*TALN 2004, Session Poster, Fès, 19–21 avril 2004*

# NLP Applications Based on Weighted Multi-Tape Automata

André Kempe
Xerox Research Centre Europe – Grenoble Laboratory
6 chemin de Maupertuis – 38240 Meylan – France
andre.kempe@xrce.xerox.com – www.xrce.xerox.com/people/kempe/

## Abstract

This article describes two practical applications of *weighted multi-tape automata* (WMTAs) in Natural Language Processing, that demonstrate the augmented descriptive power of WMTAs compared to weighted 1-tape and 2-tape automata. The two examples concern the preservation of intermediate results in transduction cascades and the search for similar words in two languages. As a basis for these applications, the article proposes a number of operations on WMTAs. Among others, it (re-)defines multi-tape intersection, where a number of tapes of one WMTA are intersected with the same number of tapes of another WMTA. In the proposed approach, multi-tape intersection is not an atomic operation but rather a sequence of more elementary ones, which facilitates its implementation.

## Keywords

finite-state automaton, weighted multi-tape automaton, transduction cascade, lexicon

## 1 Introduction

Finite state automata (FSAs) and weighted finite state automata (WFSAs) are widely used in language and speech processing (Kaplan & Kay, 1981; Mohri, 1997; Beesley & Karttunen, 2003). They permit, among others, the fast processing of input strings and can be easily modified and combined by well defined operations. Most systems and applications deal with *1-tape* and *2-tape automata*, also called acceptors and transducers, respectively. *Multi-tape automata* (MTAs) (Elgot & Mezei, 1965) offer additional advantages such as storing different types of information on different tapes. MTAs have been implemented and used, e.g., in the morphological analysis of Semitic languages, where the vowels, consonants, pattern, and surface form of words have been represented on different tapes (Kay, 1987; Kiraz & Grimley-Evans, 1998).

This article describes two practical applications of *weighted multi-tape automata* (WMTAs) and MTAs in *Natural Language Processing* (NLP). The first example shows how intermediate results can be preserved in transduction cascades so that they can be accessed by any of the following transductions (Sec. 4.1). The second one deals with the search for words that are similar in two languages, and in particular in French and Spanish (Sec. 4.2). To support these applications, the article defines WMTAs (Sec. 2) and some WMTA operations (Sec. 3). Efficient algorithms for the these operations have been implemented in the WFSC toolkit (Kempe *et al.*, 2003) and will be presented in future publications.

# 2 Weighted Multi-Tape Automata

In the following we build on basic definitions of a *monoid*, a *semiring*, a *weighted automaton*, and a *multi-tape automaton* (Elgot & Mezei, 1965; Eilenberg, 1974; Kuich & Salomaa, 1986; Mohri *et al.*, 1998) which we do not recall for reasons of space.

A *weighted multi-tape automaton* (WMTA), $A^{(n)}$, also called weighted $n$-tape automaton, over a semiring $\mathcal{K}$ is defined as a six-tuple

$$A^{(n)} =_{\text{def}} \langle \Sigma, Q, I, F, E^{(n)}, \mathcal{K} \rangle \tag{1}$$

with $\Sigma$ being a finite alphabet, $Q$ the finite set of states, $I \subseteq Q$ the set of initial states, $F \subseteq Q$ the set of final states, $n$ the arity, i.e., the number of tapes in $A^{(n)}$, $E^{(n)} \subseteq (Q \times (\Sigma^*)^n \times \mathbb{K} \times Q)$ the finite set of $n$-tape transitions, and $\mathcal{K} = \langle \mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1} \rangle$ the semiring of weights. For any state $q \in Q$, we denote by $\lambda(q) \in \mathcal{K}$ its initial weight and by $\varrho(q) \in \mathcal{K}$ its final weight. For any transition $e^{(n)} \in E^{(n)}$, $e^{(n)} = \langle p, \ell^{(n)}, w, n \rangle$, we denote by $p(e^{(n)}) \in Q$ its source state, by $w(e^{(n)}) \in \mathcal{K}$ its weight, by $n(e^{(n)}) \in Q$ its target state, and by $\ell(e^{(n)})$ its label which is an $n$-tuple of strings, $\ell : E^{(n)} \rightarrow (\Sigma^*)^n$. A *path* $\pi^{(n)}$ of length $r = |\pi^{(n)}|$ is a sequence of transitions $e_1^{(n)} e_2^{(n)} \cdots e_r^{(n)}$ such that $n(e_i^{(n)}) = p(e_{i+1}^{(n)}), \forall i \in [\![1, r-1]\!]$. A path is said to be *successful* iff $p(e_1^{(n)}) \in I$ and $n(e_r^{(n)}) \in F$. Its label $\ell(\pi^{(n)})$ is an $n$-tuple of strings and equals the concatenation of the labels of its transitions:

$$\ell(\pi^{(n)}) \;=\; s^{(n)} \;=\; \langle s_1, s_2, \ldots, s_n \rangle \;=\; \ell(e_1^{(n)})\, \ell(e_2^{(n)}) \;\cdots\; \ell(e_r^{(n)}) \tag{2}$$

Its weight $w(\pi^{(n)})$ is

$$w(\pi^{(n)}) \;=\; \lambda(p(e_1^{(n)})) \otimes \left( \bigotimes_{j=[\![1,r]\!]} w\left(e_j^{(n)}\right) \right) \otimes \varrho(n\left(e_r^{(n)}\right)) \tag{3}$$

We denote by $\Pi(A^{(n)})$ the set of successful paths of $A^{(n)}$ and by $\mathcal{R}^{(n)} = \mathcal{R}(A^{(n)})$ the $n$-tape relation of $A^{(n)}$. It is the set of $n$-tuples of strings $s^{(n)}$ having successful paths in $A^{(n)}$:

$$\mathcal{R}(A^{(n)}) \;=\; \{\; s^{(n)} \mid \exists \pi^{(n)} \in \Pi(A^{(n)}) \;\wedge\; \ell(\pi^{(n)}) = s^{(n)} \;\} \tag{4}$$

The weight of any $s^{(n)} \in \mathcal{R}(A^{(n)})$ is the semiring sum of the weights of all paths labeled with $s^{(n)}$ :

$$w(s^{(n)}) \;=\; \bigoplus_{\pi^{(n)} \mid \ell(\pi^{(n)}) = s^{(n)}} w(\pi^{(n)}) \tag{5}$$

# 3 Operations

**Pairing and Concatenation:** We define the *pairing* of two string tuples, $s^{(n)} : v^{(m)} = u^{(n+m)}$, as

$$\langle s_1, \ldots, s_n \rangle : \langle v_1, \ldots, v_m \rangle \quad =_{\text{def}} \quad \langle s_1, \ldots, s_n, v_1, \ldots, v_m \rangle \tag{6}$$

$$w\,(\, \langle s_1, \ldots, s_n \rangle : \langle v_1, \ldots, v_m \rangle \,) \quad =_{\text{def}} \quad w\,(\, \langle s_1, \ldots, s_n \rangle \,) \otimes w\,(\, \langle v_1, \ldots, v_m \rangle \,) \tag{7}$$

The *concatenation* of two string tuples of equal arity, $s^{(n)} v^{(n)} = u^{(n)}$, is defined as

$$\langle s_1, \ldots, s_n \rangle \langle v_1, \ldots, v_n \rangle \quad =_{\text{def}} \quad \langle s_1 v_1, \ldots, s_n v_n \rangle \tag{8}$$

$$w\,(\, \langle s_1, \ldots, s_n \rangle \langle v_1, \ldots, v_n \rangle \,) \quad =_{\text{def}} \quad w\,(\, \langle s_1, \ldots, s_n \rangle \,) \otimes w\,(\, \langle v_1, \ldots, v_n \rangle \,) \tag{9}$$

**Projection and Complementary Projection:** *Projection*, $\mathcal{P}_{j,k,\ldots}(s^{(n)})$, of a string tuple is defined as

$$\mathcal{P}_{j,k,\ldots}(\, \langle s_1, \ldots, s_n \rangle \,) \quad =_{\text{def}} \quad \langle s_j, s_k, \ldots \rangle \tag{10}$$

It retains only those strings (i.e., tapes) of the tuple that are specified by the indices $j, k, \ldots \in [\![1, n]\!]$, and places them in the specified order. The weight of the tuple is not modified (if we

consider it not as a member of a relation). The projection of an $n$-tape relation is the projection of all its string tuples:

$$\mathcal{P}_{j,k,\ldots}(\mathcal{R}^{(n)}) \quad =_{\text{def}} \quad \{\ v^{(m)} \ \mid\ \exists s^{(n)} \in \mathcal{R}^{(n)} \wedge \mathcal{P}_{j,k,\ldots}(s^{(n)}) = v^{(m)}\ \} \tag{11}$$

The weight of each $v^{(m)} \in \mathcal{P}_{j,k,\ldots}(\mathcal{R}^{(n)})$ is the semiring sum of the weights of each $s^{(n)} \in \mathcal{R}^{(n)}$ leading, when projected, to $v^{(m)}$:

$$w(v^{(m)}) \quad =_{\text{def}} \bigoplus_{s^{(n)} \mid \mathcal{P}_{j,k,\ldots}(s^{(n)}) = v^{(m)}} w(s^{(n)}) \tag{12}$$

*Complementary projection*, $\overline{\mathcal{P}}_{j,k,\ldots}(s^{(n)})$, removes those strings of the tuple $s^{(n)}$ that are specified by the indices $j, k, \ldots \in [\![1, n]\!]$. It is defined as

$$\overline{\mathcal{P}}_{j,k,\ldots}(\ \langle s_1, \ldots, s_n \rangle\ ) \quad =_{\text{def}} \quad \langle \ldots, s_{j-1}, s_{j+1}, \ldots, s_{k-1}, s_{k+1}, \ldots \rangle \tag{13}$$

$$\overline{\mathcal{P}}_{j,k,\ldots}(\mathcal{R}^{(n)}) \quad =_{\text{def}} \quad \{\ v^{(m)} \ \mid\ \exists s^{(n)} \in \mathcal{R}^{(n)} \wedge \overline{\mathcal{P}}_{j,k,\ldots}(s^{(n)}) = v^{(m)}\ \} \tag{14}$$

$$w(v^{(m)}) \quad =_{\text{def}} \bigoplus_{s^{(n)} \mid \overline{\mathcal{P}}_{j,k,\ldots}(s^{(n)}) = v^{(m)}} w(s^{(n)}) \tag{15}$$

**Cross-Product:** The *cross-product* of two $n$-tape relations is based on pairing and is defined as

$$\mathcal{R}_1^{(n)} \times \mathcal{R}_2^{(m)} \quad =_{\text{def}} \quad \{\ s^{(n)} : v^{(m)} \mid s^{(n)} \in \mathcal{R}_1^{(n)}, v^{(m)} \in \mathcal{R}_2^{(m)}\ \} \tag{16}$$

**Auto-Intersection:** We define the *auto-intersection* of a relation, $\mathcal{I}_{j,k}(\mathcal{R}^{(n)})$, on the tapes $j$ and $k$ as the subset of $\mathcal{R}^{(n)}$ that contains all $s^{(n)}$ with equal $s_j$ and $s_k$:

$$\mathcal{I}_{j,k}(\mathcal{R}^{(n)}) \quad =_{\text{def}} \quad \{s^{(n)} \in \mathcal{R}^{(n)} \mid s_j = s_k\} \tag{17}$$

The weight of any $s^{(n)} \in \mathcal{I}_{j,k}(\mathcal{R}^{(n)})$ is not modified. For example (Figure 1)

$$\mathcal{R}_1^{(3)} \ = \ \langle a, x, \varepsilon \rangle\ \langle b, y, a \rangle^*\ \langle \varepsilon, z, b \rangle \ = \ \{\langle ab^k, xy^kz, a^kb \rangle \mid k \in \mathbb{N}\} \tag{18}$$

$$\mathcal{I}_{1,3}(\mathcal{R}_1^{(3)}) \ = \ \{\langle ab^1, xy^1z, a^1b \rangle\} \tag{19}$$

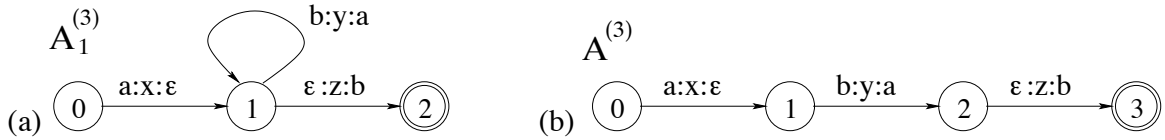


Figure 1: (a) A WMTA $A_1^{(3)}$ and (b) its auto-intersection $A^{(3)} = \mathcal{I}_{1,3}(A_1^{(3)})$. (Weights omitted)

**Single-Tape and Multi-Tape Intersection:** *Multi-tape intersection* of two relations, $\mathcal{R}_1^{(n)}$ and $\mathcal{R}_2^{(m)}$, uses $r$ tapes in each relation, and intersects them pair-wise. We (re-)define it as

$$\mathcal{R}_1^{(n)} \underset{\substack{j_1,\,k_1 \\ \cdots \\ j_r,\,k_r}}{\cap} \mathcal{R}_2^{(m)} \quad =_{\text{def}} \quad \overline{\mathcal{P}}_{n+k_1,\ldots,n+k_r} \left( \mathcal{I}_{j_r,n+k_r}( \cdots \mathcal{I}_{j_1,n+k_1}( \mathcal{R}_1^{(n)} \times \mathcal{R}_2^{(m)} ) \cdots ) \right) \tag{20}$$

The operation pairs each $s^{(n)} \in \mathcal{R}_1^{(n)}$ with each $v^{(m)} \in \mathcal{R}_2^{(m)}$ iff $s_{j_1} = v_{k_1}$ until $s_{j_r} = v_{k_r}$. We speak about *single-tape intersection* if only one tape is used in each relation ($r = 1$). All tapes $k_i$ of $\mathcal{R}_2^{(m)}$ that are used in the intersection are afterwards equal to the tapes $j_i$ of $\mathcal{R}_1^{(n)}$, and are removed. The operation is conceptually similar to composition of two relations, except that the tapes $j_i$ are preserved and hence can be (re-)used in subsequent operations. The result is

$$\begin{aligned} \mathcal{R}^{(n+m-r)} \ = \ \{\ u^{(n+m-r)} \ \mid\ & \exists s^{(n)} \in \mathcal{R}_1^{(n)} \wedge \exists v^{(m)} \in \mathcal{R}_2^{(m)} \wedge s_{j_i} = v_{k_i},\ \forall i \in [\![1, r]\!] \\ & \wedge\ u^{(n+m-r)} = \overline{\mathcal{P}}_{n+k_1,\ldots,n+k_r}(s^{(n)} : v^{(m)})\ \} \end{aligned} \tag{21}$$

$$w(u^{(n+m-r)}) \ = \ w(s^{(n)}) \otimes w(v^{(m)}) \tag{22}$$

Although single-tape and multi-tape intersection include complementary projection, Eq. 22 is not in conflict with Eq. 15 because any two $u^{(n+m)} = s^{(n)}{:}v^{(m)}$ that differ in $v_{k_i}$, differ also in $s_{j_i}$, and hence cannot become equal when the $v_{k_i}$ are removed.

Two well-known special cases are the intersection of two acceptors leading to an acceptor, and the composition of two transducers leading to a transducer:

$$A_1^{(1)} \cap A_2^{(1)} \;=\; A_1^{(1)} \underset{1,1}{\cap} A_2^{(1)} \;=\; \overline{\mathcal{P}}_2\left( \mathcal{I}_{1,2}(\, A_1^{(1)} \times A_2^{(1)} \,) \right) \tag{23}$$

$$A_1^{(2)} \diamond A_2^{(2)} \;=\; \overline{\mathcal{P}}_2(\, A_1^{(2)} \underset{2,1}{\cap} A_2^{(2)} \,) \;=\; \overline{\mathcal{P}}_{2,3}\left( \mathcal{I}_{2,3}(\, A_1^{(2)} \times A_2^{(2)} \,) \right) \tag{24}$$

# 4 Applications

## 4.1 Preserving Intermediate Transduction Results

Transduction cascades are frequently used in language and speech processing. In a (classical) weighted transduction cascade, $T_1^{(2)} \ldots\ T_r^{(2)}$, a set of weighted input strings, encoded as a weighted acceptor, $L_0^{(1)}$, is composed with the first transducer, $T_1^{(2)}$, on its input tape (Figure 2). The output projection of this composition is the first intermediate result, $L_1^{(1)}$, of the cascade. It is further composed with the second transducer, $T_2^{(2)}$, which leads to the second intermediate result, $L_2^{(1)}$, etc. The output projection of the last transducer is the final result, $L_r^{(1)}$ :

$$L_i^{(1)} \;=\; \mathcal{P}_2(\, L_{i-1}^{(1)} \diamond T_i^{(2)} \,) \qquad\qquad \text{for} \quad i \in [\![1, r]\!] \tag{25}$$

At any point in the cascade, previous results cannot be accessed.

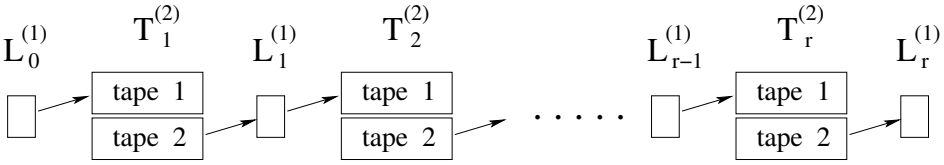


Figure 2: Weighted transduction cascade (classical)

In a weighted transduction cascade, $A_1^{(n_1)} \ldots\ A_r^{(n_r)}$, that uses WMTAs and multi-tape intersection, intermediate results can be preserved and (re-)used by all subsequent transductions. Suppose, we want to use the two previous results at each point in the cascade (except in the first transduction) which requires all intermediate results, $L_i^{(2)}$, to have two tapes (Figure 3) :

$$L_1^{(2)} \;=\; L_0^{(1)} \underset{1,1}{\cap} A_1^{(2)} \tag{26}$$

$$L_i^{(2)} \;=\; \mathcal{P}_{2,3}(\, L_{i-1}^{(2)} \underset{\substack{1,1\\2,2}}{\cap} A_i^{(3)} \,) \qquad\qquad \text{for} \quad i \in [\![2, r{-}1]\!] \tag{27}$$

$$L_r^{(2)} \;=\; \mathcal{P}_3(\, L_{r-1}^{(2)} \underset{\substack{1,1\\2,2}}{\cap} A_r^{(3)} \,) \tag{28}$$

This augmented descriptive power is also available if the whole cascade is intersected into a single WMTA, $A^{(2)}$ (although $A^{(2)}$ has only two tapes in our example). Each of the "incorporated" multi-tape sub-relations in $A^{(2)}$ (except the first one) will still refer to its two predecessors:

$$A_{1\ldots i}^{(3)} \;=\; \mathcal{P}_{1,n-1,n}(\, A_{1\ldots i-1}^{(m)} \underset{\substack{n-1,\,1\\n,\,2}}{\cap} A_i^{(3)} \,) \qquad\qquad \text{for} \quad i \in [\![2, r]\!]\,,\ m \in \{2, 3\} \tag{29}$$

$$A^{(2)} \;=\; \mathcal{P}_{1,n}(\, A_{1\ldots r} \,) \tag{30}$$

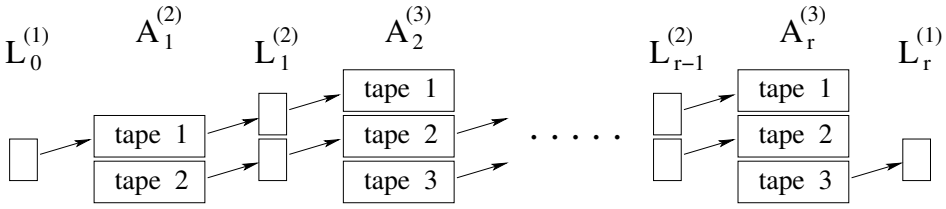


Figure 3: Weighted transduction cascade using multi-tape intersection

## 4.2 Extracting Similar Words in French and Spanish

To extract, in general, from a relation $\mathcal{R}_1^{(n)}$ all string tuples $s^{(n)}$ whose strings $s_{j_1}$ to $s_{j_r}$ are similar to its strings $s_{k_1}$ to $s_{k_r}$, respectively, we can compare each pair of tapes, $j_i$ and $k_i$, independently from all other pairs. Hence the task can be reduced to comparing two tapes, $j$ and $k$. This can be done by means of a weighted 2-tape relation, $\mathcal{R}_S^{(2)}$, that describes the required similarity between tape $j$ and $k$ of $\mathcal{R}_1^{(n)}$:

$$\mathcal{R}_2^{(n)} \;=\; \mathcal{R}_1^{(n)} \underset{\substack{j,\,1\\ k,\,2}}{\cap} \mathcal{R}_S^{(2)} \tag{31}$$

For example, if we have an French-Spanish 3-tape lexicon, $FrEs^{(3)}$, with entries of the form $s^{(3)} = \langle FrenchWord, SpanishWord, PosTag\rangle$, and want to find all words that are similar in the two languages, we create a 2-tape automaton, $S^{(2)}$, describing this similarity. For that we compile a WMTA $G_c^{(2)}$ that encodes various synchronic consonant correspondences, $G_v^{(2)}$ and $G_{v\text{fin}}^{(2)}$ that describe alternations between sequences of vowels, and $G_d^{(2)}$ that admits any diacritization (insertion of accents, cedille, tilde, etc.) :

$$G_c^{(2)} \;=\; \{\texttt{b:v}\} \cup \{\texttt{ph:f}\} \cup \{\texttt{ch:c}\} \cup \{\texttt{qu:c}\} \cup \;\ldots \tag{32}$$

$$G_v^{(2)} \;=\; {A_v^{(1)}}^{+} \times {A_v^{(1)}}^{+} \quad \text{with} \quad A_v^{(1)} = \{\texttt{a}\} \cup \{\texttt{e}\} \cup \{\texttt{i}\} \cup \{\texttt{o}\} \cup \{\texttt{u}\} \cup \{\texttt{y}\} \tag{33}$$

$$G_{v\text{fin}}^{(2)} \;=\; {A_v^{(1)}}^{*} \times {A_v^{(1)}}^{*} \tag{34}$$

$$G_d^{(2)} \;=\; \{\texttt{a:à}\} \cup \{\texttt{a:â}\} \cup \;\ldots\; \{\texttt{e:é}\} \cup \{\texttt{e:è}\} \cup \;\ldots\; \{\texttt{n:ñ}\} \cup \{\texttt{c:ç}\} \cup \;\ldots \tag{35}$$

From these sub-relations we compile $S^{(2)}$ describing the relation between any (hypothetical) French word and its potential Spanish form:[1]

$$S^{(2)} \;=\; (\;(G_d^{(2)})^{-1} \cup ?:_i? \;)^{+} \diamond \Big(\;(\;G_c^{(2)} \cup G_v^{(2)} \cup ?:_i? \;)^{+}\; G_{v\text{fin}}^{(2)}\Big) \diamond (\;G_d^{(2)} \cup ?:_i? \;)^{+} \tag{36}$$

To extract similar words with equal meaning from $FrEs^{(3)}$, we intersect it on the tapes of French and Spanish with $S^{(2)}$:

$$FrEs_{sim}^{(3)} \;=\; FrEs^{(3)} \underset{\substack{1,\,1\\ 2,\,2}}{\cap} S^{(2)} \tag{37}$$

For better illustration, we explain this approach on a tiny example: a French-Spanish lexicon containing only the four entries

$$\mathcal{R}(FrEs^{(3)}) \;=\; \{\langle\texttt{chanter},\texttt{cantar},\texttt{VB}\rangle, \langle\texttt{manger},\texttt{comer},\texttt{VB}\rangle, \langle\texttt{piquer},\texttt{mangar},\texttt{VB}\rangle, \langle\texttt{piquer},\texttt{picar},\texttt{VB}\rangle\}$$

With the above approach (Eq. 37), we can extract a sub-lexicon of similar words with equal meaning:

$$\mathcal{R}(FrEs_{sim}^{(3)}) \;=\; \{\langle\texttt{chanter},\texttt{cantar},\texttt{VB}\rangle, \langle\texttt{piquer},\texttt{picar},\texttt{VB}\rangle\}$$

Classical composition cannot accomplish this task, even if we had only a 2-tape lexicon $FrEs^{(2)}$.

[1] Here ? means any symbol, i.e., ? $\in \{\texttt{a},\texttt{b},\texttt{c},\ldots\}$, and $:_i$ is an identity pairing such that $(?:_i?) \in \{\texttt{a:a},\texttt{b:b},\texttt{c:c},\ldots\}$, whereas $(?:?) \in \{\texttt{a:a},\texttt{a:b},\texttt{b:a},\ldots\}$.

For example
$$D_1^{(2)} = \mathcal{P}_1(FrEs^{(2)}) \diamond S^{(2)} \diamond \mathcal{P}_2(FrEs^{(2)}) \quad (38)$$
$$\mathcal{R}(D_1^{(2)}) = \{\langle \texttt{chanter}, \texttt{cantar}\rangle, \langle \texttt{manger}, \texttt{mangar}\rangle, \langle \texttt{piquer}, \texttt{picar}\rangle\}$$

From a full-size French-Spanish lexicon with 45,578 entries, generated from lexical data from ELRA, we extracted 5,624 similar entries, containing among others

| | | |
|---|---|---|
| ⟨blanche, blanca, S⟩ | ⟨cheval, caballo, S⟩ | ⟨oeuvre, obra, S⟩ |
| ⟨brusque, brusco, ADJ⟩ | ⟨grumeau, grumo, S⟩ | ⟨ouvrier, obrero, S⟩ |
| ⟨approuver, aprobar, V⟩ | ⟨nid, nido, S⟩ | ⟨poire, pera, S⟩ |
| ⟨chaleur, calor, S⟩ | ⟨noeud, nudo, S⟩ | ⟨pont, puente, S⟩ |
| ⟨chanter, cantar, V⟩ | ⟨oeil, ojo, S⟩ | ⟨trois, tres, NUM⟩ |

# 5 Conclusion

We have described two practical applications of WMTAs and MTAs in NLP, demonstrating their augmented descriptive power compared to 1-tape and 2-tape automata, namely the preservation of intermediate results in transduction cascades and the search for similar words in two languages. None of these tasks can be accomplished with 1-tape or 2-tape automata, in general.

We recalled some basic operations for WMTAs and MTAs and proposed some others such as auto-intersection of one WMTA and multi-tape intersection of two WMTAs. In our approach, multi-tape intersection is not an atomic operation but rather a sequence of more elementary ones, which facilitates its implementation.

**Acknowledgments** I wish to thank Kenneth R. Beesley, Jean-Marc Champarnaud, Franck Guingne, and Florent Nicart for their help.

# References

BEESLEY K. R. & KARTTUNEN L. (2003). *Finite State Morphology*. Palo Alto, CA, USA: CSLI Publications.

EILENBERG S. (1974). *Automata, Languages, and Machines*, volume A. San Diego, CA, USA: Academic Press.

ELGOT C. C. & MEZEI J. E. (1965). On relations defined by generalized finite automata. *IBM Journal of Research and Development*, 9, 47–68.

KAPLAN R. M. & KAY M. (1981). Phonological rules and finite state transducers. In *Winter Meeting of the Linguistic Society of America*, New York, NY, USA.

KAY M. (1987). Nonconcatenative finite-state morphology. In *Proc. 3rd Int. Conf. EACL*, p. 2–10.

KEMPE A., BAEIJS C., GAÁL T., GUINGNE F. & NICART F. (2003). WFSC – A new weighted finite state compiler. In O. H. IBARRA & Z. DANG, Eds., *Proc. 8th Int. Conf. CIAA*, volume 2759 of *Lecture Notes in Computer Science*, p. 108–119, Santa Barbara, CA, USA: Springer Verlag.

KIRAZ G. A. & GRIMLEY-EVANS E. (1998). Multi-tape automata for speech and language systems: A prolog implementation. In D. WOODS & S. YU, Eds., *Automata Implementation*, number 1436 in Lecture Notes in Computer Science. Springer Verlag.

KUICH W. & SALOMAA A. (1986). *Semirings, Automata, Languages*. Number 5 in EATCS Monographs on Theoretical Computer Science. Springer Verlag.

MOHRI M. (1997). Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2), 269–312.

MOHRI M., PEREIRA F. C. N. & RILEY M. (1998). A rational design for a weighted finite-state transducer library. *Lecture Notes in Computer Science*, 1436, 144–158.