A NIVAT THEOREM FOR WEIGHTED ALTERNATING AUTOMATA OVER COMMUTATIVE SEMIRINGS

GUSTAV GRABOLLE ⊙

Institute of Computer Science, Leipzig University, 04109 Leipzig, Germany
e-mail address: grabolle@informatik.uni-leipzig.de

ABSTRACT. This paper connects the classes of weighted alternating finite automata (WAFA), weighted finite tree automata (WFTA), and polynomial automata (PA).

First, we investigate the use of trees in the run semantics for weighted alternating automata and prove that the behavior of a weighted alternating automaton can be characterized as the composition of the behavior of a weighted finite tree automaton and a specific tree homomorphism if weights are taken from a commutative semiring.

Based on this, we give a Nivat-like characterization for weighted alternating automata. Moreover, we show that the class of series recognized by weighted alternating automata is closed under inverses of homomorphisms, but not under homomorphisms. Additionally, we give a logical characterization of weighted alternating automata, which uses weighted MSO logic for trees.

Finally, we investigate the strong connection between weighted alternating automata and polynomial automata. We prove: A weighted language is recognized by a weighted alternating automaton iff its reversal is recognized by a polynomial automaton. Using the corresponding result for polynomial automata, we are able to prove that the ZERONESS problem for weighted alternating automata with weights taken from the rational numbers is decidable.

1. INTRODUCTION

Non-determinism, a situation with several possible outcomes, is usually interpreted as a choice. This view has deep historical and philosophical roots and is adapted to automata theory in the following way: An (existential) automaton accepts if there exists at least one successful run. However, we may as well view a situation with several possible outcomes as an obligation: A universal automaton accepts only if all possible runs are successful. While this notion of "universal non-determinism" is less prominent, without further context it is as natural as the well known (existential) non-determinism. Allowing for the simultaneous use of existential and universal non-determinism leads to the concept of alternation, such as in alternating Turing machines [CKS81] or alternating automata on finite [BL80], or infinite

structures [DS87]. States of an alternating finite automaton (AFA) are either existential, or universal. For an existential state at least one of the outgoing runs needs to be successful, for a universal state all outgoing runs need to be successful to make the entire run successful. It is even possible to mix both modes by assigning a propositional formula over the states to each pair of state and letter. Alternating finite automata have been known for a long time. They are more succinct than finite automata and constructions like the complement, or intersection are easy for them. Due to this, they have many uses such as a stepping stone between logics and automata [DGV13], or in program verification [Var95].

While alternating automata recognize the same class of languages as finite automata, the situation is different in the weighted setting. A weighted finite automaton (WFA) assigns a weight to each of its transitions. The weight of a run is computed by multiplying its transition weights. Finally, the automaton assigns to each input the sum over all weights of runs corresponding to this input. By this, a weighted automaton recognizes a quantitative language i.e. a mapping from the set of words into a weight structure. Depending on the weight structure used, we may view a quantitative language as a probability distribution over the words, as a cost or yield assignment, or as the likelihood or quantity of success for each input. To simultaneously allow for a multitude of interesting weight structures, weighted automata have been studied over arbitrary semirings [DKV09].

To adapt alternating automata into the weighted setting, it can be observed that the existence of a run in a finite automaton becomes a sum over all runs in a weighted automaton. Analogously, the demand for all runs to be successful becomes a product over all runs. More precisely, if a weighted alternating finite automaton (WAFA) is in an additive state, it will evaluate to the sum over the values of all outgoing runs. If the weighted alternating automaton is in a multiplicative state, it will evaluate to the product over the values of all outgoing runs. And again, we are able to mix both modes, this time by assigning polynomials over the states to each pair of state and letter. Weighted alternating automata over infinite words were studied in [CDH09] and in [AK11] over finite words. While these authors focused on very specific weight structures, a more recent approach defines weighted alternating automata over arbitrary commutative semirings [KM18].

Weighted alternating automata have the same expressive power as weighted automata if and only if the semiring used is locally finite [KM18]. However, for many interesting semirings such as the rational numbers, weighted alternating automata are strictly more expressive than weighted automata. While we have a fruitful framework for weighted automata, woven by results like the Nivat theorem for weighted automata [DK21], the equivalence of weighted automata and weighted rational expressions [Sch61] and weighted restricted MSO logic [DG07], or the decidability of equality due to minimization if weights are taken from a field [Sch61] and many more, no such results are known for weighted alternating automata. In this paper we will extend the results on weighted alternating automata by connecting them to known formalisms and thereby establishing further characterizations of quantitative languages recognized by weighted alternating automata. From there on, we will use these connections to prove interesting properties for weighted alternating automata and vice versa to translate known results for weighted alternating automata into other settings.

After a brief recollection of basic notions and notations in Section 2, Section 3 will establish several normal forms for weighted alternating automata (Lemma 3.1, Lemma 3.2) that are used as the basis of later proofs. Section 4 includes our core result (Theorem 4.5), a characterization of weighted alternating automata by the concatenation of weighted finite tree automata (WFTA) together with certain homomorphisms. More precisely, we

consider word-to-tree homomorphisms that translate words viewed as trees into trees over some arbitrary ranked alphabet. We can show that a quantitative language is recognized by a weighted alternating automaton if, and only if there exists word-to-tree homomorphism and a weighted tree automaton such that the evaluation of the weighted alternating automata on any given word is the same as the evaluation of the weighted tree automaton on the image of the homomorphism of this word.

In Section 5 we will use this result to prove that the class of quantitative languages recognized by weighted alternating automata is closed under inverses of homomorphisms (Corollary 5.2). However, we can prove the same is not true for homomorphisms in general (Lemma 5.3). Since the closure under homomorphisms plays a key part in the proof of the Nivat theorem for weighted automata this prohibits a one-to-one translation of the Nivat theorem for weighted automata into the setting of weighted alternating automata. Nonetheless, we will utilize the connection between weighted alternating automata and weighted tree automata, as well as a Nivat theorem for weighted tree automata, to prove an adequate result for weighted alternating automata (Theorem 5.7). This will lead us directly into a logical characterization of quantitative languages recognized by weighted alternating automata with the help of weighted restricted MSO logic for weighted tree automata (Section 6 Theorem 6.2). It is well known that recognizable tree languages are closed under inverses of tree homomorphisms. However, the same does not hold in the weighted setting for arbitrary commutative semirings. Section 7 gives a precise characterization of the class of semirings for which the respective class of recognizable weighted tree languages is closed under inverses of homomorphisms (Theorem 7.1). For this purpose, we will use our core theorem, and a result form [KM18].

Lastly, in Section 8, we investigate the connection between weighted alternating automata and recently introduced polynomial automata [BTSW17] to prove the decidability of the ZERONESS and EQUALITY problems for weighted alternating automata if weights are taken from the rational numbers (Corollary 8.2).

2. Preliminaries

Let $\mathbb{N} = \{0, 1, 2, \ldots\}$ denote the set of non-negative integers. For sets M, N we denote the cardinality of M by $|M|$, the set of subsets of M by $\mathcal{P}(M)$, the Cartesian product of M and N by $M \times N$, and the set of mappings from M to N by $N^M = \{f \mid f : M \to N\}$. If M is finite and non-empty, it is also called an *alphabet*.

For the remainder of this paper, let Σ, Γ and Λ denote alphabets. The set of all (finite) words over Σ is denoted by Σ^*. Let $|w|$ denote the length of a word w and $\Sigma^k = \{w \in \Sigma^* \mid |w| = k\}$. The unique word in Σ^0 is called the *empty word* and denoted by ε. The concatenation of words u, v is denoted by $u \cdot v$ or just uv. A mapping $h : \Lambda^* \to \Sigma^*$ is called a *homomorphism* if $h(u \cdot v) = h(u) \cdot h(v)$ and h is *non-deleting* if $h(a) \neq \varepsilon$ for all $a \in \Lambda$.

A *monoid* is an algebraic structure $(M, \cdot, 1)$, where $\cdot$ is a binary associative internal operation and $m \cdot 1 = m = 1 \cdot m$ for all $m \in M$. A monoid is *commutative* if $\cdot$ is commutative.

A *semiring* is an algebraic structure $(S, +, \cdot, 0, 1)$, where $(S, +, 0)$ is a commutative monoid, $(S, \cdot, 1)$ is a monoid, $s \cdot 0 = 0 = 0 \cdot s$ for all $s \in S$, and $s_3 \cdot (s_1 + s_2) = s_3 \cdot s_1 + s_3 \cdot s_2$ and $(s_1 + s_2) \cdot s_3 = s_1 \cdot s_3 + s_2 \cdot s_3$ for all $s_1, s_2, s_3 \in S$. A semiring is *commutative* if $(S, \cdot, 1)$ is commutative.

For the remainder of this paper, let S denote a commutative semiring.

For any set M, we denote S^M by $S\langle\!\langle M\rangle\!\rangle$. Furthermore, for $L \subseteq M$ we define the *characteristic function* $\mathbb{1}_L \in S\langle\!\langle M\rangle\!\rangle$ by $\mathbb{1}_L(w) = 1$ if $w \in L$ and $\mathbb{1}_L(w) = 0$ otherwise for all $w \in M$. An element $s \in S\langle\!\langle \Sigma^* \rangle\!\rangle$ is called a *S-weighted Σ-language* (for short: weighted language).

Let X_n always denote a linearly ordered set with $|X_n| = n \in \mathbb{N}$, we refer to the i-th element of X_n by x_i. Let $S[X_n]$ denote the *semiring of polynomials* with coefficients in S and commuting indeterminates $x_1, \ldots, x_n$. We say $m \in S[X_n]$ is a *monomial* if $m = s \cdot x_1^{k_1} \cdot \ldots \cdot x_n^{k_n}$ for some $s \in S$ and $k_1, \ldots, k_n \in \mathbb{N}$. The *degree* of m is $\sum_{i=1}^n k_i$. A monomial m is a *proper monomial* if it has a non-zero degree. Each polynomial that can be written as a sum of proper monomials is called a *polynomial without constants* and the set of all polynomials without constants is denoted by $S[X_n]_{\text{const}=0}$. For $p, p_1, \ldots, p_n \in S[X_n]$ let $p\langle p_1, \ldots, p_n\rangle$ denote the polynomial gained from the simultaneous substitution of x_i by p_i in p for all $1 \le i \le n$.

Next, we give a concise collection of definitions from the topic of ranked trees that are needed in this paper. For a more detailed introduction to trees and tree automata, we recommend [CDG$^+$08].

A *ranked alphabet* is an ordered pair (Γ, Rank), where $\text{Rank} : \Gamma \to \mathbb{N}$ is a mapping. Without loss of generality, we assume $X_n \cap \Gamma = \varnothing$. Moreover, let $\Gamma^{(r)} = \{\gamma \in \Gamma \mid \text{Rank}(\gamma) = r\}$ and $\text{Rank}(\Gamma) = \max\{\text{Rank}(\gamma) \mid \gamma \in \Gamma\}$.

The set of Γ-*terms over* X_n is the smallest set $T_\Gamma[X_n]$ such that $\Gamma^{(0)} \cup X_n \subseteq T_\Gamma[X_n]$; and $g(t_1, \ldots, t_{\text{Rank}(g)}) \in T_\Gamma[X_n]$ for all $g \in \Gamma$ and all $t_1, \ldots, t_{\text{Rank}(g)} \in T_\Gamma[X_n]$. We denote $T_\Gamma(X_0) = T_\Gamma(\varnothing)$ by T_Γ. We extend Rank by putting $\text{Rank}(x_i) = 0$ for all $i \in \mathbb{N}$. If Rank is clear from the context, we just write $g(t_1, \ldots, t_k)$. Moreover, we identify g and $g()$ for $g \in \Gamma^{(0)} \cup X_n$. Hence, all terms $t \in T_\Gamma[X_n]$ are of the form $t = g(t_1, \ldots, t_k)$ for some $g \in \Gamma \cup X_n$ and $t_1, \ldots, t_k \in T_\Gamma[X_n]$.

We define $\text{Pos} : T_\Gamma[X_n] \to \mathcal{P}(\mathbb{N}^*) : g(t_1, \ldots, t_k) \mapsto \{\varepsilon\} \cup \bigcup_{i=1}^k \{i\} \cdot \text{Pos}(t_i)$. Let $t = g(t_1, \ldots, t_k)$. The mapping $\text{Label}_t : \text{Pos}(t) \to \Gamma \cup X_n$ is defined by $\text{Label}_t(\varepsilon) = g$; and $\text{Label}_t(w) = \text{Label}_{t_i}(v)$ if $w = iv \in \text{Pos}(t)$. We will identify t and the mapping Label_t: We write $t(w)$ to denote $\text{Label}_t(w)$ and refer to terms as trees. Consequently, we have $t^{-1}(g) = \{w \in \text{Pos}(t) \mid \text{Label}_t(w) = g\}$ for all $g \in T_\Gamma[X_n]$. This coincides with the definition of a tree as a directed graph in the following way: The set of vertices is $\text{Pos}(t)$, the root is ε, and for $u, v \in \text{Pos}(t)$ we have a (u, v)-edge iff $v = ui$ for some $i \in \mathbb{N}$.

For $t = g(t_1, \ldots, t_k), t' \in T_\Gamma[X_n]$, and $w \in \text{Pos}(t)$, the *subtree of t at w*, denoted by $t|_w$ and the *substitution of t' in t at w*, denoted by $t\langle w \leftarrow t'\rangle$ are defined by $t|_\varepsilon = t$ and $t\langle \varepsilon \leftarrow t'\rangle = t'$ if $w = \varepsilon$; and $t|_w = t_i|_v$ and $t\langle w \leftarrow t'\rangle = g(t_1, \ldots, t_{i-1}, t_i\langle v \leftarrow t'\rangle, t_{i+1}, \ldots, t_k)$ for $w = iv \in \text{Pos}(t)$. Moreover, let $M \subseteq \text{Pos}(t)$ and $|M| = l$. We define $t\langle M \leftarrow (t'_1, \ldots, t'_l)\rangle = t\langle m_l \leftarrow t'_l\rangle \cdots \langle m_1 \leftarrow t'_1\rangle$, where m_i is the i-th element of M with regard to the lexicographical order on $\mathbb{N}^*$. In case $t'_1 = \ldots = t'_l = t'$, we abbreviate $t\langle M \leftarrow (t'_1, \ldots, t'_l)\rangle$ by $t\langle M \leftarrow t'\rangle$. If $M = t^{-1}(x_i)$, we write $t\langle x_i \leftarrow (t'_1, \ldots, t'_l)\rangle$ to denote $t\langle M \leftarrow (t'_1, \ldots, t'_l)\rangle$. Finally, let $t\langle t'_1, \ldots, t'_n\rangle = t\langle t^{-1}(x_1) \leftarrow t'_1\rangle \cdots \langle t^{-1}(x_n) \leftarrow t'_n\rangle$ denote the simultaneous substitution in trees.

We say a tree t is *non-deleting in l variables* if it contains at least one symbol from Γ and each of the variables $x_1, \ldots, x_l$ occurs at least once in t. We say t is *linear in l variables* if it is non-deleting in l variables and each of the variables occurs at most once in t. Moreover, let $\text{r}(t) = \sum_{i=1}^n |t^{-1}(x_i)|$ denote the number of nodes of t labeled by variables. For any $r \in \mathbb{N}$

let $T_\Gamma^{(r)}(X_n) = \{t \in T_\Gamma(X_n) \mid \mathrm{r}(t) = r\}$ be the set of Γ-trees with exactly r positions labeled by variables.

A *tree homomorphism* $h : T_\Gamma \to T_\Lambda$ is a mapping such that for all $g \in \Gamma^{(r)}$ there exists $t_g \in T_\Lambda[X_r]$ with $h(g(t_1, \ldots, t_r)) = t_g\langle h(t_1), \ldots, h(t_r)\rangle$ for all $t_1, \ldots, t_r \in T_\Gamma$. We will denote t_g by $h(g)$, even though t_g is not necessarily in T_Λ. A tree homomorphism is *non-deleting* (resp. *linear*) if each $h(g)$ is non-deleting (resp. linear) in $\mathrm{Rank}(g)$ variables. For more information on tree homomorphisms consider Paragraph 1.4 in [CDG$^+$08].

3. Weighted alternating finite automata

This section introduces weighted alternating finite automata (WAFA) and shows how to achieve desirable normal forms of WAFA (Lemma 3.1, Lemma 3.2). We will follow the definitions of [KM18].

A *weighted alternating finite automaton* (WAFA) is a 5-tuple $\mathcal{A} = (Q, \Sigma, \delta, P_0, \tau)$, where $Q = \{q_1, \ldots, q_n\}$ is a finite set of states, Σ is an alphabet, $\delta : Q \times \Sigma \to S[Q]$ is a transition function, $P_0 \in S[Q]$ an initial polynomial, and $\tau : Q \to S$ a final weight function.

Let $\mathcal{A} = (Q, \Sigma, \delta, P_0, \tau)$ be a WAFA. Its *state behavior* $[\mathcal{A}] : Q \times \Sigma^* \to S$ is the mapping defined by

$$[A](q, w) = \begin{cases} \tau(q) & \text{if } w = \varepsilon, \\ \delta(q, a)\langle [\mathcal{A}](q_1, v), \ldots, [\mathcal{A}](q_n, v)\rangle & \text{if } w = av \text{ for } a \in \Sigma \end{cases}.$$

Usually, we will write $[\mathcal{A}]_q(w)$ instead of $[\mathcal{A}](q, w)$. Now, the *behavior* of $\mathcal{A}$ is the weighted language $[\![\mathcal{A}]\!] : \Sigma^* \to S$ defined by

$$[\![A]\!](w) = P_0\langle [\mathcal{A}]_{q_1}(w), \ldots, [\mathcal{A}]_{q_n}(w)\rangle.$$

A weighted language s is *recognized* by $\mathcal{A}$ if and only if $[\![\mathcal{A}]\!] = s$. Two WAFA are said to be *equivalent* if they recognize the same weighted language.

Within the scope of this paper, we define a weighted finite automaton as follows: A *weighted finite automaton* (WFA) is a WAFA $\mathcal{A} = (\{q_1, \ldots, q_n\}, \Sigma, \delta, P_0, \tau)$ where $P_0 = \sum_{j=1}^n s_j \cdot q_j$ and $\delta(q_i, a) = \sum_{j=1}^n s_{ij}^a \cdot q_j$ for all $1 \le i \le n$, $a \in \Sigma$. This definition coincides with the well known definition cf. [DG09],[DG07]. For example $P_0 = q_1 + 3 \cdot q_4$ corresponds to initial weight 1 in q_1, initial weight 3 in q_4, and initial weight 0 in all other states; or $\delta(q_2, a) = 3 \cdot q_2 + 4 \cdot q_3$ corresponds to a situation where state q_2 has an a-loop with weight 3 and an a-transition to state q_3 with weight 4 (and vice versa). Lastly, $\tau(q_2) = 2$ means state q_2 has final weight 2.

Let $\mathcal{A}$ be a WAFA with weights taken from S. We define $M_{(q,a)}$ as the *set of monomials that appear in* $\delta(q, a)$ and M_0 as the *set of monomials that appear in* P_0. An element $s \in S$ is called a *coefficient in* $\mathcal{A}$ if s is the coefficient of a monomial in M_0, or the coefficient of a monomial in $M_{(q,a)}$ for some $q \in Q, a \in \Sigma$. Similarly, s is called a *constant in* $\mathcal{A}$ if $P_0(0, \ldots, 0) = s$, or $\delta(q, a)(0, \ldots, 0) = s$ for some $q \in Q, a \in \Sigma$.

We say $\mathcal{A}$ is *nice* if it has the following properties:

(i) $\delta(q, a)$ is a finite sum of pairwise distinct monomials of the form $s \cdot q_1^{k_1} \cdot \ldots \cdot q_n^{k_n}$ for all $q \in Q, a \in \Sigma$,

(ii) all monomials in P_0 and δ are proper,

(iii) $P_0 = q_1$.

Moreover, we say that $\mathcal{A}$ is *purely polynomial* if:

(iv) all monomials (in P_0 and δ) have coefficient 1.

We want to show that we can always assume a WAFA to be nice and purely polynomial.

Lemma 3.1. *For each WAFA $\mathcal{A}$ there exists an equivalent WAFA $\mathcal{A}'$ such that (i)-(iv) hold for $\mathcal{A}'$. The construction of $\mathcal{A}'$ is effective.*

Proof. Let $\mathcal{A} = (Q, \Sigma, \delta, P_0, \tau)$ be a WAFA.

(i) Since $\cdot$ is distributive and commutative in $S[Q]$ there exists an equivalent WAFA $\mathcal{A}'$ such that (i) holds.

(ii) Assume (i) holds for $\mathcal{A} = (Q, \Sigma, \delta, P_0, \tau)$. We define a WAFA $\mathcal{A}' = (Q', \Sigma, \delta', P_0', \tau')$ which includes a new state q_c for each constant c in $\mathcal{A}$. Furthermore, δ' and P_0' are as δ and P_0, respectively, but each occurrence of each constant c is replaced by q_c. Moreover, $\delta'(q_c, a) = q_c$ for all $a \in \Sigma$ and $\tau'(q_c) = c$. There is a finite number of constants in $\mathcal{A}$. Thus, $\mathcal{A}'$ is a WAFA. It is easy to see that $\mathcal{A}$ and $\mathcal{A}'$ are equivalent and that (i)-(ii) hold for $\mathcal{A}'$.

(iii) Assume (i)-(ii) hold for $\mathcal{A}$. Due to Lemma 6.3 of [KM18], there exists an equivalent WAFA $\mathcal{A}'$ such that (i)-(iii) hold for $\mathcal{A}'$. The construction of $\mathcal{A}'$ is straightforward: we add a new state q to $\mathcal{A}$ with $\delta(q, a) = P_0\langle\delta(q_1, a), \ldots, \delta(q_n, a)\rangle$ and $\tau(q) = P_0\langle t(q_1), \ldots, \tau(q_n)\rangle$. We rename the states of $\mathcal{A}$ such that q becomes the new q_1. Lastly, we set $P_0 = q_1$.

(iv) Assume that (i)-(iii) hold for $\mathcal{A}$. We define

$$Q' = Q \cup \{q_s \mid s \text{ is a coefficient in } \mathcal{A}\}.$$

We may assume that the two sets forming this union are disjoint. Furthermore, let δ' be defined by

$$\delta'(q, a) = \begin{cases} \displaystyle\sum_{s \cdot q_1{}^{k_1} \ldots q_n{}^{k_n} \in M_{(q,a)}} q_1{}^{k_1} \ldots q_n{}^{k_n} \cdot q_s & \text{if } q \in Q \text{ and} \\ q & \text{otherwise} \end{cases}$$

for all $q \in Q'$, $a \in \Sigma$. Moreover, let τ' be defined by

$$\tau'(q) = \begin{cases} \tau(q) & \text{if } q \in Q \text{ and} \\ s & \text{if } q = q_s \end{cases}$$

for all $q \in Q'$. Consider the WAFA $\mathcal{A}' = (Q', \Sigma, P_0, \delta', \tau')$. It is easy to see that $[\![\mathcal{A}]\!] = [\![\mathcal{A}']\!]$ and that every monomial in $\mathcal{A}'$ has 1 as coefficient.

If the appropriate order on Q' is chosen, properties (i)-(iii) hold for $\mathcal{A}'$, too. $\qquad\square$

We would like to point out: while the construction described in the proof of Property (iv) also works for WFA the resulting automaton does not have to be a WFA.

In [KM18] the transition function and the initial polynomial are not allowed to contain constants. This corresponds to the property that runs are not allowed to terminate before the entire word is read. Since it will help with several constructions, we allowed constants in our definition. Nevertheless, as Lemma 3.1 (ii) shows, the introduction of constants does not increase expressiveness since it is possible to simulate terminating transitions by "deadlock"-states.

We say a WAFA $(Q, \Sigma, \delta, P_0, \tau)$ is *equalized* if all monomials occurring in δ have the same degree.

Lemma 3.2. *For each* WAFA $\mathcal{A}$ *there exists an equivalent, equalized and nice* WAFA $\mathcal{A}'$. *The construction of* $\mathcal{A}'$ *is effective.*

Proof. Let $\mathcal{A} = (Q, \Sigma, \delta, q_1, \tau)$ be a nice WAFA and d the maximum degree of monomials occurring in δ. Let q_{n+1} be a new state and

$$\delta'(q, a) = \begin{cases} \displaystyle\sum_{sq_1^{k_1}\ldots q_n^{k_n} \in M_{(q,a)}} s \cdot q_1{}^{k_1} \ldots q_n{}^{k_n} \cdot q_{n+1}^{d - \sum_{u=1}^{n} k_u} & \text{if } q \in Q \text{ and} \\ q_{n+1}^d & \text{otherwise} \end{cases}$$

for all $q \in Q'$, $a \in \Sigma$. As well as,

$$\tau'(q) = \begin{cases} \tau(q) & \text{if } q \in Q \text{ and} \\ 1 & \text{if } q = q_{n+1} \end{cases}$$

for all $q \in Q'$. Clearly, $\mathcal{A}' = (Q \cup \{q_{n+1}\}, \Sigma, \delta', q_1, \tau')$ is equalized. Also, $\mathcal{A}$ and $\mathcal{A}'$ are equivalent and $\mathcal{A}'$ is nice.

Due to the form of the constructed polynomials and since the initial polynomial remains unchanged, we may assume that $\mathcal{A}'$ is nice. Moreover, we didn't change the coefficients, thus $\mathcal{A}'$ is purely polynomial, if $\mathcal{A}$ was purely polynomial. $\square$

Nice WAFA can be represented in the following way: As usual we depict each state by a circle. Then, each monomial $s \cdot q_1^{k_1} \ldots q_n^{k_n}$ in $\delta(q_i, a)$ is represented by a multi-arrow which is labeled by $a : s$, begins in q_i, and has k_j heads in q_j for all $1 \leq j \leq n$, respectively. In case a multi-arrow has more than one head, we join these heads by a $\bullet$. If $s = 1$, we omit the s-label. If $s = 0$, we omit the complete multi-arrow. The initial polynomial is represented analogously. The final weights are represented as usual. Note that the multi-arrows can be viewed as a parallel or simultaneous transitions and that this representation coincides with the usual representation if the automaton is a WFA. Consider the following example:

Example 3.3. Let $S = (\mathbb{N}, +, \cdot, 0, 1)$, $\Sigma = \{a, b\}$, and s the weighted language

$$\begin{aligned} s: \quad \Sigma^* \quad &\to \quad S: \\ w \quad &\mapsto \quad \begin{cases} (2^j)^{2^i} & \text{if } w = a^i b^j \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$
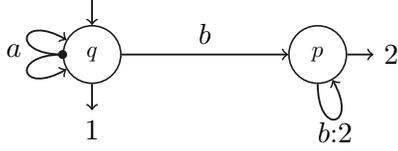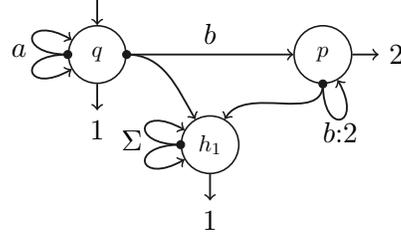
We consider the WAFA $\mathcal{A} = (\{q, p\}, \Sigma, P_0, \delta, \tau)$, defined by:

$$\begin{array}{llll} P_0 &= q & \delta(q, a) = q^2 & \delta(q, b) = p \\ \tau(q) = 1 & \tau(p) = 2 & \delta(p, a) = 0 & \delta(p, b) = 2 \cdot p \end{array}$$

A depiction of this automaton can be seen in Figure 1. One can check that $[\![\mathcal{A}]\!] = s$, for example:

$$\begin{aligned} [\![\mathcal{A}]\!](aabb) &= q\big\langle [\![\mathcal{A}]\!]_q(aabb), [\![\mathcal{A}]\!]_p(aabb) \big\rangle \\ &= [\![\mathcal{A}]\!]_q(aabb) = q^2 \big\langle [\![\mathcal{A}]\!]_q(aabb), [\![\mathcal{A}]\!]_p(aabb) \big\rangle \\ &= \big([\![\mathcal{A}]\!]_q(abb)\big)^2 = \big([\![\mathcal{A}]\!]_q(bb)\big)^{2\cdot2} = \big([\![\mathcal{A}]\!]_p(b)\big)^{2\cdot2} \\ &= \big(2 \cdot [\![\mathcal{A}]\!]_p(\varepsilon)\big)^{2\cdot2} = \big(2 \cdot \tau(p)\big)^{2\cdot2} = \big(2^2\big)^{2^2} \end{aligned}$$

Two-mode alternating automata are a subclass of alternating automata. In the weighted setting two-mode alternating automata are defined as follows (cf. [KM18]): each state q is either existential ($\delta(q, a) = \sum_{i=1}^{n} s_i^a \cdot q_i$ for all $a \in \Sigma$) or universal ($\delta(q, a) = s^a \cdot q_1^{k_1} \cdot \ldots \cdot q_n^{k_n}$

Figure 1: Representation of $\mathcal{A}$ Figure 2: Representation of equalized, nice $\mathcal{A}$

for all $a \in \Sigma$). In [KM18] run semantics were defined for two-mode alternating automata. However, they can be defined for alternating automata with mixed states, in an analogue fashion. Here, we give the definition of runs for nice WAFA. For one, as seen above, we can transform every WAFA into a nice one. For another, it is only a technicality to define runs for arbitrary WAFA, based on the definition below.

The idea of reading monomials as multi-transitions was already introduced above. Having this in mind, a run tree is a tree labeled by states such that: (i) Our run begins in the initial state, (ii) if a vertex at depth k is q labeled, then the labels of its children fit the states of an a_k labeled multi-transition beginning in state q, and (iii) our run halts after n steps.

More formally, if $\mathcal{A} = (Q, \Sigma, \delta, q_1, \tau)$ is a nice WAFA, we define the ranked alphabet $\Gamma_{\mathcal{A}} = \{(q, x) \in Q \times S[Q] \mid x = \tau(q) \text{ or } x \in M_{(q,a)} \text{ for some } a \in \Sigma\}$ where $\mathrm{Rank}(q, x)$ is the rank of x as a polynomial. A run over $w = a_1 \ldots a_n$ in $\mathcal{A}$ is a $\Gamma_{\mathcal{A}}$-tree $t_{\mathcal{R}}$ with the following properties:

(i) If $|p| = 0$, then $t_{\mathcal{R}}(p) = (q_1, x)$, for some x.
(ii) If $|p| < n$ and $t_{\mathcal{R}}(p) = (q, s \cdot q_1^{k_1} \cdot \ldots \cdot q_n^{k_n})$, then $s \cdot q_1^{k_1} \cdot \ldots \cdot q_n^{k_n} \in M_{(q, a_{|p|+1})}$ and for all $\sum_{u=1}^{l-1} k_u \leq i < \sum_{u=1}^{l} k_u$ we have $t_{\mathcal{R}}(pi) = (q_l, x)$ for some x.
(iii) If $|p| = n$, then $\mathrm{Rank}(t_{\mathcal{R}}(p)) = 0$.

The weight of a run tree $t_{\mathcal{R}}$ is defined by

$$\mathrm{Weight}(t_{\mathcal{R}}) = \prod_{(q, s \cdot q_1^{k_1} \cdot \ldots \cdot q_n^{k_n}) \in t_{\mathcal{R}}(\mathrm{Pos}(t_{\mathcal{R}}))} s \ .$$

Note that the final weights are accounted for, since they are the labels of the leaves of our run tree.

And as usual we have (Theorem 5.15 & Theorem 5.17 in [KM18]):

$$[\![\mathcal{A}]\!](w) = \sum_{t_{\mathcal{R}} \text{ run over } w} \mathrm{Weight}(t_{\mathcal{R}}) \ .$$

Example 3.4. Later we will use the connection between WAFA and trees heavily. For a better understanding of this connection we want to point out the following:

First off, let us observe that $|M_{(q,a)}| = 1$ if $\delta(q, a)$ is a monomial. Consequently, if $\delta(q, a)$ is a monomial for all $q \in Q, a \in \Sigma$, then every word has a unique run. We call a WAFA with this property *universal*. A two-mode WAFA where every state is universal is universal and vice versa. Universality for WAFA is as determinism for finite automata. However, in general universal WAFA and WFA are incomparable it terms of expressive power (cf. Corollary 3 in [AK11]).
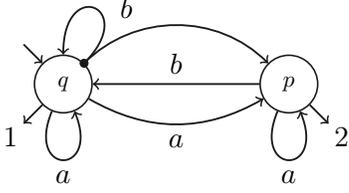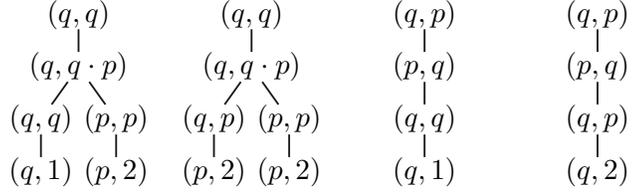
Figure 3: A non-universal WAFA   Figure 4: Runs of WAFA from Figure 3 on aba

Second, we consider the non-universal automaton from Figure 3. This automaton is not universal since $\delta(q, a) = q + p$. As a consequence, we may have several runs, whenever the letter a is to be processed in state q we can choose one of two possible children in our tree. Figure 4 shows these run-trees for the input aba. This connection becomes more clear, if we observe the behavior of the automaton on aba:

$$q \xrightarrow{a} q + p \xrightarrow{b} (q \cdot p) + (q) \xrightarrow{a} ((q + p) \cdot p) + (q + p) = q \cdot p + p \cdot p + q + p$$

Clearly, every monomial in the final polynomial corresponds to one of the run-trees. Moreover, the weight of a run-tree is the product of the coefficients in its nodes and its leaves correspond to the final weights of their state. Thus, we get that the sum of all run-weights is the behavior of the automaton.

Due to the bound on the growth of series recognized by WFA, we can see that s from Example 3.3 is not recognizable by a WFA. Thus, WAFA are more expressive than WFA when weights are taken from the non-negative integers. However, this is not the case for every semiring. A semiring S is *locally finite* if for every finite $X \subseteq S$ the generated subsemiring $\langle X \rangle$ is finite. The following result characterizes semirings on which WAFA and WFA are equally expressive:

Theorem 3.5 [KM18, Theorem 7.1]**.** *The class of S-weighted Σ-languages recognizable by* WAFA *and the class of S-weighted Σ-languages recognizable by* WFA *are equal if and only if S is locally finite.*

4. A CHARACTERIZATION OF WAFA VIA WEIGHTED FINITE TREE AUTOMATA

Our central result Theorem 4.5 is included in this section, as well as the definition of weighted finite tree automata.

The connection between alternating automata and trees was utilized before: Already in the non-weighted settings trees were used to define runs of alternating automata. As seen above, this is possible for WAFA too. We want to strengthen this connection by the use of tree automata and tree homomorphisms. In order to do so, we need some additional definitions.

An element $r \in S\langle\!\langle T_\Gamma \rangle\!\rangle$ is called a *(S-weighted) tree language*. A *weighted finite tree automaton* (WFTA) is a 4-tuple $\mathcal{A} = (Q, \Gamma, \delta, \lambda)$, where $Q = \{q_1, \ldots, q_n\}$ is a finite set of states, Γ is a ranked alphabet, $\delta = (\delta_k \mid 1 \leq k \leq \mathrm{Rank}(\Gamma))$ is a family of transition functions $\delta_k : \Gamma^{(k)} \to S^{Q^k \times Q}$, and $\lambda : Q \to S$ a root weight function.

If k is clear from the context, we will denote tuples $(p_1, \ldots, p_k)$ by $\bar{p}$. Moreover, since k in $\delta_k(g)$ is clear from g, we will denote $\delta_k(g)$ by δ_g.

Let $\mathcal{A} = (Q, \Gamma, \delta, \lambda)$ be a WFTA. Its *state behavior* $[\mathcal{A}] : Q \times T_\Gamma \to S$ is the mapping defined by

$$[A]\big(q, g(t_1, \ldots, t_k)\big) = \sum_{\bar{p} \in Q^k} \delta_g(\bar{p}, q) \cdot \prod_{i=1}^{k} [\mathcal{A}](p_i, t_i) \ .$$

Usually, we will write $[\mathcal{A}]_q(t)$ instead of $[\mathcal{A}](q, t)$. Now, the *behavior* of $\mathcal{A}$ is the weighted tree language $[\![A]\!] : T_\Gamma \to S$ defined by

$$[\![A]\!](t) = \sum_{i=1}^{n} \lambda(q_i) \cdot [\mathcal{A}]_{q_i}(t) \ .$$

A weighted tree language s is *recognized* by $\mathcal{A}$ if and only if $[\![\mathcal{A}]\!] = s$.

It is well known that a word over Σ can be represented as a 1-ary tree: Each letter of Σ is given rank one and a new end-symbol $\#$ of rank zero is added. Then $w_0 w_1 \ldots w_n$ translates to the tree $w_0(w_1(\ldots w_n(\#)\ldots))$. Here, we want to represent words as full r-ary trees for any arbitrary $r \in \mathbb{N}$. Given an alphabet Σ and $r \geq 1$, we define the ranked alphabet $\Sigma_\#^r = \Sigma \cup \{\#\}$ with $\mathrm{Rank}(\#) = 0$ and $\mathrm{Rank}(a) = r$ for all $a \in \Sigma$. For all $w \in \Sigma^*$ the tree $t_w^r \in T_{\Sigma_\#^r}$ is defined by $t_\varepsilon^r = \#$; and $t_w^r = a(t_v^r, \ldots, t_v^r)$ if $w = av$ with $a \in \Sigma$. We call $h^r : \Sigma^* \to T_{\Sigma_\#^r} : w \mapsto t_w^r$ the *generic tree homomorphism (of rank r)*. The case $r = 1$ is special since for all $t \in T_{\Sigma_\#^1}$ there exists $w \in \Sigma^*$ such that $t = t_w^1$. Therefore, if clear from the context, we will identify Σ and $\Sigma_\#^1$, Σ^* and $T_{\Sigma_\#^1}$, as well as w and t_w^1. It is well known that a weighted Σ language is recognizable by a WFA over Σ if and only if it is recognizable by a WFTA over $\Sigma_\#^1$.

The key observation is that the behavior of a WAFA $\mathcal{A}$ on w can be characterized by the behavior of a WFTA on t_w^r where r is the degree of polynomials in an equalized version of $\mathcal{A}$. Even more, the behavior of a WFTA on $h(w)$ (where h is a tree homomorphism) can be characterized by the behavior of a WAFA on w.

Lemma 4.1. *If $s \in S\langle\!\langle \Sigma^* \rangle\!\rangle$ is recognized by a WAFA, then $s = [\![\mathcal{B}]\!] \circ h^r$ for some WFTA $\mathcal{B}$ and the generic homomorphism h^r for some $r \in \mathbb{N}$.*
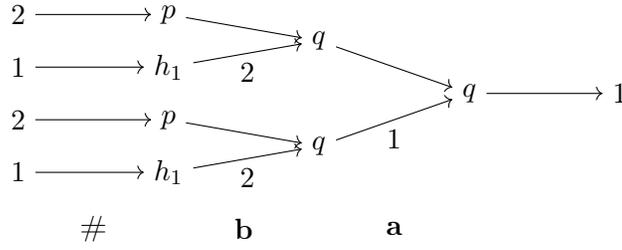
Proof. Assume s is recognized by a WAFA. Due to Lemma 3.1 and Lemma 3.2, we may assume that s is recognized by a nice and equalized WAFA $\mathcal{A} = (Q, \Sigma, \alpha, P_0, \tau)$. Let r be the unique degree of monomials in $\mathcal{A}$. Our goal is to define a WFTA $\mathcal{B}$ with $[\![\mathcal{A}]\!] = [\![\mathcal{B}]\!] \circ h^r$. To this purpose, we observe that the equalized $\mathcal{A}$ can be viewed as a WFTA over a ranked alphabet where each letter has rank r and where each multi-arrow in the representation of $\mathcal{A}$ corresponds to one transition in $\mathcal{B}$. Formally, we define the WFTA $\mathcal{B} = (Q, \Sigma_\#^r, \beta, \lambda)$ with:

$$
\begin{aligned}
\lambda &= \mathbb{1}_{\{q_1\}} \\
\beta_\#(\varepsilon, q) &= \tau(q) \ , \\
\beta_a(\bar{p}, q) &= \begin{cases} s & \text{if } p_1 \leq \ldots \leq p_r \text{ according to the order on } Q \text{ and} \\ & \quad s \cdot p_1 \cdot \ldots \cdot p_r \in M_{(q,a)} \ , \text{ and} \\ 0 & \text{otherwise} \ , \end{cases}
\end{aligned}
$$

for all $q \in Q$ and $\bar{p} \in Q^r$.

Note that the order of the $p_1, \ldots, p_r$ needs to be fixed, otherwise many runs of the WFTA may correspond to one run of the WAFA.

Now, we will show that $[\mathcal{A}]_q(w) = [\mathcal{B}]_q(t_w^r)$ for all $q \in Q$ and $w \in \Sigma^*$ by induction on $|w| \in \mathbb{N}$. If $|w| = 0$, then $w = \varepsilon$. Thus, $[\mathcal{A}]_q(w) = \tau(q) = \beta_\#(\varepsilon, q) = [\mathcal{B}]_q(t_w^r)$ for all $q \in Q$.

Figure 5: Run of translated WFTA on t_{ab}^2

Assume there exists $l \in \mathbb{N}$ such that the claim holds for all $w' \in \Sigma^l$ and all $q \in Q$. For the induction step let $w = aw' \in \Sigma \cdot \Sigma^l$. For $\bar{p} \in Q^k$ let $k_i(\bar{p})$ denote the number of q_i's in $\bar{p}$. Then, we get

$$
\begin{aligned}
& [\mathcal{A}]_q(w) \\
= \ & \alpha(q,a)\langle [\mathcal{A}]_{q_1}(w'), \ldots, [\mathcal{A}]_{q_n}(w') \rangle \\
\overset{\text{(IH)}}{=} \ & \alpha(q,a)\langle [\mathcal{B}]_{q_1}(t_{w'}^r), \ldots, [\mathcal{B}]_{q_n}(t_{w'}^r) \rangle \\
= \ & \sum_{s \cdot q_1^{k_1} \ldots q_n^{k_n} \in M_{(q,a)}} s \cdot \prod_{i=1}^{n} \left([\mathcal{B}]_{q_i}(t_{w'}^r) \right)^{k_i} \\
= \ & \sum_{s \cdot q_1^{k_1} \ldots q_n^{k_n} \in M_{(q,a)}} \beta_a (\underbrace{\overbrace{q_1, \ldots, q_1}^{}}_{k_1 \text{ times}}, \ldots, \underbrace{q_n, \ldots, q_n}_{k_n \text{ times}}, q) \cdot \prod_{i=1}^{n} \left([\mathcal{B}]_{q_i}(t_{w'}^r) \right)^{k_i} \\
= \ & \sum_{\bar{p} \in Q^r} \beta_a(\bar{p}, q) \cdot \prod_{i=1}^{n} \left([\mathcal{B}]_{q_i}(t_{w'}^r) \right)^{k_i(\bar{p})} \\
= \ & [\mathcal{B}]_q \big(a(\underbrace{t_{w'}^r, \ldots, t_{w'}^r}_{r \text{ times}}) \big) \\
= \ & [\mathcal{B}]_q(t_w^r)
\end{aligned}
$$

with the brace over the middle sum reading $\sum k_i = r$ times.

for all $q \in Q$. Which completes our induction.

Since $\mathcal{A}$ is nice and thus $P_0 = q_1$, we consequently have

$$
[\![\mathcal{A}]\!](w) = [\mathcal{A}]_{q_1}(w) = [\mathcal{B}]_{q_1}(t_w^r) = \sum_{i=1}^{n} \lambda(q_i) \cdot [\mathcal{B}]_{q_i}(t_w^r) = [\![\mathcal{B}]\!](t_w^r)
$$

for all $w \in \Sigma^*$. Since $h^r(w) = t_w^r$, this finishes our proof. $\qquad\square$

The following example illustrates this connection between WAFA and WFTA.

Example 4.2. We consider the automaton $\mathcal{A}$ from Example 3.3. It is easy to construct the corresponding WFTA $\mathcal{B} = (Q, \Gamma, \beta, \lambda)$ from the equalized version $\mathcal{A}'$ (Figure 2). First, we copy the set of states (in order) $Q = \{q, p, h_1\}$. Since the maximum degree of polynomials in $\mathcal{A}$ was 2 we get $\Gamma = \{a^{(2)}, b^{(2)}, \#^{(0)}\}$. The root weight function corresponds to the initial weights. However, $\mathcal{A}'$ is nice and thus $\lambda = \mathbb{1}_{\{q\}}$. The transition weight functions β_a and β_b can be defined using the multi arrows in Figure 2. For example, the b-labeled multi arrow in the middle corresponds to $\beta_b(ph_1, q) = 1$. Finally, the final weights in $\mathcal{A}'$ are captured by $\beta_\#(\varepsilon, q) = 1, \beta_\#(\varepsilon, h_1) = 1$, and $\beta_\#(\varepsilon, p) = 2$. The only non-zero run on t_{ab}^2 can be seen in Figure 5.

Lemma 4.3. *Let $\mathcal{B} = (Q, \Gamma, \delta, \lambda)$ a WFTA and $h : \Sigma^* \to T_\Gamma$ a tree homomorphism. Then $[\![\mathcal{B}]\!] \circ h$ is recognized by a WAFA.*

Proof. Assume $s = [\![\mathcal{B}]\!] \circ h \in S\langle\!\langle \Sigma^* \rangle\!\rangle$, where $\mathcal{B} = (Q, \Gamma, \delta, \lambda)$ is a WFTA with $|Q| = n$ and $h : \Sigma^* \to T_\Gamma$ a tree homomorphism. We want to construct a WAFA $\mathcal{A}$ such that $[\![\mathcal{A}]\!] = s$.

If h is the generic homomorphism, we can define $\delta'(q, a) = \sum_{\bar{p} \in Q^r} \delta_a(\bar{p}, q)$ and use the same proof as in the first direction. However, we want to prove this for arbitrary homomorphisms. To achieve this, we give some additional definitions.

Under h, each letter becomes a tree. Nonetheless, we are not interested in the structure of $h(a)$, but want to handle it as if it is a ranked letter. Therefore, we use $h(a)\langle x_1 \leftarrow (p_1, \ldots, p_r)\rangle$ to disambiguate its $r = \mathrm{r}(h(a))$ variables. Furthermore, we extend the family of transition functions $(\delta_k)_{0 \leq k \leq \mathrm{Rank}\,\Gamma}$ into a family $(\delta'_k)_{k \in \mathbb{N}}$ with $\delta'_k : T_\Gamma^{(k)}(\{x_1\}) \to S^{Q^k \times Q}$. We use the same notations for δ' as for δ and define δ'_k recursively as follows.

(1)   For all $g \in \Gamma^{(0)}$ let $\delta'_g(\varepsilon, q) = \delta_g(\varepsilon, q)$ for all $q \in Q$,
(2) $\delta'_{x_1}(p, q) = \mathbb{1}_{\{q\}}(p)$ for all $(p, q) \in Q \times Q$, and
(3)   if $t = g(t_1, \ldots, t_k)$ for some $g \in \Gamma^{(k)}$, we define

$$\delta'_t(\bar{p}_1, \ldots, \bar{p}_k, q) = \sum_{\bar{p}' \in Q^k} \delta_g(\bar{p}', q) \cdot \prod_{i=1}^{k} \delta'_{t_i}(\bar{p}_i, p'_i)$$

for all $(\bar{p}_1, \ldots, \bar{p}_k, q) \in Q^{\mathrm{r}(t_1)} \times \ldots \times Q^{\mathrm{r}(t_k)} \times Q$.

Please note, for $t \in T_\Gamma$ we have $\delta'_t(\varepsilon, q) = [\mathcal{A}]_q(t)$ by definition.

Now, we are well-equipped to define the WAFA $\mathcal{A} = (Q, \Sigma, \alpha, P_0, \tau)$. Let $P_0 = \sum_{i=1}^{n} \lambda(q_i) \cdot q_i$, $\tau(q_i) = \delta'_{h(\#)}(\varepsilon, q_i)$ for all $1 \leq i \leq n$, and
$\alpha(q, a) = \sum_{\bar{p} \in Q^{\mathrm{r}(h(a))}} \delta'_{h(a)}(\bar{p}, q) \cdot \prod_{i=1}^{\mathrm{r}(h(a))} p_i$ for all $q \in Q$, $a \in \Sigma$.

Next, we show $[\mathcal{B}]_q(h(w)) = [\mathcal{A}]_q(w)$ for all $q \in Q$, $w \in \Sigma^*$. Again, by induction on $|w| \in \mathbb{N}$. Before we can do so, we have to prove the following auxiliary claim about δ'.

Claim 4.4. Let $\mathcal{A} = (Q, \Gamma, \delta, \lambda)$ be a WFTA, $t \in T_\Gamma$, $\hat{t} \in T_{\Gamma \cup \{x_1\}}$, and $t_1, \ldots, t_k \in T_\Gamma$ such that $t = \hat{t}\langle x_1 \leftarrow (t_1, \ldots, t_k)\rangle$. Then

$$[\mathcal{A}]_q(t) = \sum_{\bar{p} \in Q^k} \delta'_{\hat{t}}(\bar{p}, q) \cdot \prod_{i=1}^{k} [\mathcal{A}]_{p_i}(t_i)$$

for all $q \in Q$.

Proof. We prove this claim via induction on the depth of $\hat{t}$. The claim is clear if $k = 0$. Therefore, we assume $k > 0$. If $\hat{t} = x_1$, we know $t = t_1$ and get

$$\begin{aligned}
&[\mathcal{A}]_q(t) \\
=\ &\textstyle\sum_{p \in Q} \mathbb{1}_{\{q\}}(p) \cdot [\mathcal{A}]_p(t_1) \\
=\ &\textstyle\sum_{p \in Q} \delta'_{x_1}(p, q) \cdot [\mathcal{A}]_p(t_1)\ .
\end{aligned}$$

Now, assume the claim holds for all $\hat{t} \in T_\Gamma$ with depth $\leq n$ for some $n \in \mathbb{N}$. Assume $\hat{t}$ has depth $n+1$ and $\hat{t} = g(\hat{t}_1, \ldots, \hat{t}_r)$ for some $g \in \Gamma^{(r)}$ and $\hat{t}_1, \ldots, \hat{t}_r \in T_{\Gamma \cup \{x_1\}}$. Of the k trees only the first $\mathrm{r}(\hat{t}_1)$ will be substituted in $\hat{t}_1$, only the second $\mathrm{r}(\hat{t}_2)$ will be substituted in $\hat{t}_1$, and so on. To mark this, let $k(i) = \sum_{l=1}^{i} \mathrm{r}(\hat{t}_l)$, $\bar{t}_i = (t_{k(i-1)+1}, \ldots, t_{k(i)})$, and $\bar{p}_i = (p_{k(i-1)+1}, \ldots, p_{k(i)})$

for all $1 \leq i \leq r$. We get:

$$[\mathcal{A}]_q(t)$$

$$= \sum_{\bar{p}' \in Q^r} \delta_g(\bar{p}', q) \cdot \prod_{i=1}^{r} [\mathcal{A}]_{p'_i}(\hat{t}_i \langle x_1 \leftarrow \bar{t}_i \rangle)$$

$$\stackrel{(IH)}{=} \sum_{\bar{p}' \in Q^r} \delta_g(\bar{p}', q) \cdot \prod_{i=1}^{r} \sum_{\bar{p}_i \in Q^{r(\hat{t}_i)}} \delta'_{\hat{t}_i}(\bar{p}_i, p'_i) \cdot \prod_{j=r(i-1)+1}^{r(i)} [\mathcal{A}]_{p_j}(t_j)$$

$$= \sum_{\bar{p}' \in Q^r} \delta_g(\bar{p}', q) \cdot \sum_{(\bar{p}_1,...,\bar{p}_r) \in Q^k} \prod_{i=1}^{r} \left(\delta'_{\hat{t}_i}(\bar{p}_i, p'_i) \right) \cdot \prod_{j=1}^{k} [\mathcal{A}]_{p_j}(t_j)$$

$$= \sum_{(\bar{p}_1,...,\bar{p}_r) \in Q^k} \sum_{\bar{p}' \in Q^r} \left(\delta_g(\bar{p}', q) \cdot \prod_{i=1}^{r} \left(\delta'_{\hat{t}_i}(\bar{p}_i, p'_i) \right) \cdot \prod_{j=1}^{k} [\mathcal{A}]_{p_j}(t_j) \right)$$

$$= \sum_{(\bar{p}_1,...,\bar{p}_r) \in Q^k} \sum_{\bar{p}' \in Q^r} \left(\delta_g(\bar{p}', q) \cdot \prod_{i=1}^{r} \left(\delta'_{\hat{t}_i}(\bar{p}_i, p'_i) \right) \right) \cdot \prod_{j=1}^{k} [\mathcal{A}]_{p_j}(t_j)$$

$$= \sum_{(\bar{p}_1,...,\bar{p}_r) \in Q^k} \delta'_{\hat{t}}(\bar{p}_1, \ldots, \bar{p}_r, q) \cdot \prod_{j=1}^{k} [\mathcal{A}]_{p_j}(t_j)$$

$$= \sum_{\bar{p} \in Q^k} \delta'_{\hat{t}}(\bar{p}, q) \cdot \prod_{j=1}^{k} [\mathcal{A}]_{p_j}(t_j)$$

This finishes the proof of our claim. $\square$

We return to our main proof. If $|w| = 0$, then $w = \varepsilon$. Therefore, $[\mathcal{B}]_q(w) = \tau(q) = \delta'_{h(\#)}(\varepsilon, q) \stackrel{*}{=} [\mathcal{A}]_q(h(w))$. Here $*$ holds due to Claim 4.4 with $k = 0$ and $\hat{t} = t = h(\#)$. Assume there exists some $l \in \mathbb{N}$ such that $[\mathcal{B}]_q(h(w)) = [\mathcal{A}]_q(w)$ for all $q \in Q$, $w \in \Sigma^l$. For the induction step let $w = aw' \in \Sigma \cdot \Sigma^l$ and $k = r(h(a))$. Similar to the first direction, we have

$$[\mathcal{A}]_q(aw')$$

$$= \sum_{\bar{p} \in Q^k} \delta'_{h(a)}(\bar{p}, q) \prod_{i=1}^{k} [\mathcal{A}]_{p_i}(w')$$

$$\stackrel{(IH)}{=} \sum_{\bar{p} \in Q^k} \delta'_{h(a)}(\bar{p}, q) \prod_{i=1}^{k} [\mathcal{B}]_{p_i}(h(w'))$$

$$\stackrel{C.4.4}{=} [\mathcal{B}]_q(h(a)\langle h(w') \rangle)$$

$$= [\mathcal{B}]_q(h(w))$$

for all $q \in Q$. Via induction, this finishes our proof of the second direction.

Finally, for all $w \in \Sigma^*$ we get:

$$[\![\mathcal{B}]\!](h(w)) = \sum_{i=1}^{n} \lambda(q_i) \cdot [\mathcal{B}]_{q_i}(h(w)) = \sum_{i=1}^{n} \lambda(q_i) \cdot [\mathcal{A}]_{q_i}(w)$$

$$= P_0 \langle [\mathcal{B}']_{q_1}(w), \ldots, [\mathcal{A}]_{q_n}(w) \rangle = [\![\mathcal{A}]\!](w)$$

$\square$

This leads us to our main result.

Theorem 4.5. *A weighted language $s \in S\langle\!\langle \Sigma^* \rangle\!\rangle$ is recognized by a WAFA if and only if there exists a ranked alphabet Γ, a tree homomorphism $h : \Sigma^* \to T_\Gamma$, and a WFTA $\mathcal{A} = (Q, \Gamma, \delta, \lambda)$ such that $s = [\![\mathcal{A}]\!] \circ h$.*

Proof. This is an immediate consequence of Lemma 4.1 and Lemma 4.3. □

This result allows us to transfer results from WFTA to WAFA. Moreover, additional observations in the proofs show that one can give a weight preserving, bijective mapping between the runs of $\mathcal{A}$ and $\mathcal{B}$. This allows us to translate results about runs of WFTA into results about runs of WAFA. To see how to do this, we first revisit a well known result for non-weighted WAFA.

Corollary 4.6. *Non-weighted alternating automata and finite automata are equally expressive.*

This result goes back to [CKS81] and deals with non-weighted automata. It is well known that the non-weighted setting corresponds to the weighted setting, if weights are taken from the Boolean semiring. Thus, we may still investigate the non-weighted setting with our methods. Therefore, we can reproduce this result by using the Boolean semiring. In doing so, we will get a better understanding of the translation of WAFA into WFA. In particular, we are able to identify cases, in which this translation is efficient.

Clearly, every WFA is a WAFA, hence we only have to concern ourselves with one direction. First, we observe that $\mathbb{1}_L \circ h = \mathbb{1}_{h^{-1}(L)}$ for any tree language $L \subseteq T_\Lambda$ and tree homomorphism $h : T_\Gamma \to T_\Lambda$. Assume $\mathbb{1}_L$ is recognized by a WAFA. By Lemma 4.1, we get $\mathbb{1}_L = \mathbb{1}_{L'} \circ h = \mathbb{1}_{h^{-1}(L')}$ with $h = h^r$ for some $r \in \mathbb{N}$ and $L' \subseteq T_{\Sigma_\#^r}$ regular. Since regular tree languages are closed under inverses of homomorphisms, we know $L = h^{-1}(L')$ is a regular $\Sigma_\#^1$ tree language. It is also known that WFA and WFTA are equally expressive over $\Sigma_\#^1$, thus $\mathbb{1}_L$ is recognized by a WFA. The authors in [CKS81] also show that the translation of an alternating finite automaton into an equivalent deterministic finite automaton leads to a (worst case) doubly exponential blowup in states. By our proof we get a better understanding of where this blowup comes from. Constructing $\mathcal{B}$ (from the Proof of Lemma 4.1) is linear in states. However, to construct the tree automaton $\mathcal{B}'$ recognizing $h^{-1}(L)$, we get an exponential blowup in states. Next, $\mathcal{B}'$ viewed as a WFA is non-deteministic (even if $\mathcal{B}'$ was bottom up deterministic). Thus, another exponent is needed to determinize $\mathcal{B}'$. Immediately, we see that the translation of an alaternating automaton into a non-deterministic finite automaton is only exponential. Moreover, the first exponent is only needed, if $\mathcal{B}$ is not (bottom up) deterministic. If a nice Boolean WAFA has only one non-zero final weight and for every pair of states p, q we have $M_{(q,a)} \cap M_{(p,a)} = \varnothing$ for all $a \in \Sigma$, then $\mathcal{B}$ is bottom up deterministic. Consequently, the translation of alternating automata with this property into a non-deterministic finite automaton is linear in states.

5. A NIVAT THEOREM FOR WAFA

This section leads to the Nivat-like characterization of WAFA (Theorem 5.7), but first we will prove that weighted languages recognized by WAFA are closed under inverses of homomorphisms (Corollary 5.2), but not under homomorphisms (Lemma 5.3).

Let $s_1 \odot s_2$ denote the *Hadamard product (pointwise product)* of two weighted languages $s_1, s_2 \in S\langle\!\langle \Sigma^* \rangle\!\rangle$. Furthermore, a word homomorphism $h : \Gamma^* \to \Sigma^*$ is called non-deleting if and only if $h(a) \neq \varepsilon$ for all $a \in \Sigma$. Let $r \in S\langle\!\langle \Gamma^* \rangle\!\rangle$. For $h : \Gamma^* \to \Sigma^*$ a non-deleting homomorphism, we define $h(r) \in S\langle\!\langle \Sigma^* \rangle\!\rangle$ by $h(r)(w) = \sum_{v \in h^{-1}(w)} r(v)$ for all $w \in \Sigma^*$. This sum is always finite since h is non-deleting. For $h : \Sigma^* \to \Gamma^*$ we define $h^{-1}(r) \in S\langle\!\langle \Sigma^* \rangle\!\rangle$ by $h^{-1}(r)(w) = r(h(w))$ for all $w \in \Sigma^*$. In the boolean setting, we have $h(\mathbb{1}_L)(w) = 1$ if and only if there exists $v \in \Gamma^*$ with $h(v) = w$ and $v \in L$. Thus, $h(\mathbb{1}_L) = \mathbb{1}_{h(L)}$. Analogously, we

get $h^{-1}(\mathbb{1}_L) = \mathbb{1}_{h^{-1}(L)}$. Hence, $h(r)$ corresponds to the application of a homomorphism, while $h^{-1}(r)$ corresponds to the application of the inverse of a homomorphism in the non-weighted setting.

The original Nivat Theorem [Niv68] characterizes word-to-word transducers. A generalized version for WFA over arbitrary semirings (Theorem 6.3 in [DK21]) can be stated in the following way:

Theorem 5.1 (Nivat-like theorem for WFA [DK21]). *A weighted language $s \in S\langle\!\langle \Sigma^* \rangle\!\rangle$ is recognized by a WFA if and only if there exist an alphabet Γ, a non-deleting homomorphism $h : \Gamma^* \to \Sigma^*$, a regular language $L \subseteq \Gamma^*$, and a WFA $\mathcal{A}_w$ with exactly one state such that:*

$$s = h(\llbracket \mathcal{A}_w \rrbracket \odot \mathbb{1}_L) \ .$$

Please note, $\mathcal{A}_w$ does not depend on any input and is called $\mathcal{A}_w$ since it is responsible for the application of weights. Our goal is to generalize this result to WAFA. This Nivat-like theorem is strongly connected to the closure of weighted languages recognized by WFA under (inverses) of homomorphisms. Thus, we will investigate these properties for WAFA.

5.1. **Closure properties.** A class K of S-weighted languages is said to be *closed under homomorphisms* if $s \in S\langle\!\langle \Gamma^* \rangle\!\rangle \cap K$ and $h : \Gamma^* \to \Sigma^*$ a non-deleting homomorphism implies $h(s) \in S\langle\!\langle \Sigma^* \rangle\!\rangle \cap K$. Moreover, K is *closed under inverses of homomorphisms* if $s' \in S\langle\!\langle \Gamma^* \rangle\!\rangle \cap K$ and $h : \Sigma^* \to \Gamma^*$ a homomorphism implies $h^{-1}(s') \in S\langle\!\langle \Sigma^* \rangle\!\rangle \cap K$. The same notions are used for weighted tree languages.

The class of weighted languages recognized by WFA is closed under (inverses of) homomorphisms (Lemma 6.2 in [DK21]). WAFA are also closed under inverses of homomorphisms. In fact, this is an easy corollary of Lemma 4.1 and Lemma 4.3.

Corollary 5.2. *The class of weighted languages recognized by WAFA is closed under inverses of homomorphisms.*

Proof. Let $h' : \Lambda^* \to \Sigma^*$ be a homomorphism and $s \in S\langle\!\langle \Sigma^* \rangle\!\rangle$ recognized by a WAFA. Due to Lemma 4.1, we get $s = \llbracket \mathcal{B} \rrbracket \circ h^r$ for the generic homomorphism $h^r : \Sigma^* \to \Sigma_\#^r$ and some weighted $\Sigma_\#^r$-WFTA $\mathcal{B}$. Since homomorphisms are closed under composition, $h^r \circ h' : \Lambda^* \to \Sigma_\#^r$ is a tree homomorphism. Thus, due to Lemma 4.3, $h'^{-1}(s) = (\llbracket \mathcal{B} \rrbracket \circ h^r) \circ h' = \llbracket \mathcal{B} \rrbracket \circ (h^r \circ h')$ is recognized by a WAFA. $\square$

However, the same is not true for the closure under homomorphisms.

Lemma 5.3. *The class of weighted languages recognized by WAFA is not closed under homomorphisms.*

Proof. Let $\Sigma = \{a, b, \#\}$, $\mathbb{B}$ the Boolean semiring, and $\mathbb{B}[x]$ the semiring of polynomials in one indeterminate. Consider

$$r_B : \Sigma^* \to \mathbb{B}[x] : w \mapsto \begin{cases} \sum_{k=0}^{j} x^{ki} & \text{if } w = a^i \# b^j \ , \\ 0 & \text{otherwise.} \end{cases}$$

Due to Lemma 8.3 from [KM18], we know r_B is not recognized by a WAFA. Let $\Gamma = \{a, c, d, \#\}$, $h : \Gamma^* \to \Sigma^*$ the non-deleting homomorphism induced by $h(a) = a, h(\#) =$

$\#, h(c) = h(d) = b$, and

$$r_R : \Gamma^* \to \mathbb{B}[x] : w \mapsto \begin{cases} x^{ki} & \text{if } w = a^i \# c^k d^l \\ 0 & \text{otherwise.} \end{cases},$$
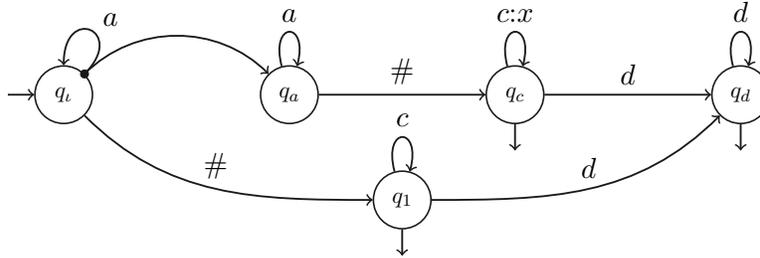
Then

$$h(r_R)(w) = \sum_{v \in h^{-1}(w)} r_R(v)$$

$$= \begin{cases} \sum_{v \in h^{-1}(w)} r_R(v) & \text{if } w = a^i \# b^j \\ 0 & \text{otherwise} \end{cases} = \begin{cases} \sum_{k=0}^{j} r_R(a^i \# c^k d^{j-k}) & \text{if } w = a^i \# b^j \\ 0 & \text{otherwise} \end{cases}$$

$$= \begin{cases} \sum_{k=0}^{j} x^{ki} & \text{if } w = a^i \# b^j \\ 0 & \text{otherwise} \end{cases} = r_B(w)$$

for all $w \in \Sigma^*$. Thus, $h(r_R)$ is not recognized by a WAFA. The weighted language r_R is recognized by the WAFA $\mathcal{A}_R = (\{q_\iota, q_1, q_a, q_c, q_d\}, \{a, \#, c, d\}, \delta_R, q_\iota, \tau_R)$ with δ_R and τ_R defined by:

	q_ι	q_1	q_a	q_c	q_d
$\delta_R(*, a)$	$q_\iota q_a$	0	q_a	0	0
$\delta_R(*, \#)$	q_1	0	q_c	0	0
$\delta_R(*, c)$	0	q_1	0	$x \cdot q_c$	0
$\delta_R(*, d)$	0	q_d	0	q_d	q_d
$\tau_R(*)$	0	1	0	1	1

A depiction of $\mathcal{A}_R$ can be seen below. This completes our proof.

Nonetheless, the proof of the second direction of Theorem 5.1 relies on the closure under homomorphisms. Thus, due to Lemma 5.3, a one to one translation of Theorem 5.1 into the framework of alternating automata is prohibited. Moreover, in the proof of the first direction of Theorem 5.1, L is defined as a language of runs of $\mathcal{A}$. As mentioned above, runs of WAFA are trees. Therefore, we will utilize a Nivat-like theorem for WFTA to prove the corresponding result for WAFA.

5.2. **A Nivat-like characterization of WFTA.** Nivat-like characterizations for weighted tree languages have been investigated in the past. Unranked trees were considered in [DG17], while a very general result for graphs can be found in [DD15]. Here, for the readers convenience, we want to restate a more restricted version for ranked trees.

Let $h : T_\Gamma \to T_\Lambda$ be a non-deleting tree homomorphism, $s \in S\langle\!\langle T_\Lambda \rangle\!\rangle$. In analogy to words, we define $h(s) \in S\langle\!\langle T_\Gamma \rangle\!\rangle$ by $h(s)(t) = \sum_{t' \in h^{-1}(t)} s(t')$ for all $t \in T_\Gamma$. For linear homomorphisms the following is known:

Lemma 5.4 [FV09, Theorem 3.8]. *The class of weighted tree languages recognized by WFTA is closed under linear homomorphisms.*

Based on this, it is easy to prove the following result:

Theorem 5.5 (Nivat-like theorem for WFTA [DG17, Theorem 12]). *A weighted tree language $s \in S\langle\!\langle T_\Gamma \rangle\!\rangle$ is recognized by a WFTA if and only if there exist a ranked alphabet Λ, a linear tree homomorphism $h : T_\Lambda \to T_\Gamma$, a regular tree language $L \subseteq T_\Lambda$, and a WFTA $\mathcal{A}_w$ with exactly one state such that:*

$$s = h(\llbracket \mathcal{A}_w \rrbracket \odot \mathbb{1}_L) \ .$$

Proof. A proof can be found in [DG17]. There, the $\Rightarrow$-direction is proved based on a WFTA $\mathcal{A}$ recognizing s. The components Λ, h, L, and $\mathcal{A}_w$ are chosen to be the set of transitions in $\mathcal{A}$, the mapping assigning each transition $(\bar{p}, \gamma, q)$ to the corresponding letter $\gamma \in \Gamma$, the tree language of runs in $\mathcal{A}$, and an automaton adding weights to the transitions, respectively. This yields the desired equation. Here, for the convenience of the reader, we want to give a precise construction, as well as a proof of correctness for the second direction of the proof of Theorem 5.5.

Let s be recognized by a WFTA $\mathcal{A} = (Q, \Gamma, \alpha, \lambda)$. First, we define Λ. Let $P = Q \cup Q_{\text{fin}}$ where Q_{fin} is a disjoint copy of Q including an element q_{fin} for each $q \in Q$. Now, $\Lambda = \bigcup_{i=0}^{\text{Rank}(\Gamma)} Q^i \times \Gamma^{(i)} \times P$. For reasons of readability, we sometimes abbreviate a letter $((q_1, \ldots, q_{\text{Rank}(g)}), g, p) \in \Lambda$ by $[\bar{q}, g, p]$. We define $\text{Rank}([\bar{q}, g, p]) = \text{Rank}(g)$. Let h be the tree homomorphism induced by $h([\bar{q}, g, p](x_1, \ldots, x_k)) = g(x_1, \ldots, x_k)$ for all $g \in \Gamma^{(k)}$. Clearly, h is linear and non-deleting.

Next, we consider the WFTA $\mathcal{B}$ with weights taken from the Boolean semiring $\mathbb{B}$. Let $\mathcal{B} = (P, \Lambda, \beta, \mathbb{1}_{Q_{\text{fin}}})$ with β defined by

$$\beta_{[\bar{q},g,p]}(\bar{q}', p') = \begin{cases} 1 & \text{if } [\bar{q}', g, p'] = [\bar{q}, g, p] \ , \\ 0 & \text{otherwise} \end{cases}$$

for all $[\bar{q}, g, p] \in \Lambda$, $(\bar{q}', p') \in Q^{k+1}$. Clearly, $\llbracket \mathcal{B} \rrbracket = \mathbb{1}_L$, for some tree language $L \subseteq T_\Lambda$. It is well known, that the support of a recognizable $\mathbb{B}$-weighted weighted tree language is a recognizable tree language (cf. Lemma 3.11 & Theorem 3.12 in [FV09]). Hence, L is a recognizable tree language.

Finally, we define the WFTA $\mathcal{A}_w = (\{q_w\}, \Lambda, \omega, \mathbb{1}_{\{q_w\}})$ with

$$\omega_{[\bar{q},g,p]}\big((\underbrace{q_w, \ldots, q_w}_{\text{Rank}(g) \text{ times}}), q_w\big) = \begin{cases} \alpha_g(\bar{q}, q') & \text{if } p = q' \in Q \\ \alpha_g(\bar{q}, q') \cdot \lambda(q') & \text{if } p = q'_{\text{fin}} \in Q_{\text{fin}} \end{cases}$$

for all $[\bar{q}, g, p] \in \Lambda$. Please note that $\llbracket \mathcal{A}_w \rrbracket = [\mathcal{A}_w]_{q_w}$. Thus, we will not distinguish between those two semantics.

To prove that

$$\llbracket \mathcal{A} \rrbracket = h^{-1} \circ (\llbracket \mathcal{A}_w \rrbracket \odot \mathbb{1}_L)$$

holds for this construction, we will prove the following claim first.

Claim 5.6. For all $q \in Q$ it holds $[\mathcal{A}]_q(t) = \left(h^{-1} \circ ([\![\mathcal{A}_w]\!] \odot [\mathcal{B}]_q)\right)(t)$ for all $t \in T_\Gamma$.

Proof. The proof is by induction on the depth of t. If $t = g$ for some $g \in \Gamma^{(0)}$, we get

$$
\begin{aligned}
& [\mathcal{A}]_q(t) \\
= \ & \alpha_g(\varepsilon, q) + 0 \\
= \ & \sum_{p \in Q} \left(\alpha_g(\varepsilon, p) \cdot \mathbb{1}_{\{q\}}(p)\right) + \sum_{p_{\text{fin}} \in Q_{\text{fin}}} \left(\alpha_g(\varepsilon, p) \cdot \lambda(p) \cdot \mathbb{1}_{\{q\}}(p_{\text{fin}})\right) \\
= \ & \sum_{[\varepsilon, g, p] \in \Lambda^{(0)}} [\![\mathcal{A}]\!]_w([\varepsilon, g, p]) \cdot [\mathcal{B}]_q([\varepsilon, g, p]) \\
= \ & \sum_{t' \in h^{-1}(g)} \left([\![\mathcal{A}]\!]_w \odot [\mathcal{B}]_q\right)(t') \\
= \ & \left(h^{-1} \circ ([\![\mathcal{A}]\!]_w \odot [\mathcal{B}]_q)\right)(g)
\end{aligned}
$$

for all $q \in Q$.

Assume the claim holds for all $t \in T_\Gamma$ of depth lower or equal to n for some $n \in \mathbb{N}$.

We consider some $t \in T_\Gamma$ of depth $n + 1$. There exist $g \in \Gamma^{(k)}$ and $t_1, \dots, t_k \in T_\Gamma$ such that $t = g(t_1, \dots, t_k)$. Clearly, t_i has a depth lower or equal to n for all $1 \leq i \leq k$. Let us denote the tuple $(t_1, \dots, t_k)$ by $\bar{t}$ and $h^{-1}(t_1) \times \dots \times h^{-1}(t_k)$ by $h^{-1}(\bar{t})$. First, we observe

$$
t' \in h^{-1}(t) \Leftrightarrow t' = [\bar{p}, g, q](\bar{t'})
$$

for some $q \in P$, $\bar{p} \in Q^k$, and $\bar{t'} \in h^{-1}(\bar{t})$. Moreover, we have

$$
[\mathcal{B}]_q\left([\bar{p}, g, q'](\bar{t'})\right) = 1 \Leftrightarrow q = q' \wedge \forall 1 \leq i \leq k. \left([\mathcal{B}]_{p_i}(t'_i) = 1\right) \ .
$$

Therefore,

$$
\sum_{t' \in h^{-1}(t)} r \odot [\mathcal{B}]_q(t') = \sum_{\bar{p} \in Q^k} \sum_{\bar{t'} \in h^{-1}(\bar{t})} r \odot [B]_q\left([\bar{p}, g, q](\bar{t'})\right) \tag{5.1}
$$

holds for all $r \in S \langle\!\langle T_\Lambda \rangle\!\rangle$.

By this, we get

$$
\begin{aligned}
& [\mathcal{A}]_q(t) \\
= \ & \sum_{\bar{p} \in Q^k} \alpha_g(\bar{p}, q) \prod_{i=1}^{k} [\mathcal{A}]_{p_i}(t_i) \\
\overset{\text{(IH)}}{=} \ & \sum_{\bar{p} \in Q^k} \alpha_g(\bar{p}, q) \prod_{i=1}^{k} \left(h^{-1} \circ ([\![\mathcal{A}]\!]_w \odot [\mathcal{B}]_{p_i})\right)(t_i) \\
= \ & \sum_{\bar{p} \in Q^k} \alpha_g(\bar{p}, q) \prod_{i=1}^{k} \left(\sum_{t'_i \in h^{-1}(t_i)} [\![\mathcal{A}]\!]_w \odot [\mathcal{B}]_{p_i}(t'_i)\right) \\
= \ & \sum_{\bar{p} \in Q^k} \alpha_g(\bar{p}, q) \sum_{\bar{t'} \in h^{-1}(\bar{t})} \prod_{i=1}^{k} [\![\mathcal{A}]\!]_w \odot [\mathcal{B}]_{p_i}(t'_i) \\
= \ & \sum_{\bar{p} \in Q^k} \sum_{\bar{t'} \in h^{-1}(\bar{t})} \left(\alpha_g(\bar{p}, q) \cdot \prod_{i=1}^{k} [\![\mathcal{A}]\!]_w(t'_i)\right) \cdot \left(1 \cdot \prod_{i=1}^{k} [\mathcal{B}]_{p_i}(t'_i)\right) \\
= \ & \sum_{\bar{p} \in Q^k} \sum_{\bar{t'} \in h^{-1}(\bar{t})} [\![\mathcal{A}]\!]_w\left([\bar{p}, g, q](\bar{t'})\right) \cdot [B]_q\left([\bar{p}, g, q](\bar{t'})\right) \\
\overset{5.1}{=} \ & \sum_{t' \in h^{-1}(t)} [\![\mathcal{A}]\!]_w \odot [\mathcal{B}]_q(t') \\
= \ & h^{-1} \circ \left([\![\mathcal{A}]\!]_w \odot [\mathcal{B}]_q\right)(t)
\end{aligned}
$$

This finishes the proof of the claim. $\qquad\square$

We return to our main proof, it remains to show that we are able to apply the final weights. We consider some $t \in T_\Gamma$. There exists $g \in \Gamma^{(k)}$ and $\bar{t} \in T_\Gamma^k$ (we use the same notation as above) with $t = g(\bar{t})$. First, we make a similar observation as in the proof of our claim. Namely,

$$\llbracket \mathcal{B} \rrbracket([\bar{q}, g, p](\bar{t'})) = 1 \Leftrightarrow p \in Q_{\text{fin}} \wedge \forall 1 \leq i \leq k.([\![\mathcal{B}]\!]_{q_i}(t'_i) = 1)$$

for all $\bar{t'} \in h^{-1}(\bar{t})$. Thus,

$$\sum_{t' \in h^{-1}(t)} r \odot \llbracket \mathcal{B} \rrbracket(t') = \sum_{p \in Q} \sum_{\bar{q} \in Q^k} \sum_{\bar{t'} \in h^{-1}(\bar{t})} r \odot \llbracket \mathcal{B} \rrbracket([\bar{q}, g, p_{\text{fin}}](\bar{t'})) \tag{5.2}$$

for all $r \in S\langle\!\langle T_\Lambda \rangle\!\rangle$. Finally, analogously to the proof of Claim 5.6, we are able to deduce

$$
\begin{aligned}
&\llbracket \mathcal{A} \rrbracket(t) \\
=\ & \sum_{p \in Q} \lambda(p) \cdot [\mathcal{A}]_p(t) \\
=\ & \sum_{p \in Q} \sum_{\bar{q} \in Q^k} \lambda(p) \cdot \alpha_g(\bar{p}, q) \cdot \prod_{i=1}^k [\mathcal{A}]_{q_i}(t_i) \\
\overset{\text{C. } 5.6}{=}\ & \sum_{p \in Q} \sum_{\bar{q} \in Q^k} \lambda(p) \cdot \alpha_g(\bar{p}, q) \cdot \prod_{i=1}^k h^{-1}\big(\llbracket \mathcal{A}_w \rrbracket \odot [\mathcal{B}]_{q_i}\big)(t_i) \\
=\ & \sum_{p \in Q} \sum_{\bar{q} \in Q^k} \lambda(p) \cdot \alpha_g(\bar{p}, q) \cdot \prod_{i=1}^k \sum_{t'_i \in h^{-1}(t_i)} \llbracket \mathcal{A}_w \rrbracket \odot [\mathcal{B}]_{q_i}(t'_i) \\
=\ & \sum_{p \in Q} \sum_{\bar{q} \in Q^k} \sum_{\bar{t'} \in h^{-1}(\bar{t})} \big(\lambda(p) \cdot \alpha_g(\bar{p}, q) \cdot \prod_{i=1}^k \llbracket \mathcal{A}_w \rrbracket(t'_i)\big) \cdot \big(1 \cdot \prod_{i=1}^k [\mathcal{B}]_{q_i}(t'_i)\big) \\
=\ & \sum_{p \in Q} \sum_{\bar{q} \in Q^k} \sum_{\bar{t'} \in h^{-1}(\bar{t})} \big(\omega_{[\bar{q}, g, p_{\text{fin}}]}(q_w, (q_w, \ldots, q_w)) \cdot \prod_{i=1}^k \llbracket \mathcal{A}_w \rrbracket(t'_i)\big) \cdot \llbracket \mathcal{B} \rrbracket([\bar{q}, g, p_{\text{fin}}](\bar{t'})) \\
=\ & \sum_{p \in Q} \sum_{\bar{q} \in Q^k} \sum_{\bar{t'} \in h^{-1}(\bar{t})} \llbracket \mathcal{A}_w \rrbracket([\bar{q}, g, p_{\text{fin}}](\bar{t'})) \cdot \llbracket \mathcal{B} \rrbracket([\bar{q}, g, p_{\text{fin}}](\bar{t'})) \\
=\ & \sum_{p \in Q} \sum_{\bar{q} \in Q^k} \sum_{\bar{t'} \in h^{-1}(\bar{t})} \llbracket \mathcal{A}_w \rrbracket \odot \llbracket \mathcal{B} \rrbracket([\bar{q}, g, p_{\text{fin}}](\bar{t'})) \\
\overset{5.2}{=}\ & \sum_{t' \in h^{-1}(t)} \llbracket \mathcal{A}_w \rrbracket \odot \llbracket \mathcal{B} \rrbracket(t') \\
=\ & h^{-1}\big(\llbracket \mathcal{A}_w \rrbracket \odot \llbracket \mathcal{B} \rrbracket\big)(t) = h^{-1}\big(\llbracket \mathcal{A}_w \rrbracket \odot \mathbb{1}_L\big)(t)\ .
\end{aligned}
$$

Since t was arbitrary, this completes our proof. $\qquad\square$

Based on this result and Theorem 4.5 a characterization of WAFA via a Nivat-like Theorem is immediate.

Theorem 5.7 (Nivat-like theorem for WAFA). *A weighted language $s \in S\langle\!\langle \Sigma \rangle\!\rangle$ is recognized by a WAFA if and only if there exist a rank $r \in \mathbb{N}$, a ranked alphabet Λ, a linear tree homomorphism $h : T_\Lambda \to T_{\Sigma_\#^r}$, a regular tree language $L \subseteq T_\Lambda$, and a WFTA $\mathcal{A}_w$ with exactly one state such that for all $w \in \Sigma^*$, it holds:*

$$s(w) = h(\llbracket \mathcal{A}_w \rrbracket \odot \mathbb{1}_L)(t_w^r)\ .$$

Proof. $\Rightarrow$: Let $\mathcal{A}$ be a nice, equalized WAFA such that $\llbracket \mathcal{A} \rrbracket = s$. Due to Lemma 4.1, $r \in \mathbb{N}$ and a WFTA $\mathcal{B}$ exist such that $s(w) = (\llbracket \mathcal{B} \rrbracket \circ h^r)(w) = \llbracket \mathcal{B} \rrbracket(t_w^r)$. Applying Theorem 5.5 to $\llbracket \mathcal{B} \rrbracket$ gives us the desired result.

$\Leftarrow$: By Theorem 5.5, there exists a WFTA $\mathcal{B}$ such that $[\![\mathcal{B}]\!] = h([\![\mathcal{A}_w]\!] \odot \mathbb{1}_L)$. Let $h^r : \Sigma^* \to T_{\Sigma^r_\#}$ be the generic homomorphism. In consequence of Theorem 4.5, a WAFA $\mathcal{A}$ exists such that $[\![\mathcal{A}]\!](w) = [\![\mathcal{B}]\!](h^r(w)) = h([\![\mathcal{A}_w]\!] \odot \mathbb{1}_L)(h^r(w)) = h([\![\mathcal{A}_w]\!] \odot \mathbb{1}_L)(t^r_w)$ for all $w \in \Sigma$. This finishes our proof. $\qquad\square$

6. A logical characterization of WAFA

Based on Theorem 4.5 we are able to give a logical characterization of WAFA (Theorem 6.2). For this purpose, we will use the logical characterization by weighted MSO logic for trees which was introduced in [DV06].

Weighted MSO logic over trees is an extension of MSO logic over trees. It allows for the use of usual MSO formulas, but also incorporates quantitative aspects such as semiring elements and operations, as well as weighted quantifiers. In the end, every weighted MSO formula defines a weighted tree language. More precisely, let Γ be a ranked alphabet, each weighted MSO formula $\varphi \in \text{MSO}(\Gamma, S)$ defines a weighted tree language $[\![\varphi]\!] : T_\Gamma \to S$. Weighted MSO logic is strictly more expressive than WFTA. Nevertheless, it is possible to restrict the syntax of weighted MSO in such a way that it characterizes weighted tree languages recognized by WFTA. This fragment is called weighted syntactically restricted MSO (srMSO). Since it is not needed to understand the following proofs, we have omitted the formal definition of srMSO. We will use syntax and semantics of weighted srMSO without any changes and refer the interested reader to [FV09] or [DV11]. Our characterization of WAFA will be fully based on the following characterization theorem for WFTA:

Theorem 6.1 [FV09, Theorem 3.49 (A)]. *A weighted tree language $s \in S\langle\!\langle T_\Gamma \rangle\!\rangle$ is recognized by a WFTA if and only if there exist $\varphi \in \text{srMSO}(\Gamma, S)$ such that $s = [\![\varphi]\!]$.*

However, we still have to handle the homomorphism used in Theorem 4.5. This will be done by choosing an appropriate way of representing words as relational structures.

By definition $[\![\varphi]\!] \in S\langle\!\langle T_\Gamma \rangle\!\rangle$ for all $\varphi \in \text{srMSO}(\Gamma, S)$. However, we want to use weighted srMSO on trees to define weighted languages on words. To this end, we define $[\![\varphi]\!]_\Sigma \in S\langle\!\langle T_\Gamma \rangle\!\rangle$ by $[\![\varphi]\!]_\Sigma(w) = [\![\varphi]\!](t_w^{\text{Rank}(\Gamma)})$ for all $\varphi \in \text{srMSO}(\Gamma, S)$, $w \in \Sigma^*$. Since $\text{srMSO}(\Gamma, S) \subseteq \text{srMSO}(\Gamma \cup \Sigma_\#^{\text{Rank}(\Gamma)}, S)$, we can assume without loss of generality $\varphi \in \text{srMSO}(\Gamma \cup \Sigma_\#^{\text{Rank}(\Gamma)}, S)$. Hence, $[\![\varphi]\!]_\Sigma$ is well defined for all Σ. It is easy to see that $[\![\varphi]\!]_\Sigma = [\![\varphi]\!] \circ h$ where $h : \Sigma \to \Sigma_\#^{\text{Rank}(\Gamma)} \cup \Gamma$ is the generic homomorphism.

Theorem 6.2. *A weighted language $s \in S\langle\!\langle \Sigma \rangle\!\rangle$ is recognized by a WAFA if and only if there exist a ranked alphabet Γ and $\varphi \in \text{srMSO}(\Gamma, S)$ such that $s = [\![\varphi]\!]_\Sigma$.*

Proof. $\Rightarrow$: Assume $s \in S\langle\!\langle \Sigma \rangle\!\rangle$ is recognized by a WAFA. By 4.1, there exists $r \in \mathbb{N}$ and a WFTA $\mathcal{B}$ such that $s = [\![\mathcal{B}]\!] \circ h^r$. By Theorem 6.1, $\varphi \in \text{srMSO}(\Sigma_\#^r, S)$ exists such that $[\![\mathcal{B}]\!] = [\![\varphi]\!]$. Thus $s = [\![\mathcal{B}]\!] \circ h^r = [\![\varphi]\!] \circ h^r = [\![\varphi]\!]_\Sigma$.

$\Leftarrow$: If $s = [\![\varphi]\!]_\Sigma$ for some $\varphi \in \text{srMSO}(\Gamma \cup \Sigma_\#^{\text{Rank}(\Gamma)}, S)$, we get $s = [\![\varphi]\!] \circ h^{\text{Rank}(\Gamma)}$. By Theorem 6.1, a WFTA $\mathcal{B}$ exists such that $[\![\varphi]\!] = [\![\mathcal{B}]\!]$. Therefore, $s = [\![\mathcal{B}]\!] \circ h^{\text{Rank}(\Gamma)}$. Since $\mathcal{B}$ is a WFTA and $h^{\text{Rank}(\Gamma)}$ a homomorphism, a WAFA $\mathcal{A}$ with $s = [\![A]\!]$ exists by Lemma 4.3. $\qquad\square$

While mirroring the branching behavior of WAFA in the logic gives a natural characterization of WAFA, the question arises how WAFA relate to the weighted MSO logic for words.

It is well known that WFTA are not capable of characterizing the entirety of weighted MSO logic for words, simply because MSO logic can define series which grow doubly exponential in the size of the input. While WAFA have this ability (cf. Example 3.3), they still are incapable of capturing the entirety of weighted MSO. It can be shown that the series r_B from the proof of Lemma 5.3 can be defined in weighted MSO logic for words, while it is not recognized by WAFA. If it is possible to characterize WAFA by a fragment of weighted MSO logic for words remains open.

7. Closure of WFTA under inverses of homomorphisms

It is well known (cf. Theorem 1.4.4 in [CDG$^+$08]) that regular tree languages are closed under inverses of homomorphisms. Sadly, this is not true in the weighted case, at least not for arbitrary semirings. This raises the question if it is possible to give a precise description of the class of semirings for which recognizable weighted tree languages are closed under inverses of homomorphisms. This question will be answered by Theorem 7.1.

Theorem 5.7 and Theorem 6.1 used Theorem 4.5 to apply known results for WFTA to WAFA. Vice versa, we can use Theorem 4.5 and Theorem 3.5 to characterize the semirings for which the class of recognizable S-weighted tree languages is closed under inverses of homomorphisms.

Theorem 7.1. *The class of S-weighted tree languages recognized by* WFTA *is closed under inverses of homomorphisms if and only if S is locally finite.*

To prove this result we will use the notion of *recognizable step functions*: A weighted tree language $r \in S\langle\!\langle T_\Gamma \rangle\!\rangle$ is a recognizable step function if there exist recognizable tree languages $L_1, \ldots, L_k$ and values $l_1, \ldots, l_k \in S$ such that $r = \sum_{i=1}^{k} l_i \cdot \mathbb{1}_{L_i}$. Due to [DV06], we know the following about recognizable step functions:

Lemma 7.2 [DV06, Lemma 3.1]. *If $r \in S\langle\!\langle T_\Gamma \rangle\!\rangle$ is a recognizable step function, then a partition $L_1, \ldots, L_k$ of T_Γ exits such that $r = \sum_{i=1}^{k} l_i \cdot \mathbb{1}_{L_i}$ for some $l_1, \ldots, l_k \in S$.*

Note, this lemma is not redundant since the definition of recognizable step functions does not demand that the recognizable tree languages are pairwise disjoint. Due to Lemma 7.2, we know that a weighted tree language is a recognizable step function if and only if it has a finite image and each preimage is a recognizable tree language. The next Lemma characterizes recognizable weighted tree languages over locally finite semirings.

Lemma 7.3 [DV06, Lemma 3.3 & Lemma 6.1]. *Let S be locally finite. A weighted tree language $r \in S\langle\!\langle T_\Gamma \rangle\!\rangle$ is recognizable if and only if r is a recognizable step function.*

Finally, we can proceed with the proof of Theorem 7.1.

Proof of Theorem 7.1. $\Rightarrow$: Assume the class of S-weighted tree languages recognized by WFTA is closed under inverses of homomorphisms.

Claim 7.4. The class of S-weighted Σ languages recognizable by WAFA and the class of S-weighted Σ languages recognizable by WFA are equal.

Clearly every WFA is a WAFA. Thus, for the proof of the claim, it remains to show that every weighted language which is recognized by a WAFA is recognized by a WFA. For this purpose, assume $s \in S\langle\!\langle \Sigma^* \rangle\!\rangle$ is recognized by a WAFA $\mathcal{A}$. Due to Theorem 4.5, there exists a WFTA $\mathcal{B}$ and a homomorphism $h : \Sigma^* \to T_\Gamma$ such that $[\![\mathcal{A}]\!] = [\![\mathcal{B}]\!] \circ h$. However, by

our assumption there exists a WFTA $\mathcal{C}$ over $\Sigma_{\#}^1$ such that $[\![\mathcal{C}]\!] = [\![\mathcal{B}]\!] \circ h$ and hence a WFA $\mathcal{C}'$ over Σ such that $[\![\mathcal{A}]\!] = [\![\mathcal{B}]\!] \circ h = [\![\mathcal{C}]\!] = [\![\mathcal{C}']\!]$. Thereby, s is recognized by a WFA. This proves our claim.

By Claim 7.4 and Theorem 3.5 it follows that S is locally finite. This finishes the proof of the first direction.

$\Leftarrow$: Assume S is locally finite. Let $r \in S\langle\!\langle T_\Gamma \rangle\!\rangle$ be recognizable and $h : T_\Lambda \to T_\Gamma$ a homomorphism. Due to Lemma 7.2 and Lemma 7.3, we have $r = \sum_{i=1}^k l_i \cdot \mathbb{1}_{L_i}$ for some partition $L_1, \ldots, L_k$ of T_Γ and values $l_1, \ldots, l_k \in S$. We claim $r \circ h = \sum_{i=1}^k l_i \cdot \mathbb{1}_{h^{-1}(L_i)}$. To prove this, consider some arbitrary $t \in T_\Lambda$. Since the L_i form a partition of T_Γ there exists a unique $j \in \{1, \ldots, k\}$ such that $h(t) \in L_i$. Therefore, we have

$$
\begin{aligned}
& (r \circ h)(t) \\
= & \textstyle\sum_{i=1}^k l_i \cdot \mathbb{1}_{L_i}(h(t)) = l_j \\
= & l_j \cdot \mathbb{1}_{h^{-1}(L_j)}(t) \\
\overset{l_j \text{ unique}}{=} & \textstyle\sum_{i=1}^k l_i \cdot \mathbb{1}_{h^{-1}(L_i)}(t) \ .
\end{aligned}
$$

Since recognizable tree languages are closed under inverses of homomorphisms, we know that $h^{-1}(L_1), \ldots, h^{-1}(L_k) \subseteq T_\Lambda$ are recognizable. Thus, $r \circ h$ is a recognizable step function. Again, by Lemma 7.3, we get $r \circ h$ is recognizable. This completes our proof. $\qquad\square$

8. WAFA and polynomial automata

We will use known results for polynomial automata, to prove the decidability of the Zeroness Problem for WAFA if weights are taken from the rationals (Lemma 8.2).

Informally a polynomial automaton is a set of registers which get updated by polynomial funcions according to some input. Historically this principle was studied from many perspectives. For example: [Sén07] studies it in the form of "polynomial recurrent relations", or the "cost register automata" from [ADD$^+$13] which allow for a very broad class of updates and weight structures. We will follow the terminology and definitions of [BTSW17] where "polynomial automata" were considered as a generalization of both vector addition systems and weighted automata. Polynomial automata are quite similar to WAFA, the authors of [BTSW17] even prove that the characteristic function of the reversal of each language recognized by a non-weighted alternating automaton is recognized by a polynomial automaton of the same size. We want to strengthen this connection. In [BTSW17] polynomial automata are defined over the rational numbers. However, it is easy to give a more general definition of arbitrary commutative semirings.

A *polynomial automaton* (PA) is a 5-tuple $\mathcal{A} = (n, \Sigma, \alpha, p, \gamma)$, where $n \in \mathbb{N}$ is the number of states, Σ is an alphabet, $\alpha \in S^n$ is an initial weight vector, $p : \Sigma \to S[X_n]^n$ the transition function, and $\gamma \in S[X_n]$ an output polynomial. We denote the i-th entry of $p(a)$ by $p_i(a)$.

Let $\mathcal{A} = (n, \Sigma, \alpha, p, \gamma)$ be a PA. Its *state behavior* $[\mathcal{A}] : \{1, \ldots, n\} \times \Sigma^* \to S$ is the mapping defined by

$$
[\mathcal{A}](i, w) = \begin{cases} \alpha_i & \text{if } w = \varepsilon, \\ p_i \langle [\mathcal{A}](1, v), \ldots, [\mathcal{A}](n, v) \rangle & \text{if } w = va \text{ for } a \in \Sigma. \end{cases}
$$

Usually we will denote $[\mathcal{A}](i, w)$ by $[\mathcal{A}]_i(w)$. Now, the *behavior* of $\mathcal{A}$ is the weighted language $[\![\mathcal{A}]\!] : \Sigma^* \to S$ defined by

$$[\![\mathcal{A}]\!](w) = \gamma\big([\mathcal{A}]_1(w), \ldots, [\mathcal{A}]_n(w)\big).$$

It is easy to check that this definition is a reformulation of the definition found in [BTSW17].

Let the reversal of a weighted language $s \in S\langle\!\langle\Sigma\rangle\!\rangle$ be defined by $s^R(w) = s(w^R)$ for all $w = w_1 \ldots w_n$ with $w_1, \ldots, w_n \in \Sigma$, where $w^R = w_n \ldots w_1$. Comparing the definition of state behavior for WAFA and PA already yields the following lemma:

Lemma 8.1. *A weighted language $s \in S\langle\!\langle\Sigma\rangle\!\rangle$ is recognized by a WAFA if and only if s^R is recognized by a PA.*

Proof. Assume s is recognized by $\mathcal{A} = (Q, \Sigma, \delta, P_0, \tau)$.

Let $\mathcal{B} = (|Q|, \Sigma, \big(\tau(q_1), \ldots, \tau(q_n)\big), p, P_0)$ be a PA with $p_i(a) = \delta(q_i, a)$ for all $1 \le i \le |Q|, a \in \Sigma$. Then, a straightforward induction on $|w|$ shows $[\![A]\!](w) = [\![\mathcal{B}]\!](w^R)$ for all $w \in \Sigma$. The second direction is proven analogously to the first one. $\qquad\square$

Let $\mathcal{A}, \mathcal{A}'$ be two WAFA. We observe $[\![\mathcal{A}]\!](w) = 0$ for all $w \in \Sigma^*$ if and only if $[\![\mathcal{A}]\!](w^R) = 0$ for all $w \in \Sigma^*$. Moreover, we have $[\![\mathcal{A}]\!](w) = [\![\mathcal{A}']\!](w)$ for all $w \in \Sigma^*$ if and only if $[\![\mathcal{A}]\!](w^R) = [\![\mathcal{A}']\!](w^R)$ for all $w \in \Sigma^*$. This allows us to derive the following corollary from Lemma 8.1:

Corollary 8.2. *The Zeroness Problem and the Equivalence Problem for WAFA with weights taken from the rationals are in the complexity class ACKERMANN and hard for the complexity class ACKERMANN.*

Proof. Using the respective results for polynomial automata (Theorem 1, Theorem 4, and Corollary 1 in [BTSW17]) together with Lemma 8.1 yields this result. $\qquad\square$

In general, weighted languages recognized by WAFA are not closed under reversal (Theorem 8.4 in [KM18]). Thus, the class of weighted languages recognized by WAFA and the class of languages recognized by PA differ. Moreover, series recognized by polynomial automata are (in general) not close under reversal. However, if weights are taken from a commutative semiring, series recognized by WFA are closed under reversal, this would allow for a direct translation of Theorem 3.5 into the setting of PA.

9. Conclusion

We were able to connect WAFA to a variety of formalisms, giving a better understanding of their expressive power and characterizing the class of quantitative languages recognized by WAFA. From here, there are various routes to take. It could be of great practical use to find a logical characterization of WAFA via a linear formalism such as a weighted linear logic, or weighted rational expressions tailored to the expressive power of WAFA, or a fitting fragment of weighted MSO logic for words.

Similar to the work in [BTSW17], one could investigate subclasses of WAFA allowing for more efficient decision procedures. An interesting candidate is strictly alternating WAFA who have a bounded number of alternations within any run. While these use the ability to recognize very fast growing series, they still add in expressive power due to the ability to multiply over subruns.

A different direction would aim to use the expressive power added by alternation to achieve an automata model which is as expressive weighted MSO for words. While WAFA can not fulfill this role a generalization of WAFA might be. After all, the use of product and sum within runs is very similar, to the use of product- and sum-quantifiers in weighted MSO for words.

Alternatively, one could approach the concept of alternation in weighted automata dealing with more complex structures than words, such as weighted alternating tree automata. And of course, having the universal interpretation of nondeterminism in mind, one may take several of these routes at once!

References

[ADD⁺13] R. Alur, L. DAntoni, J. Deshmukh, M. Raghothaman, and Y. Yuan. Regular functions and cost register automata. In *2013 28th Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 13–22, 2013. `doi:10.1109/LICS.2013.65`.

[AK11] S. Almagor and O. Kupferman. Max and sum semantics for alternating weighted automata. In T. Bultan and P. Hsiung, editors, *Automated Technology for Verification and Analysis*, pages 13–27. Springer, 2011. `doi:10.1007/978-3-642-24372-1_2`.

[BL80] J.A. Brzozowski and E. Leiss. On equations for regular languages, finite automata, and sequential networks. *Theoretical Computer Science*, 10(1):19–35, 1980. URL: `sciencedirect.com/science/article/pii/0304397580900699`, `doi:10.1016/0304-3975(80)90069-9`.

[BTSW17] M. Benedikt, Duffm T., A. Sharad, and J. Worrell. Polynomial automata: Zeroness and applications. In *ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–12, June 2017. URL: `ora.ox.ac.uk/objects/uuid:f341421b-2130-42d7-bb21-52442bad0b80#citeForm`, `doi:10.1109/LICS.2017.8005101`.

[CDG⁺08] H. Comon, M. Dauchet, R. Gilleron, F.Jacquemard, D. Lugiez, et al. *Tree Automata Techniques and Applications*. HAL open science, 2008. URL: `hal.inria.fr/hal-03367725`.

[CDH08] K. Chatterjee, L. Doyen, and T. A. Henzinger. Quantitative languages. In M. Kaminski and S. Martini, editors, *Computer Science Logic*, pages 385–400. Springer, 2008. URL: `doi.org/10.1007/978-3-540-87531-4_28`, `doi:10.1007/978-3-540-87531-4_28`.

[CDH09] K. Chatterjee, L. Doyen, and T. A. Henzinger. Alternating weighted automata. In M. Kutyłowski, W. Charatonik, and M. Gębala, editors, *Fundamentals of Computation Theory*, pages 3–13. Springer, 2009. `doi:10.1007/978-3-642-03409-1_2`.

[CKS81] A. K. Chandra, D. C. Kozen, and L. J. Stockmeyer. Alternation. *J. ACM*, 28(1):114–133, January 1981. URL: `doi.acm.org/10.1145/322234.322243`, `doi:10.1145/322234.322243`.

[DD15] M. Droste and S. Dück. Weighted automata and logics on graphs. In G. F. Italiano, G. Pighizzini, and D. T. Sannella, editors, *Mathematical Foundations of Computer Science 2015*, pages 192–204. Springer, 2015. `doi:10.1007/978-3-662-48057-1_15`.

[DG07] M. Droste and P. Gastin. Weighted automata and weighted logics. *Theoretical Computer Science*, 380(1):69–86, 2007. Automata, Languages and Programming. URL: `sciencedirect.com/science/article/pii/S0304397507001582`, `doi:10.1016/j.tcs.2007.02.055`.

[DG09] M. Droste and P. Gastin. *Weighted Automata and Weighted Logics*, chapter 5. Volume 1 of Droste et al. [DKV09], 2009. `doi:10.1007/978-3-642-01492-5_5`.

[DG17] M. Droste and D. Götze. A nivat theorem for quantitative automata on unranked trees. In *Models, Algorithms, Logics and Tools - Essays Dedicated to Kim Guldstrand Larsen on the Occasion of His 60th Birthday*, pages 22–35, 2017. URL: `doi.org/10.1007/978-3-319-63121-9_2`, `doi:10.1007/978-3-319-63121-9\_2`.

[DGV13] G. De Giacomo and M. Y. Vardi. Linear temporal logic and linear dynamic logic on finite traces. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, pages 854–860. AAAI Press, 2013. URL: `dl.acm.org/citation.cfm?id=2540128.2540252`.

[DK21] M. Droste and D. Kuske. *Weighted automata*, chapter 4, pages 113–150. Volume 1 of Pin [Pin21], 2021. `doi:10.4171/automata`.

[DKV09]   M. Droste, W. Kuich, and H. Vogler, editors. *Handbook of Weighted Automata*. Springer, 2009. `doi:10.1007/978-3-642-01492-5`.

[DS87]   E. Muller D and P. E. Schupp. Alternating automata on infinite trees. *Theoretical Computer Science*, 54(2):267–276, 1987. URL: `sciencedirect.com/science/article/pii/0304397587901332`, `doi:10.1016/0304-3975(87)90133-2`.

[DV06]   M. Droste and H. Vogler. Weighted tree automata and weighted logics. *Theoretical Computer Science*, 366(3):228 – 247, 2006. Automata and Formal Languages. URL: `sciencedirect.com/science/article/pii/S0304397506005676`, `doi:10.1016/j.tcs.2006.08.025`.

[DV11]   M. Droste and H. Vogler. Weighted logics for unranked tree automata. *Theory of Computing Systems*, 48(1):23–47, 2011. `doi:10.1007/s00224-009-9224-4`.

[FV09]   Z. Fülöp and H. Vogler. *Weighted Tree Automata and Tree Transducers*, chapter 9. Volume 1 of Droste et al. [DKV09], 2009. `doi:10.1007/978-3-642-01492-5_9`.

[Gra21]   G. Grabolle. A nivat theorem for weighted alternating automata over commutative semirings. *Electronic Proceedings in Theoretical Computer Science*, 346:241–257, Sep 2021. URL: `dx.doi.org/10.4204/EPTCS.346.16`, `doi:10.4204/eptcs.346.16`.

[KM18]   P. Kostolányi and F. Mišún. Alternating weighted automata over commutative semirings. *Theoret. Comput. Sci.*, 740:1 – 27, 2018. URL: `sciencedirect.com/science/article/pii/S0304397518303098`, `doi:10.1016/j.tcs.2018.05.003`.

[Niv68]   M. Nivat. *Transduction des langages de Chomsky*. PhD thesis, Univ. Paris, 1968. `doi:10.5802/aif.287`.

[Pin21]   Jean-Éric Pin, editor. *Handbook of Automata Theory*. European Mathematical Society, 2021. `doi:10.4171/automata`.

[Sch61]   M.P. Schützenberger. On the definition of a family of automata. *Inform. and Control*, 4(2):245 – 270, 1961. URL: `sciencedirect.com/science/article/pii/S001999586180020X`, `doi:10.1016/S0019-9958(61)80020-X`.

[Sén07]   G. Sénizergues. Sequences of level 1, 2, 3,..., k,... In Volker Diekert, Mikhail V. Volkov, and Andrei Voronkov, editors, *Computer Science – Theory and Applications*, pages 24–32. Springer, 2007. URL: `rdcu.be/c1YaX`.

[Var95]   M. Y. Vardi. *Alternating automata and program verification*, pages 471–485. Springer, 1995. `doi:10.1007/BFb0015261`.