

Weighted Automata and Logics for Infinite Nested Words

Manfred Droste and Stefan Dück*

Institut für Informatik, University Leipzig, D-04109 Leipzig, Germany
{droste, dueck}@informatik.uni-leipzig.de

Abstract. Nested words introduced by Alur and Madhusudan are used to capture structures with both linear and hierarchical order, e.g. XML documents, without losing valuable closure properties. Furthermore, Alur and Madhusudan introduced automata and equivalent logics for both finite and infinite nested words, thus extending Büchi's theorem to nested words. Recently, average and discounted computations of weights in quantitative systems found much interest. Here, we will introduce and investigate weighted automata models and weighted MSO logics for infinite nested words. As weight structures we consider valuation monoids which incorporate average and discounted computations of weights as well as the classical semirings. We show that under suitable assumptions, two resp. three fragments of our weighted logics can be transformed into each other. Moreover, we show that the logic fragments have the same expressive power as weighted nested word automata.

Keywords: nested words, weighted automata, weighted logics, quantitative automata, valuation monoids

1 Introduction

Nested words, introduced by Alur and Madhusudan [2], capture models with both a natural sequence of positions and an hierarchical nesting of these positions. Prominent examples include XML documents and executions of recursively structured programs. Automata on nested words, logical specifications, and corresponding languages of nested words have been intensively studied, see [1], [2], [17]. Recently, there has been much interest in quantitative features for the specification and analysis of systems. Quantitative automata modeling the long-time average or discounted behavior of systems were investigated by Chatterjee, Doyen, and Henzinger [6], [7]. It is the goal of this paper to present quantitative logics for such quantitative automata on nested words.

The connection between MSO logic and automata due to Büchi, Elgot, and Trakhenbrot [5], [15], [21] has proven most fruitful. Weighted automata over semirings (like $(\mathbb{N}, +, \cdot, 0, 1)$) were already investigated by Schützenberger [20]

* supported by Deutsche Forschungsgemeinschaft (DFG), project DR 202/11-1 and Graduiertenkolleg 1763 (QuantLA)

and soon developed a flourishing theory, cf. the books [3], [14], [16], [19] and the recent handbook [8]. However, an expressively equivalent weighted MSO logic was developed only recently [9]. This was extended to semiring-weighted automata and logics over finite nested words in [18], and further to strong bimonoids as weight structures in [12]. For quantitative automata and logics, incorporating average and discounting computations of weights over words, such an equivalence was given in [11].

In this paper, we will investigate quantitative nested word automata and suitable quantitative MSO logics. We will concentrate on infinite nested words, although our results also hold for finite nested words. We employ the stair Muller nested word automata of [2], [17], since these can be determinized without losing expressive power. As weight structures we take the valuation monoids of [11]. These include infinite products as in totally complete semirings [13], but also computations of long-time averages or discountings of weights. As example for such a setting we give the calculation of the long-time ratio of bracket-free positions in prefixes of an infinite nested word. As our first main result, we show that under suitable assumptions on the valuation monoid D , two resp. three versions of our weighted MSO logic have the same expressive power. In particular, if D is commutative, then any weighted MSO-formula is equivalent to one in which conjunctions occur only between 'classical' boolean formulas and constants. In contrast to [11], our proof uses direct conversions of the formulas and thus has much better complexity than using the automata-theoretic constructions of [11]. These conversions are new even for the case of weighted logics on words.

In our second main result, we show under suitable assumptions on the valuation monoid that our weighted MSO logics have the same expressive power as weighted nested automata. These assumptions on the valuation monoid are satisfied by long-time average resp. discounted computations of weights; therefore our results apply to these settings. All our constructions of automata from formulas or conversely are effective.

2 Automata and Logics for Nested ω -Words

In this section we describe basic background for classical (unweighted) automata and logics on nested- ω -words. We denote by Σ an alphabet and by Σ^ω the set of all ω -words over Σ . \mathbb{N} is the set of all natural numbers without zero. For a binary relation R , we denote with $R(x, y)$ that $(x, y) \in R$.

Definition 1. *A matching relation ν over \mathbb{N} is a subset of $(\{-\infty\} \cup \mathbb{N}) \times (\mathbb{N} \cup \{\infty\})$ such that:*

- (i) $\nu(i, j) \Rightarrow i < j$,
- (ii) $\forall i \in \mathbb{N} : |\{j : \nu(i, j)\}| \leq 1 \wedge |\{j : \nu(j, i)\}| \leq 1$,
- (iii) $\nu(i, j) \wedge \nu(i', j') \wedge i < i' \Rightarrow j < i' \vee j > j'$,
- (iv) $(-\infty, \infty) \notin \nu$.

A nested ω -word nw over Σ is a pair $(w, \nu) = (a_1 a_2 \dots, \nu)$ where $w = a_1 a_2 \dots$ is an ω -word over Σ and ν is a matching relation over \mathbb{N} . We denote by $NW^\omega(\Sigma)$ the

set of all nested ω -words over Σ and we call every subset of $NW^\omega(\Sigma)$ a language of nested ω -words.

If $\nu(i, j)$ holds, we call i a *call position* and j a *return position*. In case of $j = \infty$, i is a *pending call* otherwise a *matched call*. In case of $i = -\infty$, j is a *pending return* otherwise a *matched return*. If i is neither call nor return, then we say i is an *internal*.

Definition 2. A deterministic stair Muller nested word automaton (sMNWA) over Σ is a quadruple $\mathcal{A} = (Q, q_0, \delta, \mathfrak{F})$, where $\delta = (\delta_{\text{call}}, \delta_{\text{int}}, \delta_{\text{ret}})$, consisting of:

- a finite set of states Q ,
- an initial state $q_0 \in Q$,
- a set $\mathfrak{F} \subseteq 2^Q$ of accepting sets of states,
- the transition functions $\delta_{\text{call}}, \delta_{\text{int}} : Q \times \Sigma \rightarrow Q$,
- the transition function $\delta_{\text{ret}} : Q \times Q \times \Sigma \rightarrow Q$.

A run r of the sMNWA \mathcal{A} on the nested ω -word $nw = (a_1 a_2 \dots, \nu)$ is an infinite sequence of states $r = (q_0, q_1, \dots)$ where $q_i \in Q$ for each $i \in \mathbb{N}$ and q_0 is the initial state of \mathcal{A} such that for each $i \in \mathbb{N}$ the following holds:

$$\begin{cases} \delta_{\text{call}}(q_{i-1}, a_i) = q_i & , \text{ if } \nu(i, j) \text{ for some } j > i \text{ (or } j = \infty) \\ \delta_{\text{int}}(q_{i-1}, a_i) = q_i & , \text{ if } i \text{ is an internal} \\ \delta_{\text{ret}}(q_{i-1}, q_{j-1}, a_i) = q_i & , \text{ if } \nu(j, i) \text{ for some } 1 \leq j < i \\ \delta_{\text{ret}}(q_{i-1}, q_0, a_i) = q_i & , \text{ if } \nu(-\infty, i) . \end{cases}$$

We call $i \in \mathbb{N}$ a *top-level position* if there exist no positions $j, k \in \mathbb{N}$ with $j < i < k$ and $\nu(j, k)$. We define

$$Q_\infty^t(r) = \{q \in Q \mid q = q_i \text{ for infinitely many top-level positions } i\} .$$

A run r of an sMNWA is *accepted* if $Q_\infty^t(r) \in \mathfrak{F}$. An sMNWA \mathcal{A} *accepts* the nested ω -word nw if there is an accepted run of \mathcal{A} on nw . We denote with $L(\mathcal{A})$ the set of all accepted nested ω -words of \mathcal{A} . We call a language L of nested ω -words *regular* if there is an sMNWA \mathcal{A} with $L(\mathcal{A}) = L$.

Alur and Madhusudan [2] considered nondeterministic Büchi NWA and nondeterministic Muller NWA. They showed that the deterministic versions of these automata have strictly less expressive power than the nondeterministic automata. However, referring to Löding, Madhusudan and Serre [17], Alur and Madhusudan stated that deterministic stair Muller NWA have the same expressive power as their nondeterministic versions as well as nondeterministic Büchi NWA. Moreover, the class of regular languages of nested- ω -words is closed under union, intersection and complement ([2]).

Definition 3. The monadic second order logic for nested words $MSO(NW(\Sigma))$ contains exactly all formulas φ which are given by the following syntax:

$$\varphi ::= \text{Lab}_a(x) \mid \text{call}(x) \mid \text{ret}(x) \mid x \leq y \mid \nu(x, y) \mid x \in X \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists x.\varphi \mid \exists X.\varphi$$

where $a \in \Sigma$ and x, y are first order variables and X is a second order variable.

The semantics of these formulas is given in a natural way, cf. [2]. Later we give a full definition of the semantics of *weighted* MSO-formulas. We call φ a *sentence* if φ contains no free variables. If φ is a sentence, then $L(\varphi) = \{nw \in NW^\omega(\Sigma) \mid nw \models \varphi\}$ is *the language defined by φ* .

Theorem 4 (Alur, Madhusudan [2]). *Let L be a language of nested ω -words over Σ . Then L is regular if and only if L is definable by some MSO($NW(\Sigma)$)-sentence φ .*

3 Weighted Stair Muller Nested Word Automata

In this section, we introduce weighted versions of stair Muller nested word automata. As weight structures, we will employ ω -valuation monoids introduced in [11]. We recall the definitions.

A monoid $(D, +, 0)$ is *complete* if it has infinitary sum operations $\sum_I : D^I \rightarrow D$ for any index set I such that

- $\sum_{i \in \emptyset} d_i = 0$, $\sum_{i \in \{k\}} d_i = d_k$, $\sum_{i \in \{j, k\}} d_i = d_j + d_k$ for $j \neq k$,
- $\sum_{j \in J} (\sum_{i \in I_j} d_i) = \sum_{i \in I} d_i$ if $\bigcup_{j \in J} I_j = I$ and $I_j \cap I_k = \emptyset$ for $j \neq k$.

Note that in every complete monoid the operation $+$ is commutative. We let D^ω comprise all infinite sequences of elements of D .

Definition 5 (Droste, Meinecke [11]). *An ω -valuation monoid $(D, +, \text{Val}^\omega, 0)$ is a complete monoid $(D, +, 0)$ equipped with an ω -valuation function $\text{Val}^\omega : D^\omega \rightarrow D$ with $\text{Val}^\omega((d_i)_{i \in \mathbb{N}}) = 0$ if $d_i = 0$ for some $i \in \mathbb{N}$.*

A product ω -valuation monoid $(D, +, \text{Val}^\omega, \diamond, 0, 1)$ (*short ω -pv-monoid*) is an ω -valuation monoid $(D, +, \text{Val}^\omega, 0)$ with a constant $1 \in D$ and an operation $\diamond : D^2 \rightarrow D$ satisfying $\text{Val}^\omega(1^\omega) = 1$, $0 \diamond d = d \diamond 0 = 0$ and $1 \diamond d = d \diamond 1 = d$ for all $d \in D$.

Let $(D, +, \text{Val}^\omega, \diamond, 0, 1)$ be an ω -pv-monoid. D is called *associative* resp. *commutative* if \diamond is associative resp. commutative. D is *left-+-distributive* if for all $d \in D$, for any index set I and $(d_i)_{i \in I} \in D^I$:

$$d \diamond \sum_{i \in I} d_i = \sum_{i \in I} (d \diamond d_i) .$$

Right-+-distributivity is defined analogously. We call D *+-distributive* if D is left- and right-+-distributive. D is *left- Val^ω -distributive* if for all $d \in D$ and $(d_i)_{i \in \mathbb{N}} \in D^\omega$:

$$d \diamond \text{Val}^\omega((d_i)_{i \in \mathbb{N}}) = \text{Val}^\omega((d \diamond d_i)_{i \in \mathbb{N}}) .$$

D is *left-multiplicative* if for all $d \in D$ and $(d_i)_{i \in \mathbb{N}} \in D^\omega$:

$$d \diamond \text{Val}^\omega((d_i)_{i \in \mathbb{N}}) = \text{Val}^\omega(d \diamond d_1, (d_i)_{i \geq 2}) .$$

D is called *conditionally commutative*, if for all $(d_i)_{i \in \mathbb{N}}, (d'_i)_{i \in \mathbb{N}} \in D^\omega$ with $d_i \diamond d'_j = d'_j \diamond d_i$ for all $j < i$, the following holds:

$$\text{Val}^\omega((d_i)_{i \in \mathbb{N}}) \diamond \text{Val}^\omega((d'_i)_{i \in \mathbb{N}}) = \text{Val}^\omega((d_i \diamond d'_i)_{i \in \mathbb{N}}) .$$

We call D *left-distributive* if D is left-+-distributive and, additionally, left- Val^ω -distributive or left-multiplicative. If D is +-distributive and associative, then $(D, +, \diamond, 0, 1)$ is a complete semiring and we call $(D, +, \text{Val}^\omega, \diamond, 0, 1)$ an ω -valuation semiring. A *cc- ω -valuation semiring* is an ω -valuation semiring D which is conditionally commutative and left-distributive.

Example 1 ([11]). We set $\bar{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$ and $-\infty + \infty = -\infty$. We let

$$(D_1, +, \text{Val}^\omega, \diamond, 0, 1) = (\bar{\mathbb{R}}, \text{sup}, \lim \text{avg}, +, -\infty, 0),$$

$$\text{where} \quad \lim \text{avg}((d_i)_{i \in \mathbb{N}}) = \liminf_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n d_i .$$

Let $0 < \lambda < 1$ and $\bar{\mathbb{R}}_+ = \{x \in \bar{\mathbb{R}} \mid x \geq 0\} \cup \{-\infty\}$. We put

$$(D_2, +, \text{Val}^\omega, \diamond, 0, 1) = (\bar{\mathbb{R}}_+, \text{sup}, \text{disc}_\lambda, +, -\infty, 0),$$

$$\text{where} \quad \text{disc}_\lambda((d_i)_{i \in \mathbb{N}}) = \lim_{n \rightarrow \infty} \sum_{i=1}^n \lambda^{i-1} d_i .$$

Then D_1 is a left-+-distributive and left- Val^ω -distributive ω -valuation monoid but not conditionally commutative. Furthermore, D_2 is a left-multiplicative cc- ω -valuation semiring.

Definition 6. A weighted stair Muller nested word automaton (wsMNWA) $\mathcal{A} = (Q, I, \delta, \mathfrak{F})$, where $\delta = (\delta_{\text{call}}, \delta_{\text{int}}, \delta_{\text{ret}})$, over the alphabet Σ and the ω -valuation monoid $(D, +, \text{Val}^\omega, 0)$ consists of:

- a finite set of states Q ,
- a set $I \subseteq Q$ of initial states,
- a set $\mathfrak{F} \subseteq 2^Q$ of accepting sets of states,
- the weight functions $\delta_{\text{call}}, \delta_{\text{int}} : Q \times \Sigma \times Q \rightarrow D$,
- the weight function $\delta_{\text{ret}} : Q \times Q \times \Sigma \times Q \rightarrow D$.

A run r of the wsMNWA \mathcal{A} on the nested ω -word $nw = (a_1 a_2 \dots, \nu)$ is an infinite sequence of states $r = (q_0, q_1, \dots)$. We denote with $wt_{\mathcal{A}}(r, nw, i)$ the weight of the transition of r used at position $i \in \mathbb{N}$, defined as follows

$$wt_{\mathcal{A}}(r, nw, i) = \begin{cases} \delta_{\text{call}}(q_{i-1}, a_i, q_i) & , \text{ if } \nu(i, j) \text{ for some } j > i \\ \delta_{\text{int}}(q_{i-1}, a_i, q_i) & , \text{ if } i \text{ is an internal} \\ \delta_{\text{ret}}(q_{i-1}, q_{j-1}, a_i, q_i) & , \text{ if } \nu(j, i) \text{ for some } 1 \leq j < i \\ \delta_{\text{ret}}(q_{i-1}, q_I, a_i, q_i) & , \text{ if } \nu(-\infty, i) \text{ for some } q_I \in I . \end{cases} \quad (1)$$

Then we define the *weight* $wt_{\mathcal{A}}(r, nw)$ of r on nw by letting

$$wt_{\mathcal{A}}(r, nw) = \text{Val}^\omega((wt_{\mathcal{A}}(r, nw, i))_{i \in \mathbb{N}}) .$$

We define top-level positions and the set $Q_\infty^t(r)$ as before. A run r is *accepted* if $q_0 \in I$ and $Q_\infty^t(r) \in \mathfrak{F}$. We denote with $\text{acc}(\mathcal{A})$ the set of all accepted runs in \mathcal{A} . We define the *behavior of the automaton* \mathcal{A} as the function $\|\mathcal{A}\| : NW^\omega(\Sigma) \rightarrow D$ given by (where as usual, empty sums are defined to be 0)

$$\begin{aligned} \|\mathcal{A}\|(nw) &= \sum_{r \in \text{acc}(\mathcal{A})} \text{wt}_{\mathcal{A}}(r, nw) \\ &= \sum_{r \in \text{acc}(\mathcal{A})} \text{Val}^\omega((\text{wt}_{\mathcal{A}}(r, nw, i))_{i \in \mathbb{N}}) . \end{aligned}$$

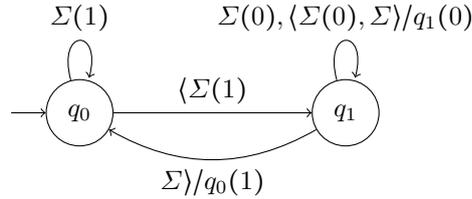
We call every function $S : NW^\omega(\Sigma) \rightarrow D$ a *nested ω -word series* (short: *series*). We call a series S *regular* if there exists an automaton \mathcal{A} with $\|\mathcal{A}\| = S$.

Example 2. Within the following example we call a position i of a nested ω -word $nw = (w, \nu)$ *bracketfree* if there are no positions $j, k \in (\mathbb{N} \cup \{-\infty, \infty\})$ with $j < i < k$ and $\nu(j, k)$. This requirement is stronger than i being a top-level position because it contains $-\infty$ and ∞ thus also banning i being in the scope of pending calls and pending returns. Only for well-matched nested ω -words, i.e. nested ω -words without pending edges, the two properties coincide.

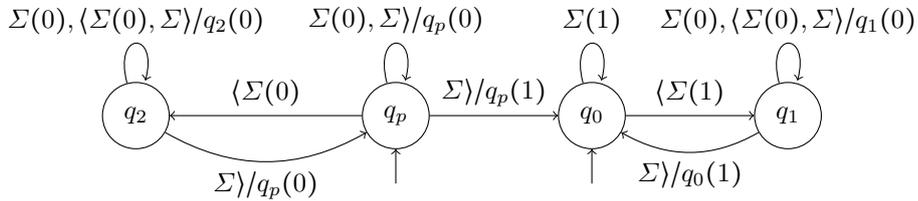
We consider the series S assigning to every nested ω -word nw the greatest accumulation point of the ratio of bracketfree positions in finite prefixes of nw .

To model S we use the ω -valuation monoid $D = (\mathbb{R}, \text{sup}, \text{lim avg}, -\infty)$. If we want to analyze this property for well-matched nested ω -words only, then automaton \mathcal{A}_1 given below recognizes S . In the general case including pending edges, automaton \mathcal{A}_2 recognizes S . Note that we denote the call transitions with $\langle \Sigma$ and the return transitions with $\Sigma \rangle / q$ where q has to be the state where the last open call was encountered. The weights 1 resp. 0 are given in brackets.

Automaton 1: wsMNWA \mathcal{A}_1 with $\mathfrak{F}_1 = \{\{q_0\}\}$



Automaton 2: wsMNWA \mathcal{A}_2 with $\mathfrak{F}_1 = \{\{q_2\}, \{q_p\}, \{q_2, q_p\}, \{q_0, q_1\}, \{q_0\}, \{q_1\}\}$



As usual, we extend the operation $+$ and \diamond to series $S, T : NW^\omega(\Sigma) \rightarrow D$ by means of pointwise definitions as follows:

$$(S \star T)(nw) = S(nw) \star T(nw) \text{ for each } nw \in NW^\omega(\Sigma), \star \in \{+, \diamond\} .$$

We let $d \in D$ also denote the constant series with value d , i.e. $\|d\|(nw) = d$ for each $nw \in NW^\omega(\Sigma)$. For $L \subseteq NW^\omega(\Sigma)$, we define the *characteristic series* $\mathbb{1}_L : NW^\omega(\Sigma) \rightarrow D$ by letting $\mathbb{1}_L(nw) = 1$ if $nw \in L$, and $\mathbb{1}_L(nw) = 0$ otherwise. We call a series S a *regular step function* if

$$S = \sum_{i=1}^k d_i \diamond \mathbb{1}_{L_i} , \quad (2)$$

where L_i are regular languages of nested- ω -words forming a partition of $NW^\omega(\Sigma)$ and $d_i \in D$ for each $i \in \{1, \dots, k\}$; so $S(nw) = d_i$ iff $nw \in L_i$ for each $i \in \{1, \dots, k\}$.

An ω -pv-monoid D is *regular* if for any alphabet Σ we have: For each $d \in D$ there exists a wsMNAWA \mathcal{A}_d with $\|\mathcal{A}_d\| = d$. Analogously to Droste and Meinecke [11] we can show that every left-distributive ω -pv-monoid is regular.

Proposition 7. *Let D be a regular ω -pv-monoid. Then each regular step function $S : NW^\omega(\Sigma) \rightarrow D$ is regular. Furthermore, the set of all regular step functions is closed under $+$ and \diamond .*

Next we show that regular series are closed under projections. Consider a mapping $h : \Sigma \rightarrow \Gamma$ between two alphabets. Then h extends uniquely to an homomorphism between Σ^ω and Γ^ω , also denoted by h . Hence h is length-preserving and we can extend h to a function $h : NW^\omega(\Sigma) \rightarrow NW^\omega(\Gamma)$ by defining $h(nw) = h(w, \nu) = (h(w), \nu)$ for each $nw \in NW^\omega(\Sigma)$. Let $S : NW^\omega(\Sigma) \rightarrow D$ be a series. Then we define $h(S) : NW^\omega(\Gamma) \rightarrow D$ for each $nv \in NW^\omega(\Gamma)$ by

$$h(S)(nv) = \sum (S(nw) \mid nw \in NW^\omega(\Sigma), h(nw) = nv) .$$

Proposition 8. *Let D be an ω -valuation monoid, $S : NW^\omega(\Sigma) \rightarrow D$ regular and $h : \Sigma \rightarrow \Gamma$. Then $h(S) : NW^\omega(\Gamma) \rightarrow D$ is regular.*

4 Weighted MSO-Logic for Nested ω -Words

In this section, we will present different fragments of our weighted MSO logic, and we give our first main result on the equivalence of these fragments. In the following D is always an ω -pv-monoid. We combine ideas of Alur and Madhusudan [2], Droste and Gastin [9], and Bollig and Gastin [4], and divide the syntax of the weighted logic into a boolean part and a weighted part.

Definition 9 (Syntax). *The weighted monadic second order logic for nested words $MSO(D, NW(\Sigma))$ is given by the following syntax*

$$\begin{aligned} \beta &::= \text{Lab}_a(x) \mid \text{call}(x) \mid \text{ret}(x) \mid x \leq y \mid \nu(x, y) \mid x \in X \mid \neg\beta \mid \beta \wedge \beta \mid \forall x.\beta \mid \forall X.\beta \\ \varphi &::= d \mid \beta \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \forall x.\varphi \mid \exists x.\varphi \mid \exists X.\varphi \end{aligned}$$

where $d \in D$, $a \in \Sigma$ and x, y, X are first resp. second order variables. We call all formulas β boolean formulas.

The set of all positions of $nw \in NW^\omega(\Sigma)$ is \mathbb{N} . Let $\varphi \in MSO(D, NW(\Sigma))$. We denote the set of free variables of φ by $\text{free}(\varphi)$. Let \mathcal{V} be a finite set of variables containing $\text{free}(\varphi)$. As usual, we define a (\mathcal{V}, nw) -assignment γ as function assigning to every first order variable of \mathcal{V} a position of nw and to every second order variable a subset of positions of nw . We let $\gamma[x \rightarrow i]$ (resp. $\gamma[X \rightarrow I]$) be the $(\mathcal{V} \cup \{x\}, nw)$ -assignment (resp. $(\mathcal{V} \cup \{X\}, nw)$ -assignment) mapping x to i (resp. X to I) and equaling γ anywhere else.

We encode a pair (nw, γ) as nested ω -word as usual over the extended alphabet $\Sigma_{\mathcal{V}} = \Sigma \times \{0, 1\}^{\mathcal{V}}$ with the same matching relation ν (cf. [9], [12]). We call $(nw, \sigma) \in NW^\omega(\Sigma_{\mathcal{V}})$ *valid* if σ emerges from a (\mathcal{V}, nw) -assignment. Clearly the language $N_{\mathcal{V}}$ of all valid words is regular.

Definition 10 (Semantics). *The semantics of φ is a series $\llbracket \varphi \rrbracket_{\mathcal{V}} : NW^\omega(\Sigma_{\mathcal{V}}) \rightarrow D$. If (nw, σ) is not valid, we set $\llbracket \varphi \rrbracket_{\mathcal{V}}(nw, \sigma) = 0$. Otherwise we define $\llbracket \varphi \rrbracket_{\mathcal{V}}(nw, \sigma)$ for $(nw, \sigma) = ((a_1 a_2 \dots, \nu), \sigma)$ inductively as follows:*

$$\begin{aligned} \llbracket \text{Lab}_a(x) \rrbracket_{\mathcal{V}}(nw, \sigma) &= \begin{cases} 1, & \text{if } a_{\sigma(x)} = a \\ 0, & \text{otherwise,} \end{cases} & \llbracket \text{call}(x) \rrbracket_{\mathcal{V}}(nw, \sigma) &= \begin{cases} 1, & \text{if } \sigma(x) \text{ is a call} \\ 0, & \text{otherwise,} \end{cases} \\ \llbracket \text{ret}(x) \rrbracket_{\mathcal{V}}(nw, \sigma) &= \begin{cases} 1, & \text{if } \sigma(x) \text{ is a return} \\ 0, & \text{otherwise,} \end{cases} & \llbracket x \leq y \rrbracket_{\mathcal{V}}(nw, \sigma) &= \begin{cases} 1, & \text{if } \sigma(x) \leq \sigma(y) \\ 0, & \text{otherwise,} \end{cases} \\ \llbracket \nu(x, y) \rrbracket_{\mathcal{V}}(nw, \sigma) &= \begin{cases} 1, & \text{if } \nu(\sigma(x), \sigma(y)) \\ 0, & \text{otherwise,} \end{cases} & \llbracket x \in X \rrbracket_{\mathcal{V}}(nw, \sigma) &= \begin{cases} 1, & \text{if } \sigma(x) \in \sigma(X) \\ 0, & \text{otherwise,} \end{cases} \\ \llbracket \neg \beta \rrbracket_{\mathcal{V}}(nw, \sigma) &= \begin{cases} 1, & \text{if } \llbracket \beta \rrbracket_{\mathcal{V}}(nw, \sigma) = 0 \\ 0, & \text{otherwise,} \end{cases} & \llbracket d \rrbracket_{\mathcal{V}}(nw, \sigma) &= d \quad \text{for all } d \in D, \\ \llbracket \varphi \vee \psi \rrbracket_{\mathcal{V}}(nw, \sigma) &= \llbracket \varphi \rrbracket_{\mathcal{V}}(nw, \sigma) + \llbracket \psi \rrbracket_{\mathcal{V}}(nw, \sigma), \\ \llbracket \varphi \wedge \psi \rrbracket_{\mathcal{V}}(nw, \sigma) &= \llbracket \varphi \rrbracket_{\mathcal{V}}(nw, \sigma) \diamond \llbracket \psi \rrbracket_{\mathcal{V}}(nw, \sigma), \\ \llbracket \exists x. \varphi \rrbracket_{\mathcal{V}}(nw, \sigma) &= \sum_{i \in \mathbb{N}} (\llbracket \varphi \rrbracket_{\mathcal{V} \cup \{x\}}(nw, \sigma[x \rightarrow i])), \\ \llbracket \exists X. \varphi \rrbracket_{\mathcal{V}}(nw, \sigma) &= \sum_{I \subseteq \mathbb{N}} (\llbracket \varphi \rrbracket_{\mathcal{V} \cup \{X\}}(nw, \sigma[X \rightarrow I])), \\ \llbracket \forall x. \varphi \rrbracket_{\mathcal{V}}(nw, \sigma) &= \text{Val}^\omega((\llbracket \varphi \rrbracket_{\mathcal{V} \cup \{x\}}(nw, \sigma[x \rightarrow i]))_{i \in \mathbb{N}}), \\ \llbracket \forall X. \beta \rrbracket_{\mathcal{V}}(nw, \sigma) &= \begin{cases} 1, & \text{if } \llbracket \beta \rrbracket_{\mathcal{V} \cup \{X\}}(nw, \sigma[X \rightarrow I]) = 1 \text{ for all } I \subseteq \mathbb{N} \\ 0, & \text{otherwise.} \end{cases} \end{aligned}$$

We write $\llbracket \varphi \rrbracket$ for $\llbracket \varphi \rrbracket_{\text{free}(\varphi)}$, so $\llbracket \varphi \rrbracket : NW^\omega(\Sigma_{\text{free}(\varphi)}) \rightarrow D$. If φ contains no free variables, φ is a *sentence* and $\llbracket \varphi \rrbracket : NW^\omega(\Sigma) \rightarrow D$.

Example 3. Continuing Example 2 with $D = (\overline{\mathbb{R}}, \text{sup}, \text{lim avg}, +, -\infty, 0)$ we define

$$\begin{aligned} \text{pcall}(x) &= \text{call}(x) \wedge \forall w. \neg \nu(x, w), & \text{pret}(z) &= \text{ret}(z) \wedge \forall u. \neg \nu(u, z), \\ \text{bfr}(y) &= \forall x \forall z. (\neg(x < y < z \wedge \nu(x, z)) \wedge \neg(x < y \wedge \text{pcall}(x)) \wedge \neg(y < z \wedge \text{pret}(z))), \end{aligned}$$

where $x < y < z = \neg(y \leq x) \wedge \neg(z \leq y)$. Then $\llbracket \forall y. ((\text{bfr}(y) \wedge 1) \vee 0) \rrbracket = S = \|\mathcal{A}_2\|$.

Analogously to [9] and [12] we can show:

Proposition 11. *Let $\varphi \in MSO(D, NW(\Sigma))$ and let \mathcal{V} be a finite set of variables with $\text{free}(\varphi) \subseteq \mathcal{V}$. Then $\llbracket \varphi \rrbracket_{\mathcal{V}}(nw, \sigma) = \llbracket \varphi \rrbracket(nw, \sigma \upharpoonright \text{free}(\varphi))$ for each valid $(nw, \sigma) \in NW^\omega(\Sigma_{\mathcal{V}})$. Furthermore, $\llbracket \varphi \rrbracket$ is regular iff $\llbracket \varphi \rrbracket_{\mathcal{V}}$ is regular.*

Clearly, every boolean formula $\beta \in MSO(D, NW(\Sigma))$ can be interpreted as an unweighted MSO-formula $\psi \in MSO(NW(\Sigma))$ with $\llbracket \beta \rrbracket = \mathbb{1}_{L(\psi)}$, since $\llbracket \beta \rrbracket$ only yields the values 0 and 1. Conversely, for every formula $\psi \in MSO(NW(\Sigma))$ there exists a boolean MSO-formula $\beta \in MSO(D, NW(\Sigma))$ with $\llbracket \beta \rrbracket = \mathbb{1}_{L(\psi)}$, since we can replace disjunctions by conjunctions and negations and we can replace existential quantifiers by universal quantifiers and negations.

In order to obtain a Büchi-like theorem (as Theorem 17 below) for weighted automata on finite words, it is necessary to restrict the weighted MSO logic (cf. [9]). Therefore we introduce and study suitable fragments of $MSO(D, NW(\Sigma))$ as in the following.

Definition 12. *The set of almost boolean formulas is the smallest set of all formulas of $MSO(D, NW(\Sigma))$ containing all constants $d \in D$ and all boolean formulas, which is closed under disjunction and conjunction.*

Proposition 13. (a) *If $\varphi \in MSO(D, NW(\Sigma))$ is an almost boolean formula, then $\llbracket \varphi \rrbracket$ is a regular step function.*
 (b) *For every regular step function $S : NW^\omega(\Sigma) \rightarrow D$, there exists an almost boolean sentence φ with $S = \llbracket \varphi \rrbracket$.*

Definition 14. *Let $\varphi \in MSO(D, NW(\Sigma))$. We denote by $\text{const}(\varphi)$ the set of all elements of D occurring in φ . We call φ*

1. *strongly- \wedge -restricted if for all subformulas $\psi \wedge \theta$ of φ :
 Either ψ and θ are almost boolean or ψ is boolean or θ is boolean.*
2. *\wedge -restricted if for all subformulas $\psi \wedge \theta$ of φ :
 Either ψ is almost boolean or θ is boolean.*
3. *commutatively- \wedge -restricted if for all subformulas $\psi \wedge \theta$ of φ :
 Either $\text{const}(\psi)$ and $\text{const}(\theta)$ commute or ψ is almost boolean.*
4. *\forall -restricted if for all subformulas $\forall x.\psi$ of φ : ψ is almost boolean.*

We call a formula of $MSO(D, NW(\Sigma))$ *syntactically restricted* if it is both \forall -restricted and strongly- \wedge -restricted. Note that every subformula of a syntactically restricted formula is syntactically restricted itself.

Now we show that under suitable assumptions on the ω -pv-monoid D , particular classes of $MSO(D, NW(\Sigma))$ -formulas have the same expressive power. In [11] these equivalences (for unnested words) followed from the main result and thus needed constructions of automata. Here we show the equivalence of the logic fragments directly.

Theorem 15. (a) *Let D be left-distributive and $\varphi \in MSO(D, NW(\Sigma))$ be \wedge -restricted. Then there exists a strongly- \wedge -restricted formula $\varphi' \in MSO(D, NW(\Sigma))$ with $\llbracket \varphi \rrbracket = \llbracket \varphi' \rrbracket$. Moreover, if φ is also \forall -restricted, then φ' can also be chosen to be \forall -restricted.*
 (b) *Let D be a cc- ω -valuation semiring and let $\varphi \in MSO(D, NW(\Sigma))$ be commutatively- \wedge -restricted. Then there exists a strongly- \wedge -restricted formula $\varphi' \in MSO(D, NW(\Sigma))$ with $\llbracket \varphi \rrbracket = \llbracket \varphi' \rrbracket$. Moreover, if φ is also \forall -restricted, then φ' can also be chosen to be \forall -restricted.*

Proof (sketch). We use an induction on the structure of φ . The interesting case is $\varphi = \psi \wedge \theta$ and ψ is almost boolean. By induction we can assume that ψ and θ are strongly- \wedge -restricted (and resp. \forall -restricted). As an example, we consider the case of the universal quantification in (a) as follows. Assume $\theta = \forall x.\theta_1$ and ψ does not contain x . By the induction hypothesis, we obtain a strongly- \wedge -restricted formula φ_1 such that $\llbracket \varphi_1 \rrbracket = \llbracket \psi \wedge \theta_1 \rrbracket$.

First let D be left- Val^ω -distributive. Using this assumption at equation $*$, we get for $\mathcal{V} = \text{free}(\psi) \cup \text{free}(\forall x.\theta_1)$ and each $(nw, \sigma) \in \text{NW}^\omega(\Sigma_{\mathcal{V}})$:

$$\begin{aligned} \llbracket \varphi \rrbracket(nw, \sigma) &= \llbracket \psi \wedge \forall x.\theta_1 \rrbracket_{\mathcal{V}}(nw, \sigma) \\ &= \llbracket \psi \rrbracket_{\mathcal{V}}(nw, \sigma) \diamond \text{Val}^\omega(\left(\llbracket \theta_1 \rrbracket_{\mathcal{V} \cup \{x\}}(nw, \sigma[x \rightarrow i])\right)_{i \in \mathbb{N}}) \\ &\stackrel{*}{=} \text{Val}^\omega(\left(\llbracket \psi \rrbracket_{\mathcal{V}}(nw, \sigma) \diamond \llbracket \theta_1 \rrbracket_{\mathcal{V} \cup \{x\}}(nw, \sigma[x \rightarrow i])\right)_{i \in \mathbb{N}}) \\ &= \text{Val}^\omega(\left(\llbracket \psi \rrbracket_{\mathcal{V} \cup \{x\}}(nw, \sigma[x \rightarrow i]) \diamond \llbracket \theta_1 \rrbracket_{\mathcal{V} \cup \{x\}}(nw, \sigma[x \rightarrow i])\right)_{i \in \mathbb{N}}) \\ &= \text{Val}^\omega(\left(\llbracket \psi \wedge \theta_1 \rrbracket_{\mathcal{V} \cup \{x\}}(nw, \sigma[x \rightarrow i])\right)_{i \in \mathbb{N}}) \\ &= \llbracket \forall x.(\psi \wedge \theta_1) \rrbracket_{\mathcal{V}}(nw, \sigma) . \end{aligned}$$

So $\varphi' = \forall x.\varphi_1$ is strongly- \wedge -restricted and $\llbracket \varphi \rrbracket = \llbracket \varphi' \rrbracket$. If φ is \forall -restricted, θ_1 is almost boolean. In this case we can put directly $\varphi' = \forall x.(\psi \wedge \theta_1)$. Then φ' is strongly- \wedge -restricted and \forall -restricted because ψ and θ_1 are almost boolean formulas.

Now let D be left-multiplicative. Using the formulas $\text{min}(x) = \forall y.(x \leq y)$ and $\text{min}(x) \rightarrow \psi = \neg \text{min}(x) \vee (\text{min}(x) \wedge \psi)$ it can be shown that

$$\begin{aligned} \llbracket \varphi \rrbracket &= \llbracket \psi \wedge \forall x.\theta_1 \rrbracket \\ &= \llbracket \forall x.((\text{min}(x) \rightarrow \psi) \wedge \theta_1) \rrbracket \\ &= \llbracket \forall x.((\neg \text{min}(x) \wedge \theta_1) \vee (\text{min}(x) \wedge \psi \wedge \theta_1)) \rrbracket . \end{aligned}$$

Then $\varphi' = \forall x.((\neg \text{min}(x) \wedge \theta_1) \vee (\text{min}(x) \wedge \varphi_1))$ is strongly- \wedge -restricted since $\text{min}(x)$ is boolean. Furthermore, $\llbracket \varphi \rrbracket = \llbracket \varphi' \rrbracket$. If φ is \forall -restricted, we can put directly $\varphi' = \forall x.((\text{min}(x) \rightarrow \psi) \wedge \theta_1)$. Then φ' is strongly- \wedge -restricted and \forall -restricted because $\text{min}(x) \rightarrow \psi$ and θ_1 are almost boolean formulas. \square

If D is a $\text{cc-}\omega$ -valuation semiring, clearly almost boolean formulas can be written as disjunctions of conjunctions of boolean formulas or constants from D . Our proof of Theorem 15 (b) shows the following corollary.

Corollary 16. *Let D be a commutative $\text{cc-}\omega$ -valuation semiring. Then for any formula $\varphi \in \text{MSO}(D, \text{NW}(\Sigma))$ there exists a formula $\varphi' \in \text{MSO}(D, \text{NW}(\Sigma))$ in which conjunctions occur only between boolean formulas and constants such that $\llbracket \varphi \rrbracket = \llbracket \varphi' \rrbracket$.*

This follows also from a slightly modified proof of Theorem 17, but the present proof gives direct and efficient conversions of the formulas.

5 Characterization of Regular Series

In this section, we give our second main result on the expressive equivalence of weighted stair Muller nested word automata and our different fragments of weighted MSO logic.

Theorem 17. *Let D be a regular ω -pv-monoid and $S : NW^\omega(\Sigma) \rightarrow D$ a series.*

1. *The following are equivalent:*
 - (a) *S is regular.*
 - (b) *$S = \llbracket \varphi \rrbracket$ for some syntactically restricted sentence φ of $MSO(D, NW(\Sigma))$.*
2. *Let D be left-distributive. Then the following are equivalent:*
 - (a) *S is regular.*
 - (b) *$S = \llbracket \varphi \rrbracket$ for some \forall -restricted and \wedge -restricted sentence φ of $MSO(D, NW(\Sigma))$.*
3. *Let D be cc- ω -valuation semiring. Then the following are equivalent:*
 - (a) *S is regular.*
 - (b) *$S = \llbracket \varphi \rrbracket$ for some \forall -restricted and commutatively- \wedge -restricted sentence φ of $MSO(D, NW(\Sigma))$.*

Proof. '(i) \Rightarrow (ii)': We construct a syntactically restricted MSO-sentence simulating the given wsMNWA, thus showing all three statements.

'(ii) \Rightarrow (i)': By Theorem 15 we may assume φ to be syntactically restricted. We prove the regularity of $\llbracket \varphi \rrbracket$ by induction on the structure of φ as follows. If φ is almost boolean, by Propositions 13(a) and 7, $\llbracket \varphi \rrbracket$ is regular. Next we have to prove that the regularity is preserved under the non-boolean operations. We only sketch the ideas. Closure under disjunction follows from Proposition 8 and a union construction of automata. If φ is a conjunction, the regularity of $\llbracket \varphi \rrbracket$ follows from a product construction of automata. The regularity of $\llbracket \exists x.\varphi \rrbracket$ and $\llbracket \exists X.\varphi \rrbracket$ follows from Proposition 8. For $\forall x.\varphi$, φ is almost boolean. Then $\llbracket \forall x.\varphi \rrbracket$ can also be shown to be regular. \square

6 Conclusion

We have introduced a weighted automaton model for infinite nested words and weighted MSO logics. We could show that under suitable assumptions on the valuation monoids, two resp. three fragments of the weighted logics have the same expressive power with efficient conversions into the smallest fragment. Moreover, the weighted automata and our logic fragments have the same expressive power. The valuation monoids form very general weight structures; they model long-time average and discounted computations of weights as well as the classical complete semirings [9]. As in [2], we considered nested words possibly containing pending edges. We remark that our results also hold similarly for finite nested words, and our conversions of the weighted logic formulas also work, similarly, for other discrete structures like trees, cf. [10].

It would be interesting to investigate decision problems for weighted nested word automata, e.g., like done in [6], [7] for automata on words and with average or discounted computations of weights.

References

1. Alur, R., Arenas, M., Barceló, P., Etessami, K., Immerman, N., Libkin, L.: First-order and temporal logics for nested words. *Logical Methods in Computer Science* 4(4), 1–44 (2008)
2. Alur, R., Madhusudan, P.: Adding nesting structure to words. *Journal of the ACM* 56(3), 16:1–16:43 (2009)
3. Berstel, J., Reutenauer, C.: *Rational Series and Their Languages*, EATCS Monographs in Theoretical Computer Science, vol. 12. Springer (1988)
4. Bollig, B., Gastin, P.: Weighted versus probabilistic logics. In: Diekert, V., Nowotka, D. (eds.) *Developments in Language Theory. Lecture Notes in Computer Science*, vol. 5583, pp. 18–38. Springer (2009)
5. Büchi, J.R.: Weak second-order arithmetic and finite automata. *Z. Math. Logik und Grundlagen Math.* 6, 66–92 (1960)
6. Chatterjee, K., Doyen, L., Henzinger, T.A.: Quantitative languages. In: Kaminski, M., Martini, S. (eds.) *CSL. LNCS*, vol. 5213, pp. 385–400. Springer (2008)
7. Chatterjee, K., Doyen, L., Henzinger, T.A.: Expressiveness and closure properties for quantitative languages. In: *LICS*, pp. 199–208. IEEE Computer Society (2009)
8. Droste, M., Kuich, W., Vogler, H., (eds.): *Handbook of Weighted Automata. EATCS Monographs in Theoretical Computer Science*, Springer (2009)
9. Droste, M., Gastin, P.: Weighted automata and weighted logics. *Theor. Comput. Sci.* 380(1-2), 69–86 (2007)
10. Droste, M., Götze, D., Märcker, S., Meinecke, I.: Weighted tree automata over valuation monoids and their characterization by weighted logics. In: Kuich, W., Rahonis, G. (eds.) *Algebraic Foundations in Computer Science. Lecture Notes in Computer Science*, vol. 7020, pp. 30–55. Springer (2011)
11. Droste, M., Meinecke, I.: Weighted automata and weighted MSO logics for average and long-time behaviors. *Inf. Comput.* 220, 44–59 (2012)
12. Droste, M., Pibalsjommee, B.: Weighted nested word automata and logics over strong bimonoids. In: Moreira, N., Reis, R. (eds.) *17th CIAA. Lecture Notes in Computer Science*, vol. 7381, pp. 138–148. Springer (2012)
13. Droste, M., Rahonis, G.: Weighted automata and weighted logics on infinite words. In: Ibarra, O.H., Dang, Z. (eds.) *Developments in Language Theory. Lecture Notes in Computer Science*, vol. 4036, pp. 49–58. Springer (2006)
14. Eilenberg, S.: *Automata, Languages, and Machines, Volume A, Pure and Applied Mathematics*, vol. 59. Academic Press (1974)
15. Elgot, C.C.: Decision problems of finite automata design and related arithmetics. *Transactions of the American Mathematical Society* 98(1), 21–52 (1961)
16. Kuich, W., Salomaa, A.: *Semirings, Automata, Languages*, EATCS Monographs in Theoretical Computer Science, vol. 6. Springer (1986)
17. Löding, C., Madhusudan, P., Serre, O.: Visibly pushdown games. In: Lodaya, K., Mahajan, M. (eds.) *FSTTCS. Lecture Notes in Computer Science*, vol. 3328, pp. 408–420. Springer (2004)
18. Mathissen, C.: Weighted logics for nested words and algebraic formal power series. *LNCS* 6(1), 1–34 (2010), special issue of ICALP 2008
19. Salomaa, A., Soittola, M.: *Automata-Theoretic Aspects of Formal Power Series. Texts and Monographs in Computer Science*, Springer (1978)
20. Schützenberger, M.P.: On the definition of a family of automata. *Information and Control* 4(2-3), 245–270 (1961)
21. Trakhtenbrot, B.A.: Finite automata and logic of monadic predicates (in Russian). *Doklady Akademii Nauk SSR* 140, 326–329 (1961)