{\mathbf{aa}, \mathbf{aa}, \mathbf{aaaa}\}\!\}$. To provide a concrete semantics, we simply consider the aggregation operator that computes the product of sets of weights and removes duplicates in multisets. □

Example 2. As a second example, consider the alphabet $A = \{a, b\}$, and the natural semiring $(\mathbb{N}, +, \times, 0, 1)$, i.e. the aggregation operator that naturally comes with a semiring. It is a commutative semiring, thus the operator Π^{-1} becomes semantically equivalent (with respect to the concrete semantics, but not to the abstract one) to Π. Consider the series $f\colon A^* \to \mathbb{N}$ defined for all words $u \in A^*$ by $f(u) = |u|_a{}^{|u|_b}$, where $|u|_c$ denotes the number of a given letter c in the word u. This series can be defined by the following formula, where we intentionally reuse the same variable name twice (but the semantics would be unchanged if the internal variable x was renamed y): $\Pi_x\,(P_b(x)?\Sigma_x\,(P_a(x)?1 : \mathbf{0}) : \mathbf{1})$. The abstract semantics maps a word with m letters a and n letters b to the multiset containing m^n copies of the sequence 1. For instance, for the word $abbaa$, the abstract semantics computes $\{\!\{\varepsilon\}\!\} \cdot \{\!\{1, 1, 1\}\!\} \cdot \{\!\{1, 1, 1\}\!\} \cdot \{\!\{\varepsilon\}\!\} \cdot \{\!\{\varepsilon\}\!\}$, where we have decomposed it with respect to the outermost Π_x operator. Once aggregated, all sequences map to 1, and we thus count m^n as expected. □

3 Nested Two-Way Weighted Automata

Weighted automata are a well-studied model of automata equipped with weights, introduced by Schützenberger [22]. They have been extended to several weight structures (semirings, valuation monoids), once again with a unified abstract semantics introduced in [19]. They also have been extended with two-way navigation, and pebbles or nested capabilities, in order to get more power [5,20]. In order to simplify our later proofs, we first redefine the semantics of the nested two-way weighted automata with the abstract semantics seen above for the logic.

Since we consider two-way navigation in a word, it is classical to frame the finite word by markers, both on the left and on the right, so that the automaton knows the boundary of the domain. We denote by $\triangleright, \triangleleft$ the left and right markers of the input word, respectively, that are supposed to be symbols not already present in the alphabet A we consider.

Definition 2. *First, by convention, we let (-1)-nested two-way weighted automata to be constants of K. Then, for $r \geq 0$, we let r-$\mathsf{NWA}(\mathsf{K}, A)$ (or, r-NWA if K and A are clear from the context) be the class of r-nested two-way weighted automata over a finite set K of constants and alphabet A, that are all tuples $\mathcal{A} = \langle Q, \mathsf{Tr}, I, F \rangle$ where*

- *Q is a finite set of states;*
- *Tr is the transition relation split into two subsets.*
  1. *For $a \in A$, there are transitions of the form $(q, a, \mathcal{B}, d, q') \in Q \times A \times (r-1)$-$\mathsf{NWA}(\mathsf{K}, A \times \{0,1\}) \times \{\leftarrow, \rightarrow\} \times Q$, meaning that the automaton is in state q, reads the letter a, calls the $(r-1)$-nested two-way weighted automaton $\mathcal{B}$ over the same set K of weights and alphabet $A \times \{0,1\}$ (used to mark the current position), decides to move in the d-direction, and changes its state to q'.*
  2. *For $a \in A \cup \{\triangleright, \triangleleft\}$, there are some other transitions where the automaton $\mathcal{B}$ is replaced by a weight from K, or by the special constant $\mathbf{1}$ (that we used in the logic WFO) to forbid the call of a nested automaton (especially on the markers): these transitions are thus of the form*

$$\begin{aligned}(q, a, k, d, q') \in \quad & \big(Q \times A \times (\mathsf{K} \cup \{\mathbf{1}\}) \times \{\leftarrow, \rightarrow\} \times Q\big) \\ & \cup \big(Q \times \{\triangleright\} \times (\mathsf{K} \cup \{\mathbf{1}\}) \times \{\rightarrow\} \times Q\big) \\ & \cup \big(Q \times \{\triangleleft\} \times (\mathsf{K} \cup \{\mathbf{1}\}) \times \{\leftarrow\} \times Q\big)\end{aligned}$$

  where we have chosen to remove the possibility to move right on a right marker, and left on a left marker (to avoid exiting the possible positions in the input word);
- *$I \subseteq Q$ is the set of initial states;*
- *$F \subseteq Q$ is the set of final states.*

An automaton $\mathcal{B}$ that appears in the transitions of an automaton $\mathcal{A}$ is called a *child* of $\mathcal{A}$, and reciprocally $\mathcal{A}$ is a *parent* of $\mathcal{B}$ (notice that an automaton could

have several parents, since it can appear in the transitions of several automata). This describes a directed acyclic relationship of dependency between automata: we thus say that an automaton is a *descendant* of another one if they are related by a sequence of parent-child relationship. The unique automaton with no ancestors shall be called the *root*.

In the following a transition of the form (q, a, x, d, q') is said to go from state q to state q', reading letter a, having weight x, and is called a d-transition.

We now define the abstract semantics of an r-$\mathsf{NWA}(\mathsf{K}, A)$ $\mathcal{A}$, mapping each word $u \in A^*$ to a multiset of sequences of weights $\{\!|\mathcal{A}|\!\}(u) \in \mathbb{N}\langle\mathsf{K}^*\rangle$. Configurations of such an automaton are tuples (u, i, q) where $u = u_1 \cdots u_n$ is the word in $\{\varepsilon, \triangleright\}A^*\{\varepsilon, \triangleleft\}$ (that could start and end with the markers, or not, in order to be able to define subruns on an unmarked word, that we will use later), $i \in \{1, \ldots, n\}$ is a position in the word, and $q \in Q$ is the current state. We call run a sequence $\rho = (u, i_0, q_0) \xrightarrow{\delta_0, f_0} (u, i_1, q_1) \xrightarrow{\delta_1, f_1} \cdots \xrightarrow{\delta_m, f_m} (u, i_m, q_m)$, where $i_0, \ldots, i_{m-1} \in \{1, \ldots, n\}$, $i_m \in \{0, 1, \ldots, n, n+1\}$ is the final position (that could *exit* the word on left or right) $\delta_0, \ldots, \delta_m \in \mathsf{Tr}$ and $f_0, \ldots, f_m$ are multisets in $\mathbb{N}\langle\mathsf{K}^*\rangle$ such that for all $j \in \{0, \ldots, m-1\}$:

- δ_j is a transition from state q_j to state q_{j+1} reading letter u_{i_j};
- if δ_j is a $\rightarrow$-transition then $i_{j+1} = i_j + 1$, otherwise $i_{j+1} = i_j - 1$;
- if $u_{i_j} \in A$ and the transition has weight $\mathcal{B}$ that is a $(r-1)$-$\mathsf{NWA}(\mathsf{K}, A \times \{0,1\})$, then $f_j = \{\!|\mathcal{B}|\!\}(u')$ where u' is the word over alphabet $A \times \{0,1\}$, that will later be denoted by (u, i_j), whose left component is u and whose right component is the constant 0 except at position i_j where it is 1;
- if the transition has weight $k \in K$, then $f_j = \{\!\{k\}\!\}$,
- if the transition has weight $\mathbf{1}$, then $f_j = \{\!\{\varepsilon\}\!\}$.

The *initial position* of the run is i_0, and its *final position* is i_m. The run is *accepting* if $q_0 \in I$, $q_m \in F$. Notice that we do not require runs to start on the left marker and stop at the right marker. The weight $\mathsf{wt}(\rho)$ of this run is given as the product of multisets $f_0 \cdot f_1 \cdots f_m$.

A run is called *simple* if it never goes twice through the same configuration. Not all runs are simple, but we restrict ourselves to using only those in the semantics: otherwise, an infinite number of runs should be summed, which would produce an infinite multiset (and then the aggregator function should be extended to add this possible behaviour). This restriction was also considered in [5,20].

We then let $\{\!|\mathcal{A}|\!\}(u)$ be the union (as multiset) of the weights of accepting simple runs (whatever their initial and final positions). As for the logics above, we may then use an aggregation operator to obtain a *concrete semantics* $[\![\mathcal{A}]\!]$ mapping each word u to a weight structure S.

Given a series $f \in (\mathbb{N}\langle\mathsf{K}^*\rangle)\langle A^*\rangle$ we say that f is NWA-*definable* if there exists $r \geq 0$ and an r-$\mathsf{NWA}(\mathsf{K}, A)$ $\mathcal{A}$ such that for all words $u \in A^*$, $f(u) = \{\!|\mathcal{A}|\!\}(u)$.

Example 3. We describe in Figure 1 a 2-NWA $\mathcal{A}$ that recognises the series described in Example 1. Two levels of nesting are used to mark non-deterministically the positions x and y, such that $x \leq y$ (or only one of them if $x = y$).

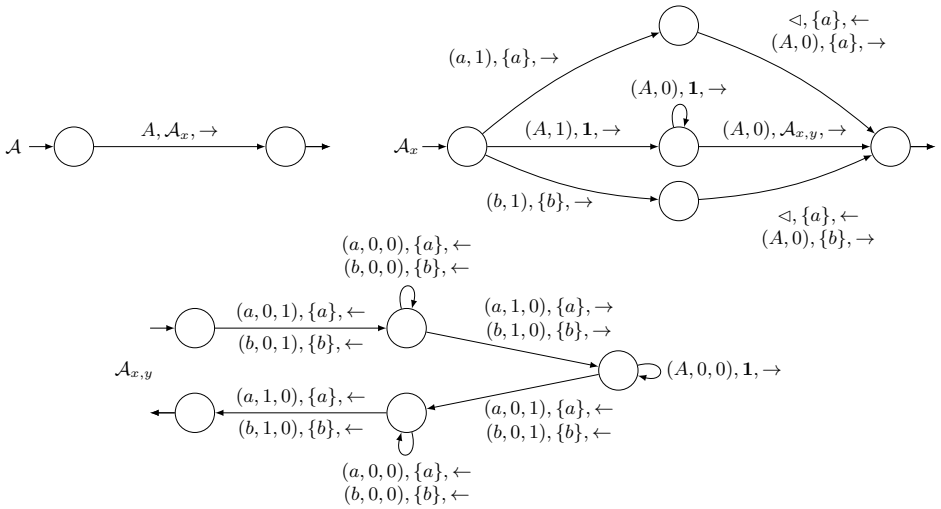

Fig. 1. An 2-NWA that recognises the series described in Example 1. The letter A is used in transitions to denote the presence of all possible letters from A. Notice that the runs of all the automata may start and stop at any position of the word, as described in the semantics.

Then, the last level of nesting is used to compute the value of the formula $\mathsf{mirror\text{-}factor}(x,y) \cdot \mathsf{mirror\text{-}factor}(x,y)$ by two passes from right to left. □

A run over a word u is called *left-to-right* (resp. *left-to-left*, *right-to-right*, *right-to-left*) if its initial position is 1 (resp. 1, $|u|$, $|u|$) and its final position is $|u|+1$ (resp. 0, $|u|+1$, 0). Intuitively, we thus use this terminology to detect if a run starts on the first or last position of the word, and if it exits the word either on the left or on the right.

Navigational restrictions. An r-NWA$^{\rightarrow}$ (respectively, r-NWA$^{\leftarrow}$) is an r-NWA where all transitions appearing in the automaton or its descendants are $\rightarrow$-transitions (resp. $\leftarrow$-transitions). Those models are called *one-way* in the following, since the head movement is fixed during the whole run.

An r-nested *sweeping* weighted automaton (r-swNWA) is an r-NWA where changes of directions are only allowed (in this automaton or its descendants) at markers. More formally, states of the automaton and each of its descendants are separated in two sets $Q^{\rightarrow}$ and $Q^{\leftarrow}$ such that for all transitions $(q, a, \mathcal{B}, d, q')$ or (q, a, k, d, q'),

- if $q, q' \in Q^{\rightarrow}$, then $d = \rightarrow$;
- if $q, q' \in Q^{\leftarrow}$, then $d = \leftarrow$;
- if $q \in Q^{\rightarrow}$ and $q' \in Q^{\leftarrow}$, then $d = \leftarrow$ and $a = \lhd$;
- if $q \in Q^{\leftarrow}$ and $q' \in Q^{\rightarrow}$, then $d = \rightarrow$ and $a = \rhd$.

Ambiguity. An r-NWA $\mathcal{A}$ is called *polynomially ambiguous* if there is a polynomial p such that over every word u on its alphabet the number of accepting

runs of $\mathcal{A}$, as well as the number of accepting runs of any of its descendants, is at most $p(|u|)$. If the polynomial p is linear, $\mathcal{A}$ is said to be *linearly ambiguous*. If the polynomial p is the constant 1, $\mathcal{A}$ is said to be *unambiguous*. Notice that the condition deals with all runs, and not only the simple ones.

Polynomial ambiguity (indeed even *finite* ambiguity, where the number of accepting runs must be finite for all words) implies that all accepting runs are simple: otherwise, there would be an infinite number of accepting runs, by allowing the loops to happen as many times as possible.

The 2-NWA of Figure 1 is linearly ambiguous since the toplevel automaton $\mathcal{A}$ has only to choose the position where to start the run, the automaton $\mathcal{A}_x$ has then only to choose the position where to call the next automaton, and the automaton $\mathcal{A}_{x,y}$ is unambiguous.

Aperiodicity. In order to define a notion of aperiodicity for NWA, we need to enhance the usual notion of aperiodicity for automata to incorporate weights, two-way navigations, and nesting. As in [10], we simply do not care about weights and thus require that the unweighted version of the automata are aperiodic. For two-way navigations, we rely on existing extensions of the notion of *transition monoid* for two-way automata and transducers [1,2,7]. Finally, for nesting, we simply require that each automaton appearing in an NWA is aperiodic.

More formally, given a NWA $\mathcal{A}$ over the alphabet A, its transition monoid is the quotient of the free monoid A^* by a congruence relation capturing the equivalence of two *behaviours* of the automaton. As for runs, we distinguish four types of behaviours: left-to-left, left-to-right, right-to-left and right-to-right. The left-to-left behaviour $\mathsf{bh}_{ll}^{\mathcal{A}}(w)$ of $w \in \{\varepsilon, \triangleright\}A^*\{\varepsilon, \triangleleft\}$ in $\mathcal{A}$ is the set of pairs of states (p, q) such that there exists a left-to-left run over w from state p to state q (notice that we do not care if the descendant automata that are called along this run are indeed "accepting" the word). The other behaviours can be defined analogously.

Definition 3. *Let $\mathcal{A} = \langle Q, \mathsf{Tr}, I, F\rangle$ be a $\mathsf{NWA}(\mathsf{K}, A)$. The transition monoid of $\mathcal{A}$ is $A^*\backslash \sim_{\mathcal{A}}$ where $\sim_{\mathcal{A}}$ is the conjunction of the following congruence relations, defined for $w, w' \in A^*$ by:*

- $w \sim_{ll}^{\mathcal{A}} w'$ *iff* $\mathsf{bh}_{ll}^{\mathcal{A}}(w) = \mathsf{bh}_{ll}^{\mathcal{A}}(w')$
- $w \sim_{lr}^{\mathcal{A}} w'$ *iff* $\mathsf{bh}_{lr}^{\mathcal{A}}(w) = \mathsf{bh}_{lr}^{\mathcal{A}}(w')$
- $w \sim_{rl}^{\mathcal{A}} w'$ *iff* $\mathsf{bh}_{rl}^{\mathcal{A}}(w) = \mathsf{bh}_{rl}^{\mathcal{A}}(w')$
- $w \sim_{rr}^{\mathcal{A}} w'$ *iff* $\mathsf{bh}_{rr}^{\mathcal{A}}(w) = \mathsf{bh}_{rr}^{\mathcal{A}}(w')$

Notice that in the previous definition, we only focus on words not containing markers. This is because we only use this monoid in order to define aperiodicity of the automata where we focus on powers of elements of the monoid, which correspond to runs on iterates of a word in which it makes no sense to duplicate some markers.

An r-NWA is *aperiodic* if its transition monoid, as well as the ones of all its descendants, are aperiodic (i.e. for all elements x of the monoid, there is a natural number n such that $x^n = x^{n+1}$).

Given an NWA $\mathcal{A}$, its *left-to-right* (resp. *right-to-left*) projection is the NWA $\overrightarrow{\mathcal{A}}$ (resp. $\overleftarrow{\mathcal{A}}$) obtained by only keeping $\to$-transitions (resp. $\leftarrow$-transitions) in the root automaton. Interestingly, when starting from sweeping automata, aperiodicity is preserved when taking such projections.

Lemma 1. *If a* swNWA $\mathcal{A}$ *is aperiodic then* $\overrightarrow{\mathcal{A}}$ *and* $\overleftarrow{\mathcal{A}}$ *are aperiodic.*

Proof. Let $\mathcal{A} = \langle Q, \mathsf{Tr}, I, F\rangle$ be a $\mathsf{swNWA}(K, A)$. We prove the result for $\overrightarrow{\mathcal{A}}$. The proof for $\overleftarrow{\mathcal{A}}$ follows analogously.

Consider the word $u \in A^*$. Then there exists a natural number k such that $\mathsf{bh}_e^{\mathcal{A}}(u^k) = \mathsf{bh}_e^{\mathcal{A}}(u^{k+1})$ for $e \in \{ll, lr, rl, rr\}$. If $u = \varepsilon$, then $\mathsf{bh}_e^{\overrightarrow{\mathcal{A}}}(u) = \{(p,p) \mid p \in Q\} = \mathsf{bh}_e^{\overrightarrow{\mathcal{A}}}(u^2)$ for $e \in \{ll, lr, rl, rr\}$. It remains to prove the lemma when u is not empty. Immediately, we have $\mathsf{bh}_{ll}^{\overrightarrow{\mathcal{A}}}(u) = \mathsf{bh}_{ll}^{\overrightarrow{\mathcal{A}}}(u^2) = \mathsf{bh}_{rl}^{\overrightarrow{\mathcal{A}}}(u) = \mathsf{bh}_{rl}^{\overrightarrow{\mathcal{A}}}(u^2) = \emptyset$. Since no run of the sweeping automaton $\mathcal{A}$ over u can change the direction of its head movement over w that does not contain end markers, we have $\mathsf{bh}_{lr}^{\overrightarrow{\mathcal{A}}}(u^k) = \mathsf{bh}_{lr}^{\mathcal{A}}(u^k) = \mathsf{bh}_{lr}^{\mathcal{A}}(u^{k+1}) = \mathsf{bh}_{lr}^{\overrightarrow{\mathcal{A}}}(u^{k+1})$. Finally, $\mathsf{bh}_{rr}^{\overrightarrow{\mathcal{A}}}(u) = \{(p,q) \mid (p, u_{|u|}, x, \to, q) \in \mathsf{Tr}\} = \mathsf{bh}_{rr}^{\overrightarrow{\mathcal{A}}}(u^2)$.

For every word $u \in A^*$, we thus have $\mathsf{bh}_e^{\overrightarrow{\mathcal{A}}}(u^{k'}) = \mathsf{bh}_e^{\overrightarrow{\mathcal{A}}}(u^{k'+1})$ for $e \in \{ll, lr, rl, rr\}$, where $k' = \max(1, k)$. Hence, $\overrightarrow{\mathcal{A}}$ is aperiodic. □

Results. The goal is to find an adequate characterisation of the automata models that recognise exactly the series WFO-definable or the fragments introduced before. Droste and Gastin [10] paved the way of this study by characterising $\mathsf{rWFO}^{\to}$ with a fragment of the classical one-way weighted automata, that are 0-$\mathsf{NWA}^{\to}$ where the semantics is computed by only considering accepting runs that start on the first letter of the word, and end on the last letter of the word: this is because of the specific type of multiset produced by formulas of $\mathsf{rWFO}^{\to}$, all elements being of the same length (the length of the input word). In particular, markers are useless in this context.

Theorem 1 ([10]). *For all series* $f \in (\mathbb{N}\langle \mathsf{K}^*\rangle)\langle A^*\rangle$*, the following conditions are equivalent:*

1. *f is definable by a polynomially ambiguous aperiodic* 0-$\mathsf{NWA}^{\to}$
2. *f is* $\mathsf{rWFO}^{\to}$*-definable.*

Here, and in the following, classes $\mathcal{C}_1$ and $\mathcal{C}_2$ of models are said to be equivalent if for all models $\mathcal{M}_1 \in \mathcal{C}_1$ and $\mathcal{M}_2 \in \mathcal{C}_2$ working on the same alphabet and the same sets of weights, and for all words u, the abstract semantics $\{\!|\mathcal{M}_1|\!\}(u)$ is equal to the abstract semantics $\{\!|\mathcal{M}_2|\!\}(u)$. This then implies that, for every aggregation function, the concrete semantics are also the same.

The proof of this theorem is constructive in both directions, and we will revisit it in the next sections, providing generalisations of it in order to get our main contribution:

Theorem 2. *For all series $f \in (\mathbb{N}\langle\mathsf{K}^*\rangle)\langle A^*\rangle$, the following conditions are equivalent:*

1. *f is definable by a linearly ambiguous aperiodic* NWA*;*
2. *f is definable by a polynomially ambiguous aperiodic* NWA*;*
3. *f is* WFO*-definable;*
4. *f is definable by a linearly ambiguous aperiodic* swNWA
5. *f is definable by a polynomially ambiguous aperiodic* swNWA*.*

The rest of the article is devoted to a sketch of the proofs of Theorem 2. We provide in Section 4 the sketch of proof of $3 \Rightarrow 4$, in Section 5 the sketch of proof of $5 \Rightarrow 3$, and in Section 6 the sketch of proof of $2 \Rightarrow 5$. We can then conclude by the trivial implications $4 \Rightarrow 1 \Rightarrow 2$.

As a side result, we also obtain a characterisation for one-way models:

Theorem 3. *For all series $f \in (\mathbb{N}\langle\mathsf{K}^*\rangle)\langle A^*\rangle$, the following conditions are equivalent:*

1. *f is definable by a linearly ambiguous aperiodic* $\mathsf{NWA}^{\rightarrow}$*;*
2. *f is definable by a polynomially ambiguous aperiodic* $\mathsf{NWA}^{\rightarrow}$*;*
3. *f is* $\mathsf{WFO}^{\rightarrow}$*-definable.*

These theorems complete the picture initiated in [5, Theorem 5.11] where it is shown that, in commutative semirings, NWA (called *pebble two-way weighted automata*, with a more operational view of dropping/lifting pebbles, but the expressive power is identical) are equivalent to an extension of the logic WFO with a bounded weighted transitive closure operator. It is also noted that, even in non commutative semirings, the whole logic $\mathsf{WFO}^{\rightarrow}$ with the bounded transitive closure operator can be translated into equivalent NWA.

4 From the Logic to Automata

In this section, we prove the implication $3 \Rightarrow 4$ of Theorem 2. This is obtained by a generalisation of the proof given by Droste and Gastin in [10, Theorem 16], where they only deal with restricted one-way logic and non-nested one-way automata. The proof is indeed simpler since we can rely on the use of nesting, contrary to them where they need a careful construction for formulas $\Pi_x \Psi$ of $\mathsf{rWFO}^{\rightarrow}$.

The construction is performed by induction on the formula of WFO, making use of nesting in automata, as originally demonstrated in [5, Proposition 5.13] to transform every formula of a logic containing WFO (as well as a bounded weighted transitive closure operator) into NWA.

As known since [23,21], from every formula φ of FO, we can obtain an equivalent classical deterministic finite state automaton that is aperiodic, starts on the marker $\triangleright$ and ends on the marker $\triangleleft$. By putting on every transition the weight $\mathbf{1}$, this results in a 1-$\mathsf{NWA}^{\rightarrow}$ $\mathcal{A}_\varphi$ that is unambiguous and aperiodic, whose abstract semantics is equal to the formula $\varphi?\mathbf{1} : \mathbf{0}$.

Consider then a formula $\Phi = \varphi ? \Phi_1 : \Phi_2$, where, by induction, we already have two r-NWA $\mathcal{A}_1$ and $\mathcal{A}_2$ for Φ_1 and Φ_2 (without loss of generality we adapt the maximal level r of nesting by adding useless levels). We can use the 1-NWA$^{\rightarrow}$ $\mathcal{A}_\varphi$ in order to produce an r-NWA equivalent to Φ: the first level consists in $\mathcal{A}_\varphi$, and once it unambiguously reach the marker $\lhd$, we continue the run by going back to the left marker, and continue either to $\mathcal{A}_1$ or to $\mathcal{A}_2$, whether the formula φ was concluded to be satisfied or not, respectively.[3]

The sum and product of two formulas can be computed by taking the disjoint union of two automata, or by starting the computation of the second after the computation of the first one (either by going back to the beginning of the word, or using a level of nesting).

For the quantification operators, we use one more level of nesting. Suppose that we have an r-NWA$(\mathsf{K}, A \times \{0,1\})$ $\mathcal{A}$ equivalent to a formula Φ with a free variable x. Then, the formula $\Sigma_x \Phi$ can be defined by the following $(r+1)$-NWA(K, A), making use of the fact that we can non-deterministically start and end a run wherever we want: the automaton thus has a single transition that calls $\mathcal{A}$. For the Π_x operator, the toplevel automaton scans the whole word from left to right, and calls $\mathcal{A}$ on each position (that is not a marker). For the Π_x^{-1} operator, we do the same but starting from the right marker and scanning the whole word from right to left. In both the cases, the root of the resulting automaton is aperiodic.

To conclude that the constructed NWA is linearly ambiguous and aperiodic, we make use of the fact that linearly ambiguous aperiodic automata are closed under disjoint union, nesting and concatenation with unambiguous (even finitely ambiguous) automata. It is indeed true for the case of disjoint unions, the individual automata still preserve the aperiodicity in their simulations and any run in the new automaton must be restricted to one of the automata. In the case of nesting, every transition of the *soon-to-be-child* automaton is replaced by all its extensions with respect to the input letter. Since the transitions are now oblivious to the marking of an input, the aperiodicity of the new child automaton is once again ensured under the extended alphabet. To understand the closure of linear ambiguity of automata under concatenation with unambiguous automata, one must just observe that the concatenation of automata essentially multiples the ambiguities of the factor automata, since every run in the concatenation is the sequence of a run in the first factor and one in the second.

5 From Nested Sweeping Weighted Automata to the Logic

This section aims at proving the implication $5 \Rightarrow 3$ of Theorem 2. We shall first prove it in the 1-way case.

[3] In the proof of Theorem 3, we replace this construction by the use of nesting that allows one to restart from the first position of the word in order to compute the behaviour of either $\mathcal{A}_1$ or $\mathcal{A}_2$.

Lemma 2. *For all polynomially ambiguous aperiodic r-$\mathsf{NWA}^{\rightarrow}$ (resp. r-$\mathsf{NWA}^{\leftarrow}$), there exists an equivalent formula of $\mathsf{WFO}^{\rightarrow}$ (resp. $\mathsf{WFO}^{\leftarrow}$).*

Proof. Once again, we only deal with the left-to-right result, the other one being obtained symmetrically. Let $\mathcal{A}_{p,q}$ denote the r-$\mathsf{NWA}^{\rightarrow}$ obtained from $\mathcal{A}$ where the initial and final states are p and q respectively. We prove by induction on r, that for all r-$\mathsf{NWA}^{\rightarrow}$ $\mathcal{A}$ and each pair of states p and q, we can construct a $\mathsf{WFO}^{\rightarrow}$ sentence $\Phi_{p,q}$ such that $\{\!|\mathcal{A}_{p,q}|\!\} = \{\!|\Phi_{p,q}|\!\}$. We then conclude by considering all initial states p and final states q.

If $r = 0$, the result follows, after trimming the root of $\mathcal{A}$ so that all states can be reached from an initial state and reach a final state (no matter if the descendant automata called on the transition indeed accept the word), directly from the construction of Droste and Gastin [10, Proposition 9 and Theorem 10]. Note that trimming the automaton does not alter its semantics. The main difference in our case is the fact that our automata can non-deterministically start and end in the middle of the word. We may however start by modifying them to force them to start on the left marker and end on the right marker: it suffices to add self-loop transitions at the beginning and the end of weight $\mathbf{1}$ (so that these additional transitions do not modify the abstract semantics).

We now suppose that $r > 0$, and assume that the result holds for $r - 1$. Consider an r-$\mathsf{NWA}^{\rightarrow}(\mathsf{K}, A)$ $\mathcal{A}$ that we suppose trimmed. As in the previous case, we can produce a formula Φ for $\mathcal{A}$, abstracting away for now the weight $k_{\mathcal{B}}$ on the transitions that stands for a $(r-1)$-$\mathsf{NWA}^{\rightarrow}(\mathsf{K}, A \times \{0,1\})$ $\mathcal{B}$.

We use the induction hypothesis to produce a formula $\Phi_{\mathcal{B}}$ of WFO for every $(r-1)$-$\mathsf{NWA}^{\rightarrow}(\mathsf{K}, A \times \{0,1\})$ $\mathcal{B}$ that appears in the transitions of $\mathcal{A}$. We modify this formula so that we incorporate a fresh first order variable x standing for the position on which $\mathcal{B}$ is called. Then, we replace every subformula $P_{(a,i)}(y)$ with $(a,i) \in A \times \{0,1\}$ by $P_a(y) \wedge y = x$ if $i = 1$, $P_a(y) \wedge y \neq x$ if $i = 0$.

In the formula Φ produced by Droste and Gastin, each weight $k_{\mathcal{B}}$ appears in a subformula with a distinguished first order variable x encoding the position of the letter read by the transition that should compute the weight $k_{\mathcal{B}}$. Thus, we simply replace every such weight $k_{\mathcal{B}}$ by the modified formula $\Phi_{\mathcal{B}}$ above. □

We then turn to the case of sweeping automata.

Lemma 3. *For every polynomially ambiguous aperiodic swNWA, there exists an equivalent formula of WFO.*

Proof. The proof also goes by induction on the level of nesting, and follows the same construction as the previous lemma. The only novelty is the treatment of change of directions in the runs. We thus only consider the case of 0-swNWA below.

For a 0-swNWA $\mathcal{A} = \langle Q, \mathsf{Tr}, I, F\rangle$, we show that for each pair of states p and q, we can construct a formula of WFO $\Phi_{p,q}$ equivalent to $\mathcal{A}_{p,q}$. As before, without loss of generality, we can suppose that every accepting run starts on the left marker, and stops on the right marker.

Given a word $w = w_1 \cdots w_m$, every run from p to q on $\rhd w \lhd$ can then be decomposed as

$$(p, \rhd, k_0, \rightarrow, p_0)\rho_1(p_1, \lhd, k_2, \leftarrow, p_2)\rho_2(p_3, \rhd, k_4, \rightarrow, p_4)\cdots \\ (p_{2n-1}, \rhd, k_{2n}, \rightarrow, p_{2n})\rho_{2n+1}(p_{2n+1}, \lhd, k_{2n+2}, \leftarrow, q)$$

where ρ_{2i+1} only contains $\rightarrow$-transitions (for $i \in \{0, \dots, n\}$), and ρ_{2i} only $\leftarrow$-transitions (for $i \in \{1, \dots, n\}$). Since we assume polynomial ambiguity of $\mathcal{A}$, we must have $n \leq |Q|$. Otherwise, there exists a position which is visited twice in the same state, thus allowing infinitely many runs over the input word by pumping this looping fragment of the run. We then immediately obtain that, for every word w, the multiset $\{\!|\mathcal{A}_{p,q}|\!\}(w)$ can be decomposed as

$$\sum_{\substack{n \leq |Q| \\ (p,\rhd,k_0,\rightarrow,p_0),\dots, \\ (p_{2n+1},\lhd,k_{2n+2},\leftarrow,q) \in \mathsf{Tr}}} \{\!\{k_0\}\!\}\{\!|\overrightarrow{\mathcal{A}}_{p_0,p_1}|\!\}(w)\{\!\{k_2\}\!\}\{\!|\overleftarrow{\mathcal{A}}_{p_2,p_3}|\!\}(w)\cdots\{\!|\overrightarrow{\mathcal{A}}_{p_{2n},p_{2n+1}}|\!\}(w)\{\!\{k_{2n+2}\}\!\}$$

It remains to show that the above decomposition can be translated into an equivalent WFO sentence. Since trimming preserves aperiodicity, using Lemma 1, we know that for every $p, q \in Q$, both $\overrightarrow{\mathcal{A}}_{p,q}$ and $\overleftarrow{\mathcal{A}}_{p,q}$ are aperiodic. By Lemma 2, we can thus construct equivalent WFO sentences $\overleftarrow{\Phi}_{p,q}$ and $\overrightarrow{\Phi}_{p,q}$, respectively. We now define,

$$\Phi_{p,q} = \sum_{\substack{n \leq |Q| \\ (p,\rhd,k_0,\rightarrow,p_0),\dots, \\ (p_{2n+1},\lhd,k_{2n+2},\leftarrow,q) \in \mathsf{Tr}}} k_0 \cdot \overrightarrow{\Phi}_{p_0,p_1} \cdot k_2 \cdot \overleftarrow{\Phi}_{p_2,p_3} \cdot \dots \cdot \overrightarrow{\Phi}_{p_{2n},p_{2n+1}} \cdot k_{2n+2}$$

It can be proved that $\{\!|\mathcal{A}_{p,q}|\!\} = \{\!|\Phi_{p,q}|\!\}$. Finally, we set $\Phi = \sum_{p \in I, q \in F} \Phi_{p,q}$ and we can check that this formula is equivalent to $\mathcal{A}$. □

6 From Nested Two-Way Weighted Automata to Nested Sweeping Weighted Automata

In this section we finally provide a sketch of the proof of $2 \Rightarrow 5$ in Theorem 2. This is the most novel and challenging part of the proof. In particular, notice that such an implication requires the use of nesting: the following example of 0-NWA does not have an equivalent 0-swNWA, even under the restriction of polynomial ambiguity and aperiodicity.

Example 4. Consider the 0-$\mathsf{NWA}(\mathsf{K}, A)$ $\mathcal{A}_{ex}$ depicted in Figure 2, over the alphabet $A = \{a, b\}$ and with weights $\mathsf{K} = \{f, g\}$. Its semantics maps every word of A^* of the form $u = a^{m_1} b \cdots a^{m_n} b$ to the multiset $\{\!\{f^{m_1} g^{m_1} \cdots f^{m_n} g^{m_n}\}\!\}$, and every word of the form $u = a^{m_1} b \cdots a^{m_n}$ to the multiset $\{\!\{f^{m_1} g^{m_1} \cdots f^{m_n}\}\!\}$. The automaton $\mathcal{A}_{ex}$ is unambiguous (even deterministic). By a computation of its transition monoid, it can also be shown to be aperiodic. We can prove (see the long version [8]) that it has no equivalent 0-swNWA, since it is crucial that the automaton switches direction several times in the middle of the word. □

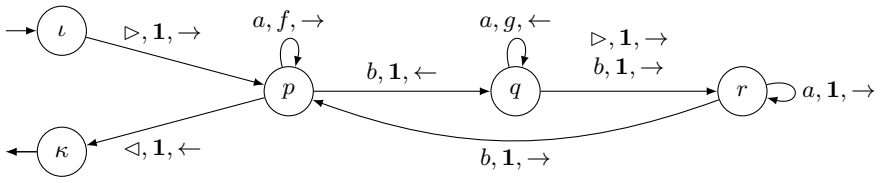

Fig. 2. A 0-NWA $\mathcal{A}_{ex}$.

Consider now a 0-NWA $\mathcal{A}$ (we will explain at the very end how to do this for an r-NWA). We build an swNWA $\overline{\mathcal{A}}$ equivalent to it, that will moreover be aperiodic and polynomially ambiguous if $\mathcal{A}$ is.

To understand our construction of $\overline{\mathcal{A}}$, consider an accepting simple run of $\mathcal{A}$ over a word u. In order to get closer to a sweeping automaton, we first split the run into subruns that go from the beginning of the run of the left marker to the right marker or the end of the run (possibly hitting in the mean time the left marker), and then to the left marker again (possibly hitting in the mean time the right marker), and so on. We get at most $|Q|$ subruns by doing so, since the run is simple (and thus cannot visit more than $|Q|$ times each marker).

For each subrun, we further decompose them as follows. We only present here the decomposition for the left-to-right case, the other one being symmetrical.

For a left-to-right run over the word $w = w_1 \cdots w_n$ (with w_1 possibly being equal to $\triangleright$, but $w_n \neq \triangleleft$), we decompose it into the interleaving of subruns with only $\rightarrow$-transitions, ending in an increasing sequence of positions $(i_j)_{1 \leq j \leq m}$, and some right-to-right subruns on the prefix words $w_1 \cdots w_{i_j}$. Formally, every left-to-right run can be written as $\rho_1 \lambda_1 \rho_2 \lambda_2 \cdots \lambda_{m-1} \rho_m$ where we have a sequence of positions $0 = i_0 < i_1 < \cdots < i_{m-1} < i_m = n + 1$ such that

- for all $j \in \{1, \ldots, m\}$, ρ_j is a run over $w_{i_{j-1}+1} \cdots w_{i_j - 1}$ with only $\rightarrow$-transitions: notice that this run can be empty if $i_j = i_{j-1} + 1$;
- for all $j \in \{1, \ldots, m-1\}$, λ_j is a right-to-right run over $w_1 \cdots w_{i_j}$ that starts with a $\leftarrow$-transition.

We exemplify the decomposition on the left of Figure 3. We thus build an NWA $\overline{\mathcal{A}}$ whose root automaton is a sweeping automaton that emulates the black ρ-parts, interleaved with some new $\rightarrow$-transitions from state p (in a position i_j) to state q (in the corresponding position $i_j + 1$): the weight of this transition is a (non-sweeping) NWA$(\mathsf{K}, A \times \{0,1\})$ $\mathcal{A}_{(p,q)}$ that is in charge of emulating the blue subrun λ_j from state p to state q of $\mathcal{A}$, keeping marked the position i_j in the second component of the alphabet $A \times \{0,1\}$.

We treat the various new automata $\mathcal{A}_{(p,q)}$ recursively to transform them to sweeping automata too. We thus similarly decompose the λ-subruns as before, by adapting the previous decomposition working only for left-to-right runs. In the decomposition of a right-to-right run over the word $w = w_1 \cdots w_n$ (with w_1 possibly being equal to $\triangleright$, but $w_n \neq \triangleleft$), the ρ-parts will be right-to-left, and we will add a special left-to-right additional run τ at the end to come back

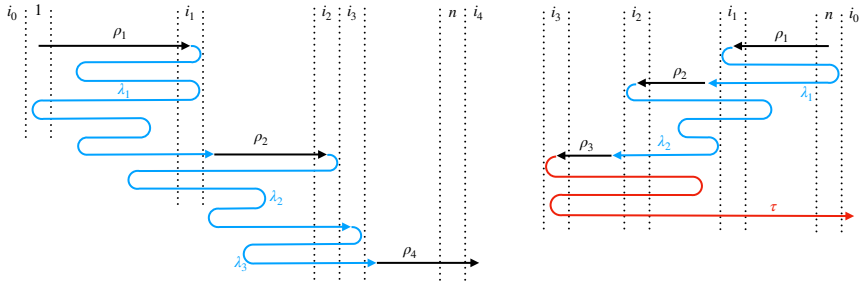

Fig. 3. On the left, the decomposition of a left-to-right run as a sequence $\rho_1\lambda_1\rho_2\lambda_2\rho_3\lambda_3\rho_4$ with ρ_3 being empty. On the right, the decomposition of a right-to-right run as a sequence $\rho_1\lambda_1\rho_2\lambda_2\rho_3\tau$.

to the right of the word. Formally, every right-to-right run can be written as $\rho_1\lambda_1\rho_2\lambda_2\cdots\lambda_{m-1}\rho_m\tau$ where we have a sequence of positions $1 \leq i_m < \cdots < i_1 < i_0 = n+1$ such that

- for all $j \in \{1,\dots,m\}$, ρ_j is a left-to-right run over $w_{i_j+1}\cdots w_{i_{j-1}-1}$ with only $\to$-transitions: notice that this run can be empty if $i_{j-1} = i_j + 1$;
- for all $j \in \{1,\dots,m-1\}$, λ_j is a left-to-left run over $w_{i_j}\cdots w_n$ that starts with a $\to$-transition;
- τ is a left-to-right run over $w_{i_m}\cdots w_n$.

We exemplify the decomposition on the right of Figure 3. Once again, the automaton $\mathcal{A}_{(p,q)}$ is transformed into a NWA where the root automaton is a sweeping automaton that emulates the black ρ-parts, interleaved with some new $\leftarrow$-transitions with a weight being a NWA that computes the λ-subruns as well as the τ one. We once again treat these NWA recursively similarly as before (new cases occur in terms of directions).

This recursive decomposition of the runs, and thus the associated construction of sweeping automata, can be terminated after a bounded number of iterations. Indeed, in all simple runs of $\mathcal{A}$, no more than $|Q|$ configurations are visited for a particular position of the word. Since each recursive step in the decomposition consumes each position in the black runs, this implies that after $|Q|$ steps, there are no remaining blue subruns to consider. At level $|Q|$ of nesting, we thus do not allow anymore the addition of new transitions that would simulate further blue λ-subruns. The previous argument is the core of the correctness proof showing that the sweeping automaton produced is equivalent to $\mathcal{A}$.

In case $\mathcal{A}$ is an r-NWA, we use the black ρ-subruns to compute the children automata of $\mathcal{A}$ with nested calls. In contrast the added transitions that are supposed to launch the emulation of the blue λ-subruns call another sweeping automaton below.

Example 5. We apply the construction on the 0-NWA of Example 4. This will produce the 2-swNWA $\overline{\mathcal{A}}$ in Figure 4. We also depict the actual decomposition

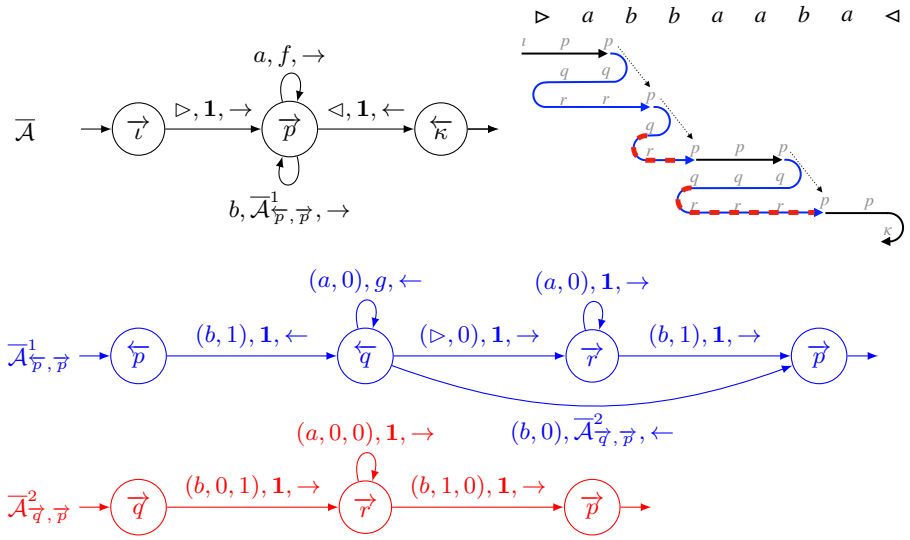

Fig. 4. 2-swNWA $\overline{\mathcal{A}}$ obtained by our construction, starting from the automaton of Example 4, and the decomposition of a run of this automaton showing which sweeping automaton computes each subrun.

of a run over the word $\triangleright abbaaba \triangleleft$. The black subruns are the ρ-parts that are computed by the sweeping root automaton (it is sweeping, and not one way, just because of the final transition). The automaton $\overline{\mathcal{A}}^1_{\overleftarrow{p},\overrightarrow{p}}$ is in charge of computing the blue λ-subruns. Notice that the subscript tells the automaton that it should start (at the marked position, which is checked by the first transition of the automaton) in state p going left, and should stop (once again at the marked position) in state p going right. There are two cases. For the leftmost λ-subrun, the sweeping automaton can entirely compute it. For the other ones, it cannot since there is a change of direction in the middle of the word. The red dotted part is thus the τ-final piece of the decomposition in the second step of the recursion, that is taken care of by automaton $\overline{\mathcal{A}}^2_{\overrightarrow{q},\overrightarrow{p}}$.

The above construction preserves the ambiguity of the automata. However, we are not able to directly show that it preserves aperiodicity. We must encode more information in the state space of the various sweeping automata in order to allow for the proof of aperiodicity. In particular, we encode some pieces of information on the behaviours allowed in the current position, allowing us to better understand the structure of the transition monoid of the built automaton. The full construction and proof is given in the long version [8], which concludes the proof of the last implication of Theorem 2.

7 Conclusion

We have extended the results of Droste and Gastin [10] relating restricted weighted first-order logic and aperiodic weighted automata with some restrictions about ambiguity. We thus have closed open questions raised by them, introducing an abstract semantics for a full fragment of weighted first-order logic, and showing the equivalence between this logic and aperiodic nested weighted automaton with linear or polynomial ambiguity.

We have only studied linear and polynomial ambiguity, contrary to Droste and Gastin that have also characterised finitely-ambiguous and unambiguous aperiodic weighted automata with fragments of the logic. We leave as future work similar study for nested weighted automata, but we do hope that similar restrictions may apply also in our more general case.

However, dropping the condition on polynomial ambiguity would certainly lead to a logical fragment beyond weighted first-order logic. In particular, the main difficulty is that the logic is not easily able to check the simplicity condition of the accepting runs in this case.

Having introduced two-way navigations (and also nesting) makes possible to ask similar questions to other input structures than words, like finite ranked or unranked trees [16], nested words [14,3], or even graphs [4]. Nested weighted automata and weighted logics have already been studied in this setting, without any characterisation of the power of first-order fragments.

Last but not least, contrary to the unweighted setting, our characterization (as well as the one by Droste and Gastin) does not yet lead to a procedure deciding if the series recognised by a given (nested two-way) weighted automaton is indeed recognisable by an aperiodic one, i.e. in the convenient first-order fragment of the logic. We still lack the algebraic tools allowing for such decision procedures.

References

1. Birget, J.C.: Concatenation of inputs in a two-way automaton. Theoretical Computer Science **63**(2), 141–156 (1989)
2. Birget, J.C.: Two-way automaton computations. RAIRO-Theoretical Informatics and Applications **24**(1), 47–66 (1990)
3. Bollig, B., Gastin, P., Monmege, B.: Weighted specifications over nested words. In: FoSSaCS'13. LNCS, vol. 7794, pp. 385–400. Springer (2013). `https://doi.org/10.1007/978-3-642-37075-5_25`
4. Bollig, B., Gastin, P., Monmege, B., Zeitoun, M.: Logical characterization of weighted pebble walking automata. In: CSL-LICS'14. ACM (2014). `https://doi.org/10.1145/2603088.2603118`
5. Bollig, B., Gastin, P., Monmege, B., Zeitoun, M.: Pebble weighted automata and weighted logics. ACM Transactions on Computational Logic **15**(2:15) (Apr 2014). `https://doi.org/10.1145/2579819`
6. Büchi, J.R.: Weak second-order arithmetic and finite automata. Zeitschrift für Mathematische Logik und Grundlagen der Mathematik **6**, 66–92 (1960)

7. Carton, O., Dartois, L.: Aperiodic two-way transducers and FO-transductions. Research Report 2103.15651, arXiv (2021)
8. Dhruv, N., Monmege, B.: An automata theoretic characterization of weighted first-order logic. Research Report 2307.14707, arXiv (2023), http://arxiv.org/abs/2307.14707
9. Droste, M., Gastin, P.: Weighted automata and weighted logics. Theoretical Computer Science **380**(1-2), 69–86 (2007)
10. Droste, M., Gastin, P.: Aperiodic weighted automata and weighted first-order logic. In: MFCS'19. No. 76 in LIPIcs, Schloss Dagstuhl–Leibniz-Zentrum für Informatik (2019). https://doi.org/10.4230/LIPIcs.MFCS.2019.76
11. Droste, M., Götze, D., Märcker, S., Meinecke, I.: Weighted tree automata over valuation monoids and their characterization by weighted logics. In: Algebraic Foundations in Computer Science. LNCS, vol. 7020. Springer (2011)
12. Droste, M., Kuich, W., Vogler, H.: Handbook of Weighted Automata. EATCS Monographs in Theoretical Computer Science, Springer (2009)
13. Droste, M., Meinecke, I.: Weighted automata and weighted MSO logics for average and long-time behaviors. Information and Computation **220-221**, 44–59 (2012)
14. Droste, M., Pibaljommee, B.: Weighted nested word automata and logics over strong bimonoids. In: CIAA'12. LNCS, vol. 7381. Springer (2012)
15. Droste, M., Rahonis, G.: Weighted automata and weighted logics on infinite words. In: DLT'06. LNCS, vol. 4036. Springer (2006)
16. Droste, M., Vogler, H.: Weighted logics for unranked tree automata. Theory of Computing Systems **48**, 23–47 (2011)
17. Elgot, C.C.: Decision problems of finite automata design and related arithmetics. Transactions of the American Mathematical Society **98**, 21–52 (1961)
18. Fülöp, Z., Stüber, T., Vogler, H.: A Büchi-like theorem for weighted tree automata over multioperator monoids. Theory of Computing Systems **50**, 241–278 (2012)
19. Gastin, P., Monmege, B.: Adding pebbles to weighted automata: Easy specification and efficient evaluation. Theoretical Computer Science **534**, 24–44 (May 2014). https://doi.org/10.1016/j.tcs.2014.02.034
20. Gastin, P., Monmege, B.: A unifying survey on weighted logics and weighted automata. Soft Computing **22**(4), 1047–1065 (Feb 2018). https://doi.org/10.1007/s00500-015-1952-6
21. McNaughton, R.F., Papert, S.A.: Counter-Free Automata. No. 65, MIT Press (1971)
22. Schützenberger, M.P.: On the definition of a family of automata. Information and Control **4**, 245–270 (1961)
23. Schützenberger, M.P.: On finite monoids having only trivial subgroups. Information and Control **8**, 190–194 (1965)
24. Trakhtenbrot, B.A.: Finite automata and logic of monadic predicates. Doklady Akademii Nauk SSSR **149**, 326–329 (1961)