A Note on Sequential Rule-Based POS Tagging

Sylvain Schmitz

LSV, ENS Cachan & CNRS, Cachan, France
schmitz@lsv.ens-cachan.fr

Abstract

Brill's part-of-speech tagger is defined through a cascade of leftmost rewrite rules. We revisit the compilation of such rules into a single sequential transducer given by Roche and Schabes (*Comput. Ling.* 1995) and provide a direct construction of the minimal sequential transducer for each individual rule.

Keywords. Brill Tagger; Sequential Transducer; POS Tagging

1 Introduction

Part-of-speech (POS) tagging consists in assigning the appropriate POS tag to a word in the context of its sentence. The program that performs this task, the *POS tagger*, can be learned from an annotated corpus in case of supervised learning, typically using hidden Markov model-based or rule-based techniques. The most famous rule-based POS tagging technique is due to Brill (1992). He introduced a three-parts technique comprising:

1. a *lexical tagger*, which associates a unique POS tag to each word from an annotated training corpus. This lexical tagger simply associates to each known word its most probable tag according to the training corpus annotation, i.e. a unigram maximum likelihood estimation;

2. an *unknown word tagger*, which attempts to tag unknown words based on suffix or capitalization features. It works like the contextual tagger, using the presence of a capital letter and bounded sized suffixes in its rules: for instance in English, a *-able* suffix usually denotes an adjective;

3. a *contextual tagger*, on which we focus in this paper. It consists of a cascade of string rewrite rules, called *contextual rules*, which correct tag assignments based on some surrounding contexts.

In this note, we revisit the proof that contextual rules can be translated into sequential transducers[1] proposed by Roche and Schabes (1995): whereas Roche and Schabes give a separate proof of sequentiality and exercise it to show that their constructed non-sequential transducer can be determinized (at the

[1] Historically, what we call here "sequential" used to be called "subsequential" (Schützenberger, 1977), but we follow the more recent practice initiated by Sakarovitch (2009).

expense of a worst-case exponential blow-up), we give a direct translation of a contextual rule into the minimal normalized sequential transducer, by adapting Simon (1994)'s string matching automaton to the transducer case. Our resulting sequential transducers are of linear size (before their composition). A similar construction can be found in (Mihov and Schultz, 2007), but no claim of minimality is made there.

2 Contextual Rules

2.1 Example

We start with an example by Roche and Schabes (1995): Let us suppose the following sentences were tagged by the lexical tagger (using the Penn Treebank tagset):

*Chapman/NNP killed/VBN John/NNP Lennon/NNP
*John/NNP Lennon/NNP was/VBD shot/VBD by/IN Chapman/NNP
He/PRP witnessed/VBD Lennon/NNP killed/VBN by/IN Chapman/NNP

with mistakes in the first two sentences: *killed* should be tagged as a past tense form "VBD", and *shot* as a past participle form "VBN".

The contextual tagger learns contextual rules over some tagset Σ of form $uav \to ubv$ (or $a \to b \,/\, u\_\_v$ using phonological rule notations (Kaplan and Kay, 1994)), meaning that the tag a rewrites to b in the context of $u\_\_v$, where the context is of length $|uv|$ bounded by some fixed k; in practice, $k = 2$ or $k = 3$ (Brill (1992) and Roche and Schabes (1995) use slightly different *templates* than the one parametrized by k we present here). For instance, a first contextual rule could be

$$\mathrm{nnp}\,\mathrm{vbn} \to \mathrm{nnp}\,\mathrm{vbd}$$

resulting in a new tagging

Chapman/NNP killed/VBD John/NNP Lennon/NNP
*John/NNP Lennon/NNP was/VBD shot/VBD by/IN Chapman/NNP
*He/PRP witnessed/VBD Lennon/NNP killed/VBD by/IN Chapman/NNP

A second contextual rule could be

$$\mathrm{vbd}\,\mathrm{in} \to \mathrm{vbn}\,\mathrm{in}$$

resulting in the correct tagging

Chapman/NNP killed/VBD John/NNP Lennon/NNP
John/NNP Lennon/NNP was/VBD shot/VBN by/IN Chapman/NNP
He/PRP witnessed/VBD Lennon/NNP killed/VBN by/IN Chapman/NNP

As stated before, our goal is to compile the entire sequence of contextual rules learned from a corpus into a single sequential function.

2.2 Cascade of Contextual Rules

Let us first formalize the semantics of Brill's contextual rules. Let $\mathcal{C} = r_1 r_2 \cdots r_n$ be a finite sequence of string rewrite rules in $\Sigma^* \times \Sigma^*$ with Σ a POS tagset of *fixed size*. In practice the rules constructed in Brill's contextual tagger are *length-preserving* and 1-*change-bounded*, i.e. they modify a single letter, but this is not a useful consideration for our transducer construction. Each rule $r_i = u_i \to v_i$ defines a *leftmost rewrite relation* $\overset{r_i}{\underset{\mathrm{lm}}{\Longrightarrow}}$ defined by

$$w \overset{r_i}{\underset{\mathrm{lm}}{\Longrightarrow}} w' \text{ iff } \exists x, y \in \Sigma^*, w = xu_iy \wedge w' = xv_iy \wedge (\forall z, z' \in \Sigma^*, w \neq zu_iz' \vee x \leq_{\mathrm{pref}} z)$$

where $x \leq_{\mathrm{pref}} z$ denotes that x is a prefix of z. Note that the domain of $\overset{r_i}{\underset{\mathrm{lm}}{\Longrightarrow}}$ is $\Sigma^* \cdot u_i \cdot \Sigma^*$. The *behavior* of a single rule is then the relation $[\![r_i]\!]$ included in $\Sigma^* \times \Sigma^*$ defined by

$$[\![r_i]\!] \stackrel{\mathrm{def}}{=} \overset{r_i}{\underset{\mathrm{lm}}{\Longrightarrow}} \cup \mathrm{Id}_{\Sigma^* \setminus (\Sigma^* \cdot u_i \cdot \Sigma^*)} \ ,$$

i.e. it applies $\overset{r_i}{\underset{\mathrm{lm}}{\Longrightarrow}}$ on $\Sigma^* \cdot u_i \cdot \Sigma^*$ and the identity on its complement $\Sigma^* \setminus (\Sigma^* \cdot u_i \cdot \Sigma^*)$. The behavior of $\mathcal{C}$ is then the composition

$$[\![\mathcal{C}]\!] \stackrel{\mathrm{def}}{=} [\![r_1]\!] \,⨾\, [\![r_2]\!] \,⨾\, \cdots \,⨾\, [\![r_n]\!] \ .$$

Note that this behavior does *not* employ the transitive closure of the rewriting rules.

A naive implementation of $\mathcal{C}$ would try to match each u_i at every position of the input string w in Σ^*, resulting in an overall complexity of $O(|w| \cdot \sum_i |u_i|)$. One often faces the problem of tagging a *set* of sentences $\{w_1, \ldots, w_m\}$, which yields $O((\sum_i |u_i|) \cdot (\sum_j |w_j|))$. As shown in Roche and Schabes' experiments, compiling $\mathcal{C}$ into a single sequential transducer $\mathcal{T}$ results in practice in huge savings, with overall complexities in $O(|w| + |\mathcal{T}|)$ and $O(|\mathcal{T}| + \sum_j |w_j|)$ respectively.

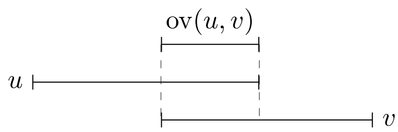
Each $[\![r_i]\!]$ is a rational function, being the union of two rational functions over disjoint domains. Let $|r_i|$ be the length $|u_iv_i| \leq k$. Roche and Schabes (1995, Section 8.2) provide a construction of an exponential-sized transducer $\mathcal{T}_{r_i}$ for each $[\![r_i]\!]$, and compute their composition $\mathcal{T}_\mathcal{C}$ of size $|\mathcal{T}_\mathcal{C}| = O(\prod_{i=1}^n 2^{|r_i|})$. As they show that each $[\![r_i]\!]$ is actually a sequential function, their composition $[\![\mathcal{C}]\!]$ is also sequential, and $\mathcal{T}_\mathcal{C}$ can be determinized to yield a sequential transducer $\mathcal{T}$ of size doubly exponential in $\sum_{i=1}^n |r_i| \leq nk$ (see Roche and Schabes, 1995, Section 9.3). By contrast, our construction directly yields linear-sized minimal sequential transducers for each $[\![r_i]\!]$, resulting in a final sequential transducer of size $O(\prod_{i=1}^n |r_i|) = O(2^{n \log k})$.

3 Sequential Transducer of a Rule

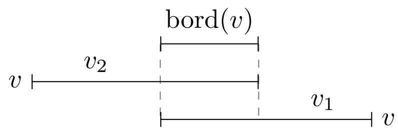
Intuitively, the sequential transducer for $[\![r_i]\!]$ is related to the *string matching automaton* (Simon, 1994; Crochemore and Hancart, 1997) for u_i, i.e. the automaton for the language $\Sigma^* u_i$. This insight yields a *direct* construction of the minimal sequential transducer of a contextual rule, with at most $|u_i| + 1$ states. Let us recall a few definitions:

3.1 Preliminaries

Overlaps, Borders (see e.g. Crochemore and Hancart, 1997, Section 6.2). The *overlap* $\mathrm{ov}(u, v)$ of two words u and v is the longest suffix of u which is simultaneously a prefix of v.



A word u is a *border* of a word v if it is both a prefix and a suffix of v, i.e. if there exist v_1, v_2 in Σ^* such that $v = uv_1 = v_2u$. For $v \neq \varepsilon$, the longest border of v different from v itself is denoted $\mathrm{bord}(v)$.



Fact 3.1. *For all u, v in Σ^* and a in Σ,*

$$\mathrm{ov}(ua, v) = \begin{cases} \mathrm{ov}(u, v) \cdot a & \textit{if } \mathrm{ov}(u, v) \cdot a \leq_{\mathrm{pref}} v \\ \mathrm{bord}(\mathrm{ov}(u, v) \cdot a) & \textit{otherwise.} \end{cases}$$

Sequential Transducers (see e.g. Sakarovitch, 2009, Section V.1.2). Formally, a sequential transducer from Σ to Δ is a tuple $\mathcal{T} = \langle Q, \Sigma, \Delta, q_0, \delta, \eta, \iota, \rho \rangle$ where $\delta : Q \times \Sigma \to Q$ is a partial transition function, $\eta : Q \times \Sigma \to \Delta^*$ a partial transition output function with the same domain as δ, i.e. $\mathrm{dom}(\delta) = \mathrm{dom}(\eta)$, $\iota \in \Delta^*$ is an initial output, and $\rho : Q \to \Delta^*$ is a partial final output function. $\mathcal{T}$ defines a partial *sequential function* $[\![\mathcal{T}]\!] : \Sigma^* \to \Delta^*$ with

$$[\![\mathcal{T}]\!](w) \stackrel{\mathrm{def}}{=} \iota \cdot \eta(q_0, w) \cdot \rho(\delta(q_0, w))$$

for all w in Σ^* for which $\delta(q_0, w)$ and $\rho(\delta(q_0, w))$ are defined, where $\eta(q, \varepsilon) = \varepsilon$ and $\eta(q, wa) = \eta(q, w) \cdot \eta(\delta(q, w), a)$ for all w in Σ^* and a in Σ.

Let us note $\mathcal{T}_{(q)}$ for the sequential transducer with q for initial state. We write $u \wedge v$ for the longest common prefix of strings u and v; the longest common prefix of all the outputs from state q can be written formally as $\bigwedge_{v \in \Sigma^*} [\![\mathcal{T}_{(q)}]\!](v)$. A sequential transducer is *normalized* if this value is ε for all $q \in Q$ such that $\mathrm{dom}([\![\mathcal{T}_{(q)}]\!]) \neq \emptyset$; any sequential transducer can be normalized.

The *translation* of a sequential function f by a word w in Σ^* is the sequential function $w^{-1}f$ with

$$\mathrm{dom}(w^{-1}f) \stackrel{\mathrm{def}}{=} w^{-1}\mathrm{dom}(f) \qquad w^{-1}f(u) \stackrel{\mathrm{def}}{=} \left(\bigwedge_{v \in \Sigma^*} f(wv) \right)^{-1} \cdot f(wu)$$

for all u in $\mathrm{dom}(w^{-1}f)$. As in the finite automata case where minimal automata are isomorphic with residual automata, the minimal sequential transducer for a sequential function f is defined as the *translation transducer* $\langle Q, \Sigma, \Delta, q_0, \delta, \eta, \iota, \rho \rangle$, where

- $Q \stackrel{\mathrm{def}}{=} \{w^{-1}f \mid w \in \Sigma^*\}$ (which is finite),
- $q_0 \stackrel{\mathrm{def}}{=} \varepsilon^{-1}f$,

- $\iota \stackrel{\text{def}}{=} \bigwedge_{v\in\Sigma^*} f(v)$ if $\text{dom}(f) \neq \emptyset$ and $\iota = \varepsilon$ otherwise,
- $\delta(w^{-1}f, a) \stackrel{\text{def}}{=} (wa)^{-1}f$,
- $\eta(w^{-1}f, a) \stackrel{\text{def}}{=} \bigwedge_{v\in\Sigma^*} (w^{-1}f)(av)$ if $\text{dom}((wa)^{-1}f) \neq \emptyset$ and $\eta(w^{-1}f, a) = \varepsilon$ otherwise, and
- $\rho(w^{-1}f) \stackrel{\text{def}}{=} (w^{-1}f)(\varepsilon)$ if $\varepsilon \in \text{dom}(w^{-1}f)$, and is otherwise undefined.

3.2 Main Construction

Here is the definition of our transducer for a contextual rule:

Definition 3.2 (Transducer of a Contextual Rule)**.** The sequential transducer $\mathcal{T}_r$ associated with a contextual rule $r = u \to v$ with $u \neq \varepsilon$ is defined as $\mathcal{T}_r \stackrel{\text{def}}{=} \langle \text{Pref}(u), \Sigma, \Sigma, \varepsilon, \delta, \eta, \varepsilon, \rho\rangle$ with the set of prefixes of u as state set, ε as initial state and initial output, and for all a in Σ and w in $\text{Pref}(u)$,

$$\delta(w, a) \stackrel{\text{def}}{=} \begin{cases} wa & \text{if } wa \leq_{\text{pref}} u \\ w & \text{if } w = u \\ \text{bord}(wa) & \text{otherwise} \end{cases}$$

$$\rho(w) \stackrel{\text{def}}{=} \begin{cases} \varepsilon & \text{if } w \leq_{\text{pref}} (u \wedge v) \\ (u \wedge v)^{-1} \cdot w & \text{if } (u \wedge v) <_{\text{pref}} w <_{\text{pref}} u \\ \varepsilon & \text{otherwise, i.e. if } w = u \end{cases}$$

$$\eta(w, a) \stackrel{\text{def}}{=} \begin{cases} a & \text{if } wa \leq_{\text{pref}} (u \wedge v) \\ \varepsilon & \text{if } (u \wedge v) <_{\text{pref}} wa <_{\text{pref}} u \\ (u \wedge v)^{-1} \cdot v & \text{if } wa = u \\ a & \text{if } w = u \\ \rho(w)a \cdot \rho(\text{bord}(wa))^{-1} & \text{otherwise.} \end{cases}$$

For instance, the sequential transducer for the rule $ababb \to abbbb$ is shown in Figure 1 (one can check that $ababb \wedge abbbb = ab$, $\text{bord}(b) = \varepsilon$, $\text{bord}(aa) = a$, $\text{bord}(abb) = \varepsilon$, $\text{bord}(abaa) = a$, and $\text{bord}(ababa) = aba$). The intuition behind the definition of $\eta(w, a)$ is to decompose the rewriting according to $u \to v$ into four phases:

1. while in the common prefix $u \wedge v$ of u and v, implement the identity function (states ε, a, and ab in Figure 1),
2. as soon as we start reading a symbol of u that does not match that of v (upon reading a in state ab in Figure 1), we stop outputting symbols and wait for the whole of u to be read,
3. if u has been read, we output the remaining rewritten string $(u \wedge v)^{-1} \cdot v$ we had been saving (upon reading b in state $abab$ in Figure 1),
4. after having read the first occurrence of u in full, we merely implement the identity again (state $ababb$ in Figure 1).

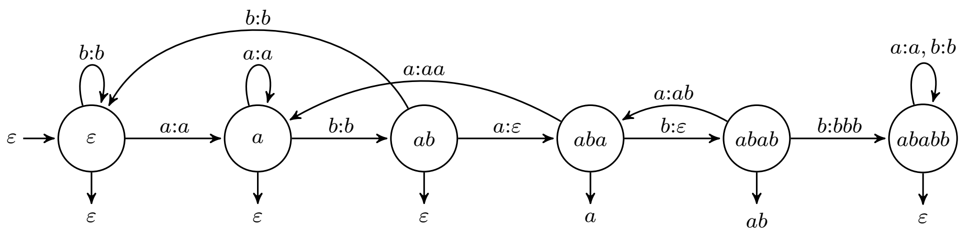


Figure 1: The sequential transducer constructed for $ababb \to abbbb$.

5. If on the other hand we realize that u cannot be read after all in some state w upon reading some a (e.g. transition on a in state aba in Figure 1), we need to flush the missing output $(u \wedge v)^{-1} \cdot w = \rho(w)$, minus the saved output if the state we reach is itself in phase 2.

It remains to show that this sequential transducer is indeed the minimal normalized sequential transducer for $[\![r]\!]$.

Proposition 3.3 (Correctness). *Let $r = u \to v$ with $u \neq \varepsilon$. Then $[\![\mathcal{T}_r]\!] = [\![r]\!]$.*

Proof. Let us first consider the case of input words in $\Sigma^* \backslash (\Sigma^* \cdot u \cdot \Sigma^*)$:

Claim 3.3.1. For all w in $\Sigma^* \backslash (\Sigma^* \cdot u \cdot \Sigma^*)$,

$$\delta(\varepsilon, w) = \mathrm{ov}(w, u) \qquad \eta(\varepsilon, w) = w \cdot \rho(\mathrm{ov}(w, u))^{-1} .$$

Proof of the claim. By induction on w: since $u \neq \varepsilon$, the base case is $w = \varepsilon$ with

$$\delta(\varepsilon, \varepsilon) = \varepsilon = \mathrm{ov}(\varepsilon, u) \qquad \eta(\varepsilon, \varepsilon) = \varepsilon = \varepsilon \cdot \varepsilon^{-1} = \varepsilon \cdot \rho(\varepsilon)^{-1} .$$

For the induction step, we consider wa in $\Sigma^* \backslash (\Sigma^* \cdot u \cdot \Sigma^*)$ for some w in Σ^* and a in Σ:

$$\begin{aligned} \delta(\varepsilon, wa) &= \delta(\delta(\varepsilon, w), a) && \text{(by def.)} \\ &= \delta(\mathrm{ov}(w, u), a) && \text{(by ind. hyp.)} \end{aligned}$$

where by definition of δ, we have $\delta(\mathrm{ov}(w, u), a) = \mathrm{ov}(w, u) \cdot a$ if $\mathrm{ov}(w, u) \cdot a \leq_{\mathrm{pref}} u$ and $\delta(\mathrm{ov}(w, u), a) = \mathrm{bord}(\mathrm{ov}(w, u) \cdot a)$ otherwise (the case $\mathrm{ov}(w, u) = u$ is impossible since w in $\Sigma^* \backslash (\Sigma^* \cdot u \cdot \Sigma^*)$). In all cases:

$$\begin{aligned} \delta(\varepsilon, wa) &= \mathrm{ov}(wa, u) && \text{(by Fact 3.1)} \\ \eta(\varepsilon, wa) &= \eta(\varepsilon, w) \cdot \eta(\delta(\varepsilon, w), a) && \text{(by def.)} \\ &= w \cdot \rho(\delta(\varepsilon, w))^{-1} \cdot \eta(\delta(\varepsilon, w), a) && \text{(by ind. hyp.)} \\ &= w \cdot \rho(w')^{-1} \cdot \eta(w', a) ; && \text{(by setting } w' = \delta(\varepsilon, w)\text{)} \end{aligned}$$

we need to do a case analysis for this last equation:

Case $w'a \not\leq_{\mathrm{pref}} u$ Then $\eta(w', a) = \rho(w') \cdot a \cdot \rho(\mathrm{border}(w'a))^{-1}$, which yields

$$\begin{aligned} \eta(\varepsilon, wa) &= w \cdot \rho(w')^{-1} \cdot \rho(w') \cdot a \cdot \rho(\delta(\varepsilon, wa))^{-1} && \text{(by Fact 3.1)} \\ &= wa \cdot \rho(\delta(\varepsilon, wa))^{-1} . \end{aligned}$$

Case $w'a <_{\mathrm{pref}} u$ Then $\delta(\varepsilon, wa) = w'a$, and we need to further distinguish between several cases:

$w'a \leq_{\text{pref}} (u \wedge v)$ then $\rho(w') = \varepsilon$, $\eta(w', a) = a$, and $\rho(w'a) = \varepsilon$, thus

$$\eta(\varepsilon, wa) = wa = wa \cdot \varepsilon^{-1} = wa \cdot \rho(w'a)^{-1} \ ,$$

$w' = (u \wedge v)$ then $\rho(w') = \varepsilon$, $\eta(w', a) = \varepsilon$, and $\rho(w'a) = (u \wedge v)^{-1} \cdot w'a = a$,

$$\eta(\varepsilon, wa) = w = wa \cdot a^{-1} = wa \cdot \rho(w'a)^{-1} \ ,$$

$(u \wedge v) <_{\text{pref}} w'$ then $\rho(w') = (u \wedge v)^{-1} \cdot w'$, $\eta(w', a) = \varepsilon$, and $\rho(w'a) = (u \wedge v)^{-1} \cdot w'a$, thus

$$\begin{aligned}\eta(\varepsilon, wa) &= w \cdot ((u \wedge v)^{-1} \cdot w')^{-1} \\ &= wa \cdot a^{-1} \cdot ((u \wedge v)^{-1} \cdot w')^{-1} \\ &= wa \cdot \rho(w'a)^{-1} \ .\end{aligned}$$

□

The claim yields that $[\![\mathcal{T}_r]\!]$ coincides with $[\![r]\!]$ on words in $\Sigma^* \setminus (\Sigma^* \cdot u \cdot \Sigma^*)$, i.e. is the identity over $\Sigma^* \setminus (\Sigma^* \cdot u \cdot \Sigma^*)$. Then, since $u \neq \varepsilon$, a word in $\Sigma^* \cdot u \cdot \Sigma^*$ can be written as waw' with w in $\Sigma^* \setminus (\Sigma^* \cdot u \cdot \Sigma^*)$, a in Σ with wa in $\Sigma^* \cdot u$, and w' in Σ^*. Let $u = u'a$; the claim implies that

$$\delta(\varepsilon, w) = u' \qquad\qquad \eta(\varepsilon, w) = w \cdot \rho(u')^{-1} \ .$$

Thus, by definition of $\mathcal{T}_r$, $\delta(\varepsilon, wa) = u'a = u$ and thus

$$\eta(\varepsilon, wa) = \eta(\varepsilon, w) \cdot \eta(u', a) = w \cdot \rho(u')^{-1} \cdot (u \wedge v)^{-1} \cdot v \ ;$$

if $(u \wedge v) <_{\text{pref}} u'$

$$\eta(\varepsilon, wa) = w \cdot ((u \wedge v)^{-1} \cdot u')^{-1} \cdot (u \wedge v)^{-1} \cdot v = w \cdot u'^{-1} \cdot v = wa \cdot u^{-1} \cdot v \ ;$$

otherwise i.e. if $u' = (u \wedge v)$:

$$\eta(\varepsilon, wa) = w \cdot u'^{-1} \cdot v = wa \cdot u^{-1} \cdot v \ .$$

Thus in all cases $[\![\mathcal{T}_r]\!](wa) = [\![r]\!](wa)$, and since $\mathcal{T}_r$ starting in state u (i.e. $\mathcal{T}_{r(u)}$) implements the identity over Σ^*, we have more generally $[\![\mathcal{T}_r]\!] = [\![r]\!]$. □

Lemma 3.4 (Normality). *Let $r = u \to v$. Then $\mathcal{T}_r$ is normalized.*

Proof. Let $w \in \text{Prefix}(u)$ be a state of $\mathcal{T}_r$; let us show that $\bigwedge [\![\mathcal{T}_{r(w)}]\!](\Sigma^*) = \varepsilon$.

If $(u \wedge v) <_{\text{pref}} w <_{\text{pref}} u$ let $u' = w^{-1}u \in \Sigma^+$, and consider the two outputs

$$\begin{aligned}[\![\mathcal{T}_{r(w)}]\!](u') &= \eta(w, u')\rho(u) = (u \wedge v)^{-1}v \\ [\![\mathcal{T}_{r(w)}]\!](\varepsilon) &= \rho(w) = (u \wedge v)^{-1}w \ .\end{aligned}$$

Since $(u \wedge v) <_{\text{pref}} u$ we can write u as $(u \wedge v)au''u'$, and either $v = (u \wedge v)bv'$ or $v = u \wedge v$, for some $a \neq b$ in Σ and u'', v' in Σ^*; this yields $w = (u \wedge v)au''$ and thus $[\![\mathcal{T}_{r(w)}]\!](u') \wedge [\![\mathcal{T}_{r(w)}]\!](\varepsilon) = \varepsilon$.

otherwise $\rho(w) = \varepsilon$, which yields the lemma. □

Proposition 3.5 (Minimality). *Let $r = u \to v$ with $u \neq \varepsilon$ and $u \neq v$. Then $\mathcal{T}_r$ is the minimal sequential transducer for $[\![r]\!]$.*

Proof. Let $w <_{\text{pref}} w'$ be two different states in $\text{Prefix}(u)$; we proceed to prove that $[\![w^{-1}\mathcal{T}_r]\!] \neq [\![w'^{-1}\mathcal{T}_r]\!]$, hence that no two states of $\mathcal{T}_r$ can be merged. By Lemma 3.4 it suffices to prove that $[\![\mathcal{T}_{r(w)}]\!] \neq [\![\mathcal{T}_{r(w')}]\!]$, thus to exhibit some $x \in \Sigma^*$ such that $[\![\mathcal{T}_{r(w)}]\!](x) \neq [\![\mathcal{T}_{r(w')}]\!](x)$. We perform a case analysis:

if $w' \leq_{\text{pref}} (u \wedge v)$ then $w <_{\text{pref}} (u \wedge v)$ thus $[\![\mathcal{T}_{r(w)}]\!](x) = x$ for all $x \notin w^{-1} \cdot \Sigma^* \cdot u \cdot \Sigma^*$; consider

$$[\![\mathcal{T}_{r(w)}]\!](w'^{-1}u) = w'^{-1}u \neq w'^{-1}v = [\![\mathcal{T}_{r(w')}]\!](w'^{-1}u) ;$$

if $w \leq_{\text{pref}} (u \wedge v)$ and $w' = u$ then $[\![\mathcal{T}_{r(w')}]\!](x) = x$ for all x and we consider

$$[\![\mathcal{T}_{r(w)}]\!](w^{-1}u) = w^{-1}v \neq w^{-1}v = [\![\mathcal{T}_{r(w')}]\!](w^{-1}u) ;$$

otherwise that is if $w \leq_{\text{pref}} (u \wedge v)$ and $(u \wedge v) <_{\text{pref}} w' <_{\text{pref}} u$, or $(u \wedge v) <_{\text{pref}} w <_{\text{pref}} w' \leq_{\text{pref}} u$, we have $\rho(w) \neq \rho(w')$ thus

$$[\![\mathcal{T}_{r(w)}]\!](\varepsilon) \neq [\![\mathcal{T}_{r(w')}]\!](\varepsilon) .$$

□

4 Conclusion

The results of the previous section yield (the cases $u = \varepsilon$ and $u = v$ are trivial):

Theorem 4.1. *Given a contextual rule $r = u \to v$, one can construct directly the minimal normalized sequential transducer $\mathcal{T}_r$ of size $O(|r|)$ for $[\![r]\!]$.*

The remaining open question is whether we can obtain good upper bounds on the size of the sequential transducer $\mathcal{T}_\mathcal{C}$ for a cascade $\mathcal{C} = r_1 \cdots r_n$: string matching automata enjoy nice combinatorial properties, but it seems unlikely for better bounds than $O(\prod_{i=1}^n |r_i|)$ to exist: indeed, the membership problem, i.e. given u, v in Σ^* and a cascade $\mathcal{C}$ whether $[\![\mathcal{C}]\!](u) = v$, is easily seen to be PTime-complete for length-preserving rules, but a LogSpace construction of a sequential transducer for $[\![\mathcal{C}]\!]$ would result in a LogSpace algorithm for the problem.

References

Brill, E., 1992. A simple rule-based part of speech tagger. In *ANLP '92, third Conference on Applied Natural Language Processing*, pages 152–155. ACL Press. doi:10.3115/974499.974526.

Crochemore, M. and Hancart, C., 1997. Automata for matching patterns. In Rozenberg, G. and Salomaa, A., editors, *Handbook of Formal Languages*, volume 2. Linear Modeling: Background and Application, chapter 9, pages 399–462. Springer. ISBN 3-540-60648-3.

Kaplan, R.M. and Kay, M., 1994. Regular models of phonological rule systems. *Computational Linguistics*, 20(3):331–378. http://www.aclweb.org/anthology/J94-3001.pdf.

Mihov, S. and Schultz, K.U., 2007. Efficient dictionary-based text rewriting using subsequential transducers. 13(4):353–381. doi:10.1017/S1351324905004092.

Roche, E. and Schabes, Y., 1995. Deterministic part-of-speech tagging with finite-state transducers. *Computational Linguistics*, 21(2):227–253. http://www.aclweb.org/anthology/J95-2004.pdf.

Sakarovitch, J., 2009. *Elements of Automata Theory*. Cambridge University Press. ISBN 978-0-521-84425-3.

Schützenberger, M.P., 1977. Sur une variante des fonctions séquentielles. *Theoretical Computer Science*, 4(1):47–57. doi:0.1016/0304-3975(77)90055-X.

Simon, I., 1994. String matching algorithms and automata. In Karhumäki, J., Maurer, H., and Rozenberg, G., editors, *Results and Trends in Theoretical Computer Science: Colloquium in Honor of Arto Salomaa*, volume 812 of *Lecture Notes in Computer Science*, pages 386–395. Springer. ISBN 978-3-540-58131-4. doi:10.1007/3-540-58131-6_61.