Transducers, Logic and Algebra for Functions of Finite Words

Emmanuel Filiot
Université Libre de Bruxelles

Pierre-Alain Reynier
Aix-Marseille Université

The robust theory of regular languages is based on three important pillars: computation (automata), logic, and algebra. In this paper, we survey old and recent results on extensions of these pillars to functions from words to words. We consider two important classes of word functions, the rational and regular functions, respectively defined by one-way and two-way automata with output words, called transducers.

1. INTRODUCTION

Important connections between computation, mathematical logic, and algebra have been established for regular languages of finite words. The class of regular languages corresponds to the class of languages recognized by finite automata, to the class of languages definable in monadic second-order logic (MSO) with one successor [Büchi 1960; Elgot 1961; Trakhtenbrot 1961], and to the class of languages whose syntactic monoid, a canonical monoid attached to every language, is finite (see for instance [Straubing 1994]). While automata are well-suited to study the algorithmic properties of regular languages, the algebraic view has provided effective characterizations of regular languages and its subclasses. Most notably, the problem of deciding whether a regular language is first-order definable amounts to checking whether its syntactic monoid, which is computable from any finite automaton recognizing the language, is aperiodic, which is decidable. See [Diekert et al. 2008] for a survey on first-order definable languages, and [Straubing 1994] for generalizations to other monoid varieties and fragments of MSO. In this paper, we want to survey some old and recent results that extend the three main pillars of language theory to functions of finite words.

At the computational level, word functions are defined by *transducers*, which extend automata with outputs on their transitions. They can be seen as Turing machines with a read-only input tape initially filled with the input word, and a write-only output tape on which to write the output word. We will consider two important classes of transducers: one-way and two-way transducers. For both classes, the output head is assumed to move only to the right, and for the class of one-way transducer, the input head is also restricted to move to the right. This yields the class of rational functions, long studied in the literature [Berstel and Boasson 1979], and the strictly more expressive class of regular functions, which has received more attention in the recent years.

At the logical level, we consider MSO transducers, as defined by Courcelle in the more general context of graph transformations [Courcelle 1994]. MSO transducers are based on MSO for words, and the main idea is to define the output word by some MSO

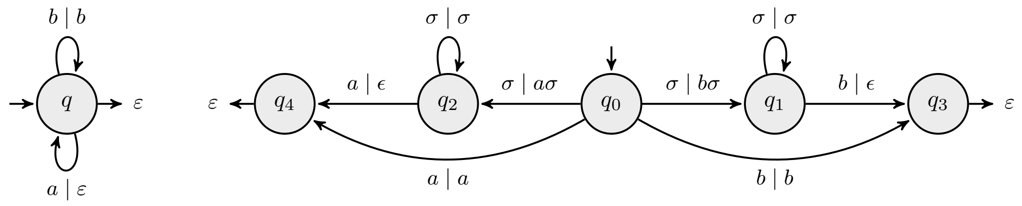


Fig. 1. One-way transducers T_0 and T_1 realizing f_{del} and f_{sw} respectively, where $\sigma \in \{a, b\}$.

interpretation on the input word. We present results establishing connections between one-way and two-way transducers and classes of MSO transducers.

At the algebraic level, we present a notion of syntactic congruence for functions introduced by Choffrut, which is of finite index iff the function is *sequential*, i.e. can be defined by some input-deterministic one-way transducer. We also present a generalization of this algebraic characterization to rational functions, a result due to Reutenauer and Schützenberger. We give an application of the latter characterization to the first-order definability problem for rational functions. For the more general class of regular functions, no algebraic characterization is known, but we present preliminary results towards it.

The paper is simply organized along the three pillars: transducers, logic, and algebra. We conclude with some extensions of the presented results and further research directions.

2. TRANSDUCERS

In this paper we assume Σ to be a finite alphabet and denote by Σ^* the set of finite words over Σ, and by ϵ the empty word. A *transduction* f is a relation on Σ^*. In this paper, we will be particularly interested in (partial) functions and for such objects, we denote by $\text{dom}(f)$ their domain.

Example 2.1. In this paper, we will use three running examples, the functions f_{del}, f_{sw} and f_{mir}, over the alphabet $\Sigma = \{a, b\}$. The function f_{del} erases the letters a of an input word: e.g. $f_{del}(abba) = bb$. The function f_{sw} puts the last symbol in a first position ('sw' stands for 'swap'). It is defined by $f_{sw}(\epsilon) = \epsilon$ and for all $u \in \Sigma^*$ and $\sigma \in \Sigma$, by $f_{sw}(u\sigma) = \sigma u$. Finally, the function f_{mir} copies an input word u and concatenates it with its mirror image $\overline{u}$, i.e. $f_{mir}(u) = u\overline{u}$ for all $u \in \Sigma^*$. E.g. $f_{mir}(ab) = abba$.

2.1. One-way transducers

One-way transducers are Turing machines with two left-to-right tapes. The first tape is read-only and contains the input word while the second tape is write-only and contains the output word. For example, to realize the function f_{del} for Example 2.1 by a one-way transducer, the machine will simply go through the input word, from left to right, and, for each input symbol σ, copy σ on the output tape iff σ is different from a.

Formally, a one-way transducer[1] is a tuple $T = (Q, I, F, \Delta, t)$ where Q is a finite set of states, $I \subseteq Q$ is a set of initial states, $F \subseteq Q$ is a set of final states, $\Delta \subseteq Q \times \Sigma \times \Sigma^* \times Q$ is a finite set of transitions, and $t : F \to \Sigma^*$ is a terminal output function. A transition $\gamma = (p, a, w, q) \in \Delta$ of T is represented as $p \xrightarrow{a|w} q$. Intuitively it means that if the

[1]More generally, a transducer is defined as a finite automaton over the monoid $\Sigma^* \times \Sigma^*$. Our definition corresponds to *real-time* transducers in the literature [Berstel and Boasson 1979]. For transductions which are functions, real-time transducers and transducers coincide in expressiveness, therefore we just call them transducers in this paper, as this paper is about functions.

transducer is in state p, and the next input letter is a, then it may go to state q and write the word w on the output tape. W.l.o.g., we assume that for all $p, q \in Q$, $a \in \Sigma$, there exists at most one $w \in \Sigma^*$ such that $(p, a, w, q) \in \Delta$. The class of these transducers is denoted NFT, standing for non-deterministic finite-state transducers.

Given a word $u = a_1 \dots a_n \in \Sigma^*$, a run of T from p to q on u is a sequence of states $\rho = (q_i)_{i=0..n}$ such that $q_0 = p$, $q_n = q$ and, for each $i \in \{1, \dots, n\}$, there is a transition of the form $q_{i-1} \xrightarrow{a_i|w_i} q_i$ in Δ. We say that a run $\rho = (q_i)_{i=0..n}$ is accepting if $q_0 \in I$ and $q_n \in F$. The output of such an accepting run ρ on u, denoted by $\mathrm{out}^u(\rho)$, is defined as the concatenation of the words produced by the transitions and the image of state q_n by the terminal function t, *i.e.* the finite word $w_1 \dots w_n t(q_n) \in \Sigma^*$.

The transduction defined (or realized) by T is the relation $[\![T]\!]$ composed of the pairs (u, w) such that there exists an accepting run ρ of T on u satisfying $w = \mathrm{out}^u(\rho)$.

An NFT T is *functional* if the transduction it defines is a function. The class of functional NFT is denoted by fNFT. The class of functions realized by fNFT is called the class of *rational functions*[2]. An NFT is *sequential* if the underlying input automaton, obtained by projecting away the output words on transitions, is deterministic. Observe that such transducers obviously recognize functions. The class of sequential NFT is denoted by SFT. The class of functions realized by SFT is called the class of *sequential functions*.

Example 2.2. We describe two examples on the alphabet $\Sigma = \{a, b\}$. The transducer T_0, depicted on the left of Figure 1, is functional, sequential, and realizes the function f_{del}. Its terminal function is constant equal to ϵ, and is depicted by a target free outgoing arrow. The transducer T_1, depicted on the right of Figure 1, is functional but not sequential. It realizes the function f_{sw}. Intuitively, the non-determinism is mandatory to guess initially the last letter of the input word.

2.2. Two-way transducers

Two-way transducers extend one-way transducers as follows: instead of being left-to-right, the input tape is two-way. As an example, this intuitively allows the transducer to move to the end of the input word, and to copy it on the output tape during a right-to-left traversal. This transformation thus maps every input word to its mirror image.

Formally, we will consider that two-way transducers take as input a word $u \in \Sigma^*$ surrounded by left and right end-markers, denoted by $\vdash$ and $\dashv$, which we suppose do not belong to the alphabet Σ. These end-markers allow the transducer to identify the beginning and end of the input word. We denote by $\Sigma_\vdash$ the alphabet $\Sigma \cup \{\vdash, \dashv\}$. A *two-way transducer* is a tuple $T = (Q, I, F, \Delta)$ where Q, I and F are as in one-way transducers. Δ is a finite set of transitions which are elements of the form $(p, a, w, q, m) \in Q \times \Sigma_\vdash \times \Sigma^* \times Q \times \{-1, +1\}$. Compared with one-way transducers, transitions contain now a component $m \in \{-1, +1\}$ which indicates the direction of the move (left or right). We also require that transitions reading the left end-marker always move to the right.

Note also that the definition does not include a terminal function. This function is indeed useless for two-way transducers as it can be simulated using the right end-marker.

The class of these transducers is denoted by 2NFT, standing for two-way non-deterministic finite-state transducers.

A *configuration* of a two-way transducer T is a pair $(q, i) \in Q \times (\mathbb{N} \setminus \{0\})$ where q is a state and i is a position on the input tape. Given an input word $u = a_1 \dots a_n \in \Sigma^*$, we

[2]They are called rational because they can be alternatively defined by the more general and classical notion of rational subsets of monoids, see [Berstel and Boasson 1979].

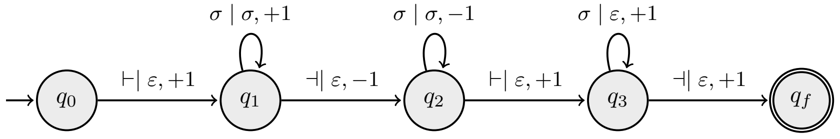

$\sigma \mid \sigma, +1$
$\sigma \mid \sigma, -1$
$\sigma \mid \varepsilon, +1$
q_0
$\vdash\mid \varepsilon, +1$
q_1
$\dashv\mid \varepsilon, -1$
q_2
$\vdash\mid \varepsilon, +1$
q_3
$\dashv\mid \varepsilon, +1$
q_f

Fig. 2. A two-way transducer T_2 for f_{mir}.

will consider an execution of T on the input $\vdash u \dashv \in \Sigma_{\vdash}^*$. A *run* of T on the input $u'{=}\vdash u \dashv$ is a finite sequence of configurations $\rho = (p_1, i_1)\dots(p_m, i_m)$ such that $i_1 = 0$, $i_m = n+2$, and for all $k \in \{1, \dots, m-1\}$, $0 \leq i_k \leq n+1$ and [3] $(p_k, u'[i_k], w_k, p_{k+1}, i_{k+1} - i_k) \in \Delta$, for some word w_k. The output of such a run ρ on $\vdash u \dashv$, denoted by $\mathrm{out}^u(\rho)$, is defined as the concatenation of the words produced by the transitions, *i.e.* the finite word $w_1 \dots w_m \in \Sigma^*$.

This run is accepting if $p_1 \in I$ and $p_m \in F$. This means that for all the configurations but the last one, the reading head should be on a letter of the input u'. For the run to be accepting, the last configuration must correspond to the reading head being immediately after the end of the input word ($i_m = n+2$).

As for NFT, the transduction defined by T is the relation $[\![T]\!]$ composed of the pairs (u, w) such that there exists an accepting run ρ of T on $\vdash u \dashv$ satisfying $w = \mathrm{out}^u(\rho)$.

As for one-way transducers, we will be interested in the subclasses of transducers *functional* and *sequential* transducers, denoted respectively by f2NFT and 2SFT, and defined respectively as the transducers whose semantics is a function, resp. the transducers whose underlying input (two-way) automaton is deterministic. The class of functions realized by f2NFT is called the class of *regular functions*. This terminology comes from the equivalence with a logical formalism, so-called MSO transducers, that will be presented in the next section.

Example 2.3. Consider the transducer T_2 for f_{mir} depicted on Figure 2. It maps every input word u to $u\bar{u}$, where $\bar{u}$ denotes the mirror image of u. The left and right end-markers are important here to allow the transducer to identify the beginning and end of the input word.

2.3. Landscape of transducer classes

Figure 3 gives the inclusion relations existing between the six classes of transducers we have defined so far. Let us first comment on these inclusions. Relations trivially extend functions. Rational functions strictly contain sequential functions, strictness being witnessed by the transduction f_{sw}. Regular functions strictly contain rational ones, as witnessed by f_{mir}. Last, unlike for one-way transducers, in presence of two-wayness, functional non-determinism does not increase expressive power (equality between f2NFT and 2SFT), as shown in [Engelfriet and Hoogeboom 2001]. This equivalence is effective.

We also depict on this picture the decidability status of several subclasses decision problems. For instance, we have proven in [Filiot et al. 2013] that it is decidable, given a f2NFT, whether there exists an equivalent one-way transducer. This decidability result is indicated on the edge from f2NFT to fNFT. Concerning this result, the overall complexity of our decision procedure is non-elementary, and a recent work provided a 2EXPSPACE procedure for the subclass of sweeping transducers [Baschenis et al.

[3]We use the following notation: $u'[0]$ denotes symbol $\vdash$, $u'[i]$ with $1 \leq i \leq n$ denotes the i-th letter of u, and $u'[n+1]$ denotes symbol $\dashv$.

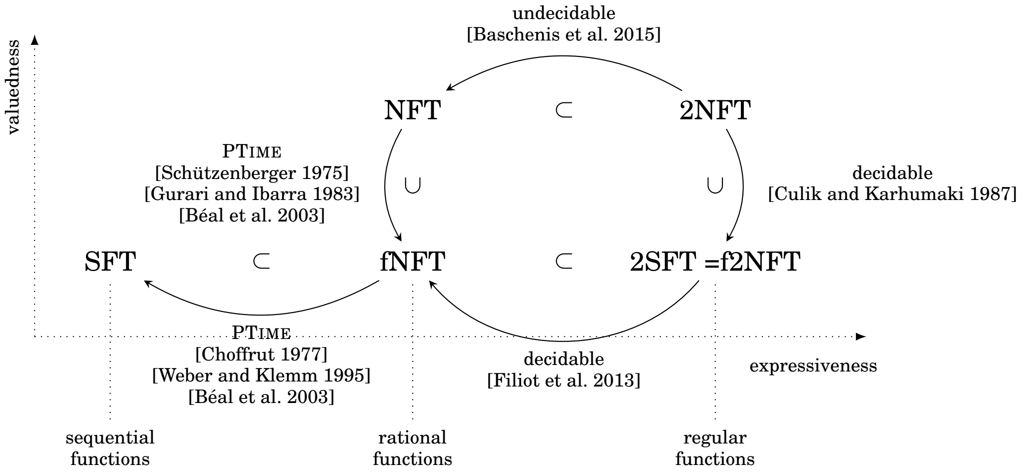


Fig. 3. A landscape of transducers of finite words.

2015]. Other decidability and undecidability results are depicted on the other edges in a similar way.

2.4. Equivalence problem

Checking whether two transducers are equivalent, *i.e.* whether they define the same transduction, is a natural and important problem, that has been studied by several authors. Unfortunately, this problem is already undecidable for NFT, as proven in [Griffiths 1968]. On the positive side, this problem is decidable for the classes of functional transducers we have presented (it is actually sufficient decide whether two functional transducers have the same domain, and then decide whether the disjoint union of the two transducers is functional), which explains the interest for these classes. The decidability frontier lies actually beyond functional transducers, as it encompasses the class of so-called *finite-valued transducers*, defined as the transducers for which there exists some natural number k such that, for every input word, the number of outputs associated with this input word is at most k. This decidability result has been proven for instance in [Culik II and Karhumäki 1986; Weber 1993; de Souza 2008] for finite-valued one-way transducers. In [Culik II and Karhumäki 1986], the authors claim that the equivalence problem is even decidable for the class of finite-valued two-way transducers.

3. LOGICS

A formalism based on monadic second-order logic (MSO) has been defined by [Courcelle 1994] to define transformations of graph structures. In this section, we cast this formalism to word to word functions, and refer the reader to [Courcelle and Engelfriet 2012] for more details and results about MSO transducers for general graph structures.

Words w are seen as logical structures on the domain $\{1, \dots, |w|\}$ over the signature consisting of unary predicates $a(x)$ for each symbol of Σ, and the binary predicate $x \preceq y$ for the total order on positions. Recall that MSO on words is the extension of first-order logic with quantification over sets of positions (see [Straubing 1994] for details). It is well-known by Büchi's theorem that a language is MSO-definable iff it is regular. We present extensions of this fundamental result to word functions defined by one-way and two-way transducers.

3.1. MSO transducers

The definition of MSO transducers is technical and in this paper, we rather want to define them intuitively and with examples. We refer the interested reader to [Courcelle 1994; Engelfriet and Hoogeboom 2001; Filiot 2015] for detailed definitions.

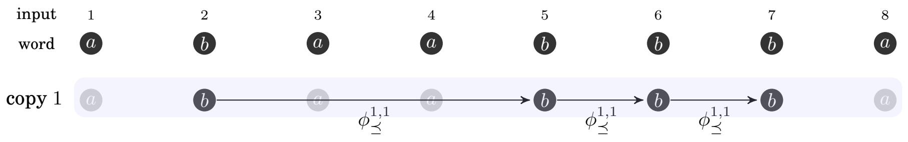
In an MSO transducer, the output word is defined as an MSO interpretation over a fixed number k of copies of the input word. Therefore, the nodes of the output word are copies 1 to k of the nodes of the input word. Output nodes are denoted x^c, for every copy c and input node x. For every copy c, only the nodes satisfying a given formula $\phi^c_{pos}(x)$ with one free first-order variable x are kept. For instance, assume one takes two copies of the input word, and in the first copy, one keeps all nodes x^1 such that x is labeled a, and in the second copy, one keeps all nodes x^2 such that x is labeled b. This is specified by the two formulas $\phi^1_{pos}(x) = a(x)$ and $\phi^2_{pos}(x) = b(x)$.

The output label and order predicates are defined by MSO formulas with one and two free first-order variables respectively, interpreted over the input structure. For instance, over the alphabet $\Sigma = \{a, b\}$, to set all the output labels to a, one just specifies the formulas $\phi^c_a(x) = \top$ and $\phi^c_b(x) = \bot$ for all copies c. The output order predicate relates input nodes of possibly different copies, and is therefore defined by formulas of the form $\phi^{c,d}_{\preceq}(x, y)$ for any copies $1 \leq c, d \leq k$.

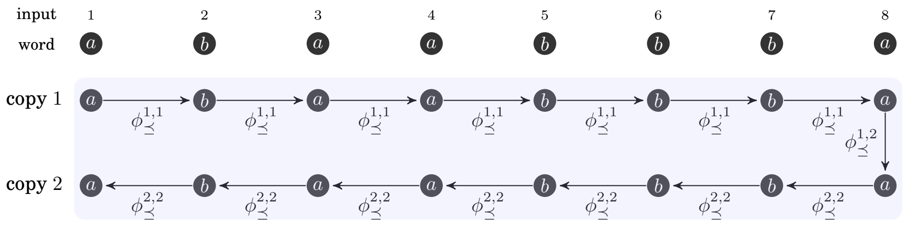
Finally, a closed MSO formula ϕ_{dom} defines the domain of the function. All in all, an MSO transducer over an alphabet Σ is a tuple

$$T = (k, \phi_{dom}, (\phi^c_{pos}(x))_{1\leq c\leq k}, (\phi^c_a(x))_{1\leq c\leq k, a\in\Sigma}, (\phi^{c,d}_{\preceq}(x, y))_{1\leq c,d\leq k})$$

The output structure may not be a word, but here we assume that an MSO transducer T outputs only word structures. It is a decidable property. Note that the length of output word of an input word of length M by an MSO transducer is bounded by kM, since one takes a fixed number k of copies of the input word.



(a) Transformation f_{del} defined by T_{del}



(b) Transformation f_{mir} defined by T_{mir}

Fig. 4. Functions defined by MSO-transducers

3.2. Examples of MSO transducers

As a first example, an MSO transducer T_{del} that realizes f_{del} is illustrated in Fig. 4(a) (only the successor relation is depicted). Input nodes filtered out by formulas $\phi^c_{pos}(x)$ are represented by fuzzy nodes. It is realized by the MSO transducer

$$T_{del}=(1, \phi_{dom} \equiv \top, \phi^1_{pos}(x) \equiv \neg a(x), (\phi^1_\sigma(x) \equiv \sigma(x))_{\sigma\in\Sigma}, \phi^{1,1}_{\preceq}(x,y) \equiv x \preceq y)$$

As a second example, consider the function f_{mir}. To realize f_{mir} with an MSO transducer (Fig. 4(b)), one needs two copies of an input word u. The labels are kept unchanged, however the order is reversed for the second copy. One also sets that all nodes of the first copy are smaller than the nodes of the second copy, as they appear before in the output word. Therefore, f_{mir} is realized by the transducer T_{mir} with $k = 2$ and

$$\begin{array}{llll} \phi_{dom} \equiv \top & \phi^c_{pos}(x) \equiv \top & \phi^c_a(x) = a(x) & \phi^c_b(x) = b(x) \\ \phi^{1,1}_{\preceq}(x,y) \equiv x \preceq y & \phi^{1,2}_{\preceq}(x,y) \equiv \top & \phi^{2,1}_{\preceq}(x,y) \equiv \bot & \phi^{2,2}_{\preceq}(x,y) \equiv y \preceq x \end{array}$$

for all copies $c \in \{1,2\}$. Note that given an input word u, the order on the output word positions is the binary relation $O = \{(i^c, j^d) \mid 1 \leq c,d \leq 2, i,j \in dom(u), u \models \phi^{c,d}_{\preceq}(i,j)\}$. Only the successor relation induced by O is depicted on the figure.

3.3. Büchi theorems for word functions

Regular functions. The first automata-logic connection for word transformations has been given by Engelfriet and Hoogeboom for the class of regular functions:

THEOREM 3.1. *[Engelfriet and Hoogeboom 2001] A function f is regular iff it is realized by some MSO transducer.*

In order to prove the direction from 2SFT to MSO transducers, one builds an MSO transducer that, given an input word u, outputs a (linear) graph that represents the run of the 2SFT on u. The converse direction is more complex, and involves an extended model of two-way transducers which can perform "MSO jumps" $\varphi(x,y)$, where $\varphi(x,y)$ is an MSO formula that defines a function from x positions to y positions. Intuitively, the machine can move from position x to position y providing $\varphi(x,y)$ holds true. 2SFT with MSO jumps are then converted, based on the Büchi theorem, into 2SFT with regular look-around. These 2SFT can move only to positions which are in the 1- neighbourhood of the current position, but their move can be based on a regular property of the current prefix and suffix of the word. Finally, it is shown that 2SFT with regular look-around are equivalent to 2SFT.

Rational functions. Using an adequate restriction of MSO transducers, one can also prove a Büchi theorem for rational functions. Intuitively, an MSO transducer is *order-preserving* if in the graphical representation of the output, there is no right-to-left edge. For instance, considering the transformations depicted on Figure 4, the transformation f_{del} satisfies this property while the transformation f_{mir} does not. Formally, this property requires that for every input word u and for every $c, d \in \{1,\dots,k\}$, if the formula $\phi^{c,d}_{\preceq}(x,y)$ evaluates to true in u, then $x \preceq y$ also evaluates to true. This property is decidable.

THEOREM 3.2. *[Bojanczyk 2014; Filiot 2015] A function f is rational iff it is realized by some order-preserving MSO transducer.*

The proof proceeds as follows: given a one-way transducer, the MSO transducer that builds as output the run of the one-way transducer naturally satisfies the order-preserving property. Conversely, the idea is to identify, for each position of the input word, the subword of the output that corresponds to this position. One can then represent the transduction as a regular language, and make use of Büchi theorem.

The power of transducer-logic connections is illustrated by the following definability problem. As shown in [Filiot et al. 2013], it is decidable whether a deterministic two-way finite state transducer T is equivalent to a one-way functional finite state transducer. As a consequence of this result and of the two previous theorems, we obtain the following corollary:

COROLLARY 3.3. *Given an MSO-transducer, it is decidable whether it is equivalent to some order-preserving MSO-transducer.*

4. ALGEBRA

Similarly to regular languages, sequential and rational functions can be characterized by the index finiteness of well-chosen (canonical) congruences. For the class of regular functions, no algebraic characterizations are known but we present partial results based on algebraic properties of the transition structure of two-way transducers (their transition monoid). For the class of regular functions with *origin information*, i.e. regular functions extended with pointers from any output position to some input position (intuitively, the input position from which they originate), an algebraic characterization is known [Bojanczyk 2014]. We present origin information at the end of this section.

We recall that a left (resp. right) congruence on Σ^* is an equivalence relation $\equiv$ such that for all $u, v \in \Sigma^*$ and $\sigma \in \Sigma$, if $u \equiv v$ then $\sigma u \equiv \sigma v$ (resp. if $u \equiv v$ then $u\sigma \equiv v\sigma$). For $u \in \Sigma^*$, we denote by $[u]_\equiv$ the class of u, or just $[u]$ when it is clear from the context. A congruence $\equiv$ (left or right) has *finite-index* if its quotient $\Sigma^*/_\equiv$ has finitely many classes.

In this section, we also denote by $\preceq$ the prefix relation on words, and by $u \wedge v$ the longest common prefix of any two words u and v. Given a language $L \subseteq \Sigma^*$ and a word $u \in \Sigma^*$, we denote by $u^{-1}L$ the residual of L by u, i.e. the set of words w such that $uw \in L$. When $u \preceq v$, we denote by $u^{-1}v$ the unique word w such that $v = uw$.

4.1. Sequential functions

It is well-known that any regular language is recognized by a unique minimal complete deterministic automaton. Intuitively, for a language L, any two words u, v which behave equivalently with respect to continuations w ($uw \in L$ iff $vw \in L$) must reach the same state in a minimal deterministic automaton. This is captured by the right congruence $\equiv_L$: $u \equiv_L v$ if for all $w \in \Sigma^*$, $uw \in L$ iff $vw \in L$. The well-known Myhill-Nerode's theorem states that L is regular iff $\equiv_L$ has finite index. Moreover, the index of $\equiv_L$ is exactly the number of states of the minimal complete deterministic automaton recognizing L.

In this section, we explain how to extend the characterization of regular languages to sequential functions, an extension due to Choffrut [Choffrut 1979; Choffrut 2003]. In a minimal sequential transducer defining a function $f : \Sigma^* \to \Sigma^*$, two (input) words u, v must go to the same state if they behave the same w.r.t. dom(f) ($u \equiv_{\mathrm{dom}(f)} v$), but also w.r.t. the outputs, in the sense that the output produced when processing a continuation w should not depend on u and v. To capture this idea, one first defines a canonical way of producing the output, via a total function $\hat{f} : \Sigma^* \to \Sigma^*$ which, given a word u, outputs the longest common prefix of all words $f(uw)$ for all continuations $w \in u^{-1}\mathrm{dom}(f)$:

$$\hat{f}(u) = \bigwedge\{f(uw) \mid w \in u^{-1}\mathrm{dom}(f)\} \quad \text{where } \bigwedge \varnothing \text{ is set to } \epsilon.$$

Then, one defines the relation $\equiv_f$ by $u \equiv_f v$ if the following two statements hold:

$$(i)\ u \equiv_{\text{dom}(f)} v \quad (ii)\ \forall w \in u^{-1}\mathbf{dom}(f),\ \hat{f}(u)^{-1}f(uw) = \hat{f}(v)^{-1}f(vw)$$

Note that $\hat{f}(u)^{-1}$ is necessarily a prefix of $f(uw)$, and similarly for $\hat{f}(v)^{-1}$, by definition of $\hat{f}$. It turns out that $\equiv_f$ is a right congruence, which allows one to construct a transducer T_f for f (possibly with infinitely many states). The set of states of T_f is $\Sigma^*/_{\equiv_f}$, with initial state $[\epsilon]$ and set of accepting states $\text{dom}(f)/_{\equiv_f}$. The transitions are $([u], a, w, [ua])$ with $u \in \Sigma^*$, $a \in \Sigma$ and $w = \hat{f}(u)^{-1}\hat{f}(u\sigma)$. The terminal function is $t : [u] \mapsto \hat{f}(u)^{-1}f(u)$ for $u \in \text{dom}(f)$.

Note that if $\equiv_f$ has finite index, then T_f is a proper sequential transducer with finitely many states, and therefore f is sequential. The converse is also true, and one gets a Myhill-Nerode theorem for sequential functions, due to Choffrut:

THEOREM 4.1. *[Choffrut 2003; Choffrut 1979] A function $f : \Sigma^* \to \Sigma^*$ is sequential iff $\equiv_f$ has finite index.*

Note that T_f is canonical. Moreover when f is sequential, then T_f is minimal, and any sequential transducer realizing f can be uniquely mapped to T_f, through a notion of transducer morphism, e.g. defined in [Choffrut 2003]. In that sense, T_f is unique.

4.2. Rational functions

In this section, we present an algebraic characterization, as well as a canonical transducer construction, for rational functions, due to Reutenauer and Schützenberger. This characterization is based on the model of *bimachines*, which are essentially sequential transducers extended with regular look-ahead. We adopt the latter view, which will hopefully allow us to give an intuitive explanation of Reutenauer and Schützenberger's result. This formalization also appeared in [Boiret et al. 2012].

Rational functions with infinite syntactic congruence. Let us see why the finiteness of $\equiv_f$ fails at characterizing rational (but non-sequential) functions. Over $\Sigma = \{a, b\}$, consider the function f_{sw} of Example 2.1. The congruence $\equiv_{\text{dom}(f_{\text{sw}})}$ has only one class Σ^*. The function $\hat{f}_{sw}$ is constant equal to ϵ: for any $u \in \Sigma^*$, $\hat{f}_{sw}(u) \preceq (f_{sw}(ua) \wedge f_{sw}(ub)) = au \wedge bu = \epsilon$. Therefore, for all $u, v \in \Sigma^*$ and all $w \in \Sigma^*$, $\hat{f}_{sw}(u)^{-1}f_{sw}(uw) = \hat{f}_{sw}(u)^{-1}(v)f_{sw}(vw)$ iff $f_{sw}(uw) = f_{sw}(vw)$ iff $u = v$. Therefore, $\equiv_{f_{\text{sw}}}$ has infinite index.

Now, consider the following two restrictions on $\text{dom}(f_{sw})$: $\text{last}_a = \{ua \mid u \in \Sigma^*\}$ and $\text{last}_b = \{ub \mid u \in \Sigma^*\}$, then the function f_{sw} is sequential on these two restricted domains. This suggests that modulo some information about the suffix (whether it ends with a or b in this case), the function is sequential. Unlike the latter example, the suffix information needed to be sequential may change along an input word. Consider for instance the iterated version of f_{sw} that we denote f_{sw}^*, which is defined over the alphabet $\Delta = \Sigma \cup \{\#\}$ by:

$$f_{sw}^* \ :\ u_1\#u_2\#\dots\#u_n \mapsto f_{sw}(u_1)\#f_{sw}(u_2)\#\dots\#f_{sw}(u_n) \qquad \text{where } u_i \in \Sigma^* \text{ and } n \geq 0$$

To realize f_{sw}^* in a sequential way, a transducer needs to know, when reading a symbol $\sigma \in \{a, b\}$, if it is the last letter of a block in $\{a, b\}^+$, and otherwise whether the last letter of this block is an a or a b. Modulo this look-ahead information, the function f_{sw}^* is sequential.

Sequentiality modulo regular look-ahead. Let us formalise the notion of sequentiality modulo (regular) look-ahead information. A *look-ahead information* $\mathcal{L}$ is a partition of Σ^* assumed to be the quotient of Σ^* by a left congruence of finite index, denoted by $\equiv_{\mathcal{L}}$. We denote by $[u]_{\mathcal{L}}$ the class of a word u. The *look-ahead extension* is

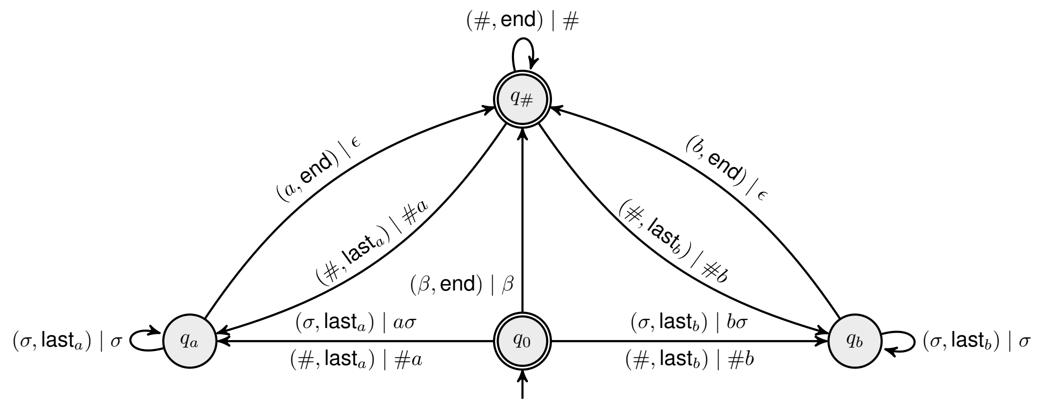

$(\#, \text{end}) \mid \#$
$q_\#$
$(a, \text{end}) \mid \epsilon$
$(b, \text{end}) \mid \epsilon$
$(\#, \text{last}_a) \mid \#a$
$(\#, \text{last}_b) \mid \#b$
$(\beta, \text{end}) \mid \beta$
$(\sigma, \text{last}_a) \mid \sigma$
q_a
$(\sigma, \text{last}_a) \mid a\sigma$
q_0
$(\sigma, \text{last}_b) \mid b\sigma$
q_b
$(\sigma, \text{last}_b) \mid \sigma$
$(\#, \text{last}_a) \mid \#a$
$(\#, \text{last}_b) \mid \#b$

Fig. 5. Sequential transducer realizing $f^*_{sw}[\mathcal{L}]$, where $\sigma \in \{a, b\}$ and $\beta \in \{a, b, \#\}$.

the function $e_\mathcal{L} : \Sigma^* \to (\Sigma \times \mathcal{L})^*$ defined for all $u = \sigma_1 \dots \sigma_n \in \Sigma^*$ by $e_\mathcal{L}(u) = (\sigma_1, [\sigma_2 \dots \sigma_n]_\mathcal{L})(\sigma_2, [\sigma_3 \dots \sigma_n]_\mathcal{L}) \dots (\sigma_n, [\epsilon]_\mathcal{L})$. A function $f : \Sigma^* \to \Sigma^*$ is *sequential modulo* $\mathcal{L}$ (or $\mathcal{L}$-sequential) if the function denoted $f[\mathcal{L}] : (\Sigma \times \mathcal{L})^* \to \Sigma^*$ defined on $e_\mathcal{L}(\text{dom}(f))$ by $f[\mathcal{L}](e_\mathcal{L}(u)) = f(u)$, is sequential.

Note that a function f over an alphabet Σ is sequential iff it is $\{\Sigma^*\}$-sequential. As a second example, the function f^*_{sw} is $\mathcal{L}$-sequential for $\mathcal{L} = \{\text{end}, \text{last}_a, \text{last}_b\}$ where $\text{end} = \epsilon + \#\Delta^*$, and for $\sigma \in \{a, b\}$, $\text{last}_\sigma = (a + b)^*\sigma(\epsilon + \#\Delta^*)$. The function $f^*_{sw}[\mathcal{L}]$ is realized by the sequential transducer of Fig. 5. Note that this transducer also checks that the look-ahead annotations are correct, i.e. it ensures that the domain is exactly $e_\mathcal{L}(\text{dom}(f^*_{sw}))$. For instance, whenever the information end is read, the transducer moves to state $q_\#$, from which only $\#$ can be read. This ensures that the annotation by look-ahead information end is correct. The only way to access state q_a is when reading the look-ahead information last_a, and the only way to leave q_a is when reading (a, end), thus ensuring that the last letter of the block is indeed a. Since the transducer is sequential, has domain $e_\mathcal{L}(\text{dom}(f^*_{sw}))$, and the look-ahead annotation is unique for each word, projecting away the look-ahead information on this transducer yields an unambiguous transducer realizing f^*_{sw}.

A canonical look-ahead. It is known [Elgot and Mezei 1965] that any rational function $f : \Sigma^* \to \Sigma^*$ equals the composition of two functions $r : \Sigma^* \to \Delta^*$ and $\ell : \Delta^* \to \Sigma^*$ for some intermediate alphabet Δ, i.e. $f = \ell \circ r$, where ℓ is sequential and r is right-sequential, i.e. the function $m \circ r \circ m$ is sequential, where m is the function that mirrors input words. In other words, r can be realized by a sequential transducer that processes input words backwards and produces output words backwards. Moreover, T can be chosen to be letter-to-letter, i.e. produce exactly one output symbol per input symbol, and hence T is nothing else than an automaton that computes some look-ahead information (the output symbols). We can rephrase this decomposition result in terms of $\mathcal{L}$-sequentiality:

THEOREM 4.2. *[Elgot and Mezei 1965] f is rational iff it is $\mathcal{L}$-sequential for some $\mathcal{L}$ of finite index.*

The strong result of Reutenauer and Schützenberger is precisely to show that, in the latter proposition, $\mathcal{L}$ can be chosen in a canonical way, that depends only on f, denoted $\mathcal{L}_f$. The idea is to identify suffixes u and v that have only a bounded "difference" on the outputs $f(wu)$ and $f(wv)$ for all w such that $wu, wv \in \text{dom}(f)$. To quantify this effect, they use the notion of *delay distance* between words, defined by $d(t_1, t_2) = |t_1| + |t_2| - 2|t_1 \wedge t_2|$. In other words, this distance only counts what remains when the longest common prefix of t_1 and t_2 has been cut out. Then, two words u, v are equivalent for $\equiv_{\mathcal{L}_f}$ if (i) for all $w \in \Sigma^*$, $wu \in \mathsf{dom}(f)$ iff $wv \in \mathsf{dom}(f)$, and (ii) the set $\{d(f(wu), f(wv)) \mid wu, wv \in \mathsf{dom}(f)\}$ is finite. It turns out that $\equiv_{\mathcal{L}_f}$ is a left congruence, and it is of finite index when f is rational.

Example 4.3. As an example, consider again the function f^*_{sw} defined before. The look-ahead given before is actually the canonical one, i.e. $\mathcal{L}_{f^*_{sw}} = \{\mathsf{end}, \mathsf{last}_a, \mathsf{last}_b\}$. Indeed, let $\sigma, \beta \in \{a, b\}$, $u, v \in \{a, b\}^*$ and $u', v' \in \mathsf{end}$. Then, any w such that $wu\sigma u' \in \mathsf{dom}(f^*_{sw})$ and $wv\beta v' \in \mathsf{dom}(f^*_{sw})$ is of the form $w_1 w_2$ where $w_1 \in \Delta^*\#$ and $w_2 \in \Sigma^*$. Then, $\{d(f^*_{sw}(w_1 w_2 u\sigma u'), f^*_{sw}(w_1 w_2 v\beta v')) \mid w_1 \in \Delta^*\#, w_2 \in \Sigma^*\} = \{d(f^*_{sw}(w_1)\sigma w_2 u f^*_{sw}(u'), f^*_{sw}(w_1)\beta w_2 v f^*_{sw}(v')) \mid w_1 \in \Delta^*\#, w_2 \in \Sigma^*\}$, which is finite iff $\beta = \sigma$. This gives the two classes last_a and last_b. Similarly, it is possible to check that all words in the set end are equivalent.

We can now state Reutenauer and Schützenberger's result:

THEOREM 4.4. *[Reutenauer and Schützenberger 1991] Let $f : \Sigma^* \to \Sigma^*$. The following three statements are equivalent:*

(*1*) *f is rational,*
(*2*) *$\mathcal{L}_f$ has finite index and f is $\mathcal{L}_f$-sequential,*
(*3*) *$\mathcal{L}_f$ and $\equiv_{f[\mathcal{L}_f]}$ have finite index.*

When f is given by a transducer, a canonical sequential transducer for $f[\mathcal{L}_f]$ can be constructed effectively. Projecting its input symbols on Σ (i.e. discarding the look-ahead information), one gets a canonical unambiguous transducer T_f realizing f.

Bimachines. The latter theorem was originally presented based on the notion of bimachines. A bimachine is made of a right deterministic automaton R reading input words from right to left, a deterministic automaton L, and an output function ω which takes states of L, states of R and symbols in Σ as arguments, and produces a word. The output produced at position i in a word $w = \sigma_1 \dots \sigma_n$ is $\omega(l, \sigma_i, r)$ where l is the state of L reached after processing the prefix $\sigma_1 \dots \sigma_i$ (if it exists), and r is the state of R reached after reading the suffix $\sigma_i \dots \sigma_n$. Reutenauer and Schützenberger's result is precisely to show that for any rational function f, there exists a canonical right automaton R_f, a left automaton $L[R_f]$ (which is canonical once the right automaton is fixed), and an output function ω, which defined a canonical bimachine for f. Translated in the look-ahead framework, R_f defines the look-ahead information $\mathcal{L}_f$, and by taking the product of R_f and $L[R_f]$, one obtains a sequential transducer realizing $f[\mathcal{L}_f]$.

4.3. First-order definable functions

If only first-order formulas are allowed in the definition of (order-preserving) MSO-transducers, one gets the subclass of (order-preserving) FO-transducers. A function is (order-preserving) first-order definable if it is definable by an (order-preserving) FO-transducer. First-order definable languages L are characterized by languages having aperiodic syntactic congruence[4] $\equiv_L$, which yields a decision procedure for automata: minimize the automaton and check the aperiodicity of its transition congruence. We present similar results for rational functions and partial results for functions definable by two-way transducers.

[4]A congruence $\equiv$ on Σ^* is aperiodic if there exists $n_0 \in \mathbb{N}$ such that for all $u, v \in \Sigma^*$ and $n \geq n_0$, $u^n \equiv v^n$ iff $u^{n+1} \equiv v^{n+1}$.

The *transition congruence* $\equiv_A$ of an automaton A is defined by $u \equiv_A v$ if for all states p, q, it holds $p \xrightarrow{u} q$ iff $p \xrightarrow{v} q$. The transition congruence of a transducer is the transition congruence of its underlying (input) automaton, and a transducer is aperiodic if its transition congruence is aperiodic. It is known that a rational function f is order-preserving first-order definable iff it is realized by some aperiodic transducer [Filiot et al. 2016]. Moreover, a function f is realizable by some aperiodic transducer iff the canonical left congruence $\mathcal{L}_f$ and the right congruence $\equiv_{f[\mathcal{L}_f]}$ are both aperiodic, which is decidable when f is given by some transducer. Therefore, one gets the following theorem:

THEOREM 4.5. *[Lhote 2015; Filiot et al. 2016] It is decidable whether a transducer defines an order-preserving first-order function.*

The problem of deciding whether a transducer is equivalent to some aperiodic and functional one has been generalized to arbitrary congruence varieties in [Filiot et al. 2016]. It is shown in this case that the congruences $\mathcal{L}_f$ and $\equiv_{f[\mathcal{L}_f]}$ may not be in the varieties, even if some transducer realizing the function is, but at least one pair of congruences $\mathcal{L}$ and $\equiv_{f[\mathcal{L}]}$ is in the variety, where those pairs are taken in a finite computable set of congruences.

No such results are known for functions definable by deterministic two-way transducers, mainly because for such functions the existence of a canonical device is still open. Nevertheless, the notion of transition congruence can be extended to two-way automata and therefore to two-way transducers. Roughly, it identifies words with the same state behaviors, where the behaviors are either left-to-right, right-to-left, left-to-left and right-to-right. For instance, left-to-right behaviors are exactly as for classical one-way automata, and a pair of states (p, q) if a left-to-left behavior of a word u if there is a run that enters u from the left in state p, and leaves u to the left in state q.

THEOREM 4.6. *[Carton and Dartois 2015] A function f is first-order definable iff it is realized by some aperiodic two-way transducer.*

In [Bojanczyk 2014], a *transduction with origin* is a function from words u to pairs (v, o), where v is a word, and o is an origin mapping that sends any position of v to a position of u, the position from which "it has been created". Most transducer models, including MSO- and FO-transducers, implicitly bear origin information. We denote by $[\![T]\!]_o$ the origin transduction defined by a transducer. For regular transductions with origin, an algebraic characterization is known (see [Bojanczyk 2014]), based on which first-order definability can be decided:

THEOREM 4.7. *[Bojanczyk 2014] Given a two-way transducer T realizing a transduction with origin $[\![T]\!]_o$, it is decidable whether there exists an FO-transducer T' such that $[\![T]\!]_o = [\![T']\!]_o$.*

5. EXTENSIONS AND PERSPECTIVES

Functions of finite words enjoy multiple presentations by means of transducers, logic and algebra. We have presented some old and recent results using these different tools, as well as some nice equivalence results. In this conclusion, we present a recent alternative automaton model, as well as some extensions to other structures than finite words.

Streaming String Transducers. Recently, an alternative model of transducers has been introduced in [Alur and Černý 2011], named streaming string transducers (SST for short). Intuitively, it consists in a deterministic one-way automaton extended with a finite number of registers, valued with finite words. These registers are updated along

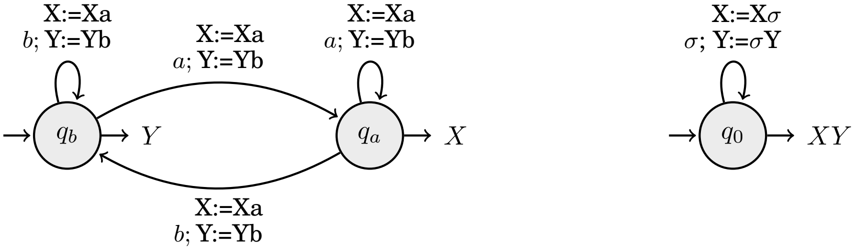


Fig. 6. Two streaming string transducers.

transitions but never tested, and they are used to define the output of a run. Two examples of SST are depicted on Figure 6. The left SST realizes the function mapping any word of the form $u\sigma$ to $\sigma^{|u|}$, and ϵ to itself, while the right SST realizes the function f_{mir}.

Interestingly, it has been proven that a simple restriction of this model, so-called copyless SST (updates should make a linear use of registers) exactly coincides with the class of regular functions [Alur and Černý 2010]. Intuitively, this means that the model allows to transfer the complexity of runs of two-way transducers to the updates of variables. Similarly, a simple restriction of SST (called right-appending SST) coincides with the class of rational functions, namely that in which updates are of the form $X := Yu$, where X and Y are registers, and u is a finite word. This model has been applied to the verification of list-processing programs [Alur and Černý 2011], and implemented [Alur et al. 2015].

Concerning first-order definable transformations, a result similar to Theorem 4.6 has been provided for streaming string transducers in [Filiot et al. 2014; Dartois et al. 2016b], using an adequate notion of transition monoid of an SST.

Another natural problem for this model is the notion of register complexity of a function, *i.e.* the minimal number of registers needed to realize a regular function. This problem has attracted a lot of attention recently. In [Daviaud et al. 2016], a solution is proposed for the class of right-appending SST using a generalization of the twinning property, a tool that has been introduced in [Choffrut 1977] to characterize sequential functions among rational ones. In [Baschenis et al. 2016], the authors consider also a register minimization problem but for the class of non-deterministic SST where concatenations of registers are forbidden in the register updates.

Infinite words. The determinization problem, i.e. deciding whether a transducer defines a sequential function, has been extended to infinite word transducers with Büchi acceptance condition in [Béal and Carton 2002]. Correspondence between MSO and SST on infinite strings have been shown in [Alur et al. 2012].

Trees and nested words. Tree transducers have been studied in numerous works, in particular by Joost Engelfriet and Sebastian Maneth, and we will not give here an exhaustive list. However, in order to echo the transducer-logic connection that we have presented in this paper, let us mention that it has been in shown in [Engelfriet and Maneth 2003] that MSO-definable tree transducers exactly coincide with Macro Tree Transducers that are linear-size increase, *i.e.* the size of the output tree is always linear in the size of the input tree. Let us also mention the survey [Maneth 2015] about the equivalence problem for tree transducers.

Recently, we have also considered transducers whose inputs are tree linearizations represented as nested words [Filiot et al. 2010]. We have defined so called visibly pushdown transducers and have proven that several positive results of one-way transducers are preserved by this model. Recently, we have shown that MSO-definable nested word-to-word transformations and a two-way model of visibly pushdown transducers are equivalent [Dartois et al. 2016a].

Some perspectives. For the class of regular functions, we have seen logical and automata models. An interesting and challenging research direction is to build an algebraic framework for regular transductions.

On the logical side, MSO transducers can be transformed into two-way or streaming transducers in non-elementary time. This blow-up is not avoidable, which raises the question of having MSO expressive logics well-suited to define transductions, and that enjoy better complexities.

Finally, an interesting direction, which generalizes the classical Church synthesis problem, is that of sequential uniformisation. The problem is to decide whether from a given word relation, one can extract a function such that (i) it has the same domain as the relation, (ii) it is included in the relation, and (iii) it belongs to some class of functions with interesting properties. The relation can be thought of as a set of good behaviours of a system, and the function as the behaviour of the synthesised system. The required properties of the targeted class of functions depend on the applications. For instance, one may target sequential functions for memory efficiency. For rational relations and the class of sequential functions, this problem is undecidable, but decidable when restricted to finite-valued rational relations or deterministic rational relations [Filiot et al. 2016].

Acknowledgments

We thank the editorial board of ACM SIGLOG newsletter for giving us the opportunity to present these exciting results. We would like to warmly thank Nathan Lhote for his careful reading on the algebra section.

This work was partially supported by the French ExStream project (ANR-13- JS02-0010), the Belgo-French PHC project VAST (35961QJ) funded by Campus France and WBI, the ARC project Transform (Federation Wallonia-Brussels) and the Belgian FNRS CDR project Flare. Emmanuel Filiot is research associate at F.R.S.-FNRS.

REFERENCES

Rajeev Alur and Pavol Černý. 2010. Expressiveness of streaming string transducers. In *Foundations of Software Technology and Theoretical Computer Science (FSTTCS) (LIPIcs)*, Vol. 8. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 1–12.

Rajeev Alur and Pavol Černý. 2011. Streaming transducers for algorithmic verification of single-pass list-processing programs. In *Proc. of the 38th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2011*. ACM, 599–610.

Rajeev Alur, Loris D'Antoni, and Mukund Raghothaman. 2015. DReX: A Declarative Language for Efficiently Evaluating Regular String Transformations. In *Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2015, Mumbai, India, January 15-17, 2015*. 125–137.

Rajeev Alur, Emmanuel Filiot, and Ashutosh Trivedi. 2012. Regular Transformations of Infinite Strings. In *Proceedings of the 27th Annual IEEE Symposium on Logic in Computer Science, LICS 2012, Dubrovnik, Croatia, June 25-28, 2012*. 65–74.

Félix Baschenis, Olivier Gauwin, Anca Muscholl, and Gabriele Puppis. 2015. One-way definability of sweeping transducers. In *35th International Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS 2015 (LIPIcs)*, Vol. 45. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 178–191.

Félix Baschenis, Olivier Gauwin, Anca Muscholl, and Gabriele Puppis. 2016. Minimizing resources of sweeping and streaming string transducers. In *In Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP'16) (LIPIcs)*. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik. To appear.

Marie-Pierre Béal and Olivier Carton. 2002. Determinization of transducers over finite and infinite words. *Theor. Comput. Sci.* 289, 1 (2002), 225–251.

Marie-Pierre Béal, Olivier Carton, Christophe Prieur, and Jacques Sakarovitch. 2003. Squaring transducers: an efficient procedure for deciding functionality and sequentiality. *Theoretical Computer Science* 292, 1 (2003), 45–63.

Jean Berstel and Luc Boasson. 1979. Transductions and context-free languages. *Ed. Teubner* (1979), 1–278.

Adrien Boiret, Aurélien Lemay, and Joachim Niehren. 2012. Learning Rational Functions. In *Developments in Language Theory - 16th International Conference, DLT 2012, Taipei, Taiwan, August 14-17, 2012. Proceedings (Lecture Notes in Computer Science)*. Springer, 273–283.

Mikolaj Bojanczyk. 2014. Transducers with Origin Information. In *41st Internationl Colloquium on Automata, Languages, and Programming (ICALP) (LNCS)*, Vol. 8573. Springer, 26–37.

J. R. Büchi. 1960. Weak second-order arithmetic and finite automata. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik* 6, 1–6 (1960), 66–92.

Olivier Carton and Luc Dartois. 2015. Aperiodic Two-way Transducers and FO-Transductions. In *Computer Science Logic (CSL) (LIPIcs)*, Vol. 41. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 160–174.

Christian Choffrut. 1977. Une Caractérisation des Fonctions Séquentielles et des Fonctions Sous-Séquentielles en tant que Relations Rationnelles. *Theor. Comput. Sci.* 5, 3 (1977), 325–337.

Christian Choffrut. 1979. A Generalization of Ginsburg and Rose's Characterization of G-S-M Mappings. In *Automata, Languages and Programming, 6th Colloquium, Graz, Austria, July 16-20, 1979, Proceedings*. 88–103.

Christian Choffrut. 2003. Minimizing subsequential transducers: a survey. *Theor. Comput. Sci.* 292, 1 (2003), 131–143.

Bruno Courcelle. 1994. Monadic Second-Order Definable Graph Transductions: A Survey. *Theor. Comput. Sci.* 126 (1994), 53–75.

Bruno Courcelle and Joost Engelfriet. 2012. *Graph Structure and Monadic Second-Order Logic - A Language-Theoretic Approach*. Encyclopedia of mathematics and its applications, Vol. 138. Cambridge University Press.

K. Culik and J. Karhumaki. 1987. The Equivalence Problem for Single-Valued Two-Way Transducers (on NPDT0L Languages) is Decidable. *SIAM J. Comput.* 16, 2 (1987), 221–230.

Karel Culik II and Juhani Karhumäki. 1986. The Equivalence of Finite Valued Transducers (On HDT0L Languages) is Decidable. *Theor. Comput. Sci.* 47, 3 (1986), 71–84.

Luc Dartois, Emmanuel Filiot, Pierre-Alain Reynier, and Jean-Marc Talbot. 2016a. Two-Way Visibly Pushdown Automata and Transducers. In *Proc. 31st Annual IEEE Symposium on Logic in Computer Science (LICS'16)*. IEEE Computer Society. To appear.

Luc Dartois, Ismaël Jecker, and Pierre-Alain Reynier. 2016b. Aperiodic String Transducers. In *Proc. 20th International Conference on Developments in Language Theory (DLT 2016) (Lecture Notes in Computer Science)*. Springer. To appear.

Laure Daviaud, Pierre-Alain Reynier, and Jean-Marc Talbot. 2016. A Generalized Twinning Property for Minimisation of Cost Register Automata. In *Proc. 31st Annual IEEE Symposium on Logic in Computer Science (LICS'16)*. IEEE Computer Society. To appear.

Rodrigo de Souza. 2008. On the Decidability of the Equivalence for k-Valued Transducers. In *Developments in Language Theory, 12th International Conference, DLT 2008, Kyoto, Japan, September 16-19, 2008. Proceedings (Lecture Notes in Computer Science)*, Vol. 5257. Springer, 252–263.

Volker Diekert, Paul Gastin, and Manfred Kufleitner. 2008. A Survey on Small Fragments of First-Order Logic over Finite Words. *Int. J. Found. Comput. Sci.* 19, 3 (2008), 513–548.

C. C. Elgot. 1961. Decision Problems of Finite Automata Design and Related Arithmetics. *In Transactions of the American Mathematical Society* 98, 1 (1961), 21–51.

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

Joost Engelfriet and Hendrik Jan Hoogeboom. 2001. MSO definable string transductions and two-way finite-state transducers. *ACM Trans. Comput. Log.* 2, 2 (2001), 216–254.

Joost Engelfriet and Sebastian Maneth. 2003. Macro Tree Translations of Linear Size Increase are MSO Definable. *SIAM J. Comput.* 32, 4 (2003), 950–1006.

Emmanuel Filiot. 2015. Logic-Automata Connections for Transformations. In *Logic and Its Applications (ICLA)*. Springer, 30–57.

Emmanuel Filiot, Olivier Gauwin, and Nathan Lhote. 2016. First-order definability of rational transductions: an algebraic approach. In *Logic in Computer Science (LICS)*. IEEE.

Emmanuel Filiot, Olivier Gauwin, Pierre-Alain Reynier, and Frédéric Servais. 2013. From Two-Way to One-Way Finite State Transducers. In *Logic in Computer Science (LICS)*. IEEE, 468–477.

Emmanuel Filiot, Ismaël Jecker, Christof Löding, and Sarah Winter. 2016. On Equivalence and Uniformisation Problems for Finite State Transducers. In *ICALP*. To appear.

Emmanuel Filiot, Shankara Narayanan Krishna, and Ashutosh Trivedi. 2014. First-order Definable String Transformations. In *Foundation of Software Technology and Theoretical Computer Science, (FSTTCS) (LIPIcs)*, Vol. 29. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 147–159.

Emmanuel Filiot, Jean-Franois Raskin, Pierre-Alain Reynier, Frédéric Servais, and Jean-Marc Talbot. 2010. Properties of Visibly Pushdown Transducers. In *Proc. 35th International Symposium on Mathematical Foundations of Computer Science (MFCS'10) (Lecture Notes in Computer Science)*, Vol. 6281. Springer, 355–367. DOI:http://dx.doi.org/10.1007/978-3-642-15155-2_32

T. V. Griffiths. 1968. The Unsolvability of the Equivalence Problem for Lambda-Free Nondeterministic Generalized Machines. *J. ACM* 15, 3 (1968), 409–413. DOI:http://dx.doi.org/10.1145/321466.321473

Eitan M. Gurari and Oscar H. Ibarra. 1983. A note on finite-valued and finitely ambiguous transducers. *Mathematical systems theory* 16, 1 (1983), 61–66.

Nathan Lhote. 2015. Towards an algebraic characterization of rational word functions. *CoRR* abs/1506.06497 (2015). http://arxiv.org/abs/1506.06497

Sebastian Maneth. 2015. A Survey on Decidable Equivalence Problems for Tree Transducers. *Int. J. Found. Comput. Sci.* 26, 8 (2015), 1069–1100. DOI:http://dx.doi.org/10.1142/S0129054115400134

Christophe Reutenauer and Marcel-Paul Schützenberger. 1991. Minimization of Rational Word Functions. *SIAM J. Comput.* 20, 4 (1991), 669–685.

Marcel Paul Schützenberger. 1975. Sur les relations rationnelles. In *Proc. 2nd GI Conference on Automata Theory and Formal Languages, Kaiserslautern, May 20-23, 1975 (Lecture Notes in Computer Science)*, Vol. 33. Springer, 209–213.

Howard Straubing. 1994. *Finite Automata, Formal Logic, and Circuit Complexity*. Birkhäuser, Boston, Basel and Berlin.

Boris Avraamovich Trakhtenbrot. 1961. Finite automata and logic of monadic predicates (in Russian). *Dokl. Akad. Nauk SSSR* 140 (1961), 326–329.

Andreas Weber. 1993. Decomposing Finite-Valued Transducers and Deciding Their Equivalence. *SIAM J. Comput.* 22, 1 (1993), 175–202.

Andreas Weber and Reinhard Klemm. 1995. Economy of Description for Single-Valued Transducers. *Information and Computation* 118, 2 (1995), 327–340.