Determinization of transducers over finite and infinite words

MARIE-PIERRE BÉAL
Institut Gaspard Monge,
Université de Marne-la-Vallée
http://www-igm.univ-mlv.fr/~beal/

OLIVIER CARTON
Institut Gaspard Monge and CNRS
Université de Marne-la-Vallée
http://www-igm.univ-mlv.fr/~carton/

March 27, 2001

Abstract

We study the determinization of transducers over finite and infinite words. The first part of the paper is devoted to finite words. We recall the characterization of subsequential functions due to Choffrut. We describe here a known algorithm to determinize a transducer.

In the case of infinite words, we consider transducers with all their states final. We give an effective characterization of sequential functions over infinite words. We describe an algorithm to determinize transducers over infinite words. This part contains the main novel results of the paper.

1 Introduction

The aim of this paper is the study of determinization of transducers, that is of machines realizing rational transductions. A transducer is a finite state automaton (or a finite state machine) whose edges are labeled by pairs of words taken in finite alphabets. The first component of each pair is called the input label. The second one the output label. The transducers that we consider have accepting (or final) states. Such transducers are sometimes called a-transducers (a for accepting). The rational relation defined by a transducer is the set of pairs of words which are labels of an accepting path in the transducer. We assume that the relations defined by our transducers are functions. This is a decidable property.

The study of transducers has many applications. Transducers are used to model coding schemes (compression schemes, convolutional coding schemes, coding schemes for constrained channels, for instance). They are widely used in computer arithmetic [17] and in natural language processing [25]. Transducers are also used in programs analysis [14]. The determinization of a transducer is the construction of another transducer which defines the same function and has a deterministic (or right resolving) input automaton. Such transducers allow a sequential encoding and thus are called sequential transducers.

In the first part of the paper, we present a short survey of the determinization of transducers realizing functions over finite words. Our transducers may have final states. We present some known results about subsequential functions, that is functions that can be realized by transducers with a deterministic input but that may have an output function defined on states. The notion of subsequential functions has been introduced by Schützenberger [28]. We recall the characterization of subsequential functions obtained by Choffrut [11, 12]. This characterization gives a decision procedure for the subsequentiality of functions defined by a transducer. It has been proved in [30, 31] that this can be decided in polynomial time. We give another proof of this result which is a consequence of the decidability in polynomial time of functionality over infinite words [10]. Another proof of the same result is given in [4]. The decidability of functionality was already proved by Gire [18]. We also describe the algorithm to determinize a transducer. This algorithm takes a real-time transducer which realizes a subsequential function and outputs a subsequential transducer. This algorithm is actually contained in the proof of Choffrut [11, 12] (see also [5, p. 109–110]). This algorithm has also been described by Mohri [22] and Roche and Shabes [25, p. 223–233].

The determinization of a transducer realizing a subsequential function f provides a subsequential transducer realizing f. If the function is sequential, this subsequential transducer can be transformed into a sequential one. This can be obtained by the normalization of a transducer introduced by Choffrut [12, 13]. Efficient algorithms that compute the normalization have been given in [21, 23], [8, 9] and [2].

In the second part of the paper, we consider transducers and functions over infinite words and our transducers have all their states final. The reason why we assume that all states are final is that the case of transducers with final states seems to be much more complex. Indeed, the determinization of automata over infinite words is already very difficult [26]. In particular, it is not true that any rational set of infinite words is recognized by a deterministic automaton with final states. Other accepting conditions, as the Muller condition for instance, must be used.

We first give an effective characterization of sequential functions over infinite words. This characterization extends to infinite words the twinning property introduced by Choffrut [11]. We prove that a function is sequential if it is a continuous map whose domain can be recognized by a deterministic Büchi automaton, and such that the transducer obtained after removing some special states has the twinning property. These conditions can be simplified in the case where the transducer has no cycling path with an empty output label. We use this characterization to describe an algorithm checking whether a function realized by a transducer is sequential. This algorithm becomes polynomial when the transducer has no cycling path with an empty output label. Finally, we give an algorithm to determinize a real-time transducer. The algorithm can be easily adapted to the case when the transducer is not real-time. The algorithm is much more complex than in the case of finite words. It is the main result of the paper.

These determinizations do not preserve the dynamic properties of the transducers as the locality of its output. We mention that in [19], an algorithm is given to determinize transducers over bi-infinite words that have a right closing input (or that are n-deterministic or deterministic with a finite delay in the input) and a local output (see also [20, p. 143] and [1, p. 110–115]). This algorithm preserves the locality of the output. These features are important for coding applications.

The paper is organized as follows. Section 2 is devoted to transducers over finite words. Basic notions of transducers of rational functions are defined at the beginning of this section. The characterization of subsequential functions is recalled in Section 2.1 while the algorithm for determinization of transducers is described in Section 2.2. The characterization of sequential functions among subsequential ones is recalled in Section 2.3. Section 3 is devoted to transducers over infinite words. We give in Section 3.1 a characterization of sequential functions while the algorithm for determinization of transducers is described in Section 3.2. In both cases of finite and infinite words, we give examples of determinization of transducers.

Part of the results of the present paper was presented at the conference ICALP'2000 [3].

2 Transducers over finite words

In the sequel, A and B denote finite alphabets. The free monoid A^* is the set of finite words or sequences of letters of A. The empty word is denoted by ε. We denote the fact that a finite word u is a prefix of a finite word v by $u \prec v$. The relation $\prec$ is a partial order. If u is a prefix of v, we denote by $u^{-1}v$ the unique word w such that $v = uw$.

A *transducer* over the monoid $A^* \times B^*$ is composed of a set Q of *states*, a set $E \subset Q \times A^* \times B^* \times Q$ of *edges* and two sets $I, F \subset Q$ of *initial* and *final* states. An edge $e = (p, u, v, q)$ from p to q is denoted by $p \xrightarrow{u|v} q$. The state p is the *origin*, u is the *input label*, v is the *output label*, and q is the *end*. Thus, a transducer is the same object as an automaton, except that the labels of the edges are pairs of words instead of letters.

A transducer is often denoted by $\mathcal{A} = (Q, E, I, F)$, or also by (Q, E, I) if all states are final, i.e., $Q = F$.

A *path* in the transducer $\mathcal{T}$ is a sequence

$$p_0 \xrightarrow{u_0|v_0} p_1 \xrightarrow{u_1|v_1} \cdots \xrightarrow{u_n|v_n} p_n$$

of consecutive edges. Its input label is the word $u = u_1u_2 \cdots u_n$ whereas its output label is the word $v = v_1v_2 \cdots v_n$. The path *leaves* p_0 and *ends* in p_n.

The path is often denoted

$$p_0 \xrightarrow{u|v} p_n.$$

A path is *successful* if it leaves an initial state and ends in a final state. The set *recognized* by the transducer is the set of labels of its successful paths, which is actually a relation $R \subset A^* \times B^*$. The transducer computes a function if for any word $u \in A^*$, there exists at most one word $v \in B^*$ such that $(u,v) \in R$. We call it the function *realized* by the transducer. A transducer which realizes a function is sometimes called *single-valued* in the literature. Thus a transducer can be seen as a machine computing nondeterministically output words from input words. We denote by $\mathrm{dom}(f)$ the domain of the function f.

A transducer is *finite* if its set of states and its set of transitions are finite. It is a consequence of Kleene's theorem that a subset of $A^* \times B^*$ is a rational relation if and only if it is the set recognized by a finite transducer.

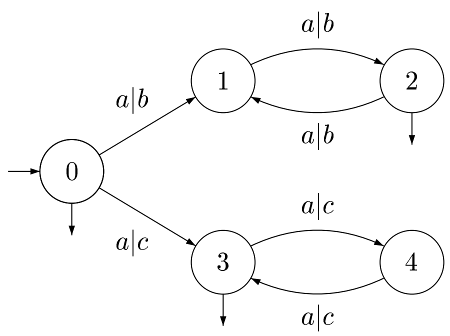


Figure 1: A transducer for the relation $(a^2,b^2)^* \cup (a^2,c^2)^*(a,c)$.

EXAMPLE 1 (from [5]) The automaton of Figure 1 recognizes the relation $(a^2,b^2)^* \cup (a^2,c^2)^*(a,c)$ over the alphabets $A = \{a\}$ and $B = \{b,c\}$. This relation is actually the function which maps a^n to b^n if n is even and to c^n if n is odd.

Let $\mathcal{T}$ be a transducer. The *underlying input automaton* (respectively *underlying output automaton*) of $\mathcal{T}$ is obtained by omitting the output label (respectively input label) of each edge.

A transducer is said to be *real-time* if it is labeled in $A \times B^*$. It can be proved that any rational function can be realized by a real-time transducer. Furthermore, from any transducer realizing a function can be computed in polynomial time an equivalent real-time transducer (see for instance [31, Prop. 1.1]). We say that a transducer $\mathcal{T}$ is *sequential* if it is real-time and if the following conditions are satisfied.

- it has a unique initial state,

- the underlying input automaton is deterministic.

These conditions ensure that for each word $u \in A^*$, there is at most one word $v \in B^*$ such that (u, v) is recognized by $\mathcal{T}$. Thus, the relation computed by $\mathcal{T}$ is a partial function from A^* into B^*. A function is *sequential* if it can be realized by a sequential transducer.

Remark 2 In [16, p. 299], [5] and [6], it is assumed that all states of a sequential transducer are final. We follow the definition of Choffrut [11, 12] where sequential transducer may have final states. Thus, some characterizations that we give below differ from those presented in [5] for this reason. When all states are final, the domain of a sequential function is prefix closed, i.e., if uv belongs to the domain then u also belongs to the domain. As our definition allows final states, the domain of a sequential function is not necessarily prefix closed.

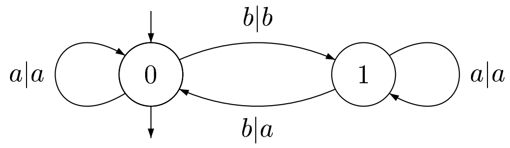


Figure 2: A sequential transducer.

Example 3 Let $A = B = \{a, b\}$ be the input and the output alphabets. The transducer of Figure 2, whose initial state is 0, is sequential. It replaces by a those b's which appear after an odd number of b. On the contrary, the transducer of Example 1 is not sequential. Actually, the function computed by this transducer is not sequential. Indeed, one may verify that if f is sequential, and if u and v are two words of $\mathrm{dom}(f)$ such that $u \prec v$, then $f(u) \prec f(v)$.

Remark 4 If f is a sequential function and if $f(\varepsilon)$ is defined, then $f(\varepsilon) = \varepsilon$. To remove this restriction, it is possible to add an *initial word* associated with the initial state. This word is output before any computation. This initial word is necessary to get the unicity of a minimal sequential transducer [28, 12].

A *subsequential transducer* $(\mathcal{A}, \rho)$ over $A^* \times B^*$ is a pair composed of a sequential transducer $\mathcal{A}$ over $A^* \times B^*$ with F as set of final states, and of a function $\rho : F \to B^*$. The function f computed by $(\mathcal{A}, \rho)$ is defined as follows. Let u be a word in A^*. The value $f(u)$ is defined if and only if there is a path $i \xrightarrow{u|v} q$ in $\mathcal{A}$ with input label u, from the initial state i to a final state q. In this case, one has $f(u) = v\rho(q)$. Thus, the function ρ is used to append a word to the output at the end of the computation. A function is *subsequential* if it can be realized by a subsequential transducer.

REMARK 5 Any sequential function is subsequential. It suffices to consider the function ρ such that $\rho(q) = \varepsilon$ for any final state q.

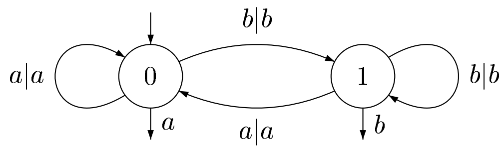


Figure 3: A subsequential transducer.

EXAMPLE 6 The function f realized by the subsequential transducer pictured in Figure 3 appends to each word its last letter. The word u is mapped to ua if it ends with an a and it is mapped to ub if it ends with a b. This function is subsequential but it is not sequential. Indeed, for any word w, $f(wa)$ is not a prefix of $f(wab)$.

2.1 Subsequential functions

In this section, we present some known results about subsequential functions. We recall the characterization of subsequential functions obtained by Choffrut [11, 12]. It is known that it is decidable whether a function realized by a transducer is subsequential. It has been proved in [30, 31] that this can be decided in polynomial time. We give here another proof of this result which is a consequence of the decidability in polynomial time of functionality over infinite words [10]. We also describe the algorithm to determinize a real-time transducer. This algorithm takes a transducer which realizes a subsequential function and outputs a subsequential transducer. This algorithm is actually contained in the proof of Choffrut [11, 12] and [5, p. 109–110]. It has also been described by Mohri [21, 23] and Roche and Shabes [25, p. 223–233].

If the function is actually sequential, this subsequential transducer is again transformed in a sequential transducer by the algorithm described in Section 2.3.

We give below two characterizations of subsequential functions that have been obtained by Choffrut (see [11, 12] and [5, p. 105]). The first characterization is intrinsic to the function. It is based on metric properties of the function. The second characterization is effective. It is based on a property called twinning property of a transducer realizing the function. As it has been shown in [30, 31], this property can be decided in polynomial time.

Some notation is needed to state the characterization of subsequential functions. We first introduce a distance d on finite words. Let u, v be two finite words, we denote by d the distance such that

$$d(u, v) = |u| + |v| - 2|u \wedge v|,$$

where $u \wedge v$ is the longest common prefix of u and v (see [5, p. 104]).

A partial function $f : A^* \to B^*$ has *bounded variation* if and only if:

$$\forall k \geq 0 \; \exists K \geq 0 \; \forall u, v \in \mathrm{dom}(f) \quad d(u,v) \leq k \Rightarrow d(f(u), f(v)) \leq K.$$

The decidability of the subsequentiality is essentially based on the following notion introduced by Choffrut [12, p. 133] (see also [5, p. 128]). Two states q and q' of a transducer are said to be *twinned* iff for any pair of paths

$$i \xrightarrow{u|u'} q \xrightarrow{v|v'} q$$
$$i' \xrightarrow{u|u''} q' \xrightarrow{v|v''} q'$$

where i and i' are two initial states, the output labels satisfy the following property. Either $v' = v'' = \varepsilon$ or there exists a finite word w such that either $u'' = u'w$ and $wv'' = v'w$, or $u' = u''w$ and $wv' = v''w$. The latter case is equivalent to the following two conditions:

(i) $|v'| = |v''|$,

(ii) $u'v'^{\omega} = u''v''^{\omega}$

A transducer has the *twinning property* if any two states are twinned.

Proposition 7 (Choffrut) *Let $f : A^* \to B^*$ be a partial function realized by a transducer $\mathcal{T}$. The following three propositions are equivalent.*

- *The function f is subsequential.*
- *The function f has bounded variation.*
- *The transducer $\mathcal{T}$ has the twinning property.*

The equivalence between the first two statements is an intrinsic characterization of subsequential functions among rational functions. It actually suffices to suppose that the inverse image by f of any rational set is still rational and that f has bounded variation to insure that f is subsequential. However, we are in his paper interested in effective matters and we always suppose that a function on words is given by a transducer which realizes it. The equivalence between the last two statements allows us to decide the subsequentiality. The proof of this equivalence is essentially the proof of Lemma 16 below.

We mention here another characterization of the subsequentiality. For a partial function $f : A^* \to B^*$, define the right congruence $\sim$ on A^* by $u \sim u'$ iff there are two words finite words v and v' such that the following two properties hold for any finite word w. First, the word uw is in the domain of f iff $u'w$ is in the domain of f. Second, if uw and $u'w$ are in the domain, then $v^{-1}f(uw) = v'^{-1}f(vw)$. The function f is then subsequential iff the right congruence $\sim$ has finite index. In that case, the congruence $\sim$ allows one to construct directly a subsequential transducer realizing f. Furthermore, this sequential transducer is minimal in the sense that any other subsequential transducer realizing f can be projected onto this one. The algorithm presented in Section 2.2 allows one to compute effectively the right congruence $\sim$.

EXAMPLE 8 We have already mentioned in Example 3 that the function $(a^2, b^2)^* \cup (a^2, c^2)^*(a, c)$ of Example 1 is not sequential. Actually, this relation is not subsequential as it can be easily shown with Proposition 7. Indeed, the function does not have bounded variation. For any integer n, one has

$$d(a^{2n}, a^{2n+1}) = 1 \quad \text{while} \quad d(b^{2n}, c^{2n+1}) = 4n + 1.$$

We now give two decidability results about rational relations. The first one is due to Schützenberger [27] (see also [7]). The second one is due to Choffrut [11, 12] (see also [5, p. 128]).

PROPOSITION 9 (SCHÜTZENBERGER) *Let $\mathcal{T}$ be a transducer over $A^* \times B^*$. It is decidable whether the relation defined by $\mathcal{T}$ is a function.*

Choffrut also proved the decidability of the subsequentiality. He showed that it suffices to check the twinning property when the lengths of the words u and v are bounded by the square of the number of states [12, p. 133] and [5, p. 128]. However, this algorithm does not seem to be polynomial.

PROPOSITION 10 (CHOFFRUT) *Let $\mathcal{T}$ be a transducer labeled in $A^* \times B^*$ which realizes a function f, then the subsequentiality of f is decidable.*

The following result is due to Weber and Klemm [30, 31].

PROPOSITION 11 *Let f be the function realized by a transducer labeled in $A^* \times B^*$. It is decidable in polynomial time whether f is subsequential.*

The proof of the proposition follows directly from Proposition 7 and from the following lemma. We give below another proof based on the decidability in polynomial time of the functionality over infinite words. A third proof is given in [4].

LEMMA 12 *The twinning property of a transducer is decidable in polynomial time.*

Proof Let $\mathcal{T} = (Q, E, I, F)$ be a transducer. We decide the twinning property of $\mathcal{T}$ in two steps. We first decide in polynomial time the condition (i) and then the condition (ii).

We define an automaton $\mathcal{A}$ whose states are the pairs of states of $\mathcal{T}$ and whose edges are labeled by integers. There is an edge $(p, p') \xrightarrow{n} (q, q')$ iff there are two edges $p \xrightarrow{a|u} q$ and $p' \xrightarrow{a|u'} q'$ in $\mathcal{A}$ such that $n = |u'| - |u|$. The label of a path in $\mathcal{A}$ is the sum of the labels of the edges of the path. We claim that the transducer $\mathcal{T}$ satisfies condition (i) iff the label of any cycle around a pair (q, q') accessible from some pair (i, i') for two initial states i and i', is equal to zero. This can be done by a depth-first search.

We assume that the transducer already satisfies condition (i). This first condition insures that the output label v' is empty iff v'' is empty. The condition (ii) is then equivalent to the functionality of the relation on infinite words defined by the transducer $\mathcal{T}$ with all states being final. Indeed, it is clear that if the relation defined by $\mathcal{T}$ is a function, then any two states are twinned. Conversely, if this relation is not a function, there exist two infinite paths labeled by $x|y$ and $x|y'$ with $y \neq y'$. Let $p_0p_1p_2\ldots$ and $p'_0p'_1p'_2\ldots$ be the states visited by the two paths. Let k an index such that $y_k \neq y'_k$. There exist indices $m > n$ such that $(p_m, p'_m) = (p_n, p'_n)$. Moreover, n may be chosen great enough such that the outputs along the paths from the initial state to p_n and p'_n have a length greater than k. Then the states p_m and p'_m are not twinned.

It is decidable in polynomial time whether a relation on infinite words realized by a transducer is a function [10]. □

2.2 Determinization of transducers over finite words

In this section, we describe an algorithm which determinizes a real-time transducer which has the twinning property. This algorithm proves that the conditions of Proposition 7 are sufficient.

Let $\mathcal{T} = (Q, E, I, F)$ be a real-time transducer, that is labeled in $A \times B^*$, realizing a function which is subsequential. We give below an algorithm to determinize the transducer $\mathcal{T}$, that is, which produces a subsequential transducer realizing f. The algorithm is exponential in the number of states of $\mathcal{T}$. The determinization of an automaton is already exponential.

We define a subsequential transducer $\mathcal{D}$ as follows. A state P of $\mathcal{D}$ is a set of pairs (q, w) where q is a state of $\mathcal{T}$ and w is a word over B. We now describe the transitions of $\mathcal{T}$. Let P be state of $\mathcal{D}$ and let a be a letter. The pair (P, a) determines a set R defined by

$$R = \{(q', wu) \mid \text{there exist } (q, w) \in P \text{ and } q \xrightarrow{a|u} q' \in E\}.$$

If R is empty, there is no transition from P input labeled by a. Otherwise, let v be the longest common prefix of the words wu for $(q', wu) \in R$ and

$$P' = \{(q', w') \mid (q', vw') \in R\}.$$

There is then a transition $P \xrightarrow{a|v} P'$. The initial state of $\mathcal{D}$ is the set $J = \{(i, \varepsilon) \mid i \in I\}$ where I is the set of initial states of $\mathcal{T}$. It follows from the definition of the transitions of $\mathcal{D}$ that if P is a state accessible from the initial state, the longest common prefix of the words w for $(q, w) \in P$ is the empty word. We only keep in $\mathcal{D}$ the accessible part from the initial state. The transducer $\mathcal{D}$ has a deterministic input automaton.

The following lemma states the main property of the transitions of $\mathcal{D}$.

LEMMA 13 *Let u be a finite word. Let $J \xrightarrow{u|v} P$ be the unique path in $\mathcal{D}$ with input label u from the initial state. Then, the state P is equal to*

$$P = \{(q, w) \mid \textit{there exists a path } i \xrightarrow{u|vw} q \textit{ in } \mathcal{T} \textit{ where } i \in I\}.$$

Proof The proof of the lemma is by induction on the length of u. Let us consider the following path in $\mathcal{D}$

$$J \xrightarrow{u|v} P \xrightarrow{a|t} P'$$

where a is a letter. Let (q', w') be a pair in P'. By the definition of the transitions of $\mathcal{D}$, there is a pair (q, w) in P and a transition $q \xrightarrow{a|t'} q'$ in $\mathcal{T}$ such that $tw' = wt'$. By the induction hypothesis, there is a path $i \xrightarrow{u|vw} q$ in $\mathcal{T}$. Finally, one has $vtw' = vwt'$. $\square$

The preceding lemma has the following consequence. If both pairs (q, w) and (q', w') belong to a state P which is accessible from the initial state and if both q and q' are final states in $\mathcal{T}$, then the equality $w = w'$ necessarily holds. Otherwise, the relation realized by $\mathcal{T}$ is not a function. This remark allows us to define the set of final states of $\mathcal{D}$ and the function ρ. A state P is final if it contains as least one pair (q, w) where q is a final state of $\mathcal{T}$. The function ρ maps such a final state P to the word w.

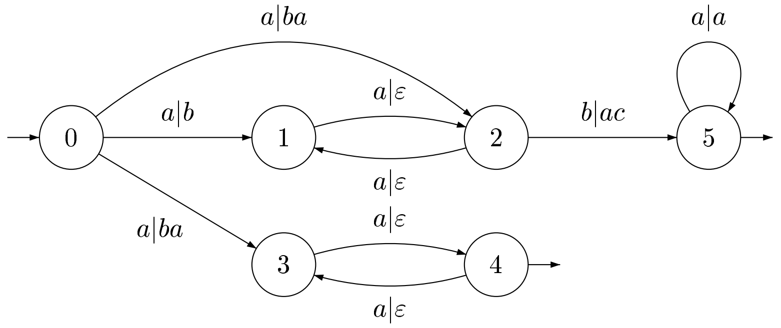


Figure 4: Transducer of Example 14

EXAMPLE 14 Consider the transducer pictured in Figure 4. If the algorithm for determinization is applied to this transducer, one gets the subsequential transducer pictured in Figure 5. This subsequential transducer is transformed into a sequential transducer in Examples 19.

This defines a subsequential transducer which may have an infinite number of states. However, we claim that the bounded variation property of $\mathcal{T}$ implies that the lengths of the words in states of $\mathcal{D}$ are bounded. Thus the number of states of $\mathcal{D}$ is actually finite.

LEMMA 15 *Let v_1, v_2, v'_1 and v'_2 be four finite words such that $|v_2| = |v'_2|$ and $v_1 v_2^\omega = v'_1 {v'_2}^\omega$. For any words v_3 and v'_3,*

$$d(v_1 v_2 v_3, v'_1 v'_2 v'_3) = d(v_1 v_3, v'_1 v'_3).$$

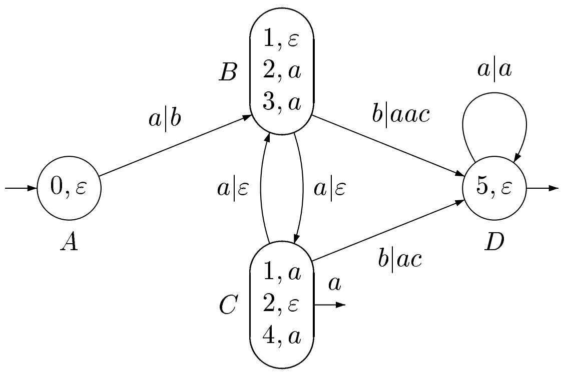


Figure 5: Determinization of the transducer of Figure 4

Proof By symmetry, we may suppose that $|v_1| \leq |v'_1|$. There is then a finite word w such that $v'_1 = v_1 w$ and $wv'_2 = v_2 w$. Thus the word $v'_1 v'_2 v'_3$ is equal to $v_1 v_2 w v'_3$. It follows that

$$d(v_1 v_2 v_3, v'_1 v'_2 v'_3) = d(v_3, wv'_3) = d(v_1 v_3, v'_1 v'_3).$$

□

The following lemma states that if a transducer $\mathcal{T}$ has the twinning property, then the outputs labels of two paths with the same input label have a long common prefix. It proves that if the relation realized by $\mathcal{T}$ is a function, it has bounded variation. The proof is very close to the proof of Proposition 6.4 in [5] but we do not assume that the relation realized by $\mathcal{T}$ is a function. This is useful when transducers realizing relations on infinite words are considered.

LEMMA 16 *Let $\mathcal{T}$ be a transducer which has the twinning property. There is a constant K such that the outputs of two paths $i \xrightarrow{u|v} q$ and $i' \xrightarrow{u|v'} q'$ from two initial states i and i' satisfy*

$$d(v, v') \leq K.$$

Proof Let K be equal to $2n^2 M$ where n is the number of states of the transducer and M is the maximal length of the output label of a transition. We prove $d(v, v') \leq K$ by induction on the length of u. If $|u| \leq n^2$, the result holds by definition of K. Otherwise, both paths can be factorized

$$i \xrightarrow{u_1|v_1} p \xrightarrow{u_2|v_2} p \xrightarrow{u_3|v_3} q$$

$$i' \xrightarrow{u_1|v'_1} p' \xrightarrow{u_2|v'_2} p' \xrightarrow{u_3|v'_3} q'$$

where $u_1 u_2 u_3 = u$, $v_1 v_2 v_3 = v$, $v'_1 v'_2 v'_3 = v'$ and $|u_2| > 0$. By the twinning property, one has $d(v_1 v_2 v_3, v'_1 v'_2 v'_3) = d(v_1 v_3, v'_1 v'_3)$ and the result follows from the induction hypothesis. □

The following lemma states that the lengths of the words w of the pairs (q, w) in the states of $\mathcal{D}$ are bounded. This implies that the number of states of $\mathcal{D}$ is finite.

LEMMA 17 *There is a constant K such that for any pair (q, w) in a state P of $\mathcal{D}$, one has $|w| \leq K$.*

Proof Let $J \xrightarrow{u|v} P$ be a path in $\mathcal{D}$. Let (q, w) be a pair in some state P. By definition of the transitions of $\mathcal{D}$, there is another pair (q', w') in $\mathcal{D}$ such that w and w' have no common prefix. By Lemma 13, there are two paths, $i \xrightarrow{u|vw} q$ and $i' \xrightarrow{u|vw'} q'$ in $\mathcal{T}$. By Lemma 16, there is a constant K such that $d(vw, vw') \leq K$ and thus $|w| \leq K$. $\square$

The following proposition finally states that the subsequential transducer $\mathcal{D}$ is equivalent to the transducer $\mathcal{T}$. It follows directly from Lemma 13 and the definition of the function ρ.

PROPOSITION 18 *The sequential transducer $\mathcal{D}$ realizes the same function f as the transducer $\mathcal{T}$.*

We have already mentioned in Proposition 11 that it can be decided in polynomial time whether a function realized by a transducer is subsequential. The algorithm described above is exponential but it provides another decision procedure. Indeed, Lemma 17 gives a upper bound of the lengths of words which can appear in states of $\mathcal{D}$. By Lemma 16, this upper bound is $2n^2M$ where n is the number of states of $\mathcal{T}$ and M is the maximal length of the output label of a transition of $\mathcal{T}$. Let $\mathcal{T}$ be a transducer realizing a function f. If the algorithm is applied to $\mathcal{T}$, either it stops and gives a subsequential transducer $\mathcal{D}$ or it creates a state P containing a pair (q, w) such that the length of w is greater than $2n^2M$. In the former case, the subsequential transducer $\mathcal{D}$ is equivalent to $\mathcal{T}$ and the function f is subsequential. In the latter case, the function f is not subsequential.

2.3 Sequential functions

The determinization of a transducer realizing a subsequential function f provides a subsequential transducer realizing f. Even if the function f is sequential, the algorithm does not give a sequential transducer but this subsequential transducer can be transformed into a sequential one.

This transformation is based on a normalization of subsequential transducers introduced by Choffrut [12, 13]. This normalization consists in pushing as much as possible the output labels from final states towards the initial state. Algorithms computing the normalized transducer are given in [21, 23], [8, 9] and [2]. The algorithms given in [21, 23] and [2] run in time $O(|E|P)$ where E is the set of transitions of the transducer, and where P is the maximal length of the greatest common prefix of the output labels of paths leaving each state of the transducer. If the normalization is applied to a subsequential transducer, the resulting transducer is sequential iff the function is sequential. Since the normalization can be performed in polynomial time, it can be checked in polynomial time whether a function realized by a subsequential transducer is sequential. It can be shown that a function realized by a subsequential transducer is sequential iff it preserves prefixes. This was already proved in [30, 31] that this property can be checked in polynomial time.

In order to transform a subsequential transducer into a sequential one, it is not necessary to push as much as possible the output labels from final states towards the initial state, as the normalization does. It suffices to push these output labels until the output of all states are empty. Therefore, the algorithm given in [2] can be adapted to meet this requirements. This gives a time complexity of $O(|E|L)$ instead of $O(|E|P)$ where L is the maximal length of the output words.

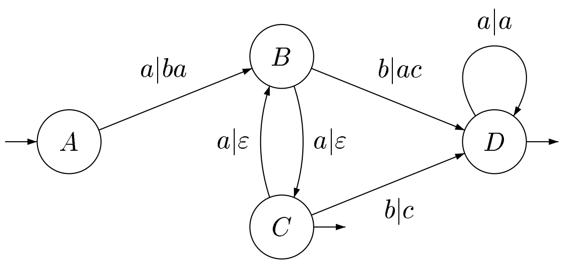


Figure 6: Sequential transducer of Example 19

Example 19 Consider the transducer pictured in Figure 5 where the states have been renamed A, B, C and D. If the normalization is applied to this subsequential transducer, one gets the sequential transducer pictured in Figure 6.

3 Transducers over infinite words

In this section, we consider transducers over infinite words with all states being final. We first give an effective characterization of sequential functions over infinite words. This characterization extends to infinite words the twinning property introduced by Choffrut [11, 12]. We use this characterization to describe an algorithm to check whether a function realized by a transducer is sequential. Finally, we give an algorithm to determinize a transducer.

In this section, we denote by A^ω the set of all (right-)infinite words over the alphabet A. We consider transducers over infinite words. The edges of the transducers are still labeled in $A^* \times B^*$. The transducer has initial states but we suppose that all states are final. Thus we omit the set F of final states in the notation. An infinite path is then *successful* if it leaves an initial state. The relation over infinite words defined by the transducer is the set $R \subset A^\omega \times B^\omega$

of labels of its successful paths. The *domain* of the transducer is the set of infinite words x such that there is some infinite word y such that (x, y) labels a successful path in the transducer. When the transducer realizes a function, its domain is also the domain of the function. A function from A^ω to B^ω is *sequential* if it is realized by a sequential transducer. We point out that the notion of subsequential function is irrelevant in the case of infinite words.

3.1 Characterization of sequential functions

In this section, we characterize functions realized by transducers with all states final that can be realized by sequential transducers. This characterization uses topological properties of the function and some twinning property of the transducer. In this section, we assume that all states of transducers are final.

We first introduce a definition. We define a subset of states which play a particular role in the sequel. We say that a state q of a transducer is non *constant* if there are two paths leaving q labelled by two pairs (x, y) and (x', y') of infinite words such that $y \neq y'$. If a state q is constant, either there is no path leaving q labelled by a pair of infinite words or there is an infinite word y_q called the constant of q such that for any pair (x, y) of infinite words labelling a path leaving q, then $y = y_q$. In the former case, the state q can be removed since it cannot occur in an accepting path labelled by a pair of infinite words. In the sequel, we always assume that such states have been removed. The constant y_q is an ultimately periodic word. It should be noticed that any state accessible from a constant state is also constant. We now state the characterization of sequential functions.

PROPOSITION 20 *Let f be a function realized by a transducer $\mathcal{T}$ with all states final. Let $\mathcal{T}'$ be the transducer obtained by removing from $\mathcal{T}$ all constant states. Then the function f is sequential iff the following three properties hold:*

- *the domain of f can be recognized by a deterministic Büchi automaton,*
- *the function f is continuous,*
- *the transducer $\mathcal{T}'$ has the twinning property.*

Since the function f is realized by a transducer, the domain of f is rational. However, it is not true that any rational set of infinite words is recognized by a deterministic Büchi automaton. Landweber's theorem states that a set of infinite words is recognized by a deterministic Büchi automaton iff it is rational and G_δ [29]. Recall that a set is said to be G_δ is it is equal to a countable intersection of open sets for the usual topology of A^ω.

It is worth pointing out that the domain of a function realized by a transducer may be any rational set although it is supposed that all states of the transducer are final. The final states of a Büchi automaton can be encoded in the outputs of a transducer in the following way. Let $\mathcal{A} = (Q, E, I, F)$ be a Büchi automaton. We construct a transducer $\mathcal{T}$ by adding an output to any transition of $\mathcal{A}$. A transition $p \xrightarrow{a} q$ of $\mathcal{A}$ becomes $p \xrightarrow{a|v} q$ in $\mathcal{T}$ where v is empty if p is not final and is equal to a fixed letter b if p is final. It is clear that the output of a path is infinite iff the path goes infinitely often through a final state. Thus the domain of the transducer $\mathcal{T}$ is the set recognized by $\mathcal{A}$. For instance, the domain of a transducer may be not recognizable by a deterministic Büchi automaton as in the following example. It is however true that the domain is closed if the transducer has no cycling path with an empty output.

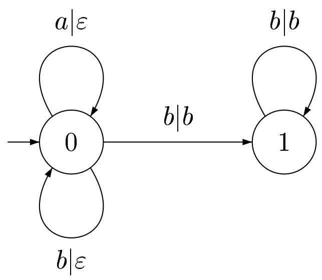


Figure 7: Transducer of Example 21

EXAMPLE 21 The domain of the function f realized by the transducer of Figure 7 is the set $(a+b)^*b^\omega$ of words having a finite number of a. The function f cannot be realized by a sequential transducer since its domain is not a G_δ set.

It must be also pointed out that a function realized by a transducer may be not continuous although it is supposed that all states of the transducer are final as it is shown in the following example.

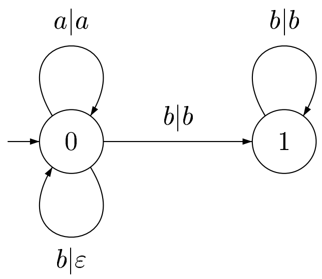


Figure 8: Transducer of Example 22

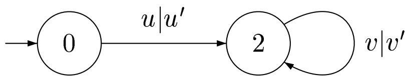
EXAMPLE 22 The image of an infinite word x by the function f realized by the transducer of Figure 8 is $f(x) = a^\omega$ if x has an infinite number of a and

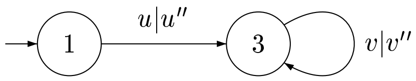
$f(x) = a^n b^\omega$ if the number of a in x is n. The function f is not continuous. For instance, the sequence $x_n = b^n a b^\omega$ converges to b^ω while $f(x_n) = ab^\omega$ does not converge to $f(b^\omega) = b^\omega$.

Proof We first explain why the above three conditions of the proposition are necessary. The fact that the conditions are sufficient follows from the algorithm that we describe in Section 3.2.

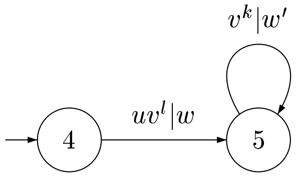
If the function f is realized by a sequential transducer $\mathcal{D}$, a deterministic Büchi automaton recognizing the domain of f can be deduced from the input automaton of $\mathcal{D}$ in the following way. Each state q is first split in two states q_1 and q_2. We distribute then the edges arriving in q between q_1 and q_2 according to the emptiness of their output. Edges with an empty output arrive in q_1 while edges with a nonempty output arrive in q_2. The state q_2 is then final and q_1 is not. If q was initial, exactly one among q_1 and q_2 is then initial. All edges going out of q are duplicated in edges going out of q_1 and q_2. In symbolic dynamics, such a transformation is called an input state splitting. It is clear that this deterministic Büchi automaton recognizes the domain of f. It is also clear that any sequential function is continuous.

We now prove that the third condition is necessary. We suppose that we have the following picture representing paths in $\mathcal{T}$.





where 0 and 1 are initial states, u, u', u'', v, v' and v'' are finite words. Let $\mathcal{D}$ be a sequential transducer realizing the same function as $\mathcal{T}$. There are in $\mathcal{D}$ paths



where 0 is the initial state, w and w' are finite words. By prolonging the path in $\mathcal{T}$ from 0 to 2 (respectively from 1 to 3) with l iterations of the path around 2 (respectively around 3), we can assume without loss of generality that $l = 0$. By replacing the cycling path around 2 (respectively around 3) by k iterations of this path, we can assume that $k = 1$.

We claim that if the state 2 is not constant, then the equality $|w| = |v'|$ holds. Since states 2 and 3 are not constant, then if $v = \varepsilon$ then $v' = v'' = \varepsilon$ and the twinning property is satisfied. We now assume that v is not empty. Let $x|x'$ and $y|y'$ be the infinite labels of two infinite paths leaving 2 such that $x' \neq y'$. There are in $\mathcal{D}$ two infinite paths labeled by $x|x''$ and $y|y''$ leaving the state 5 such that

$$\begin{aligned} u'{v'}^n x' &= w{w'}^n x'' \\ u'{v'}^n y' &= w{w'}^n y''. \end{aligned}$$

If $|v'| < |w'|$, the words x' and y' have a common prefix of length $|w| - |u'| + n(|w'| - |v'|)$ for any large n. This leads to the contradiction that $x' = y'$. If $|v'| > |w'|$, the words x'' and y'' have a common prefix of length $|u| - |w| + n(|v'| - |w'|)$ for any large n. This leads to the contradiction that $x'' = y''$ and $x' = y'$.

By symmetry, if the state 3 is not either constant, then the equality also $|w| = |v''|$ holds and therefore $|v'| = |v''|$.

If both words v' and v'' are non empty, then $f(uv^\omega) = u'{v'}^\omega = u''{v''}^\omega$. $\square$

Before describing the algorithm for determinization, we first study a particular case. It turns out that the first two conditions of the proposition are due to the fact that the transducer $\mathcal{T}$ may have cycling paths with an empty output. If the transducer $\mathcal{T}$ has no cycling path with an empty output, the previous proposition can be stated in the following way.

PROPOSITION 23 *Let f be a function realized by a transducer $\mathcal{T}$ with all states final. Suppose also that $\mathcal{T}$ has no cycling path with an empty output. Let $\mathcal{T}'$ be the transducer obtained by removing from $\mathcal{T}$ all constant states. Then the function f is sequential iff the transducer $\mathcal{T}'$ has the twinning property.*

If the transducer $\mathcal{T}$ has no cycling path with an empty output, any infinite path has an infinite output. Thus, an infinite word x belongs to the domain of f iff it is the input label of an infinite path in $\mathcal{T}$. The domain of f is then a closed set. It is then recognized by a deterministic Büchi automaton whose all states are final. This automaton can be obtained by the usual subset construction on the input automaton of $\mathcal{T}$. Furthermore, if the transducer $\mathcal{T}$ has no cycling path with an empty output, the function f is necessarily continuous. This could be proved directly but it follows from Lemma 31.

We now study the decidability of the conditions of Propositions 20 and 23. We have the following results.

PROPOSITION 24 *It is decidable if a function f given by a transducer with all states final is sequential. Furthermore, if the transducer has no cycling path with an empty output, this can be decided in polynomial time.*

Note that the result does not hold if it is not supposed that the transducer has no cycling path with an empty output. In the general case, the problem is NP-hard. For any Büchi automaton, consider the transducer obtained by replacing each transition $p \xrightarrow{a} q$ of the Büchi automaton by a transition $p \xrightarrow{a|\varepsilon} q$ if p is not final and by $p \xrightarrow{a|b} q$ for a fixed letter b if p is final. The function maps any infinite word to b^ω and its domain is exaclty the set of infinite words recognized by the Büchi automaton. This function is sequential iff its domain is deterministic. Since testing whether the set of infinite words recognized by a given non deterministic Büchi automaton is deterministic is an NP-hard problem, testing whether a function is sequential is also NP-hard.

Proof As explained in the proof of Proposition 20, a Büchi automaton recognizing the domain of the function can be easily deduced from the transducer. It is then decidable if this set can be recognized by a deterministic Büchi automaton [29, Thm 5.3].

It is decidable in polynomial time if a function given by a transducer with final states is continuous [24].

We now show that the third condition of Proposition 20 can be decided in polynomial time. Since we have already proved in Lemma 12 that the twinning property can be decided in a polynomial time, it suffices to prove that the transducer $\mathcal{T}'$ can be computed in polynomial time. We claim that it can be decided in polynomial time whether a given state is constant.

Let $\mathcal{A}$ be the output automaton of the transducer. By a depth first search, it can be found two finite words u and v such that $|u|+|v| \leq n$ and such that uv^ω labels a path leaving q. One constructs a complete deterministic automaton $\mathcal{B}$ recognizing uv^ω with a sink state 0 which is the only non accepting state. We then consider the synchronized product automaton of $\mathcal{A}$ and $\mathcal{B}$. There is a transition from (p,r) to (p',r') labelled by a finite word w (perhaps empty) iff there is a transition from p to p' in $\mathcal{A}$ and a path from r to r' in $\mathcal{B}$. The infinite word uv^ω is the label of all paths leaving q iff no state $(q',0)$ is accessible from $(q,i_{\mathcal{B}})$ where $i_{\mathcal{B}}$ is the initial of $\mathcal{B}$. This naive algorithm runs in quadratic time for each state q. Therefore the constant states of a transducer can be computed in cubic time. It turns out that they can be computed in linear time [10]. $\Box$

3.2 Determinization of transducers over infinite words

In this section, we describe an algorithm to determinize a real-time transducer which satisfies the properties of Proposition 20. This algorithm can easily be adapted to the case when the transducer is not real-time. This algorithm proves that the conditions of the proposition are sufficient.

Let $\mathcal{T} = (Q,E,I)$ be a transducer and let $\mathcal{T}'$ be the transducer obtained by removing from $\mathcal{T}$ all constant states. We assume that $\mathcal{T}'$ has the twinning property. We denote by S the set of constant states. For a state q of S, we denote by y_q, the single output of q which is an ultimately periodic word. We suppose that the domain of f is recognized by the deterministic Büchi automaton $\mathcal{A}$. This automaton is used in the constructed transducer to insure that the output is infinite only when the input belongs to the domain of the function.

We describe the deterministic transducer $\mathcal{D}$ realizing the function f. A state of $\mathcal{D}$ is a pair (p,P) where p is a state of $\mathcal{A}$ and P is a set containing two kinds of pairs. The first kind are pairs (q,z) where q belong to $Q \setminus S$ and z is a finite word over B. The second kind are pairs (q,z) where q belongs to S and z is an ultimately periodic infinite word over B. We now describe the transitions of $\mathcal{D}$. Let (p, P) be a state of $\mathcal{D}$ and let a be a letter. Let R be equal to the set defined as follows

$$\begin{aligned} R = \{(q', zw) \mid q' \notin S \text{ and } \exists (q, z) \in P,\ q \notin S \text{ and } q \xrightarrow{a|w} q' \in E\} \\ \cup \{(q', zwy_{q'}) \mid q' \in S \text{ and } \exists (q, z) \in P,\ q \notin S \text{ and } q \xrightarrow{a|w} q' \in E\} \\ \cup \{(q', z) \mid q' \in S \text{ and } \exists (q, z) \in P,\ q \in S \text{ and } q \xrightarrow{a|w} q' \in E\}. \end{aligned}$$

We now define the transition from the state (p, P) input labeled by a. If R is empty, there is no transition from (p, P) input labeled by a. Otherwise, the output of this transition is the word v defined as follows. Let $p \xrightarrow{a} p'$ be the transition in $\mathcal{A}$ from p labeled by a. If p' is not a final state of $\mathcal{A}$, we define v as the empty word. If p' is a final state, we define v as the first letter of the words z if R only contains pairs (q', z) with $q' \in S$ and if all the infinite words z are equal. Otherwise, we define v as the longest common prefix of all the finite or infinite words z for $(q', z) \in R$. The state P' is then defined as follows

$$P' = \{(q', z) \mid (q', vz) \in R\}.$$

There is then a transition $(p, P) \xrightarrow{a|v} (p', P')$ in $\mathcal{D}$. The initial state of $\mathcal{D}$ is the pair $(i_{\mathcal{A}}, J)$ where $i_{\mathcal{A}}$ is the initial state of $\mathcal{A}$ and where $J = \{(i, \varepsilon) \mid i \in I \text{ and } i \notin S\} \cup \{(i, y_i) \mid i \in I \text{ and } i \in S\}$. If the state p' is not final in $\mathcal{A}$, the output of the transition from (p, P) to (p', P') is empty and the words z of the pairs (q, z) in P, may have a nonempty common prefix. We only keep in $\mathcal{D}$ the accessible part from the initial state. The transducer $\mathcal{D}$ has a deterministic input automaton. It turns out that the transducer $\mathcal{D}$ has a finite number of states. This will be proved in Lemma 33. It will be also proved in Proposition 34 that the transducer $\mathcal{D}$ realizes the same function as $\mathcal{T}$.

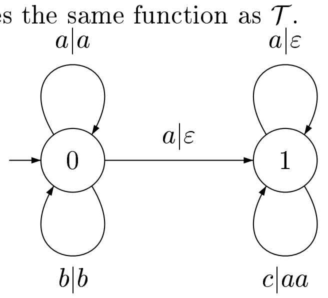


Figure 9: Transducer of Example 25

EXAMPLE 25 Consider the transducer pictured in Figure 9. A deterministic Büchi automaton recognizing the domain is pictured in Figure 10. If the algorithm for determinization is applied to this transducer, one gets the transducer pictured in Figure 11.

The following lemma states the main property of the transitions of $\mathcal{D}$.

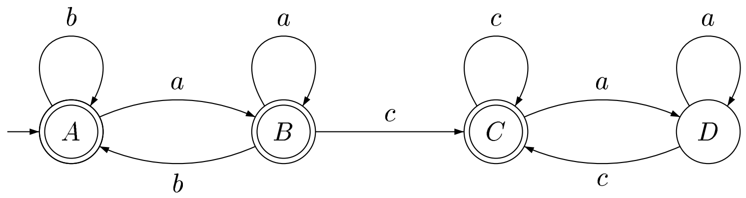


Figure 10: A deterministic Büchi automaton for the domain

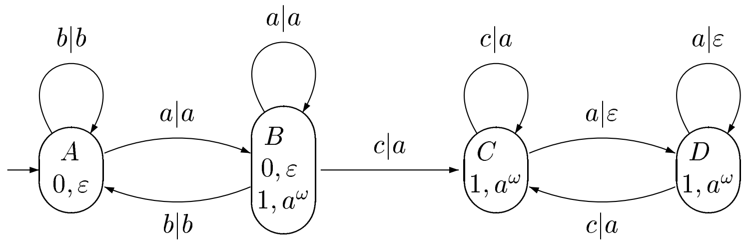


Figure 11: Determinization of the transducer of Figure 9

Lemma 26 *Let u be a finite word. Let $(i_{\mathcal{A}}, J) \xrightarrow{u|v} (p, P)$ be the unique path in $\mathcal{D}$ with input label u from the initial state. Then, the state p is the unique state of $\mathcal{A}$ such that $i_{\mathcal{A}} \xrightarrow{u} p$ is a path in $\mathcal{A}$ and the set P is equal to*

$$P = \{(q, z) \mid \exists\, i \xrightarrow{u|v'} q \text{ in } \mathcal{T} \text{ such that } v' = vz \text{ if } q \notin S \\ v'y_q = vz \text{ if } q \in S\}.$$

Proof The proof of the lemma is by induction on the length of u. Let us consider the following path in $\mathcal{D}$

$$(i_{\mathcal{A}}, J) \xrightarrow{u|v} (p, P) \xrightarrow{a|t} (p', P')$$

where a is a letter. Let (q', z') be a pair in P'. If $q' \notin S$, there is a pair (q, z) in P and a transition $q \xrightarrow{a|t'} q'$ in $\mathcal{T}$. If both states q and q' do not belong to S, the proof is similar to the proof of Lemma 13. If $q \notin S$ and $q' \in S$, one has $tz' = zt'y_{q'}$. By the induction hypothesis, there is a path $i \xrightarrow{u|vz} q$ in $\mathcal{T}$. One finally gets $vtz' = vzt'y_{q'}$. If $q \in S$ and $q' \in S$, one has $tz' = z$. By the induction hypothesis, there is a path $i \xrightarrow{u|v'} q$ in $\mathcal{T}$ such that $v'y_q = vz$. Since $y_q = t'y_{q'}$, one finally gets $vtz' = v't'y_{q'}$. □

The previous lemma has the corollary which states that each state q is the first component of at most one pair (q, z) in the second component P of a state (p, P) of $\mathcal{D}$.

Corollary 27 *Let q be a state of $\mathcal{T}$ and let (p, P) be a state of $\mathcal{D}$. The subset P contains at most one pair (q, z).*

Proof Let $(i_{\mathcal{A}}, J) \xrightarrow{u|v} (p, P)$ be a path in $\mathcal{D}$ and let (q, z) and (q, z') be two pairs in P.

We first suppose that q is not constant. Let $x|y$ and $x'|y'$ be two pairs of infinite words which label two paths leaving q such that $y \neq y'$. By the previous lemma, there are two paths $i \xrightarrow{u|vz} q$ and $i' \xrightarrow{u|vz'} q$ in $\mathcal{T}$. One has $f(ux) = vzy = vz'y$ and $f(ux') = vzy' = vz'y'$. If $z \neq z'$, it may be assumed by symmetry that $|z'| > |z|$ and that $z' = zw$ for some finite word w. This leads to the contradiction $y = y' = w^\omega$.

We now suppose that q is constant. Let $x|y$ be a pair of infinite words which labels a path leaving q. By the previous lemma, there are two paths $i \xrightarrow{u|w} q$ and $i' \xrightarrow{u|w'} q$ in $\mathcal{T}$ such that $wy = vz$ and $w'y = vz'$. Furthermore, one has $f(ux) = wy = w'y$ and thus $z = z'$. $\Box$

We now introduce some technical property of the paths of a transducer. This property is a kind of twinning property when the output of one of the cycling paths is empty. Its turns out that this property is equivalent to the continuity of the function realized by the transducer when it is already supposed that the transducer has the twinning property. Let $\mathcal{T}$ be a transducer and let S be its set of constant states. The transducer $\mathcal{T}$ is said to have the *ε-compatibility property* iff for any pair of paths

$$i \xrightarrow{u|u'} q \xrightarrow{v|v'} q$$
$$i' \xrightarrow{u|u''} q' \xrightarrow{v|\varepsilon} q'$$

such that i and i' are two initial states and v' is a nonempty word, the state q' is constant and its constant $y_{q'}$ satisfies $u''y_{q'} = u'v'^{\omega}$. If the states q and q' are twinned, there cannot be a pair of such paths. If the output along the second cycling path is empty, the output along the first cycling path should also be empty. The above conditions add some compatibility of the outputs when q and q' are not twinned.

The following lemma states that if the function realized by the transducer is continuous, then the transducer has the ε-compatibility property. The converse is established in Lemma 31.

LEMMA 28 *Let $\mathcal{T}$ be transducer realizing a function f on infinite words. If the function f is continuous, then the transducer $\mathcal{T}$ has the ε-compatibility property.*

Proof Let $x|y$ be a pair of infinite words which labels a path leaving q'. For any integer n, one has $f(uv^n x) = u''y$ and $f(uv^\omega) = u''y$ by continuity of f. Since $f(uv^\omega) = u'v'^{\omega}$, the state q' is constant and its constant $y_{q'}$ satisfies $u''y_{q'} = u'v'^{\omega}$. $\Box$

For a finite word w and an infinite word x, we denote by $d(w, x)$ the integer $|w| - |w \wedge x|$ where $w \wedge x$ is the longest common prefix of w and x. Remark that d is not a distance but Lemma 15 still holds when v'_3 is an infinite word.

LEMMA 29 *Let $\mathcal{T}$ be a transducer. Suppose that $\mathcal{T}$ has the ε-compatibility property and that $\mathcal{T}'$ has the twinning property. There is a constant K such that for any two paths $i \xrightarrow{u|v} q$ and $i' \xrightarrow{u|v'} q'$ where i and i' are initial states, $q \notin S$ and $q' \in S$, one has*

$$d(v, v'y_{q'}) \leq K.$$

Proof Let K be equal to n^2M where n is the number of states of the transducer and M is the maximal length of the output label of a transition. We prove $d(v, v'y_{q'}) \leq K$ by induction on the length of u. If $|u| \leq n^2$, the result holds by definition of K. Otherwise, both paths can be factorized

$$i \xrightarrow{u_1|v_1} p \xrightarrow{u_2|v_2} p \xrightarrow{u_3|v_3} q$$
$$i' \xrightarrow{u_1|v_1'} p' \xrightarrow{u_2|v_2'} p' \xrightarrow{u_3|v_3'} q'.$$

where $|u_2| > 0$ and $|u_3| \leq n^2$. If both words v_2 and v_2' are empty, the result follows directly from the induction hypothesis. Thus, we may assume that one the words v_2 or v_2' is not empty. Since q does not belong to S, p does not belong to S either. The ε-compatibility property implies then that v_2 cannot be empty.

We first suppose that $p' \notin S$. By the twinning property, Lemma 15 and the above remark, one has $d(v_1v_2v_3, v_1'v_2'v_3'y_{q'}) = d(v_1v_3, v_1'v_3'y_{q'})$ and the result follows from the induction hypothesis.

We now suppose that $p' \in S$ and we claim that $v_1'v_2'v_3'y_{q'} = v_1v_2^{\omega}$. Since p' is constant, $y_{p'} = v_3'y_{q'}$. If the word v_2' is empty, the ε-compatibility property implies that $v_1'y_{p'} = v_1v_2^{\omega}$. If v_2' is nonempty, $y_{p'} = {v_2'}^{\omega}$. Since $f(u_1u_2^{\omega}) = v_1v_2^{\omega} = v_1'{v_2'}^{\omega}$, the claimed equality holds. In both cases, one has $d(v_1v_2v_3, v_1'v_2'v_3'y_{q'}) = d(v_1v_2v_3, v_1v_2^{\omega}) \leq |v_3| \leq K$. □

The following lemma states some technical consequence of the ε-compatibility property.

LEMMA 30 *Let $\mathcal{T}$ be a transducer which has the ε-compatibility property and let f the function realized by $\mathcal{T}$. Then if x is in the domain of f and x is the input label of a path entirely out of S, the output of this path is infinite and is thus equal to the image of x by f.*

Proof Suppose that x is the input label of two paths γ and γ'. Suppose also that all states of γ do not belong to S and the output along γ' is an infinite word. Since the number of states is finite, both paths γ and γ' can be factorized

$$\gamma = i \xrightarrow{u_0|v_0} q \xrightarrow{u_1|v_1} q \xrightarrow{u_2|v_2} q \cdots$$
$$\gamma' = i' \xrightarrow{u_0|v_0'} q' \xrightarrow{u_1|v_1'} q' \xrightarrow{u_2|v_2'} q' \cdots$$

Furthermore, it can be assumed that each v_k' is nonempty since $v_0'v_1'v_2' \ldots$ is an infinite word. By hypothesis, this implies that each v_k is also nonempty. □

The following lemma states a kinf of converse of Lemma 28. It shows in particular that if a transducer $\mathcal{T}$ has no cycling path with an empty output and if $\mathcal{T}'$ has the twinning property, then the function realized by $\mathcal{T}$ is continuous. If x and y are two infinite words, $d(x,y)$ denotes the usual distance between x and y which makes the set A^ω of all infinite words a compact space.

LEMMA 31 *Let $\mathcal{T}$ be a transducer which has the ε-compatibility and such that $\mathcal{T}'$ has the twinning property. Then the function realized by $\mathcal{T}$ is continuous.*

Proof Let f be the function realized by the transducer $\mathcal{T}$ and let x be an infinite word in the domain of f. We claim that for any integer m there is an integer k such that for any infinite word x' also in the domain f, the inequality $d(x,x') \leq 2^{-k}$ implies the inequality $d(f(x),f(x')) \leq 2^{-m}$. Let $y = f(x)$ be the image of x. Let γ be a path labeled by $x|y$ and let i be the initial state of γ. Let γ' be a path labelled by (x',y') where $y' = f(x')$. According to the previous lemma, it can be assumed that either there is a path entirely out of S which is labeled by $x|y$ or that x is not the input label of a path entirely out of S.

We first suppose that the path γ is entirely out of S. By Lemma 15, there is a constant K such that if $i \xrightarrow{u|v} q$ and $i' \xrightarrow{u|v'} q'$ are two paths with $q \notin S$ and $q' \notin S$, then one has $d(v,v') \leq K$. By Lemmas 28 and 29, there is another constant K' such that if $i \xrightarrow{u|v} q$ and $i' \xrightarrow{u|v'} q'$ are two paths with $q \notin S$ and $q' \in S$, then one has $d(v,v'y_{q'}) \leq K$. Let k be chosen such that the output along the first k transitions of γ has a length greater then $m + \max(K,K')$. Let q be the state of γ reached after k transitions and let v be the output of γ' along the first k transitions. Suppose that x' satisfies $d(x,x') \leq k$ and that γ' is a path labeled by $x'|y'$ where $y' = f(x')$. Let i' the initial state of γ' and let q' be the state of γ' reached after k transitions. If q' does not belong to S, one has $d(v,v') \leq K$ where v' is the output of γ' along the first k transitions. Since $|v| \geq m + K$, one has $|v \wedge v'| \geq m$ and thus $d(y,y') \leq 2^{-m}$. If q' belongs to S, one has $d(v,y') \leq K'$. Since $|v| \geq m + K'$, one has $|v \wedge y'| \geq m$ and thus $d(y,y') \leq 2^{-m}$.

We now suppose that x is not the input label of a path entirely out of S. There is then an integer K such that any path input labeled by a prefix of x of length greater than K ends in a state of S. Let k be equal to $K + K'$ where K' is the length of part of γ inside S which contains at least n^2 transitions with a nonempty output. If $d(x,x') \leq 2^{-k}$, both paths γ and γ' can be factorized

$$\gamma = i \xrightarrow{u_0|v_0} q \xrightarrow{u_1|v_1} q \xrightarrow{u_2|v_2} \cdots$$
$$\gamma' = i' \xrightarrow{u_0|v_0'} q' \xrightarrow{u_1|v_1'} q' \xrightarrow{u_2'|v_2'} \cdots$$

where $u_0u_1u_2 = x$, $u_0u_1u_2' = x'$, v_1 is nonempty and q and q' belong to S. We claim that $y = y'$. One has $y = v_0y_q$ and $y' = v_0'y_{q'}$. Since v_1 is nonempty, one also has $y_q = v_1^\omega$. If v_1' is also nonempty, one has $y_{q'} = {v_1'}^\omega$ and $f(u_0u_1^\omega) = v_0v_1^\omega = v_0'{v_1'}^\omega$ and thus $y = y'$. If the word v_1' is empty, the ε-compatibility property implies $v_0y_q = v_0'y_{q'}$ and $y = y'$.

In both cases, an integer k satisfying the claimed property has been found. The function f is then continuous. $\square$

The following lemma states that the lengths of the words z of the pairs (q, z) in the states of $\mathcal{D}$ are bounded. It is essentially due to the twinning property of $\mathcal{T}'$.

LEMMA 32 *There is a constant K such that for any pair (q, z) in P of a state (p, P) of $\mathcal{D}$ where $q \notin S$ and z is a finite word, one has $|z| \leq K$.*

Proof Let m and n be the respective numbers of states of $\mathcal{A}$ and $\mathcal{T}$. By Lemma 16 and 29, there is a constant K' such that if $i \xrightarrow{u|v} q$ and $i' \xrightarrow{u|v'} q'$ are two paths such that $q \notin S$, then one has $d(v, v') \leq K'$ if $q' \notin S$ or $d(v, v'y_{q'}) \leq K'$ if $q' \in S$. Let $K = K' + mnM$ where M is the maximal length of the output label of a transition in $\mathcal{T}$. Let (p, P) be a state of $\mathcal{D}$ and consider a path

$$(i_{\mathcal{A}}, J) \xrightarrow{u'|v'} (p', P') \xrightarrow{u|v} (p, P)$$

where p' is a final state of $\mathcal{A}$. If there is no path from $(i_{\mathcal{A}}, J)$ to (p, P) which goes through a state (p', P') with p' final, we assume that (p', P') is actually $(i_{\mathcal{A}}, J)$. The proof is by induction on the length of u. If $|u| = 0$, the state p is actually a final state of $\mathcal{A}$. In the case where p is final, the longest common prefix of the words z of the pairs (q, z) in P is empty. Lemmas 16, 26 and 29 imply that $|z| \leq K'$. We now suppose that p is not final. If $|u| \leq mn$, the result follows from the definition of the transitions of $\mathcal{D}$. We now suppose that $|u| > mn$ and that (p', P') is the last state along the path from $(i_{\mathcal{A}}, J)$ to (p, P) such that p' is a final state of $\mathcal{A}$. Let (q, z) be a pair in P such that $q \notin S$ and z is a finite word. By definition of the transitions of $\mathcal{D}$, there is a pair (q', z') in P' and a path $q' \xrightarrow{u|w} q$ in $\mathcal{T}$ such that $z'w = vz$. There is also a path $p' \xrightarrow{u} p$ in $\mathcal{A}$. Since $|u| > mn$, both paths can be factorized

$$\begin{array}{ccccccc} p' & \xrightarrow{u_1} & p'' & \xrightarrow{u_2} & p'' & \xrightarrow{u_3} & p \\ q' & \xrightarrow{u_1|w_1} & q'' & \xrightarrow{u_2|w_2} & q'' & \xrightarrow{u_3|w_3} & q \end{array}$$

where $u_1u_2u_3 = u$ and $w_1w_2w_3 = w$. Since the cycling path $p'' \xrightarrow{u_2} p''$ in $\mathcal{A}$ does not contain any final state, the infinite word $u'u_1u_2^\omega$ does not belong to the domain of f. This implies that the word w_2 is empty. We then consider the path

$$(p', P') \xrightarrow{u_1u_3|v''} (p, P'')$$

in $\mathcal{D}$. The subset P'' contains a pair (q, z'') for some finite word z''. We claim that $z'' = z$. Indeed, one has $z'w_1w_2w_3 = z'w_1w_3 = vz = v''z''$. As both paths $p' \xrightarrow{u_1u_2u_3} p$ and $p' \xrightarrow{u_1u_3} p$ in $\mathcal{A}$ contain no other final state than p, both outputs v and v'' along the corresponding paths in $\mathcal{D}$ are empty. Thus one gets $z = z''$. By the induction hypothesis, one has $|z| = |z''| \leq K$. $\square$

It is now possible to prove that the transducer $\mathcal{D}$ has a finite number of states. However, the number of states of $\mathcal{D}$ can be exponential as in the case of finite words.

LEMMA 33 *The number of states of $\mathcal{D}$ is finite.*

Proof We have proved in the preceding lemma that the lengths of the finite words z are bounded. It remains to show that there is a finite number of different infinite words z which can appear in some pair (q, z). By definition of the transitions, any infinite word z of a pair is the suffix of $z'wy_q$ where (q', z') is a pair such that $q' \notin S$ and z' is finite and where $q \in S$ and $q' \xrightarrow{a|w} q$ is a transition of $\mathcal{T}$. Since the length of z' is bounded, the number of such words $z'wy_q$ is finite and they are ultimately periodic. There are then a finite number of suffixes of such words. □

The following proposition finally states that the sequential transducer $\mathcal{D}$ is equivalent to the transducer $\mathcal{T}$. Both transducers realize the same function over infinite words.

PROPOSITION 34 *The sequential transducer $\mathcal{D}$ realizes the same function f as the transducer $\mathcal{T}$.*

Proof We respectively denote by f and f' the functions realized by the transducer $\mathcal{T}$ and $\mathcal{D}$. We first prove that if an infinite word x belongs to the domain of f, it also belongs to the domain of f' and $f(x) = f'(x)$.

Let $x = a_0a_1a_2\dots$ be an infinite word which is in the domain of f. Let γ be a path

$$\gamma = i \xrightarrow{a_0|v_0} q_1 \xrightarrow{a_1|v_1} q_2 \xrightarrow{a_2|v_2} \cdots \tag{1}$$

be a path in $\mathcal{T}$ input labeled by x and whose output $v_0v_1v_2\dots$ is an infinite word. Consider the unique path Γ in $\mathcal{D}$ input labeled by x

$$\Gamma = (i_{\mathcal{A}}, J) \xrightarrow{a_0|v'_0} (p_1, P_1) \xrightarrow{a_1|v'_1} (p_2, P_2) \xrightarrow{a_2|v'_2} \cdots \tag{2}$$

By Lemma 26, each state P_n contains a pair (q_n, z_n).

We first suppose that x input labels a path in $\mathcal{T}$ entirely out of S. By Lemma 30, it can be assumed that each state q_n does not belong to S and that each z_n is finite. By Lemma 26, the equality $v_0\dots v_n = v'_0\dots v'_nz_n$ holds for any integer n. By Lemma 32, the lengths of the words z_n are bounded. This implies the equality $v_0v_1v_2\dots = v'_0v'_1v'_2\dots$ of the two outputs.

We now suppose that x is not the input label of a path entirely out of S. There is then an integer n such that for any $m \geq n$, P_m only contains pairs (q, z) with $q \in S$ and z infinite. Both path γ and Γ can be factorized

$$\begin{aligned}
\gamma &= i && \xrightarrow{u_0|v_0} q && \xrightarrow{u_1|v_1} q && \xrightarrow{u_2|v_2} q \cdots \\
\Gamma &= (i_{\mathcal{A}}, J) && \xrightarrow{u_0|v'_0} (p, P) && \xrightarrow{u_1|v'_1} (p, P) && \xrightarrow{u_2|v'_2} (p, P) \cdots
\end{aligned}$$

Furthermore, it can be assumed that each v_n is nonempty. Thus each path $p \xrightarrow{u_k} p$ in $\mathcal{A}$ contains a final state of $\mathcal{A}$. The single output of the state q is v_1^ω. By Lemma 26, the subset P contains a pair (q, z) and $v_0 y_q = v_0 v_1^\omega = v_0' z$.

Let (q_0, z') be another pair in P. By definition of the transitions of $\mathcal{D}$, there is a sequence $(q_n)_{n \geq 0}$ of states such that the pairs $(q_n, {v_1'}^n z')$ belong to P. Since there is a finite number of states, there are $n < m$ such that $q_n = q_m$. This implies that there is in $\mathcal{T}$ a cycling path around q_n input labeled by u_1^{m-n}. Let $q'' = q_n = q_m$. We first claim that ${v_1'}^n z' = z'$. If the word v_1' is empty, this is obvious. Otherwise, Corollary 27 implies that ${v_1'}^n z' = {v_1'}^m z'$. Thus $z' = {v_1'}^\omega$ and the equality ${v_1'}^n z' = z'$ also holds. The subset P contains a pair (q'', z'). By Lemma 26, there is a path $i \xrightarrow{u_0 | v''} q''$ in $\mathcal{T}$ such that $v_0' z' = v'' y_{q''}$. By construction, there is also a cycling path around q'' input labeled by u_1^{m-n}. We suppose that the output label of this cycling path is the word w. If the word w is empty, Lemma 28, states that $v'' y_{q''} = v_0 v_1^\omega$. Thus, one has $v_0' z' = v'' y_{q''} = v_0 v_1^\omega = v_0' z$ and $z = z'$. If the word w is nonempty, one has $y_{q''} = w^\omega$ and $f(u_0 u_1^\omega) = v'' w^\omega = v_0 v_1^\omega$. This implies $z = z'$.

Since we have proved that all pairs (q, z) in P share the same infinite word z and since each path $p \xrightarrow{u_i} p$ contains a final state, each word v_i' is nonempty by definition of the transitions of $\mathcal{D}$ and the equality $v_1' v_2' v_3' \ldots = z$ holds. This last equality implies that $v_0 y_q = v_0' z = v_0' v_1' v_2' \ldots$ and that $f(x) = f'(x)$.

Conversely, the definition of the transitions of $\mathcal{D}$ implies that the domain of f' is contained in the domain of f. Thus both functions f and f' have the same domain and $f = f'$. $\square$

We have already mentioned in Proposition 24 that it can be decided whether a function over infinite words realized by a transducer with all states final is sequential. As in the case of finite words, the algorithm described above provides another decision procedure. Indeed, Lemma 32 gives an upper bound K of the lengths of finite words which can appear in states of $\mathcal{D}$. Let $\mathcal{T}$ be a transducer with all states final realizing a function f. If the algorithm is applied to $\mathcal{T}$, either it stops and gives a sequential transducer $\mathcal{D}$ or it creates a state (p, P) containing a pair (q, z) such that the length of z is greater than K. In the former case, the sequential transducer $\mathcal{D}$ is equivalent to $\mathcal{T}$ and the function f is sequential. In the latter case, the function f is not sequential.

Acknowledgments

The authors would like to thank Jean Berstel for very helpful suggestions and Christian Choffrut for his relevant comments on a preliminary version of this paper.

References

[1] Béal, M.-P. *Codage Symbolique.* Masson, 1993.

[2] Béal, M.-P., and Carton, O. Computing the prefix of an automaton. Tech. Rep. 2000–08, Institut Gaspard Monge, 2000.

[3] Béal, M.-P., and Carton, O. Determinization of transducers over infinite words. In *ICALP'2000* (2000), U. Montanari et al., Eds., vol. 1853 of *Lect. Notes in Comput. Sci.*, pp. 561–570.

[4] Béal, M.-P., Carton, O., Prieur, C., and Sakarovitch, J. Squaring transducers: An efficient procedure for deciding functionality and sequentiality. In *LATIN'2000* (2000), G. Gonnet, D. Panario, and A. Viola, Eds., vol. 1776 of *Lect. Notes in Comput. Sci.*, pp. 397–406.

[5] Berstel, J. *Transductions and Context-Free Languages.* B.G. Teubner, 1979.

[6] Berstel, J., and Perrin, D. Finite and infinite words. In *Algebraic Combinatorics on Words*, M. Lothaire, Ed. Cambridge, 1999. to appear.

[7] Blattner, M., and Head, T. Singled valued a-transducers. *J. Comput. System Sci. 15* (1977), 310–327.

[8] Breslauer, D. The suffix tree of a tree and minimizing sequential transducers. In *CPM'96* (1996), vol. 1075 of *Lect. Notes in Comput. Sci.*, Springer-Verlag, pp. 116–129.

[9] Breslauer, D. The suffix tree of a tree and minimizing sequential transducers. *Theoret. Comput. Sci.*, 191 (1998), 131–144.

[10] Carton, O., Choffrut, C., and Prieur, C. How to decide functionality of rational relations on infinite words. Preprint.

[11] Choffrut, C. Une caractérisation des fonctions séquentielles et des fonctions sous-séquentielles en tant que relations rationnelles. *Theoret. Comput. Sci. 5* (1977), 325–338.

[12] Choffrut, C. *Contribution à l'étude de quelques familles remarquables de fonctions rationnelles.* Thèse d'État, Université Paris VII, 1978.

[13] Choffrut, C. A generalization of Ginsburg and Rose's characterization of gsm mappings. In *ICALP'79* (1979), vol. 71 of *Lect. Notes in Comput. Sci.*, pp. 88–103.

[14] Cohen, A., and Collard, J.-F. Instance-wise reaching definition analysis for recursive programs using context-free transductions. In *PACT'98* (1998).

[15] Cormen, T. H., Leiserson, C. E., and Rivest, R. L. *Introduction to Algorithms.* MIT Press, 1990.

[16] Eilenberg, S. *Automata, Languages and Machines*, vol. A. Academic Press, New York, 1972.

[17] Frougny, C. Numeration systems. In *Algebraic Combinatorics on Words*, M. Lothaire, Ed. Cambridge, 1999. to appear.

[18] Gire, F. Two decidability problems for infinite words. *Inform. Proc. Letters 22* (1986), 135–140.

[19] Kitchens, B. *Continuity properties of factor maps in ergodic theory.* Ph.D. thesis, University of North Carolina, Chapel Hill, 1981.

[20] Lind, D., and Marcus, B. *An Introduction to Symbolic Dynamics and Coding.* Cambridge University Press, 1995.

[21] Mohri, M. Minimization of sequential transducers. In *CPM'94* (1994), M. Crochemore and D. Gusfield, Eds., vol. 807 of *Lect. Notes in Comput. Sci.*, Springer-Verlag, pp. 151–163.

[22] Mohri, M. On some applications of finite-state automata theory to natural languages processing. *Journal of Natural Language Engineering 2* (1996), 1–20.

[23] Mohri, M. Minimization algorithms for sequential transducers. *Theoret. Comput. Sci.*, 234 (2000), 177–201.

[24] Prieur, C. How to decide continuity of rational functions on infinite words. *Theoret. Comput. Sci.* (1999).

[25] Roche, E., and Schabes, Y. *Finite-State Language Processing.* MIT Press, Cambridge, 1997, ch. 7.

[26] Safra, S. On the complexity of ω-automata. In *29th Annual Symposium on Foundations of Computer Sciences* (1988), pp. 24–29.

[27] Schützenberger, M.-P. Sur les relations rationnelles. In *Automata Theory and Formal Languages, 2nd GI Conference* (1975), H. Brakhage, Ed., vol. 33 of *Lect. Notes in Comput. Sci.*, Springer, pp. 209–213.

[28] Schützenberger, M.-P. Sur une variante des fonctions séquentielles. *Theoret. Comput. Sci. 11* (1977), 47–57.

[29] Thomas, W. Automata on infinite objects. In *Handbook of Theoretical Computer Science*, J. van Leeuwen, Ed., vol. B. Elsevier, 1990, ch. 4, pp. 133–191.

[30] Weber, A., and Klemm, R. Economy of description for single-valued transducers. In *STACS'94* (1994), vol. 775 of *Lect. Notes in Comput. Sci.*, Springer-Verlag, pp. 607–618.

[31] Weber, A., and Klemm, R. Economy of description for single-valued transducers. *Inform. Comput. 118* (1995), 327–340.