A Nivat Theorem for Weighted Alternating Automata over Commutative Semirings*

Gustav Grabolle

Institute of Computer Science, Leipzig University, 04109 Leipzig, Germany

`grabolle at infromatik.uni-leipzig.de`

In this paper, we give a Nivat-like characterization for weighted alternating automata over commutative semirings (WAFA). To this purpose we prove that weighted alternating can be characterized as the concatenation of weighted finite tree automata (WFTA) and a specific class of tree homomorphism. We show that the class of series recognized by weighted alternating automata is closed under inverses of homomorphisms, but not under homomorphisms. 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. Using the corresponding result for polynomial automata, we are able to prove that the ZERONESS problem for weighted alternating automata with the rational numbers as weights is decidable.

1 Introduction

Non-determinism, a situation with several possible outcomes, is usually interpreted as a choice. An (existential) automaton accepts if there exists at least one successful run. Contrary to this, one can view non-determinism as an obligation. A universal automaton would accept if all possible runs are successful. While this notion of non-determinism is less prominent, it is equally natural. Allowing for the simultaneous use of existential and universal determinism leads to the concept of alternation, such as in alternating Turing machines [4] or alternating automata on finite [3], or infinite structures [7]. 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 of the 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 [8], or in program verification [23].

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 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 [13].

To adapt alternating automata into the weighted setting, we observe 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 where studied in [6] and in [1] over finite words. While these authors focused on very specific weight structures, a more recent approach defines weighted alternating automata over arbitrary commutative semirings [18].

Weighted alternating automata have the same expressive power as weighted automata if and only if the semiring used is locally finite [18]. 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 [14], the equivalence of weighted automata and weighted rational expressions [22] and weighted restricted MSO logic [10], or the decidability of equality due to minimization if weights are taken from a fields [22] 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, but also 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 4.4), 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 automata 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 5.6). 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 weighted tree automata are closed under inverses of homomorphisms (Theorem 7.1). For this purpose, we will use our core theorem, and a result form [18].

Lastly, in Section 8, we investigate the connection between weighted alternating automata and recently introduced polynomial automata [2] 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).

Due to the limitation of space, we moved the technical parts of some proofs into the Appendix which can be found in the long version of this paper on the authors website.

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 *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 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 *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 $m \cdot 1 = 1 \cdot m$ 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 $S\langle\!\langle M \rangle\!\rangle$. 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 *S-weighted Σ-language* (for short: weighted language).

Let X_n always denote a linear ordered set with $|X_n| = n \in \mathbb{N}$, we refer to the i-th element of X_n by x_i. Let $S\langle X_n \rangle$ denote the *semiring of polynomials* with coefficients in S and commuting indeterminates $x_1, \ldots, x_n$. We say $m \in S\langle X_n \rangle$ 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 *constant* monomial if its degree is zero. Each polynomial which can be written as a sum of distinct non-constant monomials is called a non-constant polynomial and the set of all non-constant polynomials is denoted by $S\langle X_n \rangle_{\text{const}=0}$. For $p, p_1, \ldots, p_n \in S\langle X_n \rangle$ let $p\langle p_1, \ldots, p_n \rangle$ denote the simultaneous substitution of x_i by p_i in p for all $1 \leq i \leq n$.

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 $g \in T_\Gamma[X_n]$.

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 $w = iv \in \mathrm{Pos}(t)$. Moreover, let $M \subseteq \mathrm{Pos}(t)$ 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 regarding 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 $\mathrm{r}(t) = \sum_{i=1}^n |t^{-1}(x_i)|$ and $T_\Gamma^{(r)}(X_n) = \{t \in T_\Gamma(X_n) \mid \mathrm{r}(t) = r\}$ for all $r \in \mathbb{N}$.

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.

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 [18].

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, $\delta : Q \times \Sigma \to S\langle Q \rangle$ is a transition function, $P_0 \in S\langle Q \rangle$ 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. To ease late proofs, let $M_{(q,a)}$ denote the set of monomials that appear in $\delta(q, a)$.

We say a WAFA $(Q, \Sigma, \delta, P_0, \tau)$ with $Q = \{q_1, \ldots, q_n\}$ is a *weighted finite automaton* (WFA) if $P_0 = \sum_{j=1}^n s_j \cdot q_j$ and $\delta(q_i, a) = \sum_{j=1}^n s_{ij}^a \cdot q_j$ for all $1 \leq i \leq n$, $a \in \Sigma$. This definition coincides with the usual definition [11],[10]. We can see this by defining the initial weight function by $\lambda(q_i) = s_i$, the transition weight function by $\mu(a)(q_i, q_j) = s_{ij}^a$, and the final weight function by $\gamma(q_i) = \tau(q_i)$ for all $1 \leq i, j \leq n$ and all $a \in \Sigma$.

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 non-constant,

(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* $\mathcal{A}'$.

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

(i) Since $\cdot$ is distributive and commutative in $S\langle Q \rangle$ 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 occurring 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 [18], there exists an equivalent WAFA $\mathcal{A}'$ such that (i)-(iii) hold for $\mathcal{A}'$.

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

$$Q' = Q \cup \{q_s \mid s \text{ is the coefficient of some monomial in } \mathcal{A}\}.$$

Without loss of generality, we can assume that these sets are disjoint. Furthermore, let δ' be defined by

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

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. However, if this construction is applied to a WFA $\mathcal{A}$, the resulting $\mathcal{A}'$ does not have to be a WFA.

$\square$

In [18] 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 ease later 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 and equalized* WAFA $\mathcal{A}'$.

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, it is easy to see that $\mathcal{A}$ and $\mathcal{A}'$ are equivalent and that $\mathcal{A}'$ is nice.

Please note that the coefficients of monomials in $\mathcal{A}$ where not changed, thus we also can assume that all these coefficients are 1.   □

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 •. 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

$$
\begin{aligned}
s : \quad \Sigma^* &\rightarrow S : \\
w &\mapsto \begin{cases} (2^j)^{2^i} & \text{if } w = a^i b^j \\ 0 & \text{otherwise.} \end{cases}
\end{aligned}
$$

We consider the WAFA $\mathcal{A} = (\{q, p\}, \Sigma, P_0, \delta, \tau)$, defined by:

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

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

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

It is easy to 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 3.4 (Theorem 7.1 in [18]). *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.*
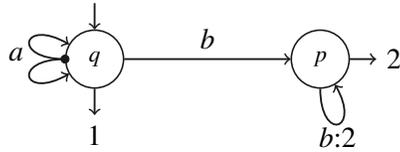
Figure 1: Representation of $\mathcal{A}$
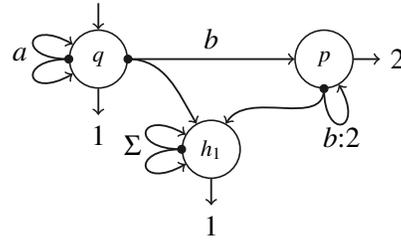
Figure 2: Representation of equalized, nice $\mathcal{A}$

4 A characterization of WAFA via weighted finite tree automata

Our central result Theorem 4.4 is included in this section, as well as the definition for weighted finite tree automata.

The connection between alternating automata and trees is well known. Often trees are used to define the runs of alternating automata. This is possible for WAFA too (see Appendix in the long version). 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 *(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, $\delta = (\delta_k \mid 1 \leq k \leq \text{Rank}(\Gamma))$ 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 $[\![\mathcal{A}]\!] : T_\Gamma \to S$ 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 arbitary $r \in \mathbb{N}$. Given an alphabet Σ and $r \geq 1$, we define the ranked alphabet $\Sigma_\#^r = \Sigma \cup \{\#\}$ with $\text{Rank}(\#) = 0$ and $\text{Rank}(a) = r$ 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
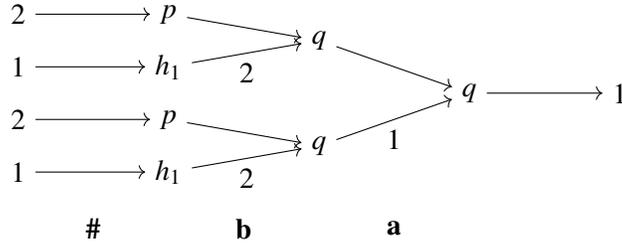
Figure 3: Run of translated WFTA on t_{ab}^2

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 $r \in \mathbb{N}$.*

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 $\mathcal{A}$. We define the WFTA $\mathcal{B} = (Q, \Sigma_\#^r, \beta, \lambda)$ with $\lambda = \mathbb{1}_{\{q_1\}}$ and

$$
\begin{aligned}
\beta_\#(\varepsilon, q) &= \tau(q) , \\
\beta_a(\bar{p}, q) &= \begin{cases} s & \text{if } s \cdot p_1 \cdot \ldots \cdot p_r \in M_{(q,a)} , \\ 0 & \text{otherwise} . \end{cases}
\end{aligned}
$$

Please note that the order of $p_1, \ldots, p_r$ is of importance. Thus, due to the fact that $\mathcal{A}$ is nice, $\beta_a(\bar{p}, q)$ becomes always zero if the p_i are not ordered according to the linear order on the states of $\mathcal{A}$.

By induction over the length of $w \in \Sigma^*$ (see Appendix in the long version), we get

$$
[\mathcal{A}]_q(w) = [\mathcal{B}]_q(t_w^r)
$$

for all $q \in Q$, $w \in \Sigma^*$. 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. $\square$

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 3.

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 would be the generic homomorphism, we could 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 $q \in Q$, $a \in \Sigma$.

By induction over the length of $w \in \Sigma^*$ we get (see Appendix in the long version)

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

for all $q \in Q$, $w \in \Sigma^*$. 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 4.4. *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. $\square$

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 of runs of WAFA.

5 A Nivat theorem for WAFA

This section leads to the Nivat-like characterization of WAFA (Theorem 5.6), 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 $w \in \Sigma^*$. 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 $w \in \Sigma^*$. Note that we have $h(\mathbb{1}_L)(w) = 1$ if and only if there exists $v \in \Gamma^*$ with $h(v) = w$ and $v \in L$ in the boolean setting. 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 [20] characterizes word-to-word transducers. A generalized version for WFA over arbitrary semirings (Theorem 6.3 in [14]) can be stated in the following way:

Theorem 5.1 (Nivat-like theorem for WFA [14]). *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 up 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 $s \in S\langle\!\langle \Sigma^* \rangle\!\rangle \cap K$ and $h : \Gamma^* \to \Sigma^*$ a homomorphism implies $h(s) \in K$. Moreover, K is *closed under inverses of homomorphisms* if $s' \in S\langle\!\langle \Gamma^* \rangle\!\rangle \cap K$ and $h : \Gamma^* \to \Sigma^*$ a homomorphism implies $h^{-1}(s') \in K$. 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 [14]). 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 $s = \llbracket B \rrbracket \circ h^r$. Clearly, $h^r \circ h' : \Lambda^* \to \Sigma^r_\#$ is a tree homomorphism. Thus, due to Lemma 4.3, $h'^{-1}(s) = (\llbracket B \rrbracket \circ h^r) \circ h' = \llbracket B \rrbracket \circ (h^r \circ h')$ 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

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

Due to Lemma 8.3 from [18], 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

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

for all $w \in \Sigma^*$. Thus, $h(r_R)$ is not recognized by a WAFA. The series r_R is recognized by the WAFA $\mathcal{A}_R$ which can be found in the appendix of the long version. This completes our proof. $\square$

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 [12], while a very general result for graphs can be found in [9]. 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 (Theorem 3.8 in [17])**.** *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 (Theorem 12 in [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 [12]. 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 transition in $\mathcal{A}$, the mapping to the letters in Γ, the tree language of runs in $\mathcal{A}$, and an automaton adding weights to the transitions, respectively. This yields the desired equation. See Appendix in the long version for a proof of correctness of this construction.   □

Based on this result and Theorem 4.4 a characterization of WAFA via a Nivat-like Theorem is immediate.

Theorem 5.6 (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. And 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 $\llbracket \mathcal{B} \rrbracket = h(\llbracket \mathcal{A}_w \rrbracket \odot \mathbb{1}_L)$. Let $h^r : \Sigma^* \to T_{\Sigma_\#^r}$ be the generic homomorphism. In consequence of Theorem 4.4, a WAFA $\mathcal{A}$ exists such that $\llbracket \mathcal{A} \rrbracket(w) = \llbracket \mathcal{B} \rrbracket(h^r(w)) = h(\llbracket \mathcal{A}_w \rrbracket \odot \mathbb{1}_L)(h^r(w)) = h(\llbracket \mathcal{A}_w \rrbracket \odot \mathbb{1}_L)(t_w^r)$ for all $w \in \Sigma$. This finishes our proof.   □

6   A logical characterization of WAFA

Based on Theorem 4.4 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 [15].

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 $\llbracket \varphi \rrbracket : T_\Gamma \to S$. Weighted MSO logic is strictly more expressive than WFTA. Nevertheless, it is possible 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). Due to a lack of space, we have to omit the formal definition of srMSO. We will use syntax and semantics of weighted srMSO without any changes and refer the interested reader to [17] or [16]. Our characterization of WAFA will be fully based on the following characterization theorem for WFTA:

Theorem 6.1 (Theorem 3.49 (A) in [17]).   *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 $s = \llbracket \varphi \rrbracket$.*

However, we still have to handle the homomorphism used in Theorem 4.4. 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 $\varphi \in \mathrm{srMSO}(\Gamma, S)$. 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 $[\![\varphi]\!]_\Sigma(w) = [\![\varphi]\!](t_w^{\mathrm{Rank}(\Gamma)})$ for all $\varphi \in \mathrm{srMSO}(\Gamma, S)$, $w \in \Sigma^*$. Since $\mathrm{srMSO}(\Gamma, S) \subseteq \mathrm{srMSO}(\Gamma \cup \Sigma_\#^{\mathrm{Rank}(\Gamma)}, S)$, we can assume without loss of generality $\varphi \in \mathrm{srMSO}(\Gamma \cup \Sigma_\#^{\mathrm{Rank}(\Gamma)}, S)$. Hence, $[\![\varphi]\!]_\Sigma$ is well defined for all Σ. It is easy to see that $[\![\varphi]\!]_\Sigma = [\![\varphi]\!] \circ h$ where $h : \Sigma \to \Sigma_\#^{\mathrm{Rank}(\Gamma)} \cup \Gamma$ 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 $\varphi \in \mathrm{srMSO}(\Gamma, S)$ 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, $\varphi \in \mathrm{srMSO}(\Sigma_\#^r, S)$ 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 $\varphi \in \mathrm{srMSO}(\Gamma \cup \Sigma_\#^{\mathrm{Rank}(\Gamma)}, S)$, we get $s = [\![\varphi]\!] \circ h^{\mathrm{Rank}(\Gamma)}$. By Theorem 6.1, a WFTA $\mathcal{B}$ exists such that $[\![\varphi]\!] = [\![\mathcal{B}]\!]$. Therefore $s = [\![\mathcal{B}]\!] \circ h^{\mathrm{Rank}(\Gamma)}$. Since $\mathcal{B}$ is a WFTA and $h^{\mathrm{Rank}(\Gamma)}$ a homomorphism, a WAFA $\mathcal{A}$ with $s = [\![A]\!]$ exists by Lemma 4.3. $\qquad\square$

7   Closure of WFTA under inverses of homomorphisms

It is a well known fact 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 $\mathcal{S}$ for which WFTA are closed under inverses of homomorphisms. This question will be answered by Theorem 7.1.

Theorem 5.6 and Theorem 6.1 used Theorem 4.4 to apply known results for WFTA to WAFA. Vice versa, we can use Theorem 4.4 and Theorem 3.4 to characterize $\mathcal{S}$.

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 [15], we know the following about recognizable step functions:

Lemma 7.2 (Lemma 3.1 in [15]). *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 (Lemma 3.3 & Lemma 6.1 in [15]).   *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 1.   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 4.4, there exists a WFTA $\mathcal{B}$ and a homomorphism $h : \Sigma^* \to T_\Gamma$ such that $[\![\mathcal{A}]\!] = [\![B]\!] \circ h$. However, by our assumption there exists a WFTA $\mathcal{C}$ over $\Sigma^1_\#$ such that $[\![C]\!] = [\![B]\!] \circ h$ and hence a WFA $\mathcal{C}'$ over Σ such that $[\![\mathcal{A}]\!] = [\![B]\!] \circ h = [\![C]\!] = [\![C']\!]$. Thereby, s is recognized by a WFA. This proves our claim.

By Claim 1 and Theorem 3.4 it follows that S is locally finite.

$\Leftarrow$: 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 $L_1, \dots, L_k$ of T_Γ and values $l_1, \dots, l_k \in S$. 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 $j \in \{1, \dots, k\}$ 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 $h^{-1}(L_1), \dots, h^{-1}(L_k) \subseteq T_\Lambda$ 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. $\square$

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).

Polynomial automata where introduced in [2] as a generalization of both vector addition systems and weighted automata. Polynomial automata are quite similar to WAFA, the authors of [2] 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 [2] polynomial automata are defined over the rational numbers. However, it is easy to give a more general definition for 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, $p : \Sigma \to S\langle X_n \rangle^n$ the transition function, and $\gamma \in S\langle X_n \rangle$ 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 $[\![A]\!] : \Sigma^* \to S$ defined by

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

It is easy to check that this definition is a reformulation of the definition found in [2].

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 $w = w_1, \ldots w_n \in \Sigma^*$, 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 $\mathcal{B} = (|Q|, \Sigma, (\tau(q_1), \ldots, \tau(q_n)), p, P_0)$ be a PA with $p_i(a) = \delta(q_i, a)$ for all $1 \leq i \leq |Q|, a \in \Sigma$. 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. $\square$

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. Lemma 8.1 + Theorem 1, Theorem 4, Corollary 1 in [2]. $\square$

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. Similar to the work in [2], one could investigate subclasses of WAFA allowing for more efficient decision procedures. 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

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

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

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

[4] A. K. Chandra, D. C. Kozen & L. J. Stockmeyer (1981): *Alternation*. *J. ACM* 28(1), pp. 114–133, doi:10.1145/322234.322243.

[5] K. Chatterjee, L. Doyen & T. A. Henzinger (2008): *Quantitative languages*. In M. Kaminski & S. Martini, editors: *Computer Science Logic*, Springer, pp. 385–400, doi:10.1007/978-3-540-87531-4_28.

[6] K. Chatterjee, L. Doyen & T. A. Henzinger (2009): *Alternating Weighted Automata*. In M. Kutyłowski, W. Charatonik & M. Gębala, editors: *Fundamentals of Computation Theory*, Springer, pp. 3–13, doi:10.1007/978-3-642-03409-1_2.

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

[8] G. De Giacomo & M. Y. Vardi (2013): *Linear temporal logic and linear dynamic logic on finite traces*. In: *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, AAAI Press, pp. 854–860, doi:10.5555/2540128.2540252. Available at `http://dl.acm.org/citation.cfm?id=2540128.2540252`.

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

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

[11] M. Droste & P. Gastin (2009): *Weighted Automata and Weighted Logics*, chapter 5. 1 of Droste et al. [13], doi:10.1007/978-3-642-01492-5_5.

[12] M. Droste & D. Götze (2017): *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*, pp. 22–35, doi:10.1007/978-3-319-63121-9_2.

[13] M. Droste, W. Kuich & H. Vogler, editors (2009): *Handbook of Weighted Automata*, 1st edition. 1, Springer, doi:10.1007/978-3-642-01492-5.

[14] M. Droste & D. Kuske (2021): *Weighted automata*, pp. 113—-150. 1 of Pin [21], doi:10.4171/Automata-1/4.

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

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

[17] Z. Fülöp & H. Vogler (2009): *Weighted Tree Automata and Tree Transducers*, chapter 9. 1 of Droste et al. [13], doi:10.1007/978-3-642-01492-5_9.

[18] P. Kostolányi & F. Mišún (2018): *Alternating weighted automata over commutative semirings*. Theoret. Comput. Sci. 740, pp. 1 – 27, doi:10.1016/j.tcs.2018.05.003. Available at `http://www.sciencedirect.com/science/article/pii/S0304397518303098`.

[19] J. van Leeuwen, editor (1995): *Computer Science Today: Recent Trends and Developments*. 1, Springer, doi:10.1007/BFb0015232.

[20] M. Nivat (1968): *Transductions des langages de Chomsky*. Annales de l'Institut Fourier 18(1), pp. 339–455, doi:10.5802/aif.287. Available at `http://www.numdam.org/articles/10.5802/aif.287/`.

[21] Jean-Éric Pin, editor (2021): *Handbook of Automata Theory*, 1st edition. 1, European Mathematical Society, doi:10.4171/Automata.

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

[23] M. Y. Vardi (1995): *Alternating automata and program verification*, pp. 471–485. 1 of van Leeuwen [19], doi:10.1007/BFb0015261.