



# Symbolic Weighted Language Models, Quantitative Parsing and Automated Music Transcription

Florent Jacquemard, Lydia Rodriguez-de La Nava

## ► To cite this version:

Florent Jacquemard, Lydia Rodriguez-de La Nava. Symbolic Weighted Language Models, Quantitative Parsing and Automated Music Transcription. CIAA 2022 - International Conference on Implementation and Application of Automata, Jun 2022, Rouen, France. hal-03647675v1

**HAL Id: hal-03647675**

**<https://hal.science/hal-03647675v1>**

Submitted on 20 Apr 2022 (v1), last revised 7 Jun 2022 (v2)

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Symbolic Weighted Language Models, Quantitative Parsing and Automated Music Transcription

Florent Jacquemard and Lydia Rodriguez-de la Nava

INRIA and CNAM/Cedric lab, Paris, France

**Abstract.** We study several classes of symbolic weighted formalisms: automata (**swA**), transducers (**swT**) and visibly pushdown extensions (**swVPA**, **swVPT**). They combine the respective extensions of their symbolic and weighted counterparts, allowing a quantitative evaluation of words over a large or infinite input alphabet.

We present properties of closure by composition, the computation of transducer-defined distances between nested words and languages, as well as a PTIME 1-best search algorithm for **swVPA**. These results are applied to solve in PTIME a variant of parsing over infinite alphabets. We illustrate this approach with a motivating use case in automated music transcription.

## 1 Introduction



Symbolic Weighted (**sw**) language models [10] (automata and transducers) combine two important extensions of standard models. On the one hand, symbolic extensions, like in *Symbolic Automata* (**sA** [6]), can handle an infinite input alphabet  $\Sigma$ , by guarding every transition with a predicate  $\phi : \Sigma \rightarrow \mathbb{B}$ . The ability of **sA** to compare input symbols is quite restricted, compared to other models of automata extended e.g. with registers (see [17] for a survey), however, under appropriate closure conditions on the set of predicates, all the good properties enjoyed by automata over finite alphabets are still valid for **sA**.

On the other hand, *Weighted Automata* (**wA** [7]) extend qualitative evaluation of input words to *quantitative* evaluation, by assigning to every transition a weight value in a semiring  $\mathbb{S}$ . The weights of the rules involved in a computation are combined using the product operator  $\otimes$  of  $\mathbb{S}$ , whereas the sum operator  $\oplus$  of  $\mathbb{S}$  is used to resolve ambiguity (typically,  $\oplus$  selects, amongst two computations, the best weighted one). These extensions have also been applied to evaluate hierarchical structures, like trees [7, ch. 9], or nested words, in symbolic [5], or weighted [4] extensions of Visibly Pushdown Automata (**VPA** [2]). With their ability to evaluate data sequences quantitatively, **sw** models have found various applications such as data stream processing [3], runtime verification of timed systems [20] or robustness optimization for machine learning models [13].

The **sw** models with data storage defined in [10], where their expressiveness is extensively studied, are very general, and cover all the models cited above, as

well as those considered in this paper. Here, we consider simple models of **sw**-automata and transducers whose transitions are assigned *functions*  $\phi : \Sigma \rightarrow \mathbb{S}$  from input symbols in an infinite alphabet  $\Sigma$  into a semiring  $\mathbb{S}$  (Section 3). Such functions generalize the boolean guards of symbolic models from the Boolean codomain  $\mathbb{B}$  to an arbitrary semiring  $\mathbb{S}$ , and the constant values of weighted models. We prove some properties of closure under composition for those **sw** models, and propose a polynomial time algorithm of search for a word of minimal weight for **swVPA**. We apply these results to the problem of parsing words over infinite alphabet (**sw-parsing**), whose goal is: given an (unstructured) input word  $s$ , to find a (structured) nested word  $t$  at a minimal distance from  $s$ , where the distance, following the sense of [15], is defined by  $T(s, t) \otimes A(t)$ ,  $T$  being a **sw**-transducer (**swT**) and  $A$  a **swVPA** (Section 4). The notion of transducer-based distances allows to consider different infinite alphabets for the input  $s$  and output  $t$ . Moreover, the use of **swVPA** allows to search for an output  $t$  in the form of a nested word, as a linear representation of a parse tree; **sw-parsing** is solved with a Bar-Hillel, Perles and Shamir construction [9, ch. 13], and the best-search algorithm for **swVPA**. We illustrate our approach with an application that motivated this work: *automated music transcription*, i.e. the problem of converting a linear music recording given in input into a score in *Common Western Music Notation*, a representation structured hierarchically [21].

*Example 1.* Let us consider a short input sequence  $I$  of musical events represented by symbols of the form  $e:\tau$  in an infinite alphabet  $\Sigma$ , where  $e$  is a MIDI key number in 21..108 [18], or the mark ‘start’ or ‘end’, and  $\tau \in \mathbb{Q}$  is a duration in seconds. Such inputs typically correspond to the recording of a live performance:  $I = \text{start}: 0.07, 69: 0.65, 71: 0.19, 73: 0.14, 74: 0.31, 76: 0.35, 77: 0.29, \text{end}: 0$ .

The output of parsing  $I$  is a nested word  $O$ , where separated notes are grouped into hierarchical patterns. It is made of symbols  $a:\tau'$  in an alphabet  $\Delta$ , where  $a$  is either a *note name*, (e.g.,  $A_4$ ,  $G_5$ , etc.), a *continuation* symbol ‘-’, or a *markup* symbol (opening or closing parenthesis). The time value  $\tau'$  is a musical duration. For instance, the music score  is represented by the nested word:  $O = [\text{m}: 2, [2: 1, A_4: \frac{1}{2}, [2: \frac{1}{2}, -: \frac{1}{4}, [2: \frac{1}{4}, B_4: \frac{1}{8}, C\sharp_5: \frac{1}{8}, [2: \frac{1}{4}, [2: \frac{1}{2}, [2: 1, [\text{m}: 1, [3: 1, D_5: \frac{1}{3}, E_5: \frac{1}{3}, F_5: \frac{1}{3}, [3: 1, [\text{m}: 1, [\text{m}: 2$ . The symbol  $[\text{m}$  marks the opening of a *measure* (a time interval of duration 1), while the subsequences of  $O$  between markups  $[_d: \ell$  and  $]_d: \ell$ , for some natural number  $d$ , represent the division of a duration  $\ell$  by  $d$ . The sequence  $O \in \Delta^*$  is a candidate solution for the transcription of  $I \in \Sigma^*$ . Let us consider another candidate , represented by  $O' = [\text{m}: 2, [2: 1, A_4: \frac{1}{2}, [2: \frac{1}{2}, -: \frac{1}{4}, B_4: \frac{1}{4}, [2: \frac{1}{2}, [2: 1, [\text{m}: 1, [3: 1, 'C\sharp_5': 0, D_5: \frac{1}{3}, E_5: \frac{1}{3}, F_5: \frac{1}{3}, [3: 1, [\text{m}: 1, [\text{m}: 2$ . The quoted symbol ‘ $C\sharp_5$ ’ represents an *appoggiatura*, i.e. an ornamental note with theoretical duration 0. Roughly, **sw**-parsing associates a weight value to each candidate, depending of its temporal distance to  $I$  and notational complexity. Our goal is to find the best candidate.  $\diamond$

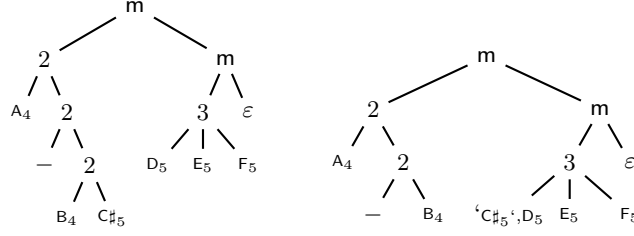


Fig. 1. Tree representation of the scores of Ex 1, linearized respectively into  $O$  and  $O'$ .

## 2 Preliminary Notions

*Semirings.* We shall consider weight values in a *semiring*  $\langle S, \oplus, \otimes, \mathbb{0}, \mathbb{1} \rangle$ : a structure of domain  $S$ , equipped with two associative binary operators  $\oplus$  and  $\otimes$ , with respective neutral elements  $\mathbb{0}$  and  $\mathbb{1}$ , such that  $\oplus$  is commutative,  $\otimes$  distributes over  $\oplus$ :  $\forall x, y, z \in S, x \otimes (y \oplus z) = (x \otimes y) \oplus (x \otimes z)$ , and  $(x \oplus y) \otimes z = (x \otimes z) \oplus (y \otimes z)$ ,  $\mathbb{0}$  is absorbing for  $\otimes$ :  $\forall x \in S, \mathbb{0} \otimes x = x \otimes \mathbb{0} = \mathbb{0}$ .

A semiring  $S$  is *commutative* if  $\otimes$  is commutative. It is *idempotent* if for every  $x \in S$ ,  $x \oplus x = x$ . Every idempotent semiring  $S$  induces a partial ordering  $\leq_\oplus$  called the *natural ordering* of  $S$  [14], defined by: for every  $x, y \in S$ ,  $x \leq_\oplus y$  iff  $x \oplus y = x$ . It is sometimes defined in the opposite direction [7, ch. 1]; we follow here the direction that coincides with the usual ordering on the Tropical semiring *min-plus* (Figure 2). An idempotent semiring  $S$  is called *total* if  $\leq_\oplus$  is total, i.e. when for every  $x, y \in S$ , either  $x \oplus y = x$  or  $x \oplus y = y$ .

**Lemma 1 (Monotony, [14]).** *If  $\langle S, \oplus, \otimes, \mathbb{0}, \mathbb{1} \rangle$  is idempotent, for every  $x, y, z \in S$ ,  $x \leq_\oplus y$  implies  $x \oplus z \leq_\oplus y \oplus z$ ,  $x \otimes z \leq_\oplus y \otimes z$  and  $z \otimes x \leq_\oplus z \otimes y$ .*

We say that  $S$  is *monotonic* wrt  $\leq_\oplus$ . Another important semiring property in the context of optimization is *superiority* ((i) of Lemma 2), which generalizes the *non-negative weights* condition in Dijkstra's shortest-path algorithm. Intuitively, it means that combining elements with  $\otimes$  always increases their weight.

**Lemma 2 (Superiority, Boundedness).** *Let  $\langle S, \oplus, \otimes, \mathbb{0}, \mathbb{1} \rangle$  be an idempotent semiring. The two following statements are equivalent: (i) for all  $x, y \in S$ ,  $x \leq_\oplus x \otimes y$  and  $y \leq_\oplus x \otimes y$  (ii) for all  $x \in S$ ,  $\mathbb{1} \oplus x = \mathbb{1}$ .*

The property (i) of superiority implies that  $\mathbb{1} \leq_\oplus x \leq_\oplus \mathbb{0}$  for all  $x \in S$  (with  $x = \mathbb{1}$  and  $y = \mathbb{0}$ ). From an optimization point of view, it means that  $\mathbb{1}$  is the best value, and  $\mathbb{0}$  the worst. A semiring  $S$  with property (ii) of Lemma 2 is called *bounded* in [14] and in the rest of the paper.

**Lemma 3 ([14], Lemma 3).** *Every bounded semiring is idempotent.*

We need to extend  $\oplus$  to infinitely many operands. A semiring  $S$  is called *complete* [7, ch. 1] if it has an operation  $\bigoplus_{i \in I} x_i$  for every family  $(x_i)_{i \in I}$  of elements of  $\text{dom}(S)$  over an index set  $I \subseteq \mathbb{N}$ , such that:

	domain	$\oplus$	$\otimes$	$\mathbb{0}$	$\mathbb{1}$
Boolean	$\{\perp, \top\}$	$\vee$	$\wedge$	$\perp$	$\top$
Viterbi	$[0, 1] \subset \mathbb{R}$	$max$	$\times$	$0$	$1$
Tropical min-plus	$\mathbb{R}_+ \cup \{\infty\}$	$min$	$+$	$\infty$	$0$

**Fig. 2.** Some commutative, bounded, total and complete semirings.

- i. *infinite sums extend finite sums*:  $\forall j, k \in \mathbb{N}, j \neq k$ ,
- $$\bigoplus_{i \in \emptyset} x_i = \mathbb{0}, \quad \bigoplus_{i \in \{j\}} x_i = x_j, \quad \bigoplus_{i \in \{j, k\}} x_i = x_j \oplus x_k,$$
- ii. *associativity and commutativity*: for all partition  $(I_j)_{j \in J}$  of  $I$ ,
- $$\bigoplus_{j \in J} \bigoplus_{i \in I_j} x_i = \bigoplus_{i \in I} x_i,$$
- iii. *distributivity of products over infinite sums*: for all  $I \subseteq \mathbb{N}$ ,  $\forall x, y \in \mathbb{S}$ ,
- $$\bigoplus_{i \in I} (x \otimes y_i) = x \otimes \bigoplus_{i \in I} y_i, \text{ and } \bigoplus_{i \in I} (x_i \otimes y) = \left(\bigoplus_{i \in I} x_i\right) \otimes y.$$

*Label Theories.* The functions labelling the transitions of **sw**-automata and transducers generalize the Boolean algebras of [6]. We consider *alphabets*, which are non-empty countable sets of symbols denoted by  $\Sigma, \Delta, \dots$  and write  $\Sigma^*$  for the set of finite sequences (*words*) over  $\Sigma$ ,  $\varepsilon$  for the empty word,  $\Sigma^+ = \Sigma^* \setminus \{\varepsilon\}$ , and  $uv$  for the concatenation of  $u, v \in \Sigma^*$ .

Given a semiring  $\langle \mathbb{S}, \oplus, \mathbb{0}, \otimes, \mathbb{1} \rangle$ , a *label theory*  $\bar{\Phi}$  over  $\mathbb{S}$  is an indexed family of sets  $\Phi_\Sigma$ , resp.  $\Phi_{\Sigma, \Delta}$ , containing recursively enumerable functions of type  $\Sigma \rightarrow \mathbb{S}$ , resp.  $\Sigma \times \Delta \rightarrow \mathbb{S}$ , and such that if  $\Phi_{\Sigma, \Delta} \in \bar{\Phi}$ , then  $\Phi_\Sigma \in \bar{\Phi}$  and  $\Phi_\Delta \in \bar{\Phi}$ , every  $\Phi_\Sigma, \Phi_{\Sigma, \Delta} \in \bar{\Phi}$  contains all the constant functions of  $\Sigma \rightarrow \mathbb{S}$ , resp.  $\Sigma \times \Delta \rightarrow \mathbb{S}$ , for all  $\Phi_{\Sigma, \Delta} \in \bar{\Phi}$ ,  $\eta \in \Phi_{\Sigma, \Delta}$ ,  $a \in \Sigma$ ,  $b \in \Delta$ , the partial application  $x \mapsto \eta(x, b)$  is in  $\Phi_\Sigma$  and the partial application  $y \mapsto \eta(a, y)$  is in  $\Phi_\Delta$ , and  $\bar{\Phi}$  is closed under the following operators, derived from the operations of  $\mathbb{S}$ :

- For all  $\Phi_\Sigma \in \bar{\Phi}$ , for all  $\phi \in \Phi_\Sigma$ , and  $\alpha \in \mathbb{S}$ ,  $\alpha \otimes \phi : x \mapsto \alpha \otimes \phi(x)$ , and  $\phi \otimes \alpha : x \mapsto \phi(x) \otimes \alpha$ , belong to  $\Phi_\Sigma$ , and similarly for  $\oplus$  and for  $\Phi_{\Sigma, \Delta}$ .
- For all  $\Phi_\Sigma \in \bar{\Phi}$ , for all  $\phi, \phi' \in \Phi_\Sigma$ ,  $\phi \otimes \phi' : x \mapsto \phi(x) \otimes \phi'(x)$  belongs to  $\Phi_\Sigma$ .
- For all  $\Phi_{\Sigma, \Delta} \in \bar{\Phi}$ , for all  $\eta, \eta' \in \Phi_{\Sigma, \Delta}$ ,  $\eta \otimes \eta' : x, y \mapsto \eta(x, y) \otimes \eta'(x, y)$  belongs to  $\Phi_{\Sigma, \Delta}$ .
- For all  $\Phi_\Sigma, \Phi_{\Sigma, \Delta} \in \bar{\Phi}$ , for all  $\phi \in \Phi_\Sigma$  and  $\eta \in \Phi_{\Sigma, \Delta}$ ,  $\phi \otimes_1 \eta : x, y \mapsto \phi(x) \otimes \eta(x, y)$  and  $\eta \otimes_1 \phi : x, y \mapsto \eta(x, y) \otimes \phi(x)$  belong to  $\Phi_{\Sigma, \Delta}$ .
- For all  $\Phi_\Delta, \Phi_{\Sigma, \Delta} \in \bar{\Phi}$ , for all  $\psi \in \Phi_\Delta$  and  $\eta \in \Phi_{\Sigma, \Delta}$ ,  $\psi \otimes_2 \eta : x, y \mapsto \psi(y) \otimes \eta(x, y)$  and  $\eta \otimes_2 \psi : x, y \mapsto \eta(x, y) \otimes \psi(y)$  belong to  $\Phi_{\Sigma, \Delta}$ .
- Analogous closures hold for  $\oplus$ .

*Example 2.* We go back to Example 1. In order to align an input in  $\Sigma^*$  with a music score in  $\Delta^*$ , we must account for the expressive timing of human performance that results in small time shifts between an input event of  $\Sigma$  and a

notational event in  $\Delta$ . These shifts can be weighted as a distance in  $\Phi_{\Sigma, \Delta}$ , defined in the tropical min-plus semiring by  $\delta(e: \tau, a: \tau') = |\tau' - \tau|$  if  $a$  corresponds to  $e$  (e.g.  $e$  is the MIDI key 69 and  $a$  is the note  $A_4$ ), or  $\emptyset$  otherwise.  $\diamond$

### 3 SW Visibly Pushdown Automata and Transducers

Let  $\langle \mathbb{S}, \oplus, \emptyset, \otimes, \mathbb{1} \rangle$  be a commutative and complete semiring and let  $\Sigma$  and  $\Delta$  be countable alphabets called *input* and *output* respectively, such that  $\Delta$  is partitioned into three disjoint subsets of symbols  $\Delta_i$ ,  $\Delta_c$  and  $\Delta_r$ , called respectively *internal*, *call* and *return* [2]. Let  $\bar{\Phi}$  be a label theory over  $\mathbb{S}$ , consisting of  $\Phi_e = \Phi_\Sigma$ ,  $\Phi_i = \Phi_{\Delta_i}$ ,  $\Phi_c = \Phi_{\Delta_c}$ ,  $\Phi_r = \Phi_{\Delta_r}$ ,  $\Phi_{ei} = \Phi_{\Sigma, \Delta_i}$  and  $\Phi_{cr} = \Phi_{\Delta_c, \Delta_r}$ .

**Definition 1 (swVPT).** A Symbolic Weighted Visibly Pushdown Transducer over  $\Sigma$ ,  $\Delta$ ,  $\mathbb{S}$ , and  $\bar{\Phi}$  is a tuple  $T = \langle Q, P, \text{in}, \bar{w}, \text{out} \rangle$ , where  $Q$  is a finite set of states,  $P$  is a finite set of stack symbols,  $\text{in} : Q \rightarrow \mathbb{S}$  (respectively  $\text{out} : Q \rightarrow \mathbb{S}$ ) are functions defining the weight for entering (respectively leaving) a state, and  $\bar{w}$  is a tuple composed of the transition functions :  $w_{10} : Q \times Q \rightarrow \Phi_e$ ,  $w_{01} : Q \times Q \rightarrow \Phi_i$ ,  $w_{11} : Q \times Q \rightarrow \Phi_{ei}$ ,  $w_c : Q \times Q \times P \rightarrow \Phi_c$ ,  $w_r : Q \times P \times Q \rightarrow \Phi_{cr}$ ,  $w_r^e : Q \times Q \rightarrow \Phi_r$ .

For convenience, we extend the above transition functions as follows, for every  $q, q' \in Q$ ,  $p \in P$ ,  $e \in \Sigma$ ,  $a \in \Delta_i$ ,  $c \in \Delta_c$ ,  $r \in \Delta_r$ , overloading their names:

$$\begin{aligned} w_{10}(q, e, \varepsilon, q') &= \phi(e) \quad \text{where } \phi = w_{10}(q, q'), \\ w_{01}(q, \varepsilon, a, q') &= \phi(a) \quad \text{where } \phi = w_{01}(q, q'), \\ w_{11}(q, e, a, q') &= \eta(e, a) \quad \text{where } \eta = w_{11}(q, q'), \\ w_c(q, \varepsilon, c, q', p) &= \phi(c) \quad \text{where } \phi = w_c(q, q', p), \\ w_r(q, c, p, \varepsilon, r, q') &= \eta(c, r) \quad \text{where } \eta = w_r(q, p, q'), \\ w_r^e(q, \varepsilon, r, q') &= \phi(r) \quad \text{where } \phi = w_r^e(q, q'). \end{aligned}$$

The swVPT  $T$  computes asynchronously on pairs  $\langle s, t \rangle \in \Sigma^* \times \Delta^*$ . Intuitively, a transition  $w_{ij}(q, e, a, q')$ , with  $i, j \in \{0, 1\}$  and  $e \in \Sigma \cup \{\varepsilon\}$ ,  $a \in \Delta_i \cup \{\varepsilon\}$ , is interpreted as follows: when reading  $e$  and  $a$  in the input and output words, it increments the current position in the input word if and only if  $i = 1$ , and in the output word iff  $j = 1$ , and changes state from  $q$  to  $q'$ . When  $e = \varepsilon$  (resp.  $a = \varepsilon$ ), the current symbol in the input (resp. output) is not read. These transitions ignore the stack.

A transition of  $w_c(q, \varepsilon, c, q', p)$  reads the call symbol  $c \in \Delta_c$  only in the output word, pushes it to the stack along with  $p \in P$ , and changes state from  $q$  to  $q'$ . As for  $w_r(q, c, p, \varepsilon, r, q')$  and  $w_r^e(q, \varepsilon, r, q')$  (used when the stack is empty), they read the return symbol  $r$  in the output word and change state from  $q$  to  $q'$ . Additionally,  $w_r$  reads and pops from the stack a pair  $\langle c, p \rangle$  and the symbol  $c$  is compared to  $r$  by the function  $\eta = w_r(q, p, q') \in \Phi_{cr}$ .

Formally, the computations of the transducer  $T$  are defined with an intermediate function  $\text{weight}_T$ . A configuration  $q[\gamma]$  is composed of a state  $q \in Q$  and a stack content  $\gamma \in \Gamma^*$ , where  $\Gamma = \Delta_c \times P$ , and  $\text{weight}_T$  is a function from

$[Q \times \Gamma^*] \times \Sigma^* \times \Delta^* \times [Q \times \Gamma^*]$  into  $\mathbb{S}$ , whose recursive definition enumerates each of the possible cases for reading  $e \in \Sigma$ ,  $a \in \Delta_i$ ,  $c \in \Delta_c$ , or  $r \in \Delta_r$  (the empty stack is denoted by  $\perp$ , and the topmost symbol is the last pushed pair):

$$\begin{aligned}
\text{weight}_T(q[\gamma], \varepsilon, \varepsilon, q'[\gamma']) &= \mathbb{1} \text{ if } q = q', \gamma = \gamma' \text{ and } \mathbb{0} \text{ otherwise} & (1) \\
\text{weight}_T(q[\gamma], e u, \varepsilon, q'[\gamma']) &= \bigoplus_{q'' \in Q} w_{10}(q, e, \varepsilon, q'') \otimes \text{weight}_T(q''[\gamma], u, \varepsilon, q'[\gamma']) \\
\text{weight}_T(q[\gamma], \varepsilon, a v, q'[\gamma']) &= \bigoplus_{q'' \in Q} w_{01}(q, \varepsilon, a, q'') \otimes \text{weight}_T(q''[\gamma], \varepsilon, v, q'[\gamma']) \\
\text{weight}_T(q[\gamma], e u, a v, q'[\gamma']) &= \bigoplus_{q'' \in Q} w_{10}(q, e, \varepsilon, q'') \otimes \text{weight}_T(q''[\gamma], u, a v, q'[\gamma']) \\
&\quad \oplus \bigoplus_{q'' \in Q} w_{01}(q, \varepsilon, a, q'') \otimes \text{weight}_T(q''[\gamma], e u, v, q'[\gamma']) \\
&\quad \oplus \bigoplus_{q'' \in Q} w_{11}(q, e, a, q'') \otimes \text{weight}_T(q''[\gamma], u, v, q'[\gamma']) \\
\text{weight}_T(q[\gamma], u, c v, q'[\gamma']) &= \bigoplus_{\substack{q'' \in Q \\ p \in P}} w_c(q, \varepsilon, c, q'', p) \otimes \text{weight}_T(q'' \left[ \begin{smallmatrix} \langle c, p \rangle \\ \gamma \end{smallmatrix} \right], u, v, q'[\gamma']) \\
\text{weight}_T(q \left[ \begin{smallmatrix} \langle c, p \rangle \\ \gamma \end{smallmatrix} \right], u, r v, q'[\gamma']) &= \bigoplus_{q'' \in Q} w_r(q, c, p, \varepsilon, r, q'') \otimes \text{weight}_T(q''[\gamma], u, v, q'[\gamma']) \\
\text{weight}_T(q[\perp], u, r v, q'[\gamma']) &= \bigoplus_{q'' \in Q} w_r^e(q, \varepsilon, r, q'') \otimes \text{weight}_T(q''[\perp], u, v, q'[\gamma'])
\end{aligned}$$

The weight associated by  $T$  to an input/output pair  $\langle s, t \rangle \in \Sigma^* \times \Delta^*$  is defined according to empty stack semantics:

$$T(s, t) = \bigoplus_{q, q' \in Q} \text{in}(q) \otimes \text{weight}_T(q[\perp], s, t, q'[\perp]) \otimes \text{out}(q') \quad (2)$$

Since  $\mathbb{0}$  is absorbing for  $\otimes$ , and neutral for  $\oplus$  in  $\mathbb{S}$ , if a transition's weight is equal to  $\mathbb{0}$ , then the entire term is  $\mathbb{0}$ , meaning the transition is impossible. This is analogous to the case of a transition's guard not satisfied in symbolic models [6].

*Symbolic Weighted Visibly Pushdown Automata.* **swVPA** are the particular case of **swVPT** that do not read in the input word, i.e. where all  $w_{10}$  and  $w_{11}$  are constant functions equal to  $\mathbb{0}$  (see Appendix C). They are a weighted extension of **svPA** [5], from Boolean semirings to arbitrary semiring domains. The relationship between **swVPA** and **sw-Tree Automata** is discussed in Appendix F.

*Example 3.* We consider a **swVPA**  $A$  over  $\Delta^*$ , with  $P = Q$ , computing a value of *notational complexity* for a given score. In a sequence  $O \in \Delta^*$  like in Example 1,  $\Delta_i$  contains timed notes and continuations, and  $\Delta_c$  and  $\Delta_r$  contain respectively opening and closing parentheses. To a call symbol  $\lceil_n: \ell$ , for some duration value

$\ell$ , let us associate a transition for the division of  $\ell$  by  $n$ :  $w_c(q_\ell, \varepsilon, \lceil n: \ell, q_{\frac{\ell}{n}}, q' \rceil) = \alpha_n \in \mathbb{S}$ . And to the matching return symbol  $\lfloor n: \ell$ , we associate a transition of weight  $\mathbb{1}$ :  $w_r(q_{\frac{\ell}{n}}, \lceil n: \ell, q', \varepsilon, \lfloor n: \ell, q' \rfloor) = \mathbb{1}$ , which jumps to the state  $q'$  stored in the stack. The choice of weight values for the call transitions can express some preferences in term of the expected output notation: if we want to prioritize pairs over triplets, in the Tropical min-plus semiring, then we would let  $\alpha_2 < \alpha_3$ . It is able to compute on several representations of a piece of music, estimating for each one a weight value depending on the preferences that we set. The algorithm of Theorem 4 then allows to select the best weighted representation.  $\diamond$

*Symbolic Weighted Transducers.* **swT** are the particular case of **swVPT** that do not use the stack during their computations, because all  $w_c$ ,  $w_r$  and  $w_r^e$  are constant functions equal to  $\mathbb{0}$ , or, more simply, because  $\Delta_c = \Delta_r = \emptyset$  (see App C).

The four first lines in expression (1) can be seen as a stateful definition of an edit-distance between a word  $s \in \Sigma^*$  and a word  $t \in \Delta_i^*$ , see also [15]. Intuitively, in this vision,  $w_{10}(q, e, \varepsilon, q')$  is the cost of the deletion of the symbol  $e \in \Sigma$  in  $s$ ,  $w_{01}(q, \varepsilon, a, q')$  is the cost of the insertion of  $a \in \Delta_i$  in  $t$ , and  $w_{11}(q, e, a, q')$  is the cost of the substitution of  $e \in \Sigma$  by  $a \in \Delta_i$ . Following (2), the cost of a sequence of such operations transforming  $s$  into  $t$  is the product in terms of  $\otimes$  of the individual costs of the operations involved, and the distance between  $s$  and  $t$  is the sum in terms of  $\oplus$  of all possible products.

*Example 4.* We propose a **swT** over  $\Sigma$  and  $\Delta_i$  that computes the distance between an input  $I \in \Sigma^*$  and an output  $O \in \Delta_i^*$  like in Ex. 1 (for  $\delta$ , see Ex. 2):

$$w_{11}(q_0, e: \tau, a: \tau', q_0) \text{ and } w_{11}(q_1, e: \tau, a: \tau', q_0) = \delta(e: \tau, a: \tau') \quad \text{if } a \neq - \quad (3)$$

$$w_{01}(q_0, \varepsilon, -: \tau', q_0) = \mathbb{1} \quad w_{01}(q_1, \varepsilon, -: \tau', q_0) = \mathbb{1} \quad w_{10}(q_0, e: \tau, \varepsilon, q_1) = \alpha \quad (4)$$

The continuation symbol  $'-'$  (e.g. in *ties*  $\text{♪} \text{---} \text{♪}$ , or *dots*  $\text{♪}$ ) is skipped with no cost ( $w_{01}$ ). We also want to consider performing errors, by switching to an error state  $q_1$ . Reading an extra event  $e$  is handled by  $w_{10}$  that switches to  $q_1$ , with a fixed  $\alpha \in \mathbb{S}$ , then  $w_{11}$  and  $w_{01}$  can switch back to  $q_0$ . Finally, we let  $q_0$  be the initial and final state, with  $\text{in}(q_0) = \text{out}(q_0) = \mathbb{1}$ , and  $\text{in}(q_1) = \text{out}(q_1) = 0$ .  $\diamond$

*Symbolic Weighted Automata.* **swA** are particular cases of **swT** omitting the output symbols, or equivalently, **swVPA** without markups ( $\Delta_c = \Delta_r = \emptyset$ ).

## 4 Symbolic Weighted Parsing

Parsing is the problem of structuring a linear representation (a finite word) according to a language model [9]. We shall consider in this section the problem of parsing over an infinite alphabet. Let  $\mathbb{S}$ ,  $\Sigma$ ,  $\Delta$ , and  $\bar{\Phi}$  be like in Section 3. We assume to be given the following input:

- a **swT**  $T$  over  $\Sigma$ ,  $\Delta_i$ ,  $\mathbb{S}$ , and  $\bar{\Phi}$ , defining a measure  $T: \Sigma^* \times \Delta_i^* \rightarrow \mathbb{S}$ ,

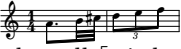
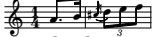


- a swVPA  $A$  over  $\Delta$ ,  $\mathbb{S}$ , and  $\bar{\Phi}$ , defining a measure  $A : \Delta^* \rightarrow \mathbb{S}$ ,
- an (unstructured) input word  $s \in \Sigma^*$ .

For every  $u \in \Sigma^*$  and  $t \in \Delta^*$ , let  $d_T(u, t) = T(u, t|_{\Delta_i})$ , where  $t|_{\Delta_i} \in \Delta_i^*$  is the projection of  $t$  onto  $\Delta_i$ , obtained from  $t$  by removing all symbols in  $\Delta \setminus \Delta_i$ . Given the above input, *symbolic weighted parsing* aims at finding a (structured) nested word  $t \in \Delta^*$  that minimizes  $d(s, t) \otimes A(t)$  wrt  $\leq_\oplus$ , i.e. such that:

$$d_T(s, t) \otimes A(t) = \bigoplus_{u \in \Delta^*} d_T(s, u) \otimes A(u) \quad (5)$$

In the terminology of [15], **sw**-parsing is the problem of computing the distance (5) between the input word  $s$  and the weighted language over the output alphabet defined by  $A$ , and returning a witness  $t$ .

*Example 5.* In the running example, the input is as follows: The **swT**  $T$  evaluates a “fitness measure”: a temporal distance between a performance and a nested-word representation of a music score (Example 4). The **swVPA**  $A$  expresses a weight related to the complexity of music notation (Example 3). The input word is  $I$  of Example 1. The notation , will be favored over  when the weight assigned to the call  $\lceil_2$  is less than the difference of weight between the appoggiatura ‘ $c_{\sharp_5}$ ’ and the standard note  $c_{\sharp_5}$ . The **sw**-parsing framework, applied to music transcription, finds an optimal solution considering both the fitness of the output to the input, and its notational complexity.  $\diamond$

Nested words in  $\Delta^*$  can represent linearizations of labeled trees, and to any Weighted Regular Tree Grammar (wRTG), we can associate in polynomial time a swVPA computing the same weight (see Appendix F). Therefore, instead of a swVPA in input, we may be given a wRTG, or a weighted CFG (wCFG), for a definition closer to conventional parsing. The **sw**-parsing problem hence generalizes the problem of searching for the best derivation tree of a wCFG  $G$  that yields a given input word  $w$ , with an infinite input alphabet instead of a finite one and transducer-defined distances instead of equality. It is however uncomparable to the related problems of *semiring parsing* [8], and *weighted parsing* [16].

In Section 5, we present results on **swVPT** and subclasses (automata construction and best-search algorithm) that can be applied for solving **sw**-parsing.

**Theorem 1.** *The problem of Symbolic Weighted Parsing can be solved in PTIME in the size of the input **swT**  $T$ , **swVPA**  $A$  and input word  $s$ , and the computation time of the functions and operators of the label theory.*

*Proof.* We follow an approach of *parsing as intersection* [9, ch. 13]. First, we associate to  $T$  and  $A$  a **swVPT** called  $(T \otimes A)$ , computing the product of the respective weights for the two models (Theorem 2): i.e.  $(T \otimes A)(u, t) = d_T(u, t) \otimes A(t)$ . Then, we construct a **swVPA** computing, for  $t \in \Delta^*$ ,  $(T \otimes A)(s, t) = d_T(s, t) \otimes A(t)$  (Theorem 3). Finally, with the algorithm of Theorem 4, we find a *best*  $t \in \Delta^*$  minimizing the latter value wrt  $\leq_\oplus$ , i.e. a solution of **sw**-parsing.  $\square$

## 5 Properties and Best-Search Algorithm

In the following results, we assume that the functions of a label theory  $\bar{\Phi}$  are given in a finite representation (e.g. Turing machine) in the definitions of **swVPT**, and provide complexity bounds parameterized by the semiring operators and the operators of Section 2 over the functions of  $\bar{\Phi}$ .

Similarly to **VPA** [2] and **sVPA** [5], the class of **swVPT** is closed under the binary operators of the underlying semiring.

**Proposition 1.** *Let  $T_1, T_2$  be two **swVPT** over the same  $\Delta$ , commutative  $\mathbb{S}$  and  $\bar{\Phi}$ . There exist two effectively constructible **swVPT**  $T_1 \oplus T_2$  and  $T_1 \otimes T_2$ , such that for every  $s \in \Sigma^*$  and  $t \in \Delta^*$ ,  $(T_1 \oplus T_2)(s, t) = T_1(s, t) \oplus T_2(s, t)$  and  $(T_1 \otimes T_2)(s, t) = T_1(s, t) \otimes T_2(s, t)$ .*

*Proof.* Classical Cartesian product construction, similar to the case of the Boolean semiring [5], see Appendix B for details.  $\square$

The following result shows how to compose, in a single **swVPT**, the two measures as input of **sw**-parsing: the **swT** computing input-output distance and the **swVPA** expressing the weight of parse trees' linearization.

**Theorem 2.** *Given a **swT**  $T$  over  $\Sigma, \Delta_i$ , commutative  $\mathbb{S}$ , and  $\bar{\Phi}$ , and a **swVPA**  $A$  over  $\Delta, \mathbb{S}, \bar{\Phi}$ , one can construct in *PTIME* a **swVPT**  $T \otimes A$ , over  $\Sigma, \Delta, \mathbb{S}, \bar{\Phi}$ , such that  $\forall s \in \Sigma^*, t \in \Delta^*, (T \otimes A)(s, t) = T(s, t|_{\Delta_i}) \otimes A(t)$ .*

*Proof.* The state set of  $T \otimes A$  is the Cartesian product of the state sets of  $T$  and  $A$ , and every transition of  $T \otimes A$  is either a transition of  $T$  or a transition of  $A$  of the same kind (in these cases the state of the other machine remains the same), or a product of two transitions  $w_{11}$  of  $T$  and  $A$ , see Appendix C.  $\square$

The next result is the construction, as a **swVPA**, for the partial application of a **swVPT**, fixing an input word  $s$  as its first argument.

**Theorem 3.** *Given a **swVPT**  $T$  over  $\Sigma, \Delta$ , commutative, complete and idempotent  $\mathbb{S}$ , and  $\bar{\Phi}$ , and given  $s \in \Sigma^*$ , there exists an effectively constructible **swVPA**  $T(s)$  over  $\Delta, \mathbb{S}$ , and  $\bar{\Phi}$ , such that for every  $t \in \Delta^*$ ,  $T(s)(t) = T(s, t)$ .*

*Proof.* (sketch, see Appendix D). We construct an automaton that simulates, while reading an output word  $t \in \Delta^*$ , the synchronized computation of  $T$  on  $s$  and  $t$ . The main difficulty comes from the transitions of  $T$  of the form  $w_{10}$ , which read in input  $s$  and ignore the output  $t$ . Since the automaton  $A(T)$  only reads the output word  $t$ , we add to  $A(T)$  a corresponding  $\varepsilon$ -transition, and show how to remove the  $\varepsilon$ -transitions from a **swVPA** while preserving its language.  $\square$

We present a procedure for searching a word of minimal weight for a **swVPA**  $A$ . First of all, for a complete semiring  $\mathbb{S}$ , we consider the following operators on the functions of a label theory  $\bar{\Phi}$ :

$$\begin{aligned} \bigoplus_{\Sigma} : \Phi_{\Sigma} &\rightarrow \mathbb{S}, \phi \mapsto \bigoplus_{a \in \Sigma} \phi(a) & \bigoplus_{\Sigma}^1 : \Phi_{\Sigma, \Delta} &\rightarrow \Phi_{\Delta}, \eta \mapsto (y \mapsto \bigoplus_{a \in \Sigma} \eta(a, y)) \\ \bigoplus_{\Delta}^2 : \Phi_{\Sigma, \Delta} &\rightarrow \Phi_{\Sigma}, \eta \mapsto (x \mapsto \bigoplus_{b \in \Delta} \eta(x, b)) \end{aligned}$$

Intuitively,  $\bigoplus_{\Sigma}$  returns the global minimum, wrt  $\leq_{\oplus}$ , of a function  $\phi$  of  $\Phi_{\Sigma}$ , and  $\bigoplus_{\Sigma}^1, \bigoplus_{\Delta}^2$  return partial minimums of a function  $\eta$  of  $\Phi_{\Sigma, \Delta}$ . A label theory is called *effective* when the three above operators applied on its functions are recursively enumerable, and there exists a function returning a witness symbol that reaches the minimum. In the complexity bounds, we assume a constant time evaluation for these operators. Effectiveness of label theories is a strong restriction, although realistic for the case study presented in this paper. It is satisfied e.g. by functions with a codomain  $\{0, \alpha\}$ , with  $\alpha <_{\oplus} 0$ , generalizing the boolean guards of [5,6] to *filters* returning null or constant weight values.

**Theorem 4.** *For a swVPA  $A$  over  $\Delta$ ,  $\mathbb{S}$  commutative, complete, bounded and total, and  $\bar{\Phi}$  effective, one can construct in PTIME a word  $t \in \Delta^*$  such that  $A(t)$  is minimal wrt the natural ordering  $\leq_{\oplus}$  for  $\mathbb{S}$ .*

*Proof.* Let  $A = \langle Q, P, \text{in}, \bar{w}, \text{out} \rangle$ . For every  $q, q' \in Q$ , let  $b_{\perp}(q, q')$  be the minimum, wrt  $\leq_{\oplus}$ , of the function  $\beta_{q, q'} : t \mapsto \text{weight}_A(q[\perp], \varepsilon, t, q'[\perp])$ . By definition of  $\leq_{\oplus}$ , and since  $\mathbb{S}$  is complete and total, it holds that:  $b_{\perp}(q, q') = \bigoplus_{t \in \Delta^*} \text{weight}_A(q[\perp], \varepsilon, t, q'[\perp])$  (see (1) for the definition of  $\text{weight}_A$ ). Following (2), and the algebraic properties of  $\otimes$  and  $\oplus$ , the minimum of  $A(t)$  is:

$$\bigoplus_{t \in \Delta^*} A(t) = \bigoplus_{t \in \Delta^*} \bigoplus_{q, q' \in Q} \text{in}(q) \otimes \beta_{q, q'}(t) \otimes \text{out}(q') = \bigoplus_{q, q' \in Q} \text{in}(q) \otimes b_{\perp}(q, q') \otimes \text{out}(q') \quad (6)$$

Hence, in order to prove Theorem 4, it is sufficient to show that for all  $q, q' \in Q$ , we can compute  $b_{\perp}(q, q')$  in PTIME. We proceed by searching for a shortest derivation in a  $\mathbb{S}$ -labeled hypergraph  $\mathcal{G}_A$  associated to the swVPA  $A$ , with set of vertices  $V_A = (Q \times \{\perp\} \times Q) \cup (Q \times \{\top\} \times Q)$ , where  $\top$  is a new symbol representing a non-empty stack, set of hyperedges  $E_A = (V_A \times V_A) \cup (V_A \times V_A \times V_A)$  and an edge labelling function  $\eta_A : E_A \rightarrow \mathbb{S}$  defined as follows, for  $q_0, q'_0, q_1, q_2, q_3 \in Q$ , ( $w_i$  is a new name for  $w_{01}$ , like in Appendix C, Definition 3):

$$\begin{aligned} \langle q_0, \perp, q_1 \rangle, \langle q'_0, \delta, q_2 \rangle &\mapsto 0 \quad \text{if } \delta = \top \text{ or } (\delta = \perp \text{ and } q'_0 \neq q_0) \\ \langle q_0, \perp, q_1 \rangle, \langle q_0, \perp, q_2 \rangle &\mapsto \bigoplus_{\Delta_i} w_i(q_1, q_2) \oplus \bigoplus_{\Delta_r} w_r^e(q_1, q_2) \\ \langle q_1, \top, q_2 \rangle, \langle q_0, \perp, q_3 \rangle &\mapsto \bigoplus_{p \in P} \bigoplus_{\Delta_c} [w_c(q_0, q_1, p) \otimes \bigoplus_{\Delta_r}^2 w_r(q_2, p, q_3)] \\ \langle q_1, \top, q_2 \rangle, \langle q_0, \top, q_3 \rangle &\mapsto \left[ \bigoplus_{q_1 = q_0} \bigoplus_{\Delta_i} w_i(q_2, q_3) \right] \oplus \\ &\quad \left[ \bigoplus_{p \in P} \bigoplus_{\Delta_c} [w_c(q_0, q_1, p) \otimes_2 \bigoplus_{\Delta_r}^2 w_r(q_2, p, q_3)] \right] \\ \langle q_0, \top, q_1 \rangle, \langle q_1, \top, q_2 \rangle, \langle q_0, \top, q_2 \rangle &\mapsto 1. \end{aligned}$$

Intuitively, a vertex  $v = \langle q, \perp, q' \rangle$  (resp.  $v = \langle q, \top, q' \rangle$ ) of  $\mathcal{G}_A$  represents computations of  $\mathcal{A}$  starting in state  $q$  with an empty stack (resp. non-empty stack  $\gamma$ ), and ending in state  $q'$  with an empty stack (resp. the same non-empty stack  $\gamma$ ). The best weight of such computations is the best cumulated weight of edges along a

derivation to  $v$ . More precisely, a *derivation* of  $\mathcal{G}_A$  is a  $V_A$ -labeled binary tree, and its weight is defined by ( $D_1$  and  $D_2$  are sub-derivations, and for  $i = 1, 2$ , the root of  $D_i$ , is labeled with  $v_i \in V_A$ , also denoted by  $v_i = \text{root}(D_i)$ ):

- $\text{weight}(\langle q, \perp, q \rangle) = \text{weight}(\langle q, \top, q \rangle) = \mathbb{1}$ ,
- $\text{weight}(\langle q, \perp, q' \rangle) = \text{weight}(\langle q, \top, q' \rangle) = \mathbb{0}$  if  $q \neq q'$ ,
- $\text{weight}(v(D_1)) = \text{weight}(D_1) \otimes \eta_A(v_1, v)$
- $\text{weight}(v(D_1, D_2)) = \text{weight}(D_1) \otimes \text{weight}(D_2) \otimes \eta_A(v_1, v_2, v)$ .

With  $\mathcal{D}(\mathcal{G}_A)$  denoting the set of derivations of  $\mathcal{G}_A$ , it holds (see Appendix E, Lemma 5) that for all  $q, q' \in Q$ ,  $b_\perp(q, q') = \bigoplus_{\substack{D \in \mathcal{D}(\mathcal{G}_A) \\ \text{root}(D) = \langle q, \perp, q' \rangle}} \text{weight}(D)$ .

Therefore, computing  $b_\perp(q, q')$  reduces to the search for a smallest weighted derivation of  $\mathcal{G}_A$  (wrt  $\leq_\oplus$ ) rooted with  $\langle q, \perp, q' \rangle$ , a problem solvable in PTIME [11], because  $\mathbb{S}$  is monotonic wrt  $\leq_\oplus$  and superior (Lemma 2). Therefore, by (6), the minimum of  $t \mapsto A(t)$ , wrt  $\leq_\oplus$ , can be computed in PTIME.

Moreover, a witness  $t \in \Delta^*$  for this minimum can be associated to the appropriate best derivation, with no additional cost. For details on the extraction of this witness, see Appendix E, the proof of Lemma 5, and Lemma 6.  $\square$

## Conclusion

We presented closure properties and one decision algorithm for three classes of Symbolic Weighted language models: **swVPT**, **swT** and **swVPA**, and applied these results to the problem of parsing with infinitely many input symbols (typically timed events). In our approach to parsing, words are compared by computing a distance between them, defined by a given **sw**-transducer, which allows to consider finer word relationships than strict equality.

The application to automated music transcription suggested in a toy example has been implemented in a C++ library [1], following the principles of the present **sw**-parsing framework, although some differences; e.g. the automata constructions are performed on-the-fly during best-search for efficiency reasons. One advantage of this **swVPA** approach is the *global view* provided by the stack during transcription, as opposed to other HMM-based approaches [19].

This work can be extended in several directions. The best-search algorithm for **swVPA** could be generalized from 1-best to  $n$ -best [12], and to  $k$ -closed semirings [14] (instead of *bounded*, which corresponds to 0-closed). One could also study the generalization of the best-search algorithm of Theorem 4 to the computation of the best possible output of a **swVPT** for a given input, or even to the more general models of [10].

Finally, the best-search algorithm presented here works offline, whereas an on-the-fly approach coupling automata construction and best-search would be interesting e.g. for online XML validation or filtering, or program monitoring [5].

## References

1. library qparse for music transcription. <https://qparse.gitlabpages.inria.fr>
2. Alur, R., Madhusudan, P.: Adding Nesting Structure to Words. *Journal of the ACM* **56**(3), 1–43 (2009)
3. Alur, R., Mamouras, K., Stanford, C.: Automata-Based Stream Processing. In: 44th International Colloquium on Automata, Languages, and Programming. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2017)
4. Caralp, M., Reynier, P.A., Talbot, J.M.: Visibly Pushdown Automata with Multiplicities: finiteness and k-boundedness. In: International Conference on Developments in Language Theory. pp. 226–238. Springer (2012)
5. D’Antoni, L., Alur, R.: Symbolic Visibly Pushdown Automata. In: International Conference on Computer Aided Verification. pp. 209–225. Springer (2014)
6. D’Antoni, L., Veanes, M.: The power of symbolic automata and transducers. In: International Conference on Computer Aided Verification. pp. 47–67. Springer (2017)
7. Droste, M., Kuich, W., Vogler, H.: Handbook of Weighted Automata. Springer Science & Business Media (2009)
8. Goodman, J.: Semiring Parsing. *Computational Linguistics* **25**(4), 573–606 (1999)
9. Grune, D., Jacobs, C.J.: Parsing Techniques. No. 2nd edition in Monographs in Computer Science, Springer (2008)
10. Herrmann, L., Vogler, H.: Weighted symbolic automata with data storage. In: International Conference on Developments in Language Theory. pp. 203–215. Springer (2016)
11. Huang, L.: Advanced dynamic programming in semiring and hypergraph frameworks. In: COLING (2008)
12. Huang, L., Chiang, D.: Better k-best parsing. In: Proceedings of the 9th International Workshop on Parsing Technology. pp. 53–64. Association for Computational Linguistics (2005)
13. Ma, M., Du, D., Liu, Y., Wang, Y., Li, Y.: Efficient Adversarial Sequence Generation for RNN with Symbolic Weighted Finite Automata. In: Proceedings of the Workshop on Artificial Intelligence Safety (SafeAI). vol. 3087 (2022)
14. Mohri, M.: Semiring frameworks and algorithms for shortest-distance problems. *Journal of Automata, Languages and Combinatorics* **7**(3), 321–350 (2002)
15. Mohri, M.: Edit-distance of weighted automata: General definitions and algorithms. *International Journal of Foundations of Computer Science* **14**(06), 957–982 (2003)
16. Mörbitz, R., Vogler, H.: Weighted parsing for grammar-based language models. In: Proceedings of the 14th International Conference on Finite-State Methods and Natural Language Processing. pp. 46–55 (2019)
17. Segoufin, L.: Automata and logics for words and trees over an infinite alphabet. In: *Computer Science Logic*. LNCS, vol. 4207. Springer (2006)
18. Selfridge-Field, E. (ed.): Beyond MIDI: the handbook of musical codes. MIT press (1997), <http://beyondmidi.ccarh.org/beyondmidi-600dpi.pdf>
19. Shibata, K., Nakamura, E., Yoshii, K.: Non-local musical statistics as guides for audio-to-score piano transcription. *Information Sciences* **566**, 262–280 (2021)
20. Waga, M.: Online quantitative timed pattern matching with semiring-valued weighted automata. In: International Conference on Formal Modeling and Analysis of Timed Systems. pp. 3–22. Springer (2019)
21. Yust, J.: Organized Time. Oxford University Press (2018)

## A Lemmata on Semirings

**Lemma 2 (Superiority, Boundedness).** *Let  $\langle \mathbb{S}, \oplus, 0, \otimes, 1 \rangle$  be an idempotent semiring. The two following statements are equivalent: (i) for all  $x, y \in \mathbb{S}$ ,  $x \leq_{\oplus} x \otimes y$  and  $y \leq_{\oplus} x \otimes y$  (ii) for all  $x \in \mathbb{S}$ ,  $1 \oplus x = 1$ .*

*Proof.* (ii)  $\Rightarrow$  (i) :  $x \oplus (x \otimes y) = x \otimes (1 \oplus y) = x$ , by distributivity of  $\otimes$  over  $\oplus$ . Hence  $x \leq_{\oplus} x \otimes y$ . Similarly,  $y \oplus (x \otimes y) = (1 \oplus x) \otimes y = y$ , hence  $y \leq_{\oplus} x \otimes y$ . (i)  $\Rightarrow$  (ii) : by the second inequality of (i), with  $y = 1$ ,  $1 \leq_{\oplus} x \otimes 1 = x$ , i.e., by definition of  $\leq_{\oplus}$ ,  $1 \oplus x = 1$ .  $\square$

**Lemma 3 ([14], Lemma 3).** *Every bounded semiring is idempotent.*

*Proof.* By boundedness,  $1 \oplus 1 = 1$ , and idempotency follows by multiplying both sides by  $x$  and distributing.  $\square$

## B Proof of Proposition 1

**Proposition 1.** *Let  $T_1, T_2$  be two swVPT over the same  $\Delta$ , commutative  $\mathbb{S}$  and  $\bar{\Phi}$ . There exist two effectively constructible swVPT  $T_1 \oplus T_2$  and  $T_1 \otimes T_2$ , such that for every  $s \in \Sigma^*$  and  $t \in \Delta^*$ ,  $(T_1 \oplus T_2)(s, t) = T_1(s, t) \oplus T_2(s, t)$  and  $(T_1 \otimes T_2)(s, t) = T_1(s, t) \otimes T_2(s, t)$ .*

*Proof.* We prove the closure under  $\otimes$  (the case of  $\oplus$  is similar).

Let  $T_1 = \langle Q_1, P_1, \text{in}_1, \bar{w}_1, \text{out}_1 \rangle$  and  $T_2 = \langle Q_2, P_2, \text{in}_2, \bar{w}_2, \text{out}_2 \rangle$ . We can build a swVPT  $T_1 \otimes T_2$  by a classical product construction. We define  $T_1 \otimes T_2 = \langle Q, P, \text{in}, \bar{w}, \text{out} \rangle$ , where  $Q = Q_1 \times Q_2$  is the set of states, and  $P = P_1 \times P_2$  is an auxiliary set of stack symbols. The state entering and leaving functions  $\text{in}$ ,  $\text{out}$  and the tuple of transition functions  $\bar{w}$  are defined using the label-theory operators of Section 2 as follows, for all  $\langle q_1, q_2 \rangle, \langle q'_1, q'_2 \rangle \in Q$  and  $\langle p_1, p_2 \rangle, \langle p'_1, p'_2 \rangle \in P$ :

$$\begin{aligned} \text{in}(\langle q_1, q_2 \rangle) &= \text{in}_1(q_1) \otimes \text{in}_2(q_2) & \text{out}(\langle q_1, q_2 \rangle) &= \text{out}_1(q_1) \otimes \text{out}_2(q_2) \\ \mathbf{w}_{10}(\langle q_1, q_2 \rangle, \langle q'_1, q'_2 \rangle) &= \mathbf{w}_{10,1}(q_1, q'_1) \otimes \mathbf{w}_{10,2}(q_2, q'_2) \\ \mathbf{w}_{01}(\langle q_1, q_2 \rangle, \langle q'_1, q'_2 \rangle) &= \mathbf{w}_{01,1}(q_1, q'_1) \otimes \mathbf{w}_{01,2}(q_2, q'_2) \\ \mathbf{w}_{11}(\langle q_1, q_2 \rangle, \langle q'_1, q'_2 \rangle) &= \mathbf{w}_{11,1}(q_1, q'_1) \otimes \mathbf{w}_{11,2}(q_2, q'_2) \\ \mathbf{w}_c(\langle q_1, q_2 \rangle, \langle q'_1, q'_2 \rangle, \langle p_1, p_2 \rangle) &= \mathbf{w}_{c,1}(q_1, q'_1, p_1) \otimes \mathbf{w}_{c,2}(q_2, q'_2, p_2) \\ \mathbf{w}_r(\langle q_1, q_2 \rangle, \langle p_1, p_2 \rangle, \langle q'_1, q'_2 \rangle) &= \mathbf{w}_{r,1}(q_1, p_1, q'_1) \otimes \mathbf{w}_{r,2}(q_2, p_2, q'_2) \\ \mathbf{w}_r^e(\langle q_1, q_2 \rangle, \langle q'_1, q'_2 \rangle) &= \mathbf{w}_{r,1}^e(q_1, q'_1) \otimes \mathbf{w}_{r,2}^e(q_2, q'_2) \end{aligned}$$

With these functions,  $T$  simulates the synchronized behaviour of  $T_1$  and  $T_2$ .  $\square$

## C Proof of Theorem 2

Before giving the details of the construction for the proof of Theorem 2, let us first state explicitly the definitions of the classes swT and swVPA.

**Definition 2 (swT).** A Symbolic Weighted Transducer over  $\Sigma$ ,  $\Delta_i$ ,  $\mathbb{S}$ , and  $\bar{\Phi}$  is a tuple  $T = \langle Q, \text{in}, \bar{w}, \text{out} \rangle$ , where  $Q$  is a finite set of states,  $\text{in} : Q \rightarrow \mathbb{S}$  (respectively  $\text{out} : Q \rightarrow \mathbb{S}$ ) are functions defining the weight for entering (respectively leaving) a state, and  $\bar{w}$  is composed of the transition functions :  $w_{10} : Q \times Q \rightarrow \Phi_e$ ,  $w_{01} : Q \times Q \rightarrow \Phi_i$ ,  $w_{11} : Q \times Q \rightarrow \Phi_{ei}$ .

We use the same extended notation for the transition functions  $w_{10}$ ,  $w_{01}$ ,  $w_{11}$  as in Section 3, Definition 1 and follow the definition of the computed weight in (1) (4 first equations) and (2).

**Definition 3 (swVPA).** A Symbolic Weighted Visibly Pushdown Automaton over  $\Delta = \Delta_i \uplus \Delta_c \uplus \Delta_r$ ,  $\mathbb{S}$  and  $\bar{\Phi}$  is a tuple  $A = \langle Q, P, \text{in}, \bar{w}, \text{out} \rangle$ , where  $Q$  is a finite set of states,  $P$  is a finite set of stack symbols,  $\text{in} : Q \rightarrow \mathbb{S}$  (respectively  $\text{out} : Q \rightarrow \mathbb{S}$ ) are functions defining the weight for entering (respectively leaving) a state, and  $\bar{w}$  is a tuple composed of the transition functions :  $w_i : Q \times Q \rightarrow \Phi_i$ ,  $w_c : Q \times Q \times P \rightarrow \Phi_c$ ,  $w_r : Q \times P \times Q \rightarrow \Phi_{cr}$ ,  $w_r^e : Q \times Q \rightarrow \Phi_r$ .

The transition function  $w_i$  is just a new name for  $w_{01}$  of Definition 1, used in the case of swVPA. In the extended notation for the transition functions after Definition 1, and the definition of the computed weight in (1) and (2), there is no input symbol for swVPA, hence for the sake of simplicity, let us restate explicitly the equations as follows.

For the transition functions in extended notation, we have with  $q, q' \in Q$  and  $a \in \Delta_i$ ,  $c \in \Delta_c$ ,  $r \in \Delta_r$ :

$$\begin{aligned} w_i(q, a, q') &= \phi(a) \quad \text{where } \phi = w_i(q, q'), \\ w_c(q, c, q', p) &= \phi(c) \quad \text{where } \phi = w_c(q, q', p), \\ w_r(q, c, p, r, q') &= \eta(c, r) \quad \text{where } \eta = w_r(q, p, q'), \\ w_r^e(q, r, q') &= \phi(r) \quad \text{where } \phi = w_r^e(q, q'). \end{aligned}$$

And for the weight function, with  $v \in \Delta^*$ :

$$\begin{aligned} \text{weight}_A(q[\gamma], \varepsilon, q'[\gamma']) &= \mathbb{1} \text{ if } q = q', \gamma = \gamma' \text{ and } \mathbb{0} \text{ otherwise} \quad (7) \\ \text{weight}_A(q[\gamma], a v, q'[\gamma']) &= \bigoplus_{q'' \in Q} w_i(q, a, q'') \otimes \text{weight}_A(q''[\gamma], v, q'[\gamma']) \\ \text{weight}_A(q[\gamma], c v, q'[\gamma']) &= \bigoplus_{\substack{q'' \in Q \\ p \in P}} w_c(q, c, q'', p) \otimes \text{weight}_A(q'' \left[ \begin{smallmatrix} \langle c, p \rangle \\ \gamma \end{smallmatrix} \right], v, q'[\gamma']) \\ \text{weight}_A(q \left[ \begin{smallmatrix} \langle c, p \rangle \\ \gamma \end{smallmatrix} \right], r v, q'[\gamma']) &= \bigoplus_{q'' \in Q} w_r(q, c, p, r, q'') \otimes \text{weight}_A(q''[\gamma], v, q'[\gamma']) \\ \text{weight}_A(q[\perp], r v, q'[\gamma']) &= \bigoplus_{q'' \in Q} w_r^e(q, r, q'') \otimes \text{weight}_A(q''[\perp], v, q'[\gamma']) \end{aligned}$$

and, for  $t \in \Delta^*$ :

$$A(t) = \bigoplus_{q, q' \in Q} \text{in}(q) \otimes \text{weight}_A(q[\perp], t, q'[\perp]) \otimes \text{out}(q') \quad (8)$$

We recall the Theorem 2 from Section 5.

**Theorem 2.** *Given a  $swTT$  over  $\Sigma$ ,  $\Delta_i$ , commutative  $\mathbb{S}$ , and  $\bar{\Phi}$ , and a  $swVPA$   $A$  over  $\Delta$ ,  $\mathbb{S}$ ,  $\bar{\Phi}$ , one can construct in  $PTIME$  a  $swVPT$   $T \otimes A$ , over  $\Sigma$ ,  $\Delta$ ,  $\mathbb{S}$ ,  $\bar{\Phi}$ , such that  $\forall s \in \Sigma^*, t \in \Delta^*, (T \otimes A)(s, t) = T(s, t|_{\Delta_i}) \otimes A(t)$ .*

*Proof.* Let  $T = \langle Q_T, \text{in}_T, \bar{w}_T, \text{out}_T \rangle$ , where  $\bar{w}_T$  contains  $w_{10}$ ,  $w_{01}$ , and  $w_{11}$ , and let  $A = \langle Q_A, \text{in}_A, \bar{w}_A, \text{out}_A \rangle$  where  $\bar{w}_A$  contains  $w_i$ ,  $w_c$ ,  $w_r$ ,  $w_r^e$ .

The set of states of  $T \otimes A$  will be  $Q' = Q_T \times Q_A$ , and its set of stack symbols  $P' = P$ . The entering, leaving and transition functions will simulate the synchronized computations of  $T$  and  $A$  on respectively  $\langle s, t|_{\Delta_i} \rangle$  and  $t$ , while reading a pair  $\langle s, t \rangle \in \Sigma^* \times \Delta^*$ . The state entering and leaving functions of  $T \otimes A$  are defined, for all  $\langle q_T, q_A \rangle \in Q'$ , by:

$$\begin{aligned} \text{in}'(\langle q_T, q_A \rangle) &= \text{in}_T(q_T) \otimes \text{in}_A(q_A) \\ \text{out}'(\langle q_T, q_A \rangle) &= \text{out}_T(q_T) \otimes \text{out}_A(q_A) \end{aligned}$$

The transition functions of  $T \otimes A$  are defined by:

$$\begin{aligned} w'_{10}(\langle q_T, q_A \rangle, \langle q'_T, q'_A \rangle) &= w_{10}(q_T, q'_T) && \text{when } q_A = q'_A \\ w'_{10}(\langle q_T, q_A \rangle, \langle q'_T, q'_A \rangle) &= 0 && \text{otherwise} \\ w'_{01}(\langle q_T, q_A \rangle, \langle q'_T, q'_A \rangle) &= w_i(q_A, q'_A) && \text{when } q_T = q'_T \\ w'_{01}(\langle q_T, q_A \rangle, \langle q'_T, q'_A \rangle) &= 0 && \text{otherwise} \\ w'_{11}(\langle q_T, q_A \rangle, \langle q'_T, q'_A \rangle) &= w_{11}(q_T, q'_T) \otimes_2 w_i(q_A, q'_A) \\ w'_c(\langle q_T, q_A \rangle, \langle q'_T, q'_A \rangle, p) &= w_c(q_A, q'_A, p) && \text{when } q_T = q'_T \\ w'_c(\langle q_T, q_A \rangle, \langle q'_T, q'_A \rangle, p) &= 0 && \text{otherwise} \\ w'_r(\langle q_T, q_A \rangle, p, \langle q'_T, q'_A \rangle) &= w_r(q_A, p, q'_A) && \text{when } q_T = q'_T \\ w'_r(\langle q_T, q_A \rangle, p, \langle q'_T, q'_A \rangle) &= 0 && \text{otherwise} \\ w'^e_r(\langle q_T, q_A \rangle, \langle q'_T, q'_A \rangle) &= w^e_r(q_A, q'_A) && \text{when } q_T = q'_T \\ w'^e_r(\langle q_T, q_A \rangle, \langle q'_T, q'_A \rangle) &= 0 && \text{otherwise} \end{aligned}$$

It means that, for all  $e \in \Sigma$ ,  $a \in \Delta_i$ ,  $c \in \Delta_c$ ,  $r \in \Delta_r$ :

$$\begin{aligned} w'_{10}(\langle q_T, q_A \rangle, e, \varepsilon, \langle q'_T, q'_A \rangle) &= \phi(e) && \text{when } q_A = q'_A \text{ where } \phi = w_{10}(q_T, q'_T) \\ w'_{10}(\langle q_T, q_A \rangle, e, \varepsilon, \langle q'_T, q'_A \rangle) &= 0 && \text{otherwise} \\ w'_{01}(\langle q_T, q_A \rangle, \varepsilon, a, \langle q'_T, q'_A \rangle) &= \phi(a) && \text{when } q_T = q'_T \text{ where } \phi = w_i(q_A, q'_A), \\ w'_{01}(\langle q_T, q_A \rangle, \varepsilon, a, \langle q'_T, q'_A \rangle) &= 0 && \text{otherwise} \\ w'_{11}(\langle q_T, q_A \rangle, e, a, \langle q'_T, q'_A \rangle) &= \eta(e, a) \otimes \phi(a) && \text{where } \eta = w_{11}(q_T, q'_T), \phi = w_i(q_A, q'_A) \\ w'_c(\langle q_T, q_A \rangle, \varepsilon, c, \langle q'_T, q'_A \rangle, p) &= \phi(c) && \text{when } q_T = q'_T \text{ where } \phi = w_c(q_A, q'_A, p), \\ w'_c(\langle q_T, q_A \rangle, \varepsilon, c, \langle q'_T, q'_A \rangle, p) &= 0 && \text{otherwise} \\ w'_r(\langle q_T, q_A \rangle, c, p, \varepsilon, r, \langle q'_T, q'_A \rangle) &= \eta(c, r) && \text{when } q_T = q'_T \text{ where } \eta = w_r(q_A, p, q'_A) \\ w'_r(\langle q_T, q_A \rangle, c, p, \varepsilon, r, \langle q'_T, q'_A \rangle) &= 0 && \text{otherwise} \\ w'^e_r(\langle q_T, q_A \rangle, \varepsilon, r, \langle q'_T, q'_A \rangle) &= \phi(r) && \text{when } q_T = q'_T \text{ where } \phi = w^e_r(q_A, q'_A) \\ w'^e_r(\langle q_T, q_A \rangle, \varepsilon, r, \langle q'_T, q'_A \rangle) &= 0 && \text{otherwise} \end{aligned}$$

The proof of the correctness of the construction, i.e. that  $\forall s \in \Sigma^*, t \in \Delta^*, (T \otimes A)(s, t) = T(s, t|_{\Delta_i}) \otimes A(t)$ , is a straightforward double induction on the length of  $s$  and  $t$ .  $\square$



## D Proof of Theorem 3

The  $\text{swVPT}$  of Definition 1 do not contain  $\varepsilon$ -transitions. However, this notion shall be convenient in the proof Theorem 3. It is defined formally as follows.

**Definition 4 ( $\text{swVPT}_\varepsilon$ ).** A Symbolic Weighted Visibly Pushdown Transducer with  $\varepsilon$ -transitions over  $\Delta$ , complete  $\mathbb{S}$ , and  $\bar{\Phi}$  is a tuple  $T = \langle Q, P, \text{in}, \bar{w}, \text{out} \rangle$ , where  $Q$ ,  $P$ ,  $\text{in}$  and  $\text{out}$  are like in Definition 1 and  $\bar{w}$  contains an additional function  $w_{00} : Q \times Q \rightarrow \mathbb{S}$ .

The function  $\text{weight}$  of a  $\text{swVPT}_\varepsilon$  is computed by adding, with  $\oplus$ , the weight of possible finite sequences  $\varepsilon$ -transitions. Formally, for a  $\text{swVPT}_\varepsilon$   $T$ , let  $\text{weight}_T^\varepsilon$  be the function defined for  $T$  by the equations (1) (for the case of  $\text{swVPT}$  without  $\varepsilon$ -transitions). Then  $\text{weight}_T$  is the function  $[Q \times \Gamma^*] \times \Sigma^* \times \Delta^* \times [Q \times \Gamma^*]$  into  $\mathbb{S}$ , defined by, for  $q, q' \in Q$ ,  $\gamma, \gamma' \in \Gamma^*$ , and  $u \in \Sigma^*$ ,  $v \in \Delta^*$ :

$$\text{weight}_T(q[\gamma], u, v, q'[\gamma']) = \bigoplus_{\substack{q_0 \dots q_n \in Q^* \\ q_0 = q, q_n = q'}} \bigotimes_{i=0}^{n-1} w_{00}(q_i, q_{i+1}) \otimes \text{weight}_T^\varepsilon(q_n[\gamma], u, v, q'[\gamma']) \quad (9)$$

The hypothesis that  $\mathbb{S}$  is complete ensures that the above (possibly infinite) sum is well defined.

**Lemma 4.** For all  $\text{swVPT}_\varepsilon$   $T_\varepsilon$  over  $\Sigma$ ,  $\Delta$ , commutative, idempotent, and complete  $\mathbb{S}$ , and  $\bar{\Phi}$ , there exists one  $\text{swVPT}$   $T$  over  $\Sigma$ ,  $\Delta$ ,  $\mathbb{S}$ , and  $\bar{\Phi}$ , of size polynomial in the size of  $T_\varepsilon$  and effectively constructible in PTIME in the size of  $T_\varepsilon$ , such that for all  $\langle s, t \rangle \in \Sigma^* \times \Delta^*$ ,  $T(s, t) = T_\varepsilon(s, t)$ .

*Proof.* Let  $T_\varepsilon = \langle Q, P, \text{in}, \bar{w}, \text{out} \rangle$ . We build  $T = \langle Q, P, \text{in}, \bar{w}', \text{out}' \rangle$ , the construction of  $\bar{w}'$  and  $\text{out}'$  following the line of the  $\varepsilon$ -removal algorithm of [App24].

For all  $q, q' \in Q$ , let

$$\ell_{00}(q, q') = \bigoplus_{\substack{q_0 \dots q_n \in Q^* \\ q_0 = q, q_n = q'}} \bigotimes_{i=0}^{n-1} w_{00}(q_i, q_{i+1})$$

Since by hypothesis,  $\mathbb{S}$  is commutative and idempotent, it holds that:

**Fact 1.** For all  $q, q' \in Q$ , there exists one sequence  $q_0 \dots q_n \in Q^*$  without repetition, such that  $q_0 = q$ ,  $q_n = q'$ , and  $\ell_{00}(q, q') = \bigotimes_{i=0}^{n-1} w_{00}(q_i, q_{i+1})$ .

Therefore, we can pre-compute every  $\ell_{00}(q, q')$  in at most  $|Q|$  iterations, Viterbi algorithm [11] for finding a shortest path in the graph defined by  $w_{00}$ .

Let, for all  $q' \in Q$ ,

$$\text{out}'(q) = \ell_{00}(q, q') \otimes \text{out}'(q')$$

and, for all  $q, q' \in Q$ ,

$$w'_{10}(q, q') = \bigoplus_{q'' \in Q} \ell_{00}(q, q'') \otimes w'_{10}(q'', q')$$

and similarly for  $w_{01}, w_{11}, w_c, w_r, w_r^e$ .

$$\text{Fact 1, implies that: } \text{out}'(q) = \bigoplus_{\substack{q_0 \dots q_n \in Q^* \\ q_0 = q}} \bigotimes_{i=0}^{n-1} w_{00}(q_i, q_{i+1}) \otimes \text{out}(q_n)$$

$$\text{and } w'_{10}(q, q') = \bigoplus_{\substack{q_0 \dots q_n \in Q^* \\ q_0 = q}} \bigotimes_{i=0}^{n-1} w_{00}(q_i, q_{i+1}) \otimes w_{10}(q_n, q').$$

By (9), it follows that  $T(s, t) = T_\varepsilon(s, t)$  for all  $\langle s, t \rangle \in \Sigma^* \times \Delta^*$ .  $\square$

**Theorem 3.** *Given a swVPT  $T$  over  $\Sigma, \Delta$ , commutative, complete and idempotent  $\mathbb{S}$ , and  $\bar{\Phi}$ , and given  $s \in \Sigma^*$ , there exists an effectively constructible swVPA  $T(s)$  over  $\Delta, \mathbb{S}$ , and  $\bar{\Phi}$ , such that for every  $t \in \Delta^*$ ,  $T(s)(t) = T(s, t)$ .*

*Proof.* Let  $T = \langle Q, P, \text{in}, \bar{w}, \text{out} \rangle$ , where  $\bar{w}$  contains  $w_{10}, w_{01}$ , and  $w_{11}$ , from  $Q \times Q$  into respectively  $\bar{\Phi}_e, \bar{\Phi}_i$ , and  $\bar{\Phi}_{ei}$ , and  $w_c : Q \times Q \times P_T \rightarrow \bar{\Phi}_c, w_r : Q \times P_T \times Q \rightarrow \bar{\Phi}_{cr}, w_r^e : Q \times Q \rightarrow \bar{\Phi}_r$  and let  $s = e_1 \dots e_k$ .

Let us construct a swVPA with  $\varepsilon$ -transitions  $T_\varepsilon(s) = \langle Q', P', \text{in}', \bar{w}', \text{out}' \rangle$ , with a state set  $Q' = [0..k] \times Q$ , a set of stack symbols  $P' = P$ . The functions  $\text{in}', \text{out}'$  and  $\bar{w}'$ , will simulate the synchronized computation of  $T$  on  $\langle s, t \rangle$ , while reading an output word  $t \in \Delta^*$ .

The state entering function of  $T_\varepsilon(s)$  is defined by, for all  $q \in Q$ :

$$\begin{aligned} \text{in}'(\langle 0, q \rangle) &= \text{in}(q) \\ \text{in}'(\langle i, q \rangle) &= \emptyset \quad \text{for } 0 < i \leq k \end{aligned}$$

and the state leaving function is defined by, for all  $q \in Q$ :

$$\begin{aligned} \text{out}'(\langle k, q \rangle) &= \text{out}(q) \\ \text{out}'(\langle i, q \rangle) &= \emptyset \quad \text{for } 0 \leq i < k. \end{aligned}$$

Regarding transition functions of  $\bar{w}'$ , for all  $q, q' \in Q$ ,

$$\begin{aligned} w_i'(\langle i, q \rangle, \langle i, q' \rangle) &= w_{01}(q, q') & \text{for } 0 \leq i \leq k \\ w_i'(\langle i, q \rangle, \langle i+1, q' \rangle) &: y \mapsto w_{11}(q, e_i, y, q') & \text{for } 0 \leq i < k \\ w_i'(\langle i, q \rangle, \langle i', q' \rangle) &= \emptyset & \text{for } 0 \leq i, i' \leq k, i' \neq i, i' \neq i+1. \end{aligned}$$

The  $\varepsilon$ -transitions of  $T_\varepsilon(s)$  are, for all  $q, q' \in Q$ ,

$$\begin{aligned} w_{00}(\langle i, q \rangle, \langle i+1, q' \rangle) &= w_{10}(q, e_i, \varepsilon, q') & \text{for } 0 \leq i < k \\ w_{00}(\langle i, q \rangle, \langle i', q' \rangle) &= \emptyset & \text{for } 0 \leq i, i' \leq k, i' \neq i+1. \end{aligned}$$

And the other transitions of  $T_\varepsilon(s)$  are, for all  $q, q' \in Q, p \in P$ ,

$$\begin{aligned} w_c'(\langle i, q \rangle, \langle i, q' \rangle, p) &= w_c(q, q') & \text{for } 0 \leq i \leq k \\ w_c'(\langle i, q \rangle, \langle i', q' \rangle, p) &= \emptyset & \text{for } 0 \leq i, i' \leq k, i' \neq i \\ w_r'(\langle i, q \rangle, p, \langle i, q' \rangle) &= w_r(q, p, q') & \text{for } 0 \leq i \leq k \\ w_r'(\langle i, q \rangle, p, \langle i', q' \rangle) &= \emptyset & \text{for } 0 \leq i, i' \leq k, i' \neq i \\ w_r^{e'}(\langle i, q \rangle, \langle i, q' \rangle) &= w_r^e(q, q') & \text{for } 0 \leq i \leq k \\ w_r^{e'}(\langle i, q \rangle, \langle i', q' \rangle) &= \emptyset & \text{for } 0 \leq i, i' \leq k, i' \neq i \end{aligned}$$

We can show that for all  $t \in \Delta^*$ ,  $T_\varepsilon(s)(t) = T(s, t)$ . Hence Theorem 3 follows, by Lemma 4.  $\square$

## E End of proof of Theorem 4

Lemma 5 below shows that the computation of  $b_\perp$  suffices to the search of a shortest path in the graph  $\mathcal{G}_A$  constructed in Section 5, and by extension the computation of the minimum of  $A$  over  $\Delta^*$ .

**Lemma 5.** *For all  $q, q' \in Q$ ,  $b_\perp(q, q') = \bigoplus_{\substack{\pi \in V_A^* \\ \text{last}(\pi) = \langle q, \perp, q' \rangle}} \text{weight}(\pi)$ .*

The direction  $\leq_\oplus$  of Lemma 5 follows from Lemma 6 below. In the following, we use the notation of Appendix C, after Definition 3, for the weights of computations of **swVPA**, as particular cases of **swVPT**, i.e. the argument in (1) corresponding to an input symbol of  $\Sigma$  (for a **swVPT**) is ignored. Note the use of the special symbol  $\top$  in configurations  $q[\top]$  in the expressions of  $\text{weight}_A$ . With such a symbol for  $\gamma$  in (1), we ensure that the computation (of  $\text{weight}_A$ ) starts with a non-empty stack and never read or pop the top of this stack.

**Lemma 6 (Correctness).** *For all paths  $\pi$  of  $\mathcal{G}_A$  such that  $\text{weight}(\pi) \neq 0$  and  $\text{last}(\pi) = \langle q, \gamma, q' \rangle$ , where  $\gamma \in \{\perp, \top\}$  and  $q, q' \in Q$ , there exists  $t \in \Delta^*$  such that  $\text{weight}_A(q[\gamma], t, q'[\gamma]) = \text{weight}(\pi)$ .*

*Proof.* By induction on the length of a path  $\pi$ .

The base case is  $\pi$  composed of a single vertex of the form  $\langle q, \perp, q \rangle$  or  $\langle q, \top, q \rangle$ , in order to ensure that  $\text{weight}(\pi) \neq 0$ . In both cases,  $\text{weight}(\pi) = \mathbb{1}$ , and by (1),  $\text{weight}_A(q[\perp], \varepsilon, q'[\perp]) = \text{weight}_A(q[\top], \varepsilon, q'[\top]) = \mathbb{1}$ . Hence the property holds with  $t = \varepsilon$ .

If  $\pi = v_0, \dots, v_n$  with  $n \geq 1$ , let us assume that Lemma 6 holds for  $\pi' = v_0, \dots, v_{n-1}$ , and a word  $t' \in \Delta^*$ . We do a case analysis on the edge  $v_{n-1} \rightarrow v_n$ .

Firstly, let us consider the case where  $v_{n-1} = \langle q_0, \perp, q_1 \rangle$  and  $v_n = \langle q_0, \perp, q_2 \rangle$  for some  $q_0, q_1, q_2 \in Q$ . By the hypothesis that  $\mathbb{S}$  is total, we are in one of the following two cases:

- $\eta_A(v_{n-1} \rightarrow v_n) = \bigoplus_{\Delta_i} w_i(q_1, q_2)$ . By effectiveness of  $\bar{\Phi}$ , there exists  $a \in \Delta_i$  such that  $\bigoplus_{\Delta_i} w_i(q_1, q_2) = w_i(q_1, a, q_2)$ . It follows that:

$$\begin{aligned} \text{weight}(\pi) &= \text{weight}(\pi') \otimes \eta_A(v_{n-1} \rightarrow v_n) \\ &= \text{weight}(\pi') \otimes w_i(q_1, a, q_2) \\ &= \text{weight}_A(q_0[\perp], t', q_1[\perp]) \otimes w_i(q_1, a, q_2) \text{ by induction hypothesis} \\ &= \text{weight}_A(q_0[\perp], t'a, q_2[\perp]) \\ &\quad \text{by (1) and associativity, commutativity of } \otimes \end{aligned}$$

and (i) holds with  $t = t'a$ .

- $\eta_A(v_{n-1} \rightarrow v_n) = \bigoplus_{\Delta_r} \mathbf{w}_r^e(q_1, q_2)$ , and we can proceed similarly as above in order to find  $t = t'r$  as expected, for some  $r \in \Delta_r$  (case of an unmatched return symbol).

Secondly, we consider the case where  $v_{n-1} = \langle q_1, \top, q_2 \rangle$  and  $v_n = \langle q_0, \perp, q_3 \rangle$  for  $q_0, q_1, q_2, q_3 \in Q$ . In this case,  $\eta_A(v_{n-1} \rightarrow v_n) = \bigoplus_{p \in P} \bigoplus_{\Delta_c} [\mathbf{w}_c(q_0, q_1, p) \otimes \bigoplus_{\Delta_r}^2 \mathbf{w}_r(q_2, p, q_3)]$ . By hypothesis, this value is not  $\emptyset$ , hence there exists a stack symbol  $p \in P$ , a call symbol  $c \in \Delta_c$ , and a return symbol  $r \in \Delta_r$  such that  $\eta_A(v_{n-1} \rightarrow v_n) = \mathbf{w}_c(q_0, c, q_1, p) \otimes \mathbf{w}_r(q_2, c, p, r, q_3)$ . Therefore,

$$\begin{aligned} \text{weight}(\pi) &= \text{weight}(\pi') \otimes \eta_A(v_{n-1} \rightarrow v_n) \\ &= \text{weight}(\pi') \otimes \mathbf{w}_c(q_0, c, q_1, p) \otimes \mathbf{w}_r(q_2, c, p, r, q_3) \\ &= \text{weight}_A(q_1[\top], t', q_2[\top]) \otimes \mathbf{w}_c(q_0, c, q_1, p) \otimes \mathbf{w}_r(q_2, c, p, r, q_3) \text{ by ind. hyp.} \\ &= \text{weight}_A(q_0[\perp], ct'r, q_3[\perp]). \end{aligned}$$

Hence, we can conclude with  $t = ct'r$ .

Finally, let us consider the case where  $v_{n-1} = \langle q_1, \top, q_2 \rangle$  and  $v_n = \langle q_0, \top, q_3 \rangle$  for  $q_0, q_1, q_2, q_3 \in Q$ . Since  $\mathbb{S}$  is total, there are two cases for the value of  $\eta_A(v_{n-1} \rightarrow v_n)$ .

- $\eta_A(v_{n-1} \rightarrow v_n) = \bigoplus_{\Delta_i} \mathbf{w}_i(q_2, q_3)$  and  $q_1 = q_0$ . By effectiveness of  $\bar{\Phi}$ , there exists  $a \in \Delta_i$  such that  $\bigoplus_{\Delta_i} \mathbf{w}_i(q_2, q_3) = \mathbf{w}_{01}(q_2, \varepsilon, a, q_3) = \mathbf{w}_i(q_2, a, q_3)$ , and

$$\begin{aligned} \text{weight}(\pi) &= \text{weight}(\pi') \otimes \eta_A(v_{n-1} \rightarrow v_n) \\ &= \text{weight}(\pi') \otimes \mathbf{w}_i(q_2, a, q_3) \\ &= \text{weight}_A(q_0[\top], t', q_2[\top]) \otimes \mathbf{w}_i(q_2, a, q_3) \text{ by induction hypothesis} \\ &= \text{weight}_A(q_0[\top], t'a, q_3[\top]) \end{aligned}$$

and the lemma holds with  $t = t'a$ .

- $\eta_A(v_{n-1} \rightarrow v_n) = \bigoplus_{p \in P} \bigoplus_{\Delta_c} [\mathbf{w}_c(q_0, q_1, p) \otimes \bigoplus_{\Delta_r}^2 \mathbf{w}_r(q_2, p, q_3)]$ . Since this value is not  $\emptyset$  by hypothesis, there exists a stack symbol  $p \in P$ , a call symbol  $c \in \Delta_c$ , and a return symbol  $r \in \Delta_r$  such that  $\eta_A(v_{n-1} \rightarrow v_n) = \mathbf{w}_c(q_0, c, q_1, p) \otimes \mathbf{w}_r(q_2, c, p, r, q_3)$ , and,

$$\begin{aligned} \text{weight}(\pi) &= \text{weight}(\pi') \otimes \eta_A(v_{n-1} \rightarrow v_n) \\ &= \text{weight}(\pi') \otimes \mathbf{w}_c(q_0, c, q_1, p) \otimes \mathbf{w}_r(q_2, c, p, r, q_3) \\ &= \text{weight}_A(q_1[\top], t', q_2[\top]) \otimes \mathbf{w}_c(q_0, c, q_1, p) \otimes \mathbf{w}_r(q_2, c, p, r, q_3) \\ &\quad \text{by induction hypothesis} \\ &= \text{weight}_A(q_0[\top], ct'r, q_3[\top]) \\ &\quad \text{by (1) and associativity, commutativity of } \otimes, \end{aligned}$$

and the lemma holds with  $t = ct'r$ .  $\square$

The direction  $\geq_\oplus$  of Lemma 5 follows from the Lemma 7 below. In this lemma, we call *safe* path a path  $\pi$  such that  $\text{fst}(\pi)$  has the form  $\langle q, \perp, q \rangle$  or  $\langle q, \top, q \rangle$  for  $q \in Q$ . Moreover, a word  $t \in \Delta^*$  is *well-parenthesised* if it is either the empty word  $\varepsilon$ , or of the form  $t = t'a$ , for  $a \in \Delta_i$ , or  $t = t'r$  or  $t = ct'r$  for some well-parenthesised word  $t'$ .

**Lemma 7 (Completeness).** *For all well-parenthesised  $t \in \Delta^*$ , for all  $q, q' \in Q$ , and  $\gamma \in \{\perp, \top\}$ , there exists a safe path  $\pi$  of  $\mathcal{G}_A$ , such that  $\text{last}(\pi) = \langle q, \gamma, q' \rangle$ , and  $\text{weight}(\pi) \leq_{\oplus} \text{weight}_A(q[\gamma], t, q'[\gamma])$ .*

*Proof.* By induction on the length of  $t$ . If the length of  $t$  is zero, then by (1),  $\text{weight}_A(q[\gamma], t, q'[\gamma]) = \mathbb{1}$  if  $q = q'$  and  $\text{weight}_A(q[\gamma], t, q'[\gamma]) = \mathbb{0}$  otherwise. In both cases, we can choose the singleton path  $\pi = \langle q, \gamma, q' \rangle$ .

Let us now assume that the length of  $t$  is strictly greater than 0. Since  $t$  is well-parenthesised by hypothesis, we are in one of the following three cases.

- $t = t' a$ , for  $a \in \Delta_i$ , and some well-parenthesised word  $t'$ . By (1), it holds that  $\text{weight}_A(q[\gamma], t, q'[\gamma]) = \text{weight}_A(q[\gamma], t', q''[\gamma]) \otimes w_i(q'', a, q')$  for some  $q'' \in Q$ . By induction hypothesis, there exists a safe path  $\pi'$  of  $\mathcal{G}_A$ , such that  $\text{last}(\pi') = \langle q, \gamma, q'' \rangle$  and  $\text{weight}(\pi') \leq_{\oplus} \text{weight}_A(q[\gamma], t', q''[\gamma])$ . Let  $\pi = \pi' \langle q, \gamma, q' \rangle$ . This path is safe and

$$\text{weight}(\pi) = \text{weight}(\pi') \otimes \left[ \bigoplus_{\Delta_i} w_i(q'', q') \oplus \bigoplus_{\Delta_r} w_r^e(q'', q') \right].$$

Following Lemma 1,  $\bigoplus_{\Delta_i} w_i(q'', q') \oplus \bigoplus_{\Delta_r} w_r^e(q'', q') \leq_{\oplus} w_i(q'', a, q')$ . Hence,

$$\begin{aligned} \text{weight}(\pi) &\leq_{\oplus} \text{weight}(\pi') \otimes w_i(q'', a, q') \\ &\leq_{\oplus} \text{weight}_A(q[\gamma], t', q''[\gamma]) \otimes w_i(q'', a, q') = \text{weight}_A(q[\gamma], t, q'[\gamma]). \end{aligned}$$

- $t = t' r$ , for  $r \in \Delta_r$ , and some well-parenthesised word  $t'$ : the proof is similar to the above case.
- $t = c t' r$  for  $c \in \Delta_c$ ,  $r \in \Delta_r$ , and some well-parenthesised word  $t'$ . We have, for some  $p \in P$ ,

$$\text{weight}_A(q[\gamma], t, q'[\gamma]) = w_c(q, c, q_1, p) \otimes \text{weight}_A(q_1[\top], t', q_2[\top]) \otimes w_r(q_2, c, p, r, q').$$

Note that in the intermediate computation from  $q_1$  to  $q_2$ , the stack must not be empty, because it contains at least the pair  $\langle c, p \rangle$  on top. By induction hypothesis, there exists a safe path  $\pi'$  of  $\mathcal{G}_A$ , such that  $\text{last}(\pi') = \langle q_1, \top, q_2 \rangle$  and  $\text{weight}(\pi') \leq_{\oplus} \text{weight}_A(q_1[\top], t', q_2[\top])$ . Let  $\pi = \pi' \langle q, \gamma, q' \rangle$ . This path is safe and  $\text{weight}(\pi) = \text{weight}(\pi') \otimes \eta_A(\langle q_1, \top, q_2 \rangle \rightarrow \langle q, \gamma, q' \rangle)$ . The edge's weight  $\eta_A(\langle q_1, \top, q_2 \rangle \rightarrow \langle q, \gamma, q' \rangle)$  can take one of the following two values:

$$\text{if } \gamma = \perp, \bigoplus_{p' \in P} \bigoplus_{\Delta_c} [w_c(q_0, q_1, p') \otimes \bigoplus_{\Delta_r}^2 w_r(q_2, p', q_3)],$$

$$\text{if } \gamma = \top, \left[ \bigoplus_{q_1=q} \bigoplus_{\Delta_i} w_i(q_2, q') \right] \oplus \left[ \bigoplus_{p' \in P} \bigoplus_{\Delta_c} [w_c(q, q_1, p') \otimes_2 \bigoplus_{\Delta_r}^2 w_r(q_2, p', q')] \right].$$

By Lemma 1, in any case, it holds that:

$$\eta_A(\langle q_1, \top, q_2 \rangle \rightarrow \langle q, \gamma, q' \rangle) \leq_{\oplus} w_c(q, c, q_1, p) \otimes w_r(q_2, c, p, r, q').$$

Hence,

$$\begin{aligned} \text{weight}(\pi) &\leq_{\oplus} w_c(q, c, q_1, p) \otimes \text{weight}(\pi') \otimes w_r(q_2, c, p, r, q') \\ &\leq_{\oplus} w_c(q, c, q_1, p) \otimes \text{weight}_A(q_1[\top], t', q_2[\top]) \otimes w_r(q_2, c, p, r, q') \\ &\leq_{\oplus} \text{weight}_A(q[\gamma], t, q'[\gamma]). \end{aligned}$$

□

Now we can complete the proof of Lemma 5, and Theorem 4. Let  $q, q' \in Q$ .

If  $\bigoplus_{\substack{\pi \in V_A^* \\ \text{last}(\pi) = \langle q, \perp, q' \rangle}} \text{weight}(\pi) = 0$ , then for all paths  $\pi$  of  $\mathcal{G}_A$  with  $\text{last}(\pi) = \langle q, \perp, q' \rangle$ ,  $\text{weight}(\pi) = 0$ , since this sum is finite and  $\mathbb{S}$  is assumed total. Lemma 7 implies that for all  $t \in \Delta^*$ ,  $\text{weight}_A(q[\perp], t, q'[\perp]) = 0$ , by contradiction. Therefore,  $b_\perp(q, q') = 0$  in this case.

Let us now assume that

$$\bigoplus_{\substack{\pi \in V_A^* \\ \text{last}(\pi) = \langle q, \perp, q' \rangle}} \text{weight}(\pi) = W \neq 0. \quad (10)$$

There exists  $\pi_{q,q'}$  of  $\mathcal{G}_A$  with  $\text{last}(\pi_{q,q'}) = \langle q, \perp, q' \rangle$ , such that  $W = \text{weight}(\pi_{q,q'})$ . By Lemma 6, there exists  $t_{q,q'} \in \Delta^*$  such that  $\text{weight}_A(q[\perp], t_{q,q'}, q'[\perp]) = W$ . We can show by contradiction that for all  $t \in \Delta^*$ ,  $W \leq_\oplus \text{weight}_A(q[\perp], t, q'[\perp])$ . Indeed, assume on the opposite that  $\text{weight}_A(q[\perp], t, q'[\perp]) <_\oplus W$  for some  $t \in \Delta^*$ . Since this weight is computed by starting and ending with an empty stack,  $t$  is well-parenthesised, and by Lemma 7, there exists a path  $\pi$  with  $\text{last}(\pi) = \langle q, \perp, q' \rangle$ , and  $\text{weight}(\pi) <_\oplus W$ , contradicting (10).

Therefore,  $b_\perp(q, q') = \text{weight}_A(q[\perp], t_{q,q'}, q'[\perp]) = W$ .

Moreover, the above word  $t_{q,q'}$  is a witness reaching the minimum of the swVPA  $A$  computed by Theorem 4.

## F Nested Words and Parse Trees

The hierarchical structure of nested words, defined with the *call* and *return* markup symbols, suggest a correspondence with trees. The lifting of this correspondence to languages, of tree automata and VPA, has been discussed in [2], and [4] for the weighted case. In this section, we describe a correspondence between the symbolic-weighted extensions of tree automata and VPA. It might be folklore knowledge but we state it explicitly for the sake of clarity.

Let  $\Omega$  be a countable ranked alphabet, such that every symbol  $a \in \Omega$  has a rank  $\text{rk}(a) \in [0..M]$  where  $M$  is a fixed natural number. We denote by  $\Omega_k$  the subset of all symbols  $a \in \Omega$  with  $\text{rk}(a) = k$ , where  $0 \leq k \leq M$ , and  $\Omega_{>0} = \Omega \setminus \Omega_0$ . The free  $\Omega$ -algebra of finite, ordered,  $\Omega$ -labeled trees is denoted by  $\mathcal{T}_\Omega$ . It is the smallest set such that  $\Omega_0 \subset \mathcal{T}_\Omega$ , and, for all  $1 \leq k \leq M$ , all  $a \in \Omega_k$ , and all  $t_1, \dots, t_k \in \mathcal{T}_\Omega$ ,  $a(t_1, \dots, t_k) \in \mathcal{T}_\Omega$ . Let us assume a commutative semiring  $\mathbb{S}$  and a label theory  $\bar{\Phi}$  over  $\mathbb{S}$  containing one set  $\Phi_{\Omega_k}$  for each  $k \in [0..M]$ .

**Definition 5.** A symbolic-weighted tree automaton (swTA) over  $\Omega$ ,  $\mathbb{S}$ , and  $\bar{\Phi}$  is a triplet  $A = \langle Q, \text{in}, \bar{\mathbf{w}} \rangle$  where  $Q$  is a finite set of states,  $\text{in} : Q \rightarrow \mathbb{S}$  is the starting weight function, and  $\bar{\mathbf{w}}$  is a tuple of transition functions containing, for each  $k \in [0..M]$ , the function  $\mathbf{w}_k : Q \times Q^k \rightarrow \Phi_{\Omega_k}$ .

We define a transition function  $\mathbf{w} : Q \times \Omega \times \bigcup_{k=0}^M Q^k \rightarrow \mathbb{S}$  by  $(q_1 \dots q_k$  is  $\varepsilon$  if  $k = 0)$ :

$$\mathbf{w}(q_0, b, q_1 \dots q_k) = \phi(b) \quad \text{where } \phi = \mathbf{w}_k(q_0, q_1 \dots q_k).$$

Every **swTA** defines a mapping from trees of  $\mathcal{T}_\Omega$  into  $\mathbb{S}$ , based on the following intermediate function  $\text{weight}_A : Q \times \mathcal{T}_\Omega \rightarrow \mathbb{S}$

$$\text{weight}_A(q_0, t) = \bigoplus_{q_1 \dots q_k \in Q^k} w(q_0, b, q_1 \dots q_k) \otimes \bigotimes_{i=1}^k \text{weight}_A(q_i, t_i) \quad (11)$$

where  $q_0 \in Q$ , and  $t = b(t_1, \dots, t_k) \in \mathcal{T}_\Omega$ , with  $0 \leq k \leq M$  (by convention, the product from 1 to  $k$  is equal to  $\mathbb{1}$  when  $k = 0$ ).

Finally, the weight associated by  $A$  to  $t \in \mathcal{T}_\Omega$  is

$$A(t) = \bigoplus_{q \in Q} \text{in}(q) \otimes \text{weight}_A(q, t) \quad (12)$$

Intuitively,  $w(q_0, b, q_1 \dots q_k)$  can be seen as the weight of a production rule  $q_0 \rightarrow b(q_1, \dots, q_k)$  of a regular tree grammar [App22], that replaces the non-terminal symbol  $q_0$  by  $b(q_1, \dots, q_k)$ . The above production rule can also be seen as a rule of a weighted CF grammar, of the form  $[b] q_0 := q_1 \dots q_k$  if  $k > 0$ , and  $[b] q_0 := b$  if  $k = 0$ . In the first case,  $b$  is a label for the rule, and in the second case, it is also a terminal symbol. The weight of a labeled derivation tree  $t$  of the weighted CF grammar associated to  $A$  as above, is  $\text{weight}_A(q, t)$ , when  $q$  is the start non-terminal.

We shall now establish a correspondence between such a derivation tree  $t$  and some word describing a linearization of  $t$ , in a way that  $\text{weight}_A(q, t)$  can be computed by a **swVPA**. Let  $\hat{\Omega}$  be the countable (unranked) alphabet obtained from  $\Omega$  by:  $\hat{\Omega} = \Delta_i \uplus \Delta_c \uplus \Delta_r$ , with  $\Delta_i = \Omega_0$ ,  $\Delta_c = \{ \langle a \mid a \in \Omega_{>0} \rangle \}$ ,  $\Delta_r = \{ a \mid a \in \Omega_{>0} \}$ . We associate to  $\hat{\Omega}$  a label theory  $\hat{\Phi}$  like in Section 3, and we define a linearization of trees of  $\mathcal{T}_\Omega$  into words of  $\hat{\Omega}^*$  as follows:

$$\begin{aligned} \text{lin}(a) &= a \text{ for all } a \in \Omega_0, \\ \text{lin}(b(t_1, \dots, t_k)) &= \langle_b \text{lin}(t_1) \dots \text{lin}(t_k) \rangle_b \text{ when } b \in \Omega_k \text{ for } 1 \leq k \leq M. \end{aligned}$$

*Example 6.* The trees in Figure 3 represent the two scores in Example 1, and their linearization are respectively  $O$  and  $O'$  in the same examples.

**Proposition 2.** *For all **swTA**  $A$  over  $\Omega$ ,  $\mathbb{S}$  commutative, and  $\bar{\Phi}$ , there exists an effectively constructible **swVPA**  $A'$  over  $\hat{\Omega}$ ,  $\mathbb{S}$  and  $\hat{\Phi}$  such that for all  $t \in \mathcal{T}_\Omega$ ,  $A'(\text{lin}(t)) = A(t)$ .*

*Proof.* We follow Definition 3 in Appendix C for **swVPA**, i.e.  $w_i = w_{01}$  and we ignore the first symbol argument (input symbol) of Definition 1. Let  $A = \langle Q, \text{in}, \bar{w} \rangle$  where  $\bar{w}$  is presented as above by a function. We build  $A' = \langle Q', P', \text{in}', \bar{w}', \text{out}' \rangle$ , where  $Q' = \bigcup_{k=0}^M Q^k$  is the set of sequences of state symbols of  $A$ , of length at most  $M$ , including the empty sequence denoted by  $\varepsilon$ , and where  $P' = Q'$  and  $\bar{w}'$  is defined by  $(\bar{u}, \bar{q} \in Q', \bar{p} \in P')$ :

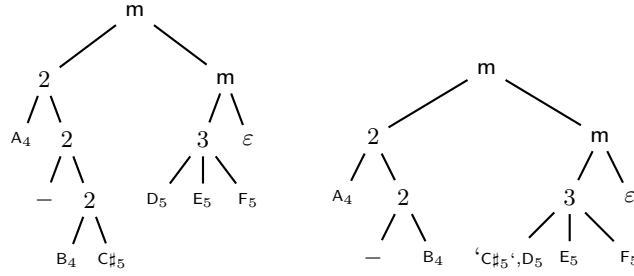
$$\begin{aligned} w_i(q_0 \bar{u}, a, \bar{u}) &= w(q_0, a, \varepsilon) \text{ for all } a \in \Omega_0 \\ w_c(q_0 \bar{u}, \langle_c \bar{q}, \bar{u} \rangle) &= w(q_0, \langle_c \bar{q} \rangle) \text{ for all } c \in \Omega_{>0} \\ w_r(\varepsilon, \langle_c \bar{p}, c \rangle, \bar{p}) &= \mathbb{1} \text{ for all } c \in \Omega_{>0} \\ w_r^e(\bar{u}, c, \bar{q}) &= \mathbb{0} \text{ for all } c \in \Omega_{>0} \end{aligned}$$

All cases not matched by one of the above equations have a weight 0, for instance  $w_r(\bar{u}, \langle c, \bar{p}, d \rangle, \bar{q}) = 0$  if  $c \neq d$  or  $\bar{u} \neq \varepsilon$  or  $\bar{q} \neq \bar{p}$ .

The entering and leaving weight functions  $\text{in}', \text{out}' : Q' \rightarrow \mathbb{S}$  are defined by:

- $\text{in}'(q) = \text{in}(q)$  for all  $q \in Q$ ,
- $\text{in}'(\bar{q}) = 0$  for every other (non-singleton) sequence  $\bar{q} \in Q'$ ,
- $\text{in}'(\bar{q}) = 1$  for all  $\bar{q} \in Q'$ .

□



**Fig. 3.** Tree representation of the scores of Ex 1, linearized respectively into  $O$  and  $O'$ .

## References for appendices

22. Comon, H., Dauchet, M., Gilleron, R., Jacquemard, F., Löding, C., Lugiez, D., Tison, S., Tommasi, M.: Tree Automata Techniques and Applications. <http://tata.gforge.inria.fr> (2007)
23. Huang, L.: Advanced dynamic programming in semiring and hypergraph frameworks. In: COLING (2008)
24. Lombardy, S., Sakarovitch, J.: The removal of weighted  $\varepsilon$ -transitions. In: International Conference on Implementation and Application of Automata. pp. 345–352. Springer (2012)