Nivat-Theorem and Logic for Weighted Pushdown Automata on Infinite Words

Manfred Droste, Sven Dziadek, Werner Kuich

Nivat-Theorem and Logic for Weighted Pushdown Automata on Infinite Words

Manfred Droste
Institut für Informatik, Universität Leipzig, Germany
droste@informatik.uni-leipzig.de

Sven Dziadek
Institut für Informatik, Universität Leipzig, Germany
dziadek@informatik.uni-leipzig.de

Werner Kuich
Institut für Diskrete Mathematik und Geometrie, Technische Unversität Wien, Austria
werner.kuich@tuwien.ac.at

Abstract

Recently, weighted ω-pushdown automata have been introduced by Droste, Ésik, Kuich. This new type of automaton has access to a stack and models quantitative aspects of infinite words. Here, we consider a simple version of those automata. The simple ω-pushdown automata do not use ϵ-transitions and have a very restricted stack access. In previous work, we could show this automaton model to be expressively equivalent to context-free ω-languages in the unweighted case. Furthermore, semiring-weighted simple ω-pushdown automata recognize all ω-algebraic series.

Here, we consider ω-valuation monoids as weight structures. As a first result, we prove that for this weight structure and for simple ω-pushdown automata, Büchi-acceptance and Muller-acceptance are expressively equivalent. In our second result, we derive a Nivat theorem for these automata stating that the behaviors of weighted ω-pushdown automata are precisely the projections of very simple ω-series restricted to ω-context-free languages. The third result is a weighted logic with the same expressive power as the new automaton model. To prove the equivalence, we use a similar result for weighted nested ω-word automata and apply our present result of expressive equivalence of Muller and Büchi acceptance.

2012 ACM Subject Classification Theory of computation → Logic and verification; Theory of computation → Quantitative automata; Theory of computation → Automata over infinite objects; Theory of computation → Grammars and context-free languages

Keywords and phrases Weighted automata, Pushdown automata, Infinite words, Weighted logic

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2020.44

Funding *Sven Dziadek*: supported by DFG Research Training Group 1763 (QuantLA).
Werner Kuich: partially supported by Austrian Science Fund (FWF): grant no. I1661 N25.

1 Introduction

Languages of infinite words or ω-languages are intensively researched due to their applications in model checking and verification [30, 3, 9]. Context-free languages of infinite words have been investigated in a fundamental study by Cohen and Gold [10].

Weighted languages allow us to model the use of resources. In formal language theory, we consider a word to be in the language or not. Contrary to this, weighted languages relate words to resources such as costs, gains, probabilities, counts, time, and of course Boolean values. There exist generalizations to several language classes (regular, context-free, star-free languages, etc.), to various structures (words, trees, pictures, nested words, infinite words, etc.) and to different weight structures (semirings, valuation monoids, etc.). See [20] for

an overview. While weighted context-free languages already date back to Chomsky and Schützenberger [8], more recently, Droste, Ésik, Kuich [27, 19, 16] generalized context-free languages of infinite words to the weighted setting.

In this paper, we investigate a type of weighted ω-pushdown automata called *simple* ω-reset pushdown automaton in [12]. They do not allow ϵ-transitions and the stack can only be altered by at most one symbol. Simple automata have been shown to be expressively equivalent to general pushdown automata in the unweighted case for finite words [4] and for infinite words [15] (i.e., the language classes accepted by these two kinds of automata coincide). For continuous commutative star-omega semirings we could show in [13, 12, 14] that for every ω-algebraic series r, there exists a simple ω-reset pushdown automaton with behavior r.

Here, we consider ω-valuation monoids as weight structures. They include complete semirings but also discounted and average behavior. Valuation monoids first appeared in [22] but their idea is based on [7]. By an example, we show how a basic web server and its average response time for requests can be modeled by a simple ω-pushdown automaton with weights in a suitable ω-valuation monoid.

Our first main result is the expressive equivalence of Büchi and Muller acceptance for weighted simple ω-pushdown automata; i.e., the classes of behaviors of these two weighted automata models coincide.

Then we show several closure properties for weighted ω-pushdown automata. Our second main result is a Nivat-like decomposition theorem [31] that shows that by the help of a morphism, we can express the behavior of every weighted ω-pushdown automaton as the intersection of an unweighted ω-pushdown automaton and a very simple ω-series. Nivat's theorem was extended to weighted automata of finite words over semirings by [21].

Büchi, Elgot, Trakhtenbrot [5, 26, 33] (BET-Theorem) proved that regular languages are exactly those languages definable by monadic second-order logic. Their result was extended by Lautemann, Schwentick, Thérien [29] to context-free languages. While both these former results are for finite words, we defined a logic that is expressively equivalent to context-free languages of infinite words (cf. [15]). The BET-Theorem has been extended to the weighted setting [17]. Weighted logics allow the logical description of weights of finite words [17, 24, 34] and also of infinite words [25, 18, 22].

In this paper, as the third main result, we extend the BET-Theorem to weighted simple ω-pushdown automata. We extend the logic in [29, 11] and prove its equivalence to weighted simple ω-pushdown automata. For the proof, we do not reinvent the wheel but use the already existing BET-Theorem for weighted nested ω-word automata [11]. The application of a projection allows us to lift the result on weighted nested ω-word automata to weighted simple ω-pushdown automata. We show how the quantitative behavior of the basic web server example mentioned above can be described in our weighted matching ω-MSO logic.

An expressive equivalence result for arbitrary weighted ω-pushdown automata, besides our Nivat-like result, remains open at present.

We structure the paper as follows. We give basic definitions and compare Muller and Büchi acceptance in Section 2. Then, we prove the Nivat-like result in Section 3. Section 4 defines the logic. Section 5 summarizes the known results about weighted nested ω-word languages and also shows the new projection. In Section 6, we prove our weighted BET-Theorem.

2 Weight Structure and Simple ω-Pushdown Automata

This section introduces our weight structure, the ω-valuation monoids (cf. [22]), and the weighted automata we want to discuss in this paper. At the end of this section, we give our first main result, the comparison of Muller and Büchi acceptance.

An *alphabet* denotes a finite set of symbols. Let $\mathbb{N}$ be the set of non-negative integers.

A monoid $(D, +, \mathbb{0})$ is called *complete*, if it is equipped with sum operations $\sum_I \colon D^I \to D$ for all families $(a_i \mid i \in I)$ of elements of D, where I is an arbitrary index set, such that the following conditions are satisfied:

(i) $\sum_{i\in\emptyset} d_i = \mathbb{0}$, $\sum_{i\in\{k\}} d_i = d_k$, $\sum_{i\in\{j,k\}} d_i = d_j + d_k$ for $j \neq k$, and

(ii) $\sum_{j\in J}\Big(\sum_{i\in I_j} d_i\Big) = \sum_{i\in I} d_i$ if $\bigcup_{j\in J} I_j = I$ and $I_j \cap I_k = \emptyset$ for $j \neq k$.

This means that a monoid D is complete if it has infinitary sum operations (i) that are an extension of the finite sums and (ii) that are associative and commutative (cf. [28]).

For a set D we denote by $C \subseteq_{\text{fin}} D$ that C is a finite subset of D. Let $(D_{\text{fin}})^\omega = \bigcup_{C\subseteq_{\text{fin}} D} C^\omega$. An *$\omega$-valuation monoid* $(D, +, \text{Val}^\omega, \mathbb{0})$ consists of a complete monoid $(D, +, \mathbb{0})$ and an ω-valuation function $\text{Val}^\omega \colon (D_{\text{fin}})^\omega \to D$ such that $\text{Val}^\omega(d_i)_{i\in\mathbb{N}} = \mathbb{0}$ whenever $d_i = \mathbb{0}$ for some $i \in \mathbb{N}$. A *product ω-valuation monoid* (*ω-pv-monoid*) is a tuple $(D, +, \text{Val}^\omega, \diamond, \mathbb{0}, \mathbb{1})$ where $(D, +, \text{Val}^\omega, \mathbb{0})$ is an ω-valuation monoid, $\diamond \colon D^2 \to D$ is a product function and further $\mathbb{1} \in D$, $Val^\omega(\mathbb{1}^\omega) = \mathbb{1}$ and $\mathbb{0} \diamond d = d \diamond \mathbb{0} = \mathbb{0}$, $\mathbb{1} \diamond d = d \diamond \mathbb{1} = d$ for all $d \in D$.

A monoid $(D, +, \mathbb{0})$ is called *idempotent* if $d + d = d$ for all $d \in D$. An ω-valuation monoid $(D, +, \text{Val}^\omega, \mathbb{0})$ is equally called *idempotent* if its underlying monoid $(D, +, \mathbb{0})$ is idempotent.

In [11, 22], ω-valuation monoids are classified by specific properties. More specific ω-valuation monoids will later lead to more loose restrictions on our logic. Due to space constraints, we omit properties on ω-valuation monoids here and refer the interested reader to [11]. Additionally, we will only present one possible restriction on our logic.

▶ **Example 1** (ω-valuation monoids)**.** The first two examples are inspired by [7].

1. Let $\bar{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$ and $-\infty + \infty = -\infty$. Then $(\bar{\mathbb{R}}, \sup, \limavg, +, -\infty, 0)$ is an ω-pv-monoid where $\limavg(d_i)_{i\in\mathbb{N}} = \liminf_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} d_i$.
2. Let $\bar{\mathbb{R}}_+ = \{x \in \mathbb{R} \mid x \geq 0\} \cup \{-\infty\}$. Then $(\bar{\mathbb{R}}_+, \sup, \text{disc}_\lambda, +, -\infty, 0)$ for $0 < \lambda < 1$ is an ω-pv-monoid where $\text{disc}_\lambda(d_i)_{i\in\mathbb{N}} = \lim_{n\to\infty} \sum_{i=0}^{n} \lambda^i d_i$.
3. Any complete semiring $(S, \oplus, \otimes, \mathbb{0}, \mathbb{1})$ is an ω-pv-monoid $(S, \oplus, \bigotimes, \otimes, \mathbb{0}, \mathbb{1})$.

As it simplifies the logical characterization, we follow [23, 15] and use a restricted type of pushdown automaton. We call it *simple ω-pushdown automaton*. For the unweighted setting, we proved in [15] that this automaton model is expressively equivalent to general ω-pushdown automata; for finite words, this equivalence is hidden in [4]. For the weighted case and for continuous semirings, we show a corresponding result for finite words in [13]. For weights in continuous semirings and for infinite words, we showed in [12, 14] that all ω-algebraic series are recognized by weighted simple ω-pushdown automata.

Simple ω-pushdown automata are realtime, i.e. they do not use ϵ-transitions. Additionally, we restrict transitions in a way to only allow either to keep the stack unaltered, to push one symbol or to pop one symbol. Thus, let $\mathcal{S}(\Gamma) = (\{\downarrow\} \times \Gamma) \cup \{\#\} \cup (\{\uparrow\} \times \Gamma)$ be the set of *stack commands* for a stack alphabet Γ. Note that this implies that the automaton can only read the top of the stack when popping it. Additionally, for technical reasons, we start runs with an empty stack and therefore allow to push onto the empty stack.

▶ **Definition 2.** *An (unweighted)* ω-pushdown automaton *(ωPDA) over the alphabet* Σ *is a tuple* $M = (Q, \Gamma, T, I, F)$ *where*

- Q *is a finite set of states,*
- Γ *is a finite stack alphabet,*
- $T \subseteq Q \times \Sigma \times Q \times \mathcal{S}(\Gamma)$ *is a set of transitions,*
- $I \subseteq Q$ *is the set of initial states,*
- $F \subseteq Q$ *is a set of (Büchi-accepting) final states.*

▶ **Definition 3.** *A* weighted ω-pushdown automaton *(ω WPDA) over the alphabet* Σ *and the* ω*-valuation monoid* $(D, +, Val^\omega, \mathbb{0})$ *is a tuple* $M = (Q, \Gamma, T, I, F, wt)$ *where*

- (Q, Γ, T, I, F) *is an unweighted* ω*-pushdown automaton over* Σ*,*
- $wt\colon T \to D$ *is a weight function.*

▶ **Definition 4.** *A* Muller-accepting *ω-pushdown automaton over the alphabet* Σ *is a tuple* $M = (Q, \Gamma, T, I, \mathcal{F})$ *where* Q, Γ, T, I *are defined as for ωPDA, but* $\mathcal{F} \subseteq 2^Q$ *is a set of Muller-accepting subsets of* Q*. Similarly, a weighted* Muller-accepting *ω-pushdown automaton over the alphabet* Σ *and the* ω*-valuation monoid* D *is a tuple* $M = (Q, \Gamma, T, I, \mathcal{F}, wt)$*.*

A *configuration* of an ωPDA or ωWPDA is a pair (q, γ), where $q \in Q$ and $\gamma \in \Gamma^*$. We define the transition relation between configurations as follows. Let $\gamma \in \Gamma^*$ and $t \in T$. For $t = (q, a, q', (\downarrow, A))$, we write $(q, \gamma) \vdash_M^t (q', A\gamma)$. For $t = (q, a, q', \#)$, we write $(q, \gamma) \vdash_M^t (q', \gamma)$. Finally, for $t = (q, a, q', (\uparrow, A))$, we write $(q, A\gamma) \vdash_M^t (q', \gamma)$. These three types of transitions are called *push*, *internal* and *pop* transitions, respectively.

We denote by $\mathrm{label}(q, a, q', s) = a$ the *label* and by $\mathrm{state}(q, a, q', s) = q$ the *state* of a transition. Both, as well as the function wt will be extended to infinite sequences of transitions by letting $\mathrm{label}((t_i)_{i\geq 0}) = (\mathrm{label}(t_i))_{i\geq 0} \in \Sigma^\omega$ for the infinite word constructed from the labels and similar for $\mathrm{state}((t_i)_{i\geq 0}) \in Q^\omega$ and for $\mathrm{wt}((t_i)_{i\geq 0}) \in D^\omega$.

An infinite sequence of transitions $\rho = (t_i)_{i\geq 0}$ with $t_i \in T$ is called a *run* of the ωWPDA or ωPDA M on $w = \mathrm{label}(\rho)$ iff there exists an infinite sequence of configurations $(p_i, \gamma_i)_{i\geq 0}$ with $p_0 \in I$ and $\gamma_0 = \epsilon$ such that $(p_i, \gamma_i) \vdash_M^{t_i} (p_{i+1}, \gamma_{i+1})$ for each $i \geq 0$. We abbreviate a run $\rho = (t_i)_{i\geq 0}$ with $(p_0, \gamma_0) \vdash_M^{t_0} (p_1, \gamma_1) \vdash_M^{t_1} \cdots$ where $\mathrm{label}(t_i) = a_i$ by $\rho\colon (p_0, \gamma_0) \xrightarrow{a_0} (p_1, \gamma_1) \xrightarrow{a_1} \cdots$ such that the word becomes visible.

For an infinite sequence of states $(q_i)_{i\geq 0}$, let $\mathrm{Inf}((q_i)_{i\geq 0}) = \{q \mid q = q_i \text{ for infinitely many } i \geq 0\}$ be the set of states that occur infinitely often. For Büchi-accepting automata, a run ρ is called *successful* if $\mathrm{Inf}(\mathrm{state}(\rho)) \cap F \neq \emptyset$. For Muller-accepting automata, a run ρ is called successful if $\mathrm{Inf}(\mathrm{state}(\rho)) \in \mathcal{F}$. For an ωPDA $M = (Q, \Gamma, T, I, F)$, the language *accepted* by M is denoted by $\mathcal{L}(M) = \{w \in \Sigma^\omega \mid \exists \text{ successful run of } M \text{ on } w\}$. A language $L \subseteq \Sigma^\omega$ is called *ω PDA-recognizable* if there exists an ωPDA M with $\mathcal{L}(M) = L$. For an ωWPDA M, we introduce the following function $\|M\|\colon \Sigma^\omega \to D$ which is called the *behavior* of M and which is defined by $\|M\|(w) = \sum(\mathrm{Val}^\omega(\mathrm{wt}(\rho)) \mid \rho \text{ successful run of } M \text{ on } w)$.

An ωPDA or ωWPDA M over Σ is called *unambiguous* if there exists at most one successful run of M on every word $w \in \Sigma^\omega$. If there exists an unambiguous ωPDA M with $\mathcal{L}(M) = L$, the language L is called unambiguous.

Any function $s\colon \Sigma^\omega \to D$ is called a *series* over Σ and D. The set of all such series is denoted by $D\langle\langle\Sigma^\omega\rangle\rangle$. Every series $s\colon \Sigma^\omega \to D$ which is the behavior of some ωWPDA over D is called *ω WPDA-recognizable.*

An ωWPDA $M = (Q, \Gamma, T, I, F, \mathrm{wt})$ that only uses internal transitions, i.e., for which $\Gamma = \emptyset$ and for all transitions $t = (q, a, q', s) \in T$ holds $s = \#$, is called a weighted finite automaton, or short *ω WFA*. Series that are the behavior of some ωWFA are called *ω WFA-recognizable.*

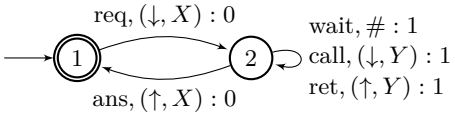

Figure 1 Example 5: Weighted ω-pushdown automaton over the alphabet $\Sigma = \{\text{req}, \text{ans}, \text{call}, \text{ret}, \text{wait}\}$ and the ω-valuation monoid $\bar{\mathbb{R}}$. The value after the ":" are the used weights 0 and 1.

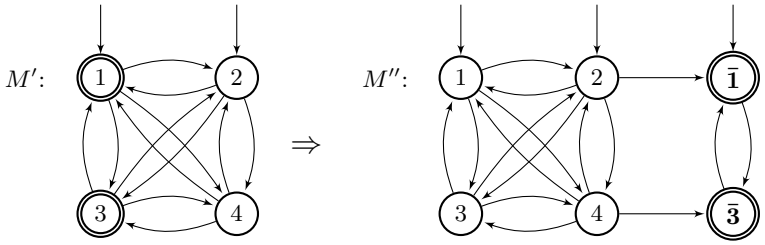

Figure 2 Proof of Theorem 6: The states 1, 2, 3, and 4 stand for the set of states that are initial and final, initial but not final, final but not initial, or neither initial nor final, respectively. Groups 1 and 3 are copied into $\bar{1}$ and $\bar{3}$. Transitions into $\bar{1}$ and $\bar{3}$ are only allowed from originally non-accepting states.

▶ **Example 5** (ωWPDA). We extend the ω-pv-monoid 1 of Example 1 as $(\bar{\mathbb{R}}, \sup, \text{specialavg}, +, -\infty, 0)$ where we define a new ω-valuation function to count and take the average of the counted values. Let h be a function that maps natural numbers to strings as follows.

$$h\colon \mathbb{N} \to \{0,1\}^*, \qquad n \mapsto 0\underbrace{11\ldots1}_{n\text{-times}}0$$

Then we extend h to infinite sequences of natural numbers $h\colon \mathbb{N}^\omega \to \{0,1\}^\omega$ in the natural way. We will consider its inverse where we have for instance $h^{-1}(011100110011110\ldots) = 324\ldots$. Then let specialavg $= \limsup \text{avg} \circ h^{-1}$. For $w \notin (01^*0)^\omega$ we set specialavg$(w) = -\infty$.

Now, we define an automaton $\mathcal{A}$ as shown in Figure 1. We let $\mathcal{A} = (\{1,2\}, \{X,Y\}, T, \{1\}, \{1\}, \text{wt})$ be an ωWPDA over the alphabet $\Sigma = \{\text{req}, \text{ans}, \text{call}, \text{ret}, \text{wait}\}$, where T is defined as shown in the Figure and the weights are indicated after the colon symbol.

The automaton simulates some kind of (web) server that takes *req*uests from clients and *ans*wers them. For every request, the server has to *call* some amount of other services and a*wait* their *ret*urns. Only when all calls have been returned, the server answers the original request. This is a context-free property. Only runs that always eventually return to state 1 to serve new clients are considered valid.

Every call, return, or wait takes one second to operate and this operation time is accounted for in the weight. The specialavg operation sums up all the waiting time per request and returns the long run average response time.

We now state our first main result.

▶ **Theorem 6.** *Let $s\colon \Sigma^\omega \to D$ be a series. The following are equivalent:*

- *s is recognizable by a Büchi-accepting ωWPDA,*
- *s is recognizable by a Muller-accepting ωWPDA.*

Proof Idea. In the direction Büchi to Muller acceptance, the standard approach works also in the weighted case.

For the other direction, the standard approach usually employs a special set of accepting states that have to be traversed to be accepted. This construction needs to be adjusted as it creates infinitely many possible runs in the Büchi automaton for every run of the Muller automaton.

A solution to this problem was presented in [25] whose construction allows exactly one entry point into the special set of accepting states. Entering the group of accepting states is forbidden from a state that is already accepting. In this way, the only successful runs are the ones that switch from the original states to the new group of accepting states at the last possible moment. In contrast to [25], we cannot assume an initially normalized automaton to solve the remaining question of the initial states that are also final.

Instead, the automaton decides non-deterministically if it will eventually see a non-final state in the run. If not, and only in this case, it already starts in the new group of accepting states. Figure 2 depicts the idea of the construction. ◀

3 Closure Properties

Let Σ, Δ be alphabets and $h\colon \Sigma \to \Delta$ a mapping. We can extend h to infinite words in the natural way by setting $h(w) = h(a_0)h(a_1)h(a_2)\cdots \in \Delta^\omega$ for $w = a_0a_1a_2\cdots \in \Sigma^\omega$.

Let now $h\colon \Delta \to \Sigma$ and let $h^{-1}(w) = \{v \in \Delta^\omega \mid h(v) = w\}$. Then for a series $s\colon \Delta^\omega \to D$, we define the series $h(s)\colon \Sigma^\omega \to D$ by $h(s)(w) = \sum_{v\in h^{-1}(w)} s(v)$ for all $w \in \Sigma^\omega$.

▶ **Lemma 7.** *Let Σ, Δ be alphabets, $(D, +, Val^\omega, \mathbb{0})$ an ω-valuation monoid and $h\colon \Delta \to \Sigma$ a mapping. If $s\colon \Delta^\omega \to D$ is ωWPDA-recognizable, then so is $h(s)\colon \Sigma^\omega \to D$.*

Let $g\colon \Sigma \to D$ be a mapping. We denote by $\mathrm{Val}^\omega \circ g\colon \Sigma^\omega \to D$ the series defined for all $w \in \Sigma^\omega$ by $(\mathrm{Val}^\omega \circ g)(w) = \mathrm{Val}^\omega(g(w))$.

▶ **Lemma 8.** *Let Σ be an alphabet, $(D, +, Val^\omega, \mathbb{0})$ an ω-valuation monoid and $g\colon \Sigma \to D$ a mapping. Then $Val^\omega \circ g$ is ωWFA-recognizable by an ωWFA with only one state.*

Let $(D, +, \mathrm{Val}^\omega, \mathbb{0})$ be an ω-valuation monoid, $s\colon \Sigma^\omega \to D$ an ωWFA-recognizable series and $L \subseteq \Sigma^\omega$ an ωPDA-recognizable language. By $s \cap L\colon \Sigma^\omega \to D$, we denote the series that assigns the weights of s to the words accepted by L. Formally, for words $u \in \Sigma^\omega$,

$$(s \cap L)(u) = \begin{cases} s(u), & \text{if } u \in L \\ \mathbb{0}, & \text{otherwise}. \end{cases}$$

▶ **Lemma 9.** *Let $(D, +, Val^\omega, \mathbb{0})$ be an ω-valuation monoid, $s\colon \Sigma^\omega \to D$ an ωWFA-recognizable series and $L \subseteq \Sigma^\omega$ an ωPDA-recognizable language.*

1. *If L is unambiguous, then the series $s \cap L\colon \Sigma^\omega \to D$ is ωWPDA-recognizable.*
2. *If D is idempotent, then the series $s \cap L\colon \Sigma^\omega \to D$ is ωWPDA-recognizable.*

Proof Idea. To allow final states of both original Büchi-accepting automata to be visited alternately, we use the standard construction for intersecting unweighted Büchi automata for infinite words, see [32] for details. The assumptions of (1), respectively (2), imply that the weights for $s \cap L$ are computed correctly (cf. [2]). ◀

Intersection with inherently ambiguous languages over non-idempotent ω-valuation monoids might not be ωWPDA-recognizable. For a counterexample, consider the ω-valuation monoid $(\mathbb{N}^\infty, +, \prod, 0)$ of natural numbers or ∞ with the natural operations, an inherently ambiguous language like e.g. $L = \{a^ib^jc^kd^\omega \mid i = j \text{ or } j = k\}$ and intersect it with the

constant series $s(u) = 1$ for all $u \in \Sigma^\omega$. But then, the intersection $(s \cap L)$ is no longer ωWPDA-recognizable which can be seen as follows. An automaton M for the series $(s \cap L)$ would yield the value 1 precisely for the words in L. Since the ω-valuation monoid $\mathbb{N}^\infty$ only allows non-negative integers or ∞ as weights, each word in L can have only one successful run in M. Stripping M of its weights while only keeping transitions with non-zero weight, we obtain an unweighted pushdown automaton M'. The new automaton M' has only one successful run for every input $u \in L$ and is thus unambiguous; moreover, M' accepts the language L. This contradicts L being inherently ambiguous.

▶ **Definition 10.** *Let Σ be an alphabet and $(D, +, Val^\omega, \mathbb{0})$ an ω-valuation monoid.*

We denote by $D^{rec}\langle\langle\Sigma^\omega\rangle\rangle$ the family of ωWPDA-recognizable series over Σ and D. Let further $D^{\mathcal{N}}\langle\langle\Sigma^\omega\rangle\rangle$ (with $\mathcal{N}$ meaning Nivat*) denote the set of series s over Σ and D such that there exist an alphabet Δ, mappings $h\colon \Delta \to \Sigma$ and $r\colon \Delta \to D$ and an ωPDA-recognizable language $L \subseteq \Delta^\omega$ such that*

$$s = h((\mathit{Val}^\omega \circ r) \cap L)\,.$$

We further define $D^{\mathcal{N}}_{UNAMB}\langle\langle\Sigma^\omega\rangle\rangle$ ($D^{\mathcal{N}}_{DET}\langle\langle\Sigma^\omega\rangle\rangle$) like $D^{\mathcal{N}}\langle\langle\Sigma^\omega\rangle\rangle$ with the difference that L is an unambiguous (deterministic, respectively) ωPDA-recognizable language.

▶ **Example 11.** We extend the ω-pv-monoid 1 of Example 1 as $(\dot{\mathbb{R}}, \sup, \text{partialavg}, +, -\infty, 0)$ where we add a new value d that will later be ignored, i.e., $\dot{\mathbb{R}} = \bar{\mathbb{R}} \cup \{d\}$. We set $\sup(-\infty, d) = d$ and $\sup(r, d) = r$ for every $r \in \mathbb{R}$. We define a new ω-valuation function to ignore d and take the average of the remaining values. Let now h be defined as follows.

$$h\colon \dot{\mathbb{R}} \to \bar{\mathbb{R}}^*, \qquad \begin{aligned} r &\mapsto r, \quad \text{for } r \in \bar{\mathbb{R}} \\ d &\mapsto \epsilon \end{aligned}$$

Then we extend h to infinite sequences $h\colon \dot{\mathbb{R}}^\omega \to \bar{\mathbb{R}}^\omega$ in the natural way. Now let $\text{partialavg} = \lim \text{avg} \circ h$ and $\Sigma = \{a, b\}$. We make the following definitions:

- $\Delta = \Sigma \times \{0, 1, \ldots, 6\}$,
- $L = \{(\sigma_1, d_1)(\sigma_2, d_2)(\sigma_3, d_3)\cdots \mid d_i = i \bmod 7,\ \sigma_i \in \Sigma\}$,
- $r(b, i) = d$ for all $i \in \{0, \ldots, 6\}$ and $r(a, i) = \begin{cases} 1, & \text{if } 5 \le i \le 6 \\ 0, & \text{otherwise}, \end{cases}$
- $h(\sigma, i) = \sigma$

The language $L \subseteq \Delta^\omega$ is obviously ωPDA-recognizable. As we will see in the following theorem, the series $s = h((\text{Val}^\omega \circ r) \cap L) \in \dot{\mathbb{R}}^{\mathcal{N}}\langle\langle\Sigma^\omega\rangle\rangle$ is ωWPDA-recognizable because $\dot{\mathbb{R}}$ is idempotent. The series s calculates the greatest accumulation point of the ratio of events a happening at the weekend (days 5 and 6) compared to all occurrences of events a.

The following is the second main Nivat-like decomposition result.

▶ **Theorem 12.** *Let Σ be an alphabet and $(D, +, Val^\omega, \mathbb{0})$ an ω-valuation monoid. Then,*

$$D^{rec}\langle\langle\Sigma^\omega\rangle\rangle = D^{\mathcal{N}}_{DET}\langle\langle\Sigma^\omega\rangle\rangle = D^{\mathcal{N}}_{UNAMB}\langle\langle\Sigma^\omega\rangle\rangle \subseteq D^{\mathcal{N}}\langle\langle\Sigma^\omega\rangle\rangle\,.$$

If D is idempotent, $D^{\mathcal{N}}\langle\langle\Sigma^\omega\rangle\rangle = D^{rec}\langle\langle\Sigma^\omega\rangle\rangle$.

Proof. First, we show $D^{\text{rec}}\langle\langle\Sigma^\omega\rangle\rangle \subseteq D^{\mathcal{N}}_{\text{DET}}\langle\langle\Sigma^\omega\rangle\rangle$: Let $s \in D^{\text{rec}}\langle\langle\Sigma^\omega\rangle\rangle$. Thus there exists an ωWPDA $M = (Q, \Gamma, T, I, F, wt)$ over Σ such that $\|M\| = s$. We will show that there exist Δ, h, r and L such that $s = h((\text{Val}^\omega \circ r) \cap L)$.

Let $\Delta = T$ and let $r = \mathrm{wt}\colon \Delta \to D$. We define $h\colon \Delta \to \Sigma$ by $h((q,a,q',s)) = a$. Note that the automaton does not allow ϵ-transitions and therefore, h is well-defined. We construct an unweighted ωPDA $M' = (Q, \Gamma, T', I, F)$ over Δ with

$$T' = \{(q,(q,a,p,s),p,s) \mid (q,a,p,s) \in T\}\,.$$

Note that M' accepts exactly the successful runs of M. As there is at most one transition of M' with label (q,a,p,s), M' is deterministic (and unambiguous). Define $L = \mathcal{L}(M')$.

Let $w \in \Sigma^\omega$. Therefore,

$$\begin{aligned} h((\mathrm{Val}^\omega \circ r) \cap L)(w) &= \sum \big(((\mathrm{Val}^\omega \circ r) \cap L)(w') \mid w' \in \Sigma^\omega \text{ and } h(w') = w\big) \\ &= \sum \big((\mathrm{Val}^\omega \circ r)(w') \mid w' \in L \text{ and } h(w') = w\big) \\ &= \sum (\mathrm{Val}^\omega(\mathrm{wt}(w')) \mid w' \text{ successful run of } M \text{ on } w) \\ &= \|M\|(w) = s(w)\,. \end{aligned}$$

The inclusions $D^{\mathcal{N}}_{\mathrm{DET}}\langle\langle\Sigma^\omega\rangle\rangle \subseteq D^{\mathcal{N}}_{\mathrm{UNAMB}}\langle\langle\Sigma^\omega\rangle\rangle \subseteq D^{\mathcal{N}}\langle\langle\Sigma^\omega\rangle\rangle$ is true by definition. The converse $D^{\mathcal{N}}_{\mathrm{UNAMB}}\langle\langle\Sigma^\omega\rangle\rangle \subseteq D^{\mathrm{rec}}\langle\langle\Sigma^\omega\rangle\rangle$ is proven by the closure properties of Lemmas 7, 8 and 9(1).

If D is idempotent, by Lemmas 7, 8 and 9(2), we get $D^{\mathcal{N}}\langle\langle\Sigma^\omega\rangle\rangle \subseteq D^{\mathrm{rec}}\langle\langle\Sigma^\omega\rangle\rangle$. ◀

The inclusion $D^{\mathcal{N}}\langle\langle\Sigma^\omega\rangle\rangle \subseteq D^{\mathrm{rec}}\langle\langle\Sigma^\omega\rangle\rangle$ does not hold in general for non-idempotent D. For the proof, one can consider an adaptation of the counterexample after Lemma 9.

4 Logic for Weighted ω-Pushdown Automata

The third main goal of this paper is a logical characterization of weighted ω-context-free languages. This section introduces this logic. It is based on [15, 17, 29].

Our logic has three components. The first component is a monadic second-order logic (MSO). By Büchi, Elgot, Trakhtenbrot [5, 6, 26, 33], MSO has the same expressive power on finite and infinite words as finite automata.

The second component adds the weights to the logic. Here, this is done by a new layer of formulas that are to be interpreted quantitatively, using the operations of the ω-pv-monoid. Formulas of the unweighted part of the logic will be interpreted as $\mathbb{0}$ or $\mathbb{1}$ in the ω-pv-monoid.

The third component is a dyadic second-order predicate – a binary relation that is called matching relation. Every formula will be allowed to use exactly one such predicate to link positions in words. A matching relation has a specific shape that makes it possible to argue about the stack in pushdown automata or the brackets in Dyck languages or even about the nesting in nested words.

Let $w \in \Sigma^\omega$. The set of all positions of w is $\mathbb{N}$. A binary relation $M \subseteq \mathbb{N} \times \mathbb{N}$ is a *matching* (cf. [29]) if M is compatible with $<$, i.e., $(i,j) \in M$ implies $i < j$, if each element i belongs to at most one pair in M, and if M is noncrossing, i.e., $(i,j) \in M$ and $(k,l) \in M$ with $i < k < j$ imply $i < l < j$. Let $\mathrm{Match}(\mathbb{N})$ denote the set of all matchings in $\mathbb{N} \times \mathbb{N}$.

Let V_1, V_2 denote countable and pairwise disjoint sets of first-order and second-order variables, respectively. We fix a *matching variable* $\mu \notin V_1 \cup V_2$. Let $\mathcal{V} = V_1 \cup V_2 \cup \{\mu\}$. Furthermore, D is always an ω-pv-monoid $(D, +, \mathrm{Val}^\omega, \diamond, \mathbb{0}, \mathbb{1})$.

▶ **Definition 13.** *Let Σ be an alphabet. The set $\omega MSO(D,\Sigma)$ of weighted matching ω-MSO formulas over Σ and D is defined by the extended Backus-Naur form*

$$\beta ::= P_a(x) \mid x \le y \mid x \in X \mid \mu(x,y) \mid \neg\beta \mid \beta \vee \beta \mid \exists x.\,\beta \mid \exists X.\,\beta$$
$$\varphi ::= d \mid \beta \mid \varphi \oplus \varphi \mid \varphi \otimes \varphi \mid \bigoplus\nolimits_x \varphi \mid \bigoplus\nolimits_X \varphi \mid \mathit{Val}_x\ \varphi$$

where $a \in \Sigma$, $d \in D$, $x, y \in V_1$ and $X \in V_2$. We call all formulas β boolean formulas.

Table 1 The semantics of boolean (left) and weighted (right) $\omega\mathrm{MSO}(D,\Sigma)$ formulas.

$(w,\sigma) \models P_a(x)$ iff $a_{\sigma(x)} = a$	$[\![\varphi \oplus \psi]\!](w,\sigma) = [\![\varphi]\!](w,\sigma) + [\![\psi]\!](w,\sigma)$
$(w,\sigma) \models x \le y$ iff $\sigma(x) \le \sigma(y)$	$[\![\varphi \otimes \psi]\!](w,\sigma) = [\![\varphi]\!](w,\sigma) \diamond [\![\psi]\!](w,\sigma)$
$(w,\sigma) \models x \in X$ iff $\sigma(x) \in \sigma(X)$	$[\![\bigoplus_x \varphi]\!](w,\sigma) = \sum_{i\in\mathbb{N}}([\![\varphi]\!](w,\sigma[x/i]))$
$(w,\sigma) \models \mu(x,y)$ iff $(\sigma(x),\sigma(y)) \in \sigma(\mu)$	$[\![\bigoplus_X \varphi]\!](w,\sigma) = \sum_{I\subseteq\mathbb{N}}([\![\varphi]\!](w,\sigma[X/I]))$
$(w,\sigma) \models \neg\varphi$ iff $(w,\sigma) \not\models \varphi$	$[\![\mathrm{Val}_x\ \varphi]\!](w,\sigma) = \mathrm{Val}^\omega(([\![\varphi]\!](w,\sigma[x/i]))_{i\in\mathbb{N}})$
$(w,\sigma) \models \varphi \vee \psi$ iff $(w,\sigma) \models \varphi$ or $(w,\sigma) \models \psi$	$[\![\beta]\!](w,\sigma) = \begin{cases} \mathbb{1}, & \text{if } (w,\sigma) \models \beta, \\ \mathbb{0}, & \text{otherwise} \end{cases}$
$(w,\sigma) \models \exists x.\, \varphi$ iff $\exists j \in \mathbb{N}.\, (w,\sigma[x/j]) \models \varphi$	
$(w,\sigma) \models \exists X.\, \varphi$ iff $\exists J \subseteq \mathbb{N}.\, (w,\sigma[X/J]) \models \varphi$	$[\![d]\!](w,\sigma) = d$

Variables denote positions in the word. $P_a(x)$ is a predicate indicating that the x-th letter of the word is a. Furthermore, $\mu(x,y)$ says that x and y will be matched. The operations $\oplus$ and $\otimes$ evaluate to the operations $+$ and $\diamond$ of the ω-pv-monoid D, respectively. The formulas $\bigoplus_x$ and $\bigoplus_X$ sum up over all possible instances of x and X, respectively. $\mathrm{Val}_x\ \varphi$ applies Val^ω to the sequence of infinitely many φ, each of them instantiated with a position $x \in \mathbb{N}$.

Let $\bar{\mathcal{V}}$ be the collection of all $\mathcal{V}$-*assignments*, i.e., mappings $\sigma\colon \mathcal{V} \to \mathbb{N} \cup 2^{\mathbb{N}} \cup \mathrm{Match}(\mathbb{N})$ where $\sigma(V_1) \subseteq \mathbb{N}$, $\sigma(V_2) \subseteq 2^{\mathbb{N}}$ and $\sigma(\mu) \in \mathrm{Match}(\mathbb{N})$. Let σ be a $\mathcal{V}$-assignment. For $x \in V_1$ and $j \in \mathbb{N}$, the update $\sigma[x/j]$ is the $\mathcal{V}$-assignment σ' with $\sigma'(x) = j$ and $\sigma'(y) = \sigma(y)$ for all $y \in \mathcal{V} \setminus \{x\}$. The updates $\sigma[X/J]$ and $\sigma[\mu/M]$ are defined similarly.

Let $\varphi \in \omega\mathrm{MSO}(D,\Sigma)$ be a formula, $w = a_0a_1a_2\ldots \in \Sigma^\omega$ and let σ be a $\mathcal{V}$-assignment. We inductively define $(w,\sigma) \models \varphi$ if φ is boolean and $[\![\varphi]\!]\colon \Sigma^\omega \times \bar{\mathcal{V}} \to D$ if φ is non-boolean over the structure of φ as shown in the Table 1, where $a \in \Sigma$, $d \in D$, $x,y \in V_1$ and $X \in V_2$. The logical counterparts $\wedge$, $\to$, $\forall x.\,\varphi$, $\forall X.\,\varphi$, $x \neq y$, $x < y$ and $i < j < k$ can be gained from negation and the existing operators.

Note how formulas $\phi \otimes \psi$ are evaluated by the product operation $\diamond$ in the ω-pv-monoid and also note that our ωWPDAs do not have direct access to this operation. However, the first two layers of our logic, the $\omega\mathrm{MSO}(D,\Sigma)$ formulas, will be translated into weighted nested ω-word automata and simple series of those automata are closed under intersection and therefore, $\diamond$ can be translated by a product construction.

We now define $\mathrm{MATCHING}(\mu) \in \omega\mathrm{MSO}(D,\Sigma)$ which ensures that $\mu \in \mathrm{Match}(\mathbb{N})$. Let

$$\begin{aligned}\mathrm{MATCHING}(\mu) = {} & \forall x\forall y.\,(\mu(x,y) \to x < y) \wedge \\ & \forall x\forall y\forall k.\,\big((\mu(x,y) \wedge k \neq x \wedge k \neq y) \to \neg\mu(x,k) \wedge \neg\mu(k,x) \wedge \neg\mu(y,k) \wedge \neg\mu(k,y)\big) \wedge \\ & \qquad \forall x\forall y\forall k\forall l.\,\big((\mu(x,y) \wedge \mu(k,l) \wedge x < k < y) \to x < l < y\big)\,.\end{aligned}$$

▶ **Definition 14.** *The set of formulas of* weighted matching ω-logic *over* Σ *and* D, $\omega\mathrm{ML}(D,\Sigma)$, *denotes the set of all formulas* ψ *of the form*

$$\psi = \bigoplus\nolimits_\mu \big(\varphi \otimes \mathit{MATCHING}(\mu)\big)\,,$$

for short $\psi = \bigoplus_\mu^{match} \varphi$, *where* $\varphi \in \omega\mathit{MSO}(D,\Sigma)$.

Let $\psi = \bigoplus_\mu^{\mathrm{match}} \varphi$, $w \in \Sigma^\omega$ and let σ be a $\mathcal{V}$-assignment. Then,

$$[\![\psi]\!](w,\sigma) = \sum_{M\in\mathrm{Match}(\mathbb{N})} ([\![\varphi]\!](w,\sigma[\mu/M]))\,.$$

Let $\psi \in \omega\mathrm{ML}(D, \Sigma)$. We denote by $\mathrm{Free}(\psi) \subseteq \mathcal{V}$ the set of *free variables* of ψ. A formula ψ with $\mathrm{Free}(\psi) = \emptyset$ is called a *sentence*. For a sentence ψ, $[\![\psi]\!](w, \sigma)$ does not depend on σ. It will therefore be omitted and we only write $[\![\psi]\!](w)$ where the series $[\![\psi]\!] \colon \Sigma^\omega \to D$ is called *defined* by ψ. A series $s \colon \Sigma^\omega \to D$ is *weighted-ωML-definable* if there exists a sentence $\psi \in \omega\mathrm{ML}(D, \Sigma)$ such that $[\![\psi]\!] = s$.

▶ **Example 15.** Here we define a logical sentence that defines the same series as in Example 5. Consider the same ω-pv-monoid $(\bar{\mathbb{R}}, \sup, \mathrm{specialavg}, +, -\infty, 0)$ as there.

The subformula $p_{\text{structure}}$ ensures that the first symbol is a request and that requests occur directly after answers. The formula p_{matching} relates corresponding call and returns and forbids calls without returns and vice versa. Furthermore, calls must be returned before giving the answer to the clients. Finally, the server has to serve clients infinitely often.

$$
\begin{aligned}
\mathrm{next}(x, y) &= x < y \wedge \neg(\exists z.\, x \leq z \leq y) \\
\mathrm{first}(x) &= \forall y.\, x \leq y \\
p_{\text{structure}} &= \forall x.\, (\mathrm{first}(x) \to P_{\text{req}}(x)) \wedge \forall x \forall y.\, \mathrm{next}(x, y) \to (P_{\text{ans}}(x) \leftrightarrow P_{\text{req}}(y)) \\
p_{\text{matching}} &= \forall x.\, P_{\text{call}}(x) \to \exists y.\, P_{\text{ret}}(y) \wedge \mu(x, y) \\
&\quad \wedge \forall y.\, P_{\text{ret}}(y) \to \exists x.\, P_{\text{call}}(x) \wedge \mu(x, y) \\
&\quad \wedge \forall x.\, \forall y.\, \big[\mu(x, y) \to \neg\big(\exists z.\, x \leq z \leq y \wedge P_{\text{ans}}(z)\big)\big] \\
&\quad \wedge \forall x.\, \forall y.\, \big[\mu(x, y) \to \big((P_{\text{req}}(x) \wedge P_{\text{ans}}(y)) \vee (P_{\text{call}}(x) \wedge P_{\text{ret}}(y))\big)\big] \\
p_{\text{inf\_serving}} &= \forall x.\, \exists y.\, (x < y \wedge P_{\text{req}}(y)) \\
\varphi_{\text{unweighted}} &= p_{\text{structure}} \wedge p_{\text{matching}} \wedge p_{\text{inf\_serving}}
\end{aligned}
$$

The weights of the words are distributed depending on the symbol and the formula $\varphi_{\text{weighted}}$ also applies the Val^ω function to the resulting sequence of weights.

$$
\varphi_{\text{weighted}} = \mathrm{Val}_x \left[\big(P_{\text{req}}(x) \vee P_{\text{ans}}(x)\big) \oplus \big((P_{\text{call}}(x) \vee P_{\text{ret}}(x) \vee P_{\text{wait}}(x)) \otimes 1\big)\right]
$$

Then, we we quantify over the matching variable and only consider the weight calculated in $\varphi_{\text{weighted}}$ if the formula $\varphi_{\text{unweighted}}$ is true:

$$
\psi = \bigoplus\nolimits_{\mu}^{\text{match}} \varphi_{\text{unweighted}} \otimes \varphi_{\text{weighted}}
$$

Finally, we have $[\![\psi]\!] = \|\mathcal{A}\|$ for the ωWPDA $\mathcal{A}$ of Example 5.

The weighted matching ω-logic, $\omega\mathrm{ML}(D, \Sigma)$, contains exactly one predicate μ and exactly one quantification over it. This corresponds to the behavior of pushdown automata where exactly one pushdown tape is used. In contrast, the pushdown automaton uses the ω-valuation function Val^ω only once per run and never recursively. As formulas $\mathrm{Val}_x \mathrm{Val}_y\ \varphi \in \omega\mathrm{MSO}(D, \Sigma)$ are not always translatable into automata, we follow [11, 17, 22] and define some possible restrictions of our logic.

The set of *almost boolean formulas* is the smallest set of all formulas of $\omega\mathrm{MSO}(D, \Sigma)$ containing all constants $d \in D$ and all boolean formulas which is closed under $\oplus$ and $\otimes$.

▶ **Definition 16** ([11, 22]). *Let $\varphi \in \omega MSO(D, \Sigma)$. We call φ*

1. strongly-$\otimes$-restricted *if for all subformulas $\mu \otimes \nu$ of φ: either μ and ν are almost boolean or μ is boolean or ν is boolean.*
2. Val-restricted *if for all subformulas $Val_x\ \mu$ of φ, μ is almost boolean.*
3. syntactically restricted *if it is both Val-restricted and strongly-$\otimes$-restricted.*

Let now $\psi = \bigoplus_{\mu}^{match} \varphi \in \omega ML(D, \Sigma)$. For $X \in \{$strongly-$\otimes$, Val, syntactically$\}$, we also say that ψ is X-restricted if φ is X-restricted.

The following will be the third main result. *Regular* ω-pv-monoids will be defined in the next section on page 11 as they depend on nested ω-word automata. We will prove the following theorem in Section 6.

▶ **Theorem 17.** *Let D be a regular ω-pv-monoid and $s\colon \Sigma^\omega \to D$ be a series. The following are equivalent:*

1. *s is ωWPDA-recognizable*
2. *There is a syntactically restricted $\omega ML(D,\Sigma)$-sentence φ with $[\![\varphi]\!] = s$.*

5 Weighted Nested ω-Word Languages

The $\omega\mathrm{MSO}(D,\Sigma)$ formulas correspond exactly to weighted nested ω-word languages [11] (cf. [1]). In fact, without considering the existential quantification over the matching relation $\exists^{\mathrm{match}}\mu$, the matching must explicitly be encoded in the words; the result is a nested word. Because of limited space, we refrain from a detailed definition of weighted nested ω-word automata and refer the reader to [11].

A *nested ω-word* nw over Σ is a pair $(w,\nu) = (a_0a_1a_2\ldots,\nu)$ where $w \in \Sigma^\omega$ is an ω-word and $\nu \in \mathrm{Match}(\mathbb{N})$ is a matching relation over $\mathbb{N}$. Let $NW^\omega(\Sigma)$ denote the set of all nested ω-words over Σ. For two positions $i,j \in \mathbb{N}$ with $\nu(i,j)$, we call i a *call position* and j a *return position*. If i is neither call nor return, we call it an *internal position*. A position i for $i \in \mathbb{N}$ is called *top-level* if there exist no positions $j,k \in \mathbb{N}$ with $j < i < k$ and $\nu(j,k)$.

A *weighted stair Muller nested word automaton* (ωWNWA) as defined in [11] is a Muller automaton on nested ω-words (w,ν) that for every return position has access to the state at the corresponding call position. The stair Muller acceptance condition is a Muller acceptance condition used exclusively on top-level position, i.e., only the states occurring infinitely often in the infinite sequence of top-level positions are considered.

Every function $s\colon NW^\omega(\Sigma) \to D$ is called a *nested ω-word series* (nw-series). Every nw-series s which is the behavior of some ωWNWA over D is called *ωWNWA-recognizable*.

We will now discuss how ωMSO is an equivalent logic to ωWNWAs. Note that $\omega\mathrm{MSO}(D,\Sigma)$ formulas may contain the free variable μ. Given a nested word $nw = (w,\nu)$, we let $\sigma(\mu) = \nu$ and make no difference between $(w,\sigma) \in \Sigma^\omega \times (\{\mu\} \to \mathrm{Match}(\mathbb{N}))$ and the nested word nw. We extend the semantics definitions as follows. Let $\varphi \in \omega\mathrm{MSO}(D,\Sigma)$ and $\mathrm{Free}(\varphi) \subseteq \{\mu\}$, then we define $[\![\varphi]\!]_{\mathrm{nw}}\colon NW^\omega(\Sigma) \to D$ by letting

$$[\![\varphi]\!]_{\mathrm{nw}}(w,\nu) = [\![\varphi]\!](w,\sigma) \qquad \text{for } \sigma(\mu) = \nu\,.$$

Let $d \in D$ denote the *constant series* with value d, i.e., $d(nw) = d$ for each $nw \in NW^\omega(\Sigma)$.

An ω-pv-monoid D is called *regular* if all constant series of D are ωWNWA-recognizable. In other words, D is regular if for any alphabet Σ, we have: For each $d \in D$, there exists an ωWNWA A_d with $\|A_d\| = d$.

Note that for this paper, regularity of ω-pv-monoids is defined by the means of ωWNWAs. In the proof of Theorem 18, this is used in the structural induction as a logical formula $\varphi = d$, for a weight d, can otherwise not necessarily be translated into an automaton.

Sufficient properties for an ω-pv-monoid to be regular are shown in [22]. Especially left-multiplicative and left-Val^ω-distributive ω-pv-monoids are regular, i.e., if we have $d \diamond \mathrm{Val}^\omega((d_i)_{i\in\mathbb{N}}) = \mathrm{Val}^\omega((d \diamond d_i)_{i\in\mathbb{N}})$ or $d \diamond \mathrm{Val}^\omega((d_i)_{i\in\mathbb{N}}) = \mathrm{Val}^\omega(d \diamond d_0, (d_i)_{i\geq 1})$ for all $d \in D$ and $(d_i)_{i\in\mathbb{N}} \in D^\omega$, then D is regular because we can easily construct ωWNWAs (and even ωWFAs) for every constant series. All ω-pv-monoids in Example 1 are regular.

▶ **Theorem 18** ([11]). *Let D be a regular ω-pv-monoid and $s\colon NW^\omega(\Sigma) \to D$ be a nw-series. The following are equivalent:*

1. *s is ωWNWA-recognizable*
2. *There is a syntactically restricted ωMSO(D,Σ)-formula φ with Free$(\varphi) \subseteq \{\mu\}$ and $[\![\varphi]\!]_{nw} = s$.*

The mapping $\pi\colon NW^\omega(\Sigma) \to \Sigma^\omega$ removes the nesting relation from the nested word, i.e., for $nw = (w, \nu)$, we define $\pi(nw) = w$. This can be extended to nw-series $s\colon NW^\omega(\Sigma) \to D$ by setting $\pi(s)(w) = \sum_{nw \in \pi^{-1}(w)} s(nw)$ which equals $\pi(s)(w) = \sum_{M \in \mathrm{Match}(\mathbb{N})} s(w, \emptyset[\mu/M])$.

The following is crucial for the rest of the paper.

▶ **Lemma 19.** *Let $s\colon NW^\omega(\Sigma) \to D$ be an ωWNWA-recognizable nw-series. Then the series $\pi(s) : \Sigma^\omega \to D$ is ωWPDA-recognizable.*

For unweighted languages, there is a similar proof in [4, 15]. Here, the proof is more complicated because the acceptance conditions differ. We have to construct a Büchi-accepting pushdown automaton from a stair Muller nested-word automaton.

Proof. By Theorem 6, it suffices to construct a Muller-accepting ωWPDA from a given ωWNWA. We simulate the transitions of the ωWNWA by pushing states onto the stack. Additionally, we enrich the states by the information if we are top-level or not. This information is also pushed onto the stack for the reconstruction of the top-level property upon popping. Furthermore, we allow the new Muller-accepting ωWPDA to visit arbitrary subsets of states that are not top-level in between the original Muller-accepting states. ◀

6 Equivalence of Logic and Automata

This section proves the equivalence of $\omega\mathrm{ML}(D,\Sigma)$ and weighted simple ω-pushdown automata.

▶ **Lemma 20.** *Let D be a regular ω-pv-monoid and $s\colon \Sigma^\omega \to D$ be an ωWPDA-recognizable series. Then s is ωML-definable by a syntactically restricted $\omega\mathrm{ML}(D,\Sigma)$-sentence.*

Proof. The proof builds a syntactically restricted $\omega\mathrm{ML}(D,\Sigma)$-sentence θ such that $[\![\theta]\!] = s$. The sentence θ defines exactly the behavior of an ωWPDA. Hereby, we proceed similarly to [15] and [17, 34, 11]. ◀

▶ **Lemma 21.** *Let D be a regular ω-pv-monoid and let ψ be a syntactically restricted $\omega\mathrm{ML}(D,\Sigma)$-sentence. Then $[\![\psi]\!]\colon \Sigma^\omega \to D$ is ωWPDA-recognizable.*

Proof. Let $\psi = \bigoplus_\mu^{\mathrm{Match}} \varphi$ for $\varphi \in \omega\mathrm{MSO}(D,\Sigma)$. Apply Theorem 18 to infer that $[\![\varphi]\!]_{\mathrm{nw}}$ is ωWNWA-recognizable. Now, we use the projection $\pi\colon NW^\omega(\Sigma) \to \Sigma^\omega$ of Section 5 to get $\pi([\![\varphi]\!]_{\mathrm{nw}})(w) = \sum_{M \in \mathrm{Match}(\mathbb{N})}([\![\varphi]\!](w, \emptyset[\mu/M])) = [\![\psi]\!](w)$. By Lemma 19, $[\![\psi]\!] = \pi([\![\varphi]\!]_{\mathrm{nw}})$ is ωWPDA-recognizable. ◀

Proof of Theorem 17. This is immediate by Lemmas 20 and 21. ◀

7 Conclusion

We defined ω-pv-monoids and ω-pushdown automata with weights from ω-pv-monoids. We first generalized a fundamental result of unweighted automata theory: Büchi acceptance and Muller acceptance are expressively equivalent; we can show that this remains the case for weighted simple pushdown automata of infinite words.

For the class of languages recognized by our automata, we proved several closure properties and, as our second main result, a Nivat-like decomposition theorem. It states that the weighted languages in our class are induced by an unweighted context-free language and a very simple weighted part; the two components can be intersected and a projection of this intersection gives us the original language.

The third main result is an expressively equivalent logic. This logic has three layers. The first layer basically describes nested ω-word-languages. The first two layers together describe weighted nested ω-word-languages. The third layer existentially quantifies the matching variable and corresponds to a projection from nested words to context-free languages. In this way, we can apply the Büchi-Elgot-Trakhtenbrot-Theorem for weighted regular nested ω-word-languages to obtain our equivalence result.

The present result raises the question how weighted simple ω-pushdown automata on ω-valuation monoids and therefore also our weighted matching ω-logic relate to a corresponding notion of weighted context-free ω-grammars; for weighted simple ω-pushdown automata over commutative complete star-omega semirings, this was described in [12].

In Theorem 17, it would be desirable to generalize the notion of regular ω-pv-monoids to only require ωWPDA instead of ωWNWA. The classical inductive proof method of Theorem 18 not longer works in this case. However it seems that ω-pv-monoids where constant series are ωWPDA-recognizable but not ωWNWA-recognizable are very artificial.

References

1 R. Alur and P. Madhusudan. Visibly pushdown languages. In *ACM Symposium on Theory of Computing (STOC 2004)*, pages 202–211, 2004. doi:10.1145/1007352.1007390.

2 P. Babari and M. Droste. A Nivat theorem for weighted picture automata and weighted MSO logics. *J. Comput. Syst. Sci.*, 104:41–57, 2019. doi:10.1016/j.jcss.2017.02.009.

3 C. Baier and J.-P. Katoen. *Principles of Model Checking*. The MIT Press, 2008.

4 A. Blass and Y. Gurevich. A note on nested words. *Microsoft Research*, 2006. URL: https://www.microsoft.com/en-us/research/publication/180-a-note-on-nested-words/.

5 J. R. Büchi. Weak second-order arithmetic and finite automata. *Mathematical Logic Quarterly*, 6:66–92, 1960. doi:10.1002/malq.19600060105.

6 J. R. Büchi. Symposium on decision problems: On a decision method in restricted second order arithmetic. In *Logic, Methodology and Philosophy of Science*, volume 44 of *Studies in Logic and the Foundations of Mathematics*, pages 1–11. Elsevier, 1966. doi:10.1016/S0049-237X(09)70564-6.

7 K. Chatterjee, L. Doyen, and T. A. Henzinger. Quantitative languages. In *Computer Science Logic (CSL 2008)*, pages 385–400. Springer, 2008. doi:10.1007/978-3-540-87531-4_28.

8 N. Chomsky and M. P. Schützenberger. The algebraic theory of context-free languages. In *Studies in Logic and the Foundations of Mathematics*, volume 35: Computer Programming and Formal Systems, pages 118–161. Elsevier, 1963. doi:10.1016/S0049-237X(08)72023-8.

9 E. M Clarke, T. A Henzinger, H. Veith, and R. P. Bloem. *Handbook of Model Checking*. Springer, 2016. doi:10.1007/978-3-319-10575-8.

10 R. S. Cohen and A. Y. Gold. Theory of ω-languages I: Characterizations of ω-context-free languages. *Journal of Computer and System Sciences*, 15(2):169–184, 1977. doi:10.1016/S0022-0000(77)80004-4.

11 M. Droste and S. Dück. Weighted automata and logics for infinite nested words. *Information and Computation*, 253:448–466, 2017. doi:10.1016/j.ic.2016.06.010.

12 M. Droste, S. Dziadek, and W. Kuich. Greibach normal form for ω-algebraic systems and weighted simple ω-pushdown automata. In *Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2019)*, volume 150 of *LIPIcs*, pages 38:1–38:14, 2019. doi:10.4230/LIPIcs.FSTTCS.2019.38.

13 M. Droste, S. Dziadek, and W. Kuich. Weighted simple reset pushdown automata. *Theoretical Computer Science*, 777:252–259, 2019. doi:10.1016/j.tcs.2019.01.016.

14 M. Droste, S. Dziadek, and W. Kuich. Greibach normal form for ω-algebraic systems and weighted simple ω-pushdown automata, 2020. Submitted. arXiv:2007.08866.

15 M. Droste, S. Dziadek, and W. Kuich. Logic for ω-pushdown automata. *Information and Computation*, 2020. Special issue on "Weighted Automata", Accepted for publication.

16 M. Droste, Z. Ésik, and W. Kuich. The triple-pair construction for weighted ω-pushdown automata. In *Conference on Automata and Formal Languages (AFL 2017)*, volume 252 of *Electronic Proceedings in Theoretical Computer Science*, pages 101–113, 2017. doi:10.4204/EPTCS.252.12.

17 M. Droste and P. Gastin. Weighted automata and weighted logics. *Theoretical Computer Science*, 380(1-2):69–86, 2007. doi:10.1016/j.tcs.2007.02.055.

18 M. Droste and P. Gastin. Weighted automata and weighted logics. In M. Droste, W. Kuich, and H. Vogler, editors, *Handbook of Weighted Automata*, EATCS Monographs in Theoretical Computer Science, chapter 5, pages 175–211. Springer, 2009. doi:10.1007/978-3-642-01492-5_5.

19 M. Droste and W. Kuich. A Kleene theorem for weighted ω-pushdown automata. *Acta Cybernetica*, 23:43–59, 2017. doi:10.14232/actacyb.23.1.2017.4.

20 M. Droste, W. Kuich, and H. Vogler, editors. *Handbook of Weighted Automata*. EATCS Monographs in Theoretical Computer Science. Springer, 2009. doi:10.1007/978-3-642-01492-5.

21 M. Droste and D. Kuske. Weighted automata. In J.-E. Pin, editor, *Handbook of Automata Theory*, chapter 4. European Mathematical Society, to appear.

22 M. Droste and I. Meinecke. Weighted automata and weighted MSO logics for average and long-time behaviors. *Information and Computation*, 220:44–59, 2012. doi:10.1016/j.ic.2012.10.001.

23 M. Droste and V. Perevoshchikov. A logical characterization of timed pushdown languages. In *Computer Science Symposium in Russia (CSR 2015)*, volume 9139 of *LNCS*, pages 189–203. Springer, 2015. doi:10.1007/978-3-319-20297-6_13.

24 M. Droste and V. Perevoshchikov. Logics for weighted timed pushdown automata. In *Fields of Logic and Computation II*, pages 153–173. Springer, 2015. doi:10.1007/978-3-319-23534-9_9.

25 M. Droste and G. Rahonis. Weighted automata and weighted logics on infinite words. In *Developments in Language Theory (DLT 2006)*, volume 54, pages 49–58. Springer, 2006. doi:10.1007/11779148_6.

26 C. C. Elgot. Decision problems of finite automata design and related arithmetics. *Transactions of the American Mathematical Society*, 98:21–51, 1961. doi:10.2307/2270940.

27 Z. Ésik and W. Kuich. A semiring-semimodule generalization of ω-context-free languages. In *Theory Is Forever*, volume 3113 of *LNCS*, pages 68–80. Springer, 2004. doi:10.1007/978-3-540-27812-2_7.

28 D. Krob. Monoides et semi-anneaux complets. *Semigroup Forum*, 36:323–339, 1987. doi:10.1007/BF02575025.

29 C. Lautemann, T. Schwentick, and D. Thérien. Logics for context-free languages. In *Computer Science Logic (CSL 1994)*, volume 933 of *LNCS*, pages 205–216. Springer, 1994. doi:10.1007/BFb0022257.

30 K. L. McMillan. *Symbolic Model Checking*. Kluwer Academic Publishers, 1993. doi:10.1007/978-1-4615-3190-6.

31 M. Nivat. Transductions des langages de Chomsky. *Annales de l'Institut Fourier*, 18(1):339–455, 1968. doi:10.5802/aif.287.

32 W. Thomas. Automata on infinite objects. In *Handbook of Theoretical Computer Science*, volume B: Formal Models and Semantics, chapter 4, pages 133–191. Elsevier, 1990. doi:10.1016/B978-0-444-88074-1.50009-3.

33 B. A. Trakhtenbrot. Finite automata and the logic of single-place predicates. *Doklady Akademii Nauk*, 140(2):326–329, 1961. In Russian. URL: http://mi.mathnet.ru/dan25511.

34 H. Vogler, M. Droste, and L. Herrmann. A weighted MSO logic with storage behaviour and its Büchi-Elgot-Trakhtenbrot theorem. In *Language and Automata Theory and Applications (LATA 2016)*, volume 9618 of *LNCS*, pages 127–139. Springer, 2016. doi:10.1007/978-3-319-30000-9_10.