On Canonical Models for Rational Functions over Infinite Words

Emmanuel Filiot, Olivier Gauwin, Nathan Lhote, Anca Muscholl

On Canonical Models for Rational Functions over Infinite Words

Emmanuel Filiot
Université Libre de Bruxelles, Belgium
efiliot@ulb.ac.be

Olivier Gauwin
LaBRI, Université de Bordeaux, France
olivier.gauwin@labri.fr

Nathan Lhote
LaBRI, Université de Bordeaux, France and Université Libre de Bruxelles, Belgium
nlhote@labri.fr

Anca Muscholl
LaBRI, Université de Bordeaux, France
anca@labri.fr

Abstract

This paper investigates canonical transducers for rational functions over infinite words, i.e., functions of infinite words defined by finite transducers. We first consider sequential functions, defined by finite transducers with a deterministic underlying automaton. We provide a Myhill-Nerode-like characterization, in the vein of Choffrut's result over finite words, from which we derive an algorithm that computes a transducer realizing the function which is minimal and unique (up to the automaton for the domain).

The main contribution of the paper is the notion of a canonical transducer for rational functions over infinite words, extending the notion of canonical bimachine due to Reutenauer and Schützenberger from finite to infinite words. As an application, we show that the canonical transducer is aperiodic whenever the function is definable by some aperiodic transducer, or equivalently, by a first-order transduction. This allows to decide whether a rational function of infinite words is first-order definable.

2012 ACM Subject Classification Theory of computation → Formal languages and automata theory → Automata extensions → Transducers

Keywords and phrases transducers, infinite words, minimization, aperiodicity, first-order logic

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2018.30

Funding This work was supported by the French ANR projects *ExStream* (ANR-13-JS02-0010) and *DeLTA* (ANR-16-CE40-0007), the Belgian FNRS CDR project *Flare* (J013116) and the ARC project *Transform* (Fédération Wallonie Bruxelles).

Introduction

Machine models, such as automata and their extensions, describe mathematical objects in a finite way. Finite automata, for instance, describe languages (of words, trees, etc). A canonization function $\mathcal{C}$ is a function from and to machine models (not necessarily of the same type) such that, whenever two machines M_1, M_2 describe the same object, then $\mathcal{C}(M_1) = \mathcal{C}(M_2)$. Accordingly, $\mathcal{C}(M_1)$ is called the *canonical model* of the object described by M_1, and it does not depend on the initial representation of the object. A classical example of

38th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2018).
Editors: Sumit Ganguly and Paritosh Pandya; Article No. 30; pp. 30:1–30:34

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

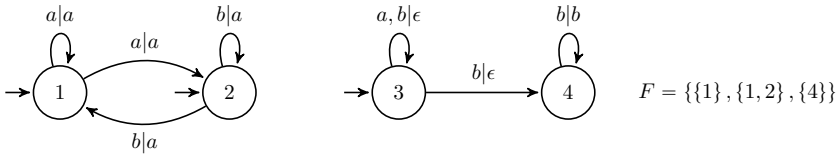

Figure 1 Transducer with Muller sets F realizing the function $f_{\#a}$ mapping any word with infinitely many a to a^ω, otherwise to b^ω.

canonization is the function which associates with a finite automaton its equivalent minimal deterministic automaton. A canonization function becomes interesting when it satisfies additional constraints like being computable, preserving some algebraic properties, and enjoying minimal models. Canonical models not only shed light on the intrinsic characteristics of the class of objects they describe, but can also serve to decide *definability* problems. For instance, it is well-known that the minimal DFA of a word language L is aperiodic if and only if L is definable in first-order logic [18, 22]. Hence, this allows to decide whether a monadic second-order formula has an equivalent first-order one over words. This result has been extended to infinite words [24, 25, 1, 19], although there is no unique minimal automaton for languages of infinite words (see also [10] for a survey).

Rational functions are functions defined by word transducers. A canonical model for rational functions over *finite* words has been introduced in [21]. This result, which can be considered as one of the jewels of transducer theory, states the existence of a procedure that computes from a given transducer a canonical input-deterministic transducer with look-ahead, called bimachine. For the subclass of functions realized by input-deterministic transducers, called sequential functions, it is even possible to compute a unique and minimal transducer realizing the function [8]. For rational functions, the procedure of [21], though it preserves aperiodicity of the transition congruence of the transducer, does not preserve other congruence varieties, in general. In [14, 15] it was shown how to adapt [21] to obtain a canonization procedure which overcomes this issue. Later it was shown that the first-order definability problem for rational functions is PSpace-c [13]. In a different setting, functions *with origin information* realizable by two-way transducers were shown to have decidable first-order definability [4]. In this paper, we extend the results of [21] and the decidability of first-order definability of [13] to rational functions of infinite words.

Rational functions of infinite words We consider rational functions of infinite words, *i.e.* functions defined by transducers with Muller acceptance condition. Such machines map any ω-word for which there exists an accepting run to either a finite or an ω-word. Take as example the function $f_{\#a}$ over alphabet $\{a, b\}$ mapping any word containing an infinite number of a to a^ω, and to b^ω otherwise. This function is realized by the transducer of Fig. 1.

The class of *sequential functions* is of particular interest: they are realized by transducers whose underlying input Muller automaton is deterministic. Note that the function $f_{\#a}$ is not sequential, unlike the function f_{ab} of Fig. 2. Sequential functions over infinite words have been studied *e.g.* in [2]. One difference between our setting and [2] is that in the latter paper infinite words are mapped to infinite words, whereas we

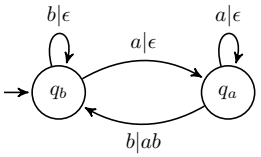

Figure 2 Sequential transducer with Muller condition $F = \{\{q_b\}\}$ realizing the function f_{ab} which maps any word containing a finite number of a's to the subsequence of ab factors, and is undefined otherwise.

need also functions that map infinite words to finite words. Deciding whether a rational function is sequential can be done in PTime, as shown in [2]. Bimachines for infinite words were introduced in [26] to define the particular class of total letter-to-letter rational functions, and in their counter-free versions, a connection with linear temporal logic was established.

To the best of our knowledge, nothing is known about canonical models for sequential and rational functions over infinite words, and their applications to definability problems in logics.

Contributions (1) We provide a characterization of sequential functions by means of the finiteness of a congruence. We give a PTime procedure which, for any sequential function f given as a transducer whose domain is topologically closed, produces *the* minimal (and hence canonical) sequential transducer $\mathcal{T}_f$ realizing f. When the domain of f is not topologically closed, we extend f to a domain-closed sequential function $\overline{f}$ which coincides with f on its domain. By intersecting $\mathcal{T}_{\overline{f}}$ with some automaton $\mathcal{D}$ recognizing the domain of f, one obtains a canonical transducer for f, as long as $\mathcal{D}$ can be obtained in a canonical way (such a procedure exists, see *e.g.* [7]).

(2) Our main contribution (Theorem 29) is a notion of canonical sequential transducer with look-ahead for any rational function. This canonical transducer is an effectively computable bimachine. Hence we lift results of Reutenauer and Schützenberger [21] on rational transductions from finite to infinite words.

(3) As a side result we lift a result by Elgot and Mezei [11] from finite to infinite words, stating that a function f is rational if and only if $f = g_1 \circ h_1$ (resp. $f = g_2 \circ h_2$) such that h_1, h_2 are letter-to-letter, g_1, h_2 are sequential and h_1, g_2 are right-sequential (*i.e.*, realized by a transducer whose underlying input automaton is prophetic [6]). The existence of such g_1, h_1 was already shown in [5], but the one of g_2, h_2 was left open.

(4) Finally, we show that our procedure which computes a canonical bimachine for any rational function given by a transducer, preserves aperiodicity. As an application, after showing some correspondences between logics and transducers, we obtain the decidability of FO-transductions in MSO-transductions over infinite words.

Overview of the canonization procedure for rational functions The main idea to get a canonical object for a rational function, inspired by [21], is to add a canonical look-ahead information to the input word, so that the function can be evaluated in a sequential (equivalently, deterministic) manner. We say that the look-ahead "makes the function sequential". By doing so, we can reduce the problem to computing canonical machines for sequential functions. The main difficulty is to define a canonical (and computable) notion of look-ahead which makes the function sequential. Over finite words, the look-ahead information is computed by a co-deterministic automaton, or equivalently, a deterministic automaton reading the input word from right to left (called a right automaton). On infinite words we need something different, so we use prophetic automata [6] to define look-aheads (called right automata in this paper). Prophetic automata are a special form of co-deterministic automata over infinite words. In Section 3, sequential transducers with look-ahead are formalized via the notion of bimachines, consisting of a left automaton and a right automaton. We show that bimachines over infinite words capture exactly the class of rational transductions. Our goal is to obtain a canonical bimachine, fine enough to realize the function, but coarse enough to preserve algebraic properties like aperiodicity.

Unlike the setting of finite words, some difficulties arise when prefix-independent properties matter (such as for instance that a suffix contains an infinite number of a's). We overcome

this issue by defining two kinds of look-ahead information which we combine later on. This decomposition simplifies the overall proof.

The first look-ahead information we define allows one to make any rational function almost sequential, in the sense that it can be implemented by a transducer model which can additionally output some infinite word after processing the whole input, depending on the run (similar to so-called *subsequential* transducers in the case of finite words). We call *quasi-sequential* functions realized by such transducers. They constitute a class with interesting properties. We show that they correspond precisely to transducers satisfying the weak twinning property, a syntactic condition defined in [2]. On the algebraic side, we exhibit a congruence having finite index exactly for quasi-sequential functions.

We then define another kind of canonical look-ahead which makes any quasi-sequential function sequential. Combined together, these two look-aheads turn any rational function into a sequential one: the first one from rational to quasi-sequential, and the second one from quasi-sequential to sequential.

The whole procedure does not yield a minimal bimachine in general. While the minimality question is an important and interesting (open) question, our procedure still has the strong advantages of being canonical, effective, and of preserving aperiodicity. This allows one to answer positively the important question of the decidability of first-order definability for rational functions of infinite words. Detailed proofs are provided in Appendix.

1 Regular languages and rational functions

Finite words, infinite words and languages An *alphabet* A is a finite set of symbols called *letters*. A *finite word* is a finite sequence of letters, the empty sequence is called the *empty word* and is denoted by ϵ. The set of (resp. non-empty) finite words over A is denoted by A^* (resp. A^+). An infinite sequence of letters is called an ω-*word* (or just an infinite word), we denote by A^ω the set of ω-words and we write $A^\infty = A^* \cup A^\omega$. For a word $x \in A^\omega$ we denote by $\mathrm{Inf}(x)$ the set of letters of x which appear an infinite number of times. The *length* of a word w is written $|w|$, with $|w| = \infty$ if $w \in A^\omega$. Throughout the paper, we often denote finite words by $u, v, \ldots$ and infinite words by $x, y \ldots$

For a non-empty word w and two integers $1 \leqslant i \leqslant j \leqslant |w|$ we denote by $w(i)$ the ith letter of w, by $w(i{:})$ the suffix of w starting at the ith position, by $w({:}i)$ the prefix of w ending at the ith position and by $w(i{:}j)$ the infix of w starting at the ith position and ending at the jth, both included. For two words $u \in A^*$ and $v \in A^\infty$, we write $u \prec v$ if u is a strict prefix of v, *i.e.* there exists a non-empty word $w \in A^\infty$ such that $uw = v$, and we write $u^{-1}v$ for w. For $u, v \in A^\infty$, we write $u \preceq v$ if either $u \prec v$, or $u = v$. We denote by $u \wedge v$ the longest common prefix of u and v. The *delay* $\mathsf{del}(u, v)$ between two words $u, v \in A^\infty$ is the unique pair (u', v') such that $u = (u \wedge v)u'$ and $v = (u \wedge v)v'$. For example, $\mathsf{del}(aab, ab) = (ab, b)$ and $\mathsf{del}(a^\omega, a^\omega) = (\epsilon, \epsilon)$.

A *language* is a set of words $L \subseteq A^\infty$, and by $\bigwedge L$ we denote the longest common prefix of all words in L (if $L \neq \varnothing$). The closure $\overline{L}$ of L is $\{u \in A^\infty \mid \forall i \in \mathbb{N},\ i \leqslant |u|,\ \exists w \text{ s.t. } u(:i)w \in L\}$, *i.e.* the set of words for which any finite prefix has a continuation in L. For instance $\overline{a^*b^\omega} = a^*b^\omega \cup a^\omega$. A word is called *regular* if it is of the form uv^ω with $u, v \in A^*$. In particular any finite word is regular (since $\epsilon^\omega = \epsilon$) and regular ω-words are also called *ultimately periodic*. We say that a regular word uv^ω is in *normal form* if v has minimal length and is minimal in the lexicographic order among all possible decompositions of uv^ω, and v is not a suffix of u (if $v \neq \epsilon$). *E.g.* the normal form of $(ba)^\omega$ is $b(ab)^\omega$. In the sequel we often assume regular words are in normal form.

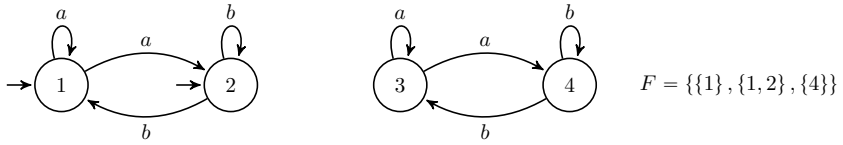

Figure 3 A right automaton (with Muller condition) recognizing $(b^*a)^\omega$. Words with finitely many b's have final run with $\{1\}$, words with finitely many a's have final run with $\{4\}$, and those with infinitely many a's and infinitely many b's have final run with $\{1,2\}$.

Automata A Muller[1] automaton over an alphabet A is a tuple $\mathcal{A} = (Q, \Delta, I, F)$ where Q is a finite set of *states*, $\Delta \subseteq Q \times A \times Q$ is the set of *transitions*, $I \subseteq Q$ is the set of *initial* states, and $F \subseteq \mathcal{P}(Q)$ is called the *final condition*. When there is no final condition, so $F = \mathcal{P}(Q)$, we will omit it. A *run* of $\mathcal{A}$ over a word $w \in A^\infty$ is itself a word $r \in Q^\infty$ of length $|w| + 1$, (with the convention that $\infty + 1 = \infty$) such that for any $1 \leqslant i < |r|$, we have $(r(i), w(i), r(i+1)) \in \Delta$. A run r is called *initial* if $r(1) \in I$, *final* if $r \in Q^\omega$ and $\mathrm{Inf}(r) \in F$, and *accepting* if it is both initial and final. For a finite word u and two states p, q, we write $p \xrightarrow{u}_{\mathcal{A}} q$ to denote that there is a run r of $\mathcal{A}$ over u such that $r(1) = p$ and $r(|r|) = q$. For an ω-word x, a state p and a subset of states $P \subseteq Q$, we write $p \xrightarrow{x}_{\mathcal{A}} P$ to denote that there is a run r of $\mathcal{A}$ over x such that $r(1) = p$ and $\mathrm{Inf}(r) = P$. A word is *accepted* by $\mathcal{A}$ if there exists an accepting run over it, and the language *recognized* by $\mathcal{A}$ is the set of words it accepts, denoted by $[\![\mathcal{A}]\!] \subseteq A^\omega$. A state p is *accessible* (resp. *co-accessible*) if there exists a finite initial (resp. infinite final) run r such that $r(|r|) = p$ (resp. $r(1) = p$), and an automaton $\mathcal{A}$ is called *accessible* (resp. *co-accessible*) if all its states are. An automaton which is both accessible and co-accessible is called *trim*. An automaton is called *deterministic* if its set of initial states is a singleton, and any word has at most one initial run. We define a *left automaton* as a deterministic automaton $\mathcal{L} = (Q, \Delta, I)$ with no acceptance condition. We call a *right automaton* an automaton for which any ω-word has exactly one final run[2]. A language is called *ω-regular* if it is recognized by an automaton. It is well-known that every ω-regular language can be recognized by a deterministic (Muller) automaton. Moreover, [6] shows that every ω-regular language can be recognized by a right automaton (even with Büchi condition). Figure 3 shows a right automaton accepting the words with infinitely many a's. Throughout the paper, all automata – except for right automata – are assumed trim, without loss of generality.

Transductions Given two alphabets A, B, we call *transduction* a relation $R \subseteq A^\omega \times B^\infty$ whose domain is denoted by $\mathrm{dom}(R)$. A transducer over A, B is a tuple $\mathcal{T} = (\mathcal{A}, i, o)$ with $\mathcal{A} = (Q, \Delta, I, F)$ the *underlying automaton*, $i : I \to B^*$ the *initial function* and $o : \Delta \to B^*$ the *output function*. Let u be a finite word of length n, let r be a run of $\mathcal{A}$ over u with $r(1) = p, r(n+1) = q$, and let v be the word $o(p, u(1), r(2)) \cdots o(r(n), u(n), q)$ then we write $p \xrightarrow{u|v}_{\mathcal{T}} q$ to denote that fact. Similarly, for $p \in Q$ and $P \subseteq Q$ we write $p \xrightarrow{x|v}_{\mathcal{T}} P$ to denote that there is a run r of $\mathcal{A}$ over the ω-word x such that $r(1) = p$, $\mathrm{Inf}(r) = P$ and $v = o(p, u(1), r(2))o(r(2), u(2), r(3)) \cdots$. In that case, if $p \in I$ and $P \in F$, let $w = i(p) \cdot v$, then we say that the pair (x, w) is *realized* by $\mathcal{T}$. We denote by $[\![\mathcal{T}]\!]$ the set of pairs realized by $\mathcal{T}$, which we call the *transduction realized* by $\mathcal{T}$. A transducer is called *functional* if it realizes a (partial) function, and in that case we write $[\![\mathcal{T}]\!](x) = w$ rather than $(x, w) \in [\![\mathcal{T}]\!]$.

[1] We consider the Muller condition since it is more general than Büchi or parity for instance, but most of our results hold for other conditions as well.
[2] Such automata are called prophetic and were introduced in [6].

Functionality is a decidable property, see *e.g.* [16], and it can be checked in PTIME (see *e.g.* [20]). *In the following all the transductions we consider are functional, and when we speak about functions, we tacitly assume that they are partial.* A transduction is *rational* if it is realized by a transducer. A transducer with a deterministic underlying automaton is called *sequential*, as well as the function it realizes. A transducer with a left (resp. right) underlying automaton is called *left-sequential* (resp. *right-sequential*), and again we extend this terminology to the function it realizes.

Congruences Given an equivalence relation $\sim$ over a set L, we denote by $[w]^{\sim}$ (or simply $[w]$) the equivalence class of an element $w \in L$. We say that $\sim$ has *finite index* if the set $L/_{\sim} = \{[w] \mid w \in L\}$ is finite. Given two equivalence relations $\sim_1, \sim_2$ over the same set we say that $\sim_1$ is *finer* than $\sim_2$ (or that $\sim_2$ is *coarser* than $\sim_1$) if for any u, v we have $u \sim_1 v \Rightarrow u \sim_2 v$. Equivalently we could say that the equivalence classes of $\sim_2$ are unions of equivalence classes of $\sim_1$ or that $\sim_1$ is included (as a set of pairs) in $\sim_2$, which we denote by $\sim_1 \sqsubseteq \sim_2$. A *right congruence* over A^* is an equivalence relation $\sim$ such that for any letter a and any words u, v we have $u \sim v \Rightarrow ua \sim va$. A *left congruence* over A^* (resp. A^ω) is an equivalence relation $\approx$ such that for any letter a and any words u, v we have $u \approx v \Rightarrow au \approx av$. We say that a left congruence is *regular* if it has finite index and any equivalence class is an ω-regular language. In the following all the left congruences will be regular. A *congruence* over A^* is a left and right congruence. A congruence $\equiv$ is *aperiodic* if there exists an integer n such that $\forall u \in A^*$, $u^n \equiv u^{n+1}$.

Given an automaton $\mathcal{A}$ with state space Q, the *right congruence associated with* $\mathcal{A}$ is defined for $u, v \in A^*$ by $u \sim_{\mathcal{A}} v$ if $\forall q \in Q$, there is an initial run of $\mathcal{A}$ over u reaching q if and only if there is one over v. Note that for a trim deterministic automaton, there is a bijection (up to adding a sink state) between Q and the equivalence classes of $\mathcal{A}$. Similarly, the *left congruence associated with* $\mathcal{A}$ is defined for $x, y \in A^\omega$ by $x \approx_{\mathcal{A}} y$ if $\forall q \in Q$ there is a final run of $\mathcal{A}$ over x from q if and only if there is one over y. Given a right automaton there is a bijection between Q and the equivalence classes of $\approx_{\mathcal{A}}$. Finally, the *transition congruence of* $\mathcal{A}$ is defined for $u, v \in A^*$ by $u \equiv_{\mathcal{A}} v$ if $\forall p, q \in Q$, there is a run over u from p to q if and only if there is one over v. An automaton is called *aperiodic* if its transition congruence is aperiodic. A language is called *aperiodic* if there exists an aperiodic automaton recognizing it. A transducer is *aperiodic* if its underlying automaton is aperiodic and in that case the transduction it realizes is called *aperiodic*.

Given a right congruence $\sim$, the *left automaton associated with* $\sim$ is $\mathcal{A}_{\sim} = (Q_{\sim}, \Delta_{\sim}, I_{\sim})$: $Q_{\sim} = A^*/_{\sim}$, $\Delta_{\sim} = \{([u], a, [ua]) \mid u \in A^*\}$, $I_{\sim} = \{[\epsilon]\}$. Given a left congruence $\approx$ and a right automaton $\mathcal{R}$, if $\approx_{\mathcal{R}} \sqsubseteq \approx$ then we say that $\mathcal{R}$ *recognizes* $\approx$. The existence of a canonical automaton for a left congruence is less obvious. From [6] we know that every ω-regular language can be recognized by a right automaton. We rely on the construction of [6] and, abusing language, we denote the right automaton obtained in the next proposition as the *canonical right automaton recognizing a left congruence*:

▸ **Proposition 1.** Given a (regular) left congruence, we can compute in 2-EXPTIME a right automaton recognizing it. Furthermore, this automaton is aperiodic if the congruence is aperiodic.

2 Sequential and quasi-sequential transductions

We define the syntactic congruence associated with any functional transduction over infinite words. Sequential functions are exactly the rational functions having a syntactic congruence

	f_{ab}	$f_{\#a}$	f_{blocks}
definition	maps a word over $\{a,b\}$ with a finite number of a's to the subsequence of ab-factors.	maps a word x over $\{a,b\}$ to a^ω if x contains an infinite number of a's, and to b^ω otherwise.	maps $u_1\#\dots\#u_n\#v$ where v does not contain $\#$, to $a_1^{\lvert u_1\rvert}\#\dots\#a_n^{\lvert u_n\rvert}\#w$ where $u_i \in \{a,b\}^*$, a_i is the last letter of u_i (if any), $w = a^\omega$ if v has an infinite number of a's, and $w = b^\omega$ otherwise.
A and B	$A = B = \{a,b\}$	$A = B = \{a,b\}$	$A = B = \{a,b,\#\}$
$\text{dom}(f)$	words over $\{a,b\}$ with a finite number of a's	$\{a,b\}^\omega$	words over $\{a,b,\#\}$ with a finite (non-zero) number of $\#$'s
examples	$f_{ab}(abbab^\omega) = abab$, $f_{ab}(b^\omega) = \epsilon$	$f_{\#a}(ab^\omega) = b^\omega$, $f_{\#a}((ab)^\omega) = a^\omega$	$f_{\text{blocks}}((ab\#)^n b^\omega) = (bb\#)^n b^\omega$, $f_{\text{blocks}}(\#(ab)^\omega) = a^\omega$.
$\widehat{f}$	$\widehat{f_{ab}}$ extracts the ab-factors, for instance $\widehat{f_{ab}}(abbabb) = abab$.	reading a finite prefix u does not give any insight on the output, thus $\widehat{f_{\#a}}(u) = \epsilon$	$\widehat{f_{\text{blocks}}}(u_1\#\dots\#u_n\#v) = a_1^{\lvert u_1\rvert}\#\dots\#a_n^{\lvert u_n\rvert}\#$ whenever v does not contain $\#$.
$\overline{f}$	$\overline{f_{ab}}$ is defined over $\text{dom}(\overline{f_{ab}}) = \{a,b\}^\omega$ and $\overline{f_{ab}}((ba)^\omega) = \lim_n \widehat{f_{ab}}((ba)^n) = \lim_n (ab)^{n-1} = (ab)^\omega$.	$\overline{f_{\#a}}(x) = \epsilon$ for every $x \in \{a,b\}^\omega$ as it is based on $\widehat{f_{\#a}}$	$\overline{f_{\text{blocks}}}(u_1\#\dots\#u_n\#v) = a_1^{\lvert u_1\rvert}\#\dots\#a_n^{\lvert u_n\rvert}\#$ whenever v does not contain $\#$.
class	sequential	quasi-sequential	not quasi-sequential

Figure 4 Examples of rational transductions, and their associated $\widehat{f}$ and $\overline{f}$ functions.

of finite index, and being continuous over their domain. When removing this last condition on continuity, we obtain the class of quasi-sequential transductions. These transductions are also characterized by the weak twinning property [2].

We will show that for any sequential function, like in the case of finite words [8], we can define a canonical transducer, with a minimal underlying automaton. This minimal transducer extends the domain of the function to its closure.

▸ **Definition 2** ($\widehat{f}$ and $\overline{f}$)**.** Let $f : A^\omega \to B^\infty$ be a function, we define $\widehat{f} : A^* \to B^\infty$ by $\widehat{f}(u) = \bigwedge \{f(ux) \mid ux \in \text{dom}(f)\}$. In other words, $\widehat{f}$ outputs the longest possible output that f could produce on any word that begins with u. We also define $\overline{f} : A^\omega \to B^\infty$ by setting $\overline{f}(x) = \lim_n \widehat{f}(x(:n))$, for $x \in \overline{\text{dom}(f)}$.

We refer to $\overline{f}$ as the *sequential extension* of f. Note that if f is sequential, then $\overline{f}$ extends f over the closure $\overline{\text{dom}(f)}$ of the domain of f.

▸ **Example 3.** We illustrate these definitions on three rational transductions, described in Figure 4.

▸ **Definition 4** (syntactic congruence $\sim_f$)**.** The *syntactic congruence* associated with a transduction f is defined over A^* by $u \sim_f v$ if:

1. $\forall x \in A^\omega$, $ux \in \overline{\text{dom}(f)} \Leftrightarrow vx \in \overline{\text{dom}(f)}$, and
2. either $\widehat{f}(u)$ and $\widehat{f}(v)$ are both ultimately periodic with the same period (in normal form) or they are both finite and $\forall x \in A^\omega$ such that $ux, vx \in \text{dom}(f)$, $\widehat{f}(u)^{-1} f(ux) = \widehat{f}(v)^{-1} f(vx)$.

▸ **Example 5.** Let us illustrate the definition of $\sim_f$ on f_{ab}, as defined in Figure 4. The syntactic congruence $\sim_{f_{ab}}$ has only two classes: $[\epsilon]$ and $[a]$. Indeed, if we consider two

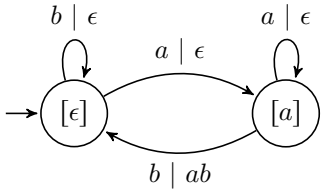

Figure 5 Transducer $\mathcal{T}_{f_{ab}}$.

finite words u and v, condition (1) on the domain is always true, and $\widehat{f_{ab}}(u)$ and $\widehat{f_{ab}}(v)$ are finite (ab-factors in u and v, respectively). Hence $u \sim_{f_{ab}} v$ if and only if $\forall x \in A^\omega$, $\widehat{f_{ab}}(u)^{-1} f_{ab}(ux) = \widehat{f_{ab}}(v)^{-1} f_{ab}(vx)$.

Let us analyze $\widehat{f_{ab}}(u)^{-1} f_{ab}(ux)$. If u does not end with an a, then $\widehat{f_{ab}}(u)^{-1} f_{ab}(ux) = ((ab)^n)^{-1}((ab)^{n+k}) = (ab)^k$ where n and k are the number of ab-factors in u and x, respectively. Now, if u ends with an a and x starts with a b, then a new ab-factor appears in ux and we get $\widehat{f_{ab}}(u)^{-1} f_{ab}(ux) = ((ab)^n)^{-1}((ab)^{n+k+1}) = (ab)^{k+1}$. This means that $\sim_{f_{ab}}$ contains exactly two classes: one for the words ending with an a, and one for the others.

The resulting transducer $\mathcal{T}_{f_{ab}}$ is depicted in Figure 5. Let us check for instance the transition from $[a]$ to $[\epsilon]$ when reading b. We have $[ab] = [\epsilon]$, so $([a], b, [\epsilon]) \in \Delta_{f_{ab}}$. From the definition, $o_{f_{ab}}([a], b, [\epsilon]) = \widehat{f_{ab}}(a)^{-1}\widehat{f_{ab}}(ab) = \epsilon^{-1}.ab = ab$.

▸ **Proposition 6.** Let f be a functional transduction, then $\sim_f$ is a right congruence.

From $\sim_f$ we define[3] the transducer $\mathcal{T}_f = (\mathcal{A}_f, i_f, o_f)$ with $\mathcal{A}_f = (Q_f, \Delta_f, I_f)$ and:

- $Q_f = A^*/_{\sim_f}$ and $I_f = \{[\epsilon]\}$
- $\Delta_f = \{([u], a, [ua]) \mid u \in A^*, a \in A, \exists x \text{ s.t. } uax \in \mathrm{dom}(f)\}$
- $o_f([u], a, [ua]) = \begin{cases} \widehat{f}(u)^{-1}\widehat{f}(ua) & \text{if } \widehat{f}(ua) \text{ is finite} \\ \beta & \text{if } \widehat{f}(u) = \alpha\beta^\omega,\ \beta \neq \epsilon \\ \alpha & \text{if } \widehat{f}(u) \text{ is finite and } \widehat{f}(u)^{-1}\widehat{f}(ua) = \alpha\beta^\omega,\ \beta \neq \epsilon \end{cases}$
- $i_f([\epsilon]) = \begin{cases} \widehat{f}(\epsilon) & \text{if } \widehat{f}(\epsilon) \text{ is finite} \\ \alpha & \text{if } \widehat{f}(\epsilon) = \alpha\beta^\omega,\ \beta \neq \epsilon \end{cases}$

▸ Remark. Note that, in general, $\sim_f$ may have an infinite index, thus $\mathcal{T}_f$ may be infinite. This is the case for f_{blocks}: for two words $u = u_0\#w$ and $v = u_0\#w'$ with u_0ww' not containing $\#$, $u \sim_{f_{\text{blocks}}} v$ if and only if $|w| = |w'|$ and they end with the same letter. We will define below a subclass of rational transductions, which captures exactly finite $\sim_f$ (Theorem 12).

As shown below, the sequential transducer $\mathcal{T}_f$ computes the sequential extension $\overline{f}$ of f. If f is sequential then f and $\overline{f}$ coincide on $\mathrm{dom}(f)$(see Proposition 32 in the appendix).

▸ **Proposition 7.** Given a function f, the transducer $\mathcal{T}_f$ realizes $\overline{f}$.

We now focus on sequential transductions, and show first that $\mathcal{T}_f$ can be built in PTime.

▸ **Proposition 8.** There is a PTime algorithm that, for a given sequential transducer $\mathcal{T}$ realizing the function f, computes the transducer $\mathcal{T}_f$.

For sequential transductions we get a characterization, as stated in the next theorem. We will see that the first condition is equivalent to the weak twinning property. Thus, the next theorem adapts a result from [2] to the case where transducers may output finite words.

[3] We check in Appendix B that $\mathcal{T}_f$ is well-defined.

▸ **Theorem 9.** *A rational function f is sequential if and only if the following conditions hold:*
- $\sim_f$ *has finite index*
- $\overline{f}_{|dom(f)} = f$

If we remove the last restriction $\overline{f}_{|\mathrm{dom}(f)} = f$ in Theorem 9, we obtain a class of transductions where the output can be still generated deterministically (as for sequential transductions), although not necessarily in a progressive manner:

▸ **Definition 10.** A function f is called *quasi-sequential* if it is rational and $\sim_f$ has finite index.

Intuitively, quasi-sequential functions generalize the so-called *subsequential* functions on finite words to infinite words. For subsequential functions there is a final output associated with final states. Quasi-sequential functions can be shown to correspond to sequential transducers where final sets may have an associated word in A^∞. The output of an accepting run with such a final set is obtained by appending the associated word to the output word obtained through the transitions (if finite). Since we do not use this model in the present paper, we do not provide more details in the following. The following property and construction are now taken directly from [2]. As in the latter article, a state is called *constant* if the set of words produced by final runs from this state is a singleton.

▸ **Definition 11** (weak twinning property)**.** A transducer $\mathcal{T}$ is said to satisfy the *weak twinning property* (WTP) if for any initial runs $p_1 \xrightarrow{u|\alpha_1} q_1 \xrightarrow{v|\beta_1} q_1$ and $p_2 \xrightarrow{u|\alpha_2} q_2 \xrightarrow{v|\beta_2} q_2$ the following property holds:
- If q_1, q_2 are not constant then $\mathsf{del}(i(p_1)\alpha_1, i(p_2)\alpha_2) = \mathsf{del}(i(p_1)\alpha_1\beta_1, i(p_2)\alpha_2\beta_2)$
- If q_1 is not constant, q_2 is constant and produces the regular word γ, then either $\beta_1 = \epsilon$ or $i(p_1)\alpha_1\beta_1^\omega = i(p_2)\alpha_2\beta_2\gamma$

Note that if q_2 is constant and $\beta_2 \neq \epsilon$ then $\gamma = \beta_2^\omega$.

The authors of [2] provide a determinization procedure – which we call *subset construction with delays*– which terminates if and only if the transducer satisfies the WTP. We show that actually the procedure gives a transducer realizing the sequential extension of the function and we use this fact in Sec. 4 in order to compute a canonical look-ahead.

▸ **Theorem 12.** *Let $\mathcal{T}$ be a transducer realizing a function f, let $\mathcal{S}$ be the transducer obtained by subset construction with delays. The following statements are equivalent:*
1. *The transducer $\mathcal{T}$ satisfies the WTP*
2. *The transducer $\mathcal{S}$ is finite*
3. *f is quasi-sequential*

Furthermore, if $\mathcal{T}$ is aperiodic then $\mathcal{S}$ is aperiodic as well.

3 Rational transductions

Bimachines over infinite words A *bimachine* over alphabets A, B is a tuple $\mathcal{B} = (\mathcal{L}, \mathcal{R}, i, o)$ where $\mathcal{L} = (Q_\mathcal{L}, \Delta_\mathcal{L}, \{l_0\})$ is a left automaton, $\mathcal{R} = (Q_\mathcal{R}, \Delta_\mathcal{R}, I, F)$ is a right automaton, $i : I \to B^*$ is the *initial function* and $o : Q_\mathcal{L} \times A \times Q_\mathcal{R} \to B^*$ is the *output function*. We have a semantic restriction that $[\![\mathcal{L}]\!] = \overline{[\![\mathcal{R}]\!]}$. The output produced on an infinite word $w \in [\![\mathcal{R}]\!]$ at position $i \geqslant 1$ is $\alpha_i = o(l, a, r)$, where l is the state reached in $\mathcal{L}$ after reading the prefix $w(:i-1)$ of w up to position $i-1$ (if defined), r is the state of the unique final run of $\mathcal{R}$ on w (if defined) reached by the suffix $w(i+1:)$ of w from position $i+1$ on, and $a = w(i)$. In other words, the output at position i is determined by the left context up to position

$i-1$, the right context from position $i+1$ onwards, and the letter at position i. The output produced on w is $i(r_0)\alpha_1\alpha_2\cdots$, with $r_0 \in I$ the state from which there is a final run of $\mathcal{R}$ on w (if defined). Thus, the right automaton $\mathcal{R}$ provides a look-ahead and the output depends both on the state of $\mathcal{L}$ and the unique final run of $\mathcal{R}$ on the given word. The transduction *realized* by $\mathcal{B}$ is denoted by $[\![\mathcal{B}]\!]$. Note that $[\![\mathcal{B}]\!]$ is defined over $[\![\mathcal{R}]\!]$. A bimachine is called *aperiodic* if both its automata are aperiodic.

▸ **Example 13.** Let us define a bimachine for f_{ab}, the function that outputs ab-factors of the input over $\{a,b\}$, if this input has a finite number of a's. We use as left automaton the underlying automaton of the transducer in Figure 2, without its Muller acceptance condition. This automaton will only be used to store the last letter read. The domain has to be checked by the right automaton, and we choose the one in Figure 3. As output functions, we let $i(q) = \epsilon$ for the initial states of the right automaton, and let $o(q_a, b, r) = ab$ for $r \in \{1,2\}$, and $o(l,c,r) = \epsilon$ for all other states l, r of the left and right automata, and letter $c \in \{a,b\}$.

Left minimization We show how to minimize the left automaton of a bimachine with respect to a right automaton $\mathcal{R}$. The procedure is very similar to the minimization for sequential transducers. The objects we use are the same as in Section 2, but relativized to the right context defined by the look-ahead provided by the right automaton $\mathcal{R}$. The bimachine with minimal left automaton with respect to the right automaton $\mathcal{R}$ is the bimachine $\mathcal{B}_f^{\mathcal{R}}$ defined below.

Recall that the left congruence $\approx_{\mathcal{R}}$ of a right automaton $\mathcal{R}$ sets $x \approx_{\mathcal{R}} y$ if the unique state from which there is a final run on x is the same as for y. Let $f : A^\omega \to B^\infty$ be a function and let $\mathcal{R} = (Q_{\mathcal{R}}, \Delta_{\mathcal{R}}, I, F)$ be a right automaton recognizing $\mathrm{dom}(f)$. We write $[x]^{\mathcal{R}}$ for the class of a word x with respect to $\approx_{\mathcal{R}}$, and, abusing notations, for the state of $Q_{\mathcal{R}}$ from which words of $[x]^{\mathcal{R}}$ have a final run. We define $\widehat{f}_x : A^* \to B^\infty$ by setting $\widehat{f}_x(u) = \bigwedge \{f(uy) \mid y \approx_{\mathcal{R}} x\}$. Note that there are finitely many functions $\widehat{f}_x$, one for each equivalence class of $\approx_{\mathcal{R}}$. We also define $\overline{f}^{\mathcal{R}} : A^\omega \to B^\infty$, by setting $\overline{f}^{\mathcal{R}}(x) = \lim_n \widehat{f}_{x(n+1:)}(x(:n))$. The transduction $\overline{f}^{\mathcal{R}}$ is defined over $\mathrm{dom}(f)$.

▸ **Definition 14** ($\mathcal{R}$-syntactic congruence)**.** The *$\mathcal{R}$-syntactic congruence of f* is defined over A^* by letting $u \sim_f^{\mathcal{R}} v$ if:

1. $\forall x \in A^\omega$, $ux \in \mathrm{dom}(f) \Leftrightarrow vx \in \mathrm{dom}(f)$, and
2. for any $x \in A^\omega$, either $\widehat{f}_x(u)$ and $\widehat{f}_x(v)$ are both infinite with the same ultimate period (in normal form) or they are both finite and $\widehat{f}_x(u)^{-1} f(ux) = \widehat{f}_x(v)^{-1} f(vx)$.

Similarly to the sequential case, we define from $\sim_f^{\mathcal{R}}$ a bimachine $\mathcal{B}_f^{\mathcal{R}} = \left(\mathcal{L}_f^{\mathcal{R}}, \mathcal{R}, i_f^{\mathcal{R}}, o_f^{\mathcal{R}}\right)$ with right automaton $\mathcal{R}$, and left automaton $\mathcal{L}_f^{\mathcal{R}} = \left(Q_f^{\mathcal{R}}, \Delta_f^{\mathcal{R}}, I_f^{\mathcal{R}}\right)$ corresponding to $\sim_f^{\mathcal{R}}$. To simplify notations we denote the congruence class of a word u with respect to $\sim_f^{\mathcal{R}}$ by $[u]$. Abusing notations we also write $[x]^{\mathcal{R}}$ for the state of $\mathcal{R}$ from which x has an accepting run.

- $Q_f^{\mathcal{R}} = A^*/_{\sim_f^{\mathcal{R}}}$ and $I_f^{\mathcal{R}} = \{[\epsilon]\}$
- $\Delta_f^{\mathcal{R}} = \{([u], a, [ua]) \mid u \in A^*, a \in A, uax \in \mathrm{dom}(f) \text{ for some } x \in A^\omega\}$
- $o_f([u], a, [x]^{\mathcal{R}}) = \begin{cases} \widehat{f}_{ax}(u)^{-1}\widehat{f}_x(ua) & \text{if } \widehat{f}_x(ua) \text{ is finite} \\ \beta & \text{if } \widehat{f}_{ax}(u) = \alpha\beta^\omega,\ \beta \neq \epsilon \\ \alpha & \text{if } \widehat{f}_{ax}(u) \text{ is finite, } \widehat{f}_{ax}(u)^{-1}\widehat{f}(ua) = \alpha\beta^\omega \\ & \quad \text{and } \beta \neq \epsilon \end{cases}$
- $i_f([x]^{\mathcal{R}}) = \begin{cases} \widehat{f}_x(\epsilon) & \text{if } \widehat{f}_x(\epsilon) \text{ is finite} \\ \alpha & \text{if } \widehat{f}_x(\epsilon) = \alpha\beta^\omega, \beta \neq \epsilon \end{cases}$

We show in appendix that $\mathcal{B}_f^{\mathcal{R}}$ is well-defined, and exhibit some of its properties. We also describe in appendix a polynomial time algorithm that computes $\mathcal{B}_f^{\mathcal{R}}$ from a bimachine with right automaton $\mathcal{R}$, with a technique similar to the sequential case(Proposition 6).

From transducers to bimachines For the theorem below, recall that $\sim_{\mathcal{A}}$ denotes the right congruence of an automaton $\mathcal{A}$. The left congruence $\approx_{\mathcal{A}}$ of an automaton $\mathcal{A}$ sets $x \approx_{\mathcal{A}} y$ if for every state q of $\mathcal{A}$, there is some final run on x from q if and only if there is one on y.

▸ **Theorem 15.** *Given a transducer with underlying automaton $\mathcal{A}$ and a right automaton $\mathcal{R}$ with $\approx_{\mathcal{R}} \sqsubseteq \approx_{\mathcal{A}}$. Then $\sim_{\mathcal{A}} \sqsubseteq \sim_f^{\mathcal{R}}$ and the bimachine $\mathcal{B}_f^{\mathcal{R}}$ realizes f.*

In particular any aperiodic transduction can be realized by an aperiodic bimachine.

The other direction also holds: from a bimachine we can build an equivalent (unambiguous) transducer, by taking the product of the left and right automata of the bimachine. The construction is not hard but given in the appendix. By Theorem 15 and Proposition 1 we obtain:

▸ **Theorem 16.** *A function is rational (resp. rational and aperiodic) if and only if it can be realized by a bimachine (resp. aperiodic bimachine).*

Labelings and bimachines We define the *labeling function* associated with a right automaton $\mathcal{R} = (Q, \Delta, I, F)$ by the right transducer $\ell(\mathcal{R}) = (\mathcal{R}, i, o)$, with $i(q) = \epsilon$ and $o(p, a, q) = (a, q)$. Intuitively, the labeling function labels each position with the look-ahead information about the suffix provided by $\mathcal{R}$. For a transduction f we define $f_{\mathcal{R}} = f \circ [\![\ell(\mathcal{R})]\!]^{-1}$. Note that $f_{\mathcal{R}}$ is a function, since the labeling is injective (because $\mathcal{R}$ is unambiguous). Thus, $f_{\mathcal{R}}$ corresponds to f defined over words enriched by the look-ahead information of $\mathcal{R}$.

▸ **Proposition 17.** Let f be a transduction and let $\mathcal{R}$ be a right automaton. There exists a bimachine $\mathcal{B}$ realizing f with $\mathcal{R}$ as a right automaton if and only if $f_{\mathcal{R}}$ is left-sequential. Furthermore, assuming that $\mathcal{R}$ is aperiodic, then $\sim_f^{\mathcal{R}}$ is aperiodic if and only if $f_{\mathcal{R}}$ is aperiodic.

We say that a transducer is *letter-to-letter* if its initial output function always outputs the empty word and its output function always outputs a single letter. The following corollary states the classical result of [11] but over infinite words, and generalizes a result of [5].

▸ **Corollary 18.** *For any rational function f, there exists a left-sequential (right-seq. resp.) function g and a letter-to-letter right-sequential (left-seq. resp.) function h such that $f = g \circ h$.*

4 Canonical machines

The goal of this section is to define a canonical bimachine for any rational function. By canonicity we mean that it should be machine-independent. Our ultimate goal is to show that the canonical bimachine suffices to decide the algebraic properties we are interested in. To get a canonical bimachine, we need a right automaton for the look-ahead that is 1) canonical, 2) coarse-grained enough to preserve algebraic properties, and 3) fine-grained enough to obtain a deterministic left automaton (and hence a bimachine).

We define the delay congruence and show that it is the coarsest left congruence such that any automaton $\mathcal{R}$ recognizing it satisfies that $f_{\mathcal{R}}$ is quasi-sequential (Proposition 21). However, this congruence is, in general, too coarse to make $f_{\mathcal{R}}$ sequential. We then introduce the ultimate congruence, and show how to combine these two congruences to build a canonical bimachine.

Let f be a transduction. We define the *delay* between $x, y \in A^\omega$ with respect to f by: $\mathsf{del}_f(x,y) = \{\mathsf{del}(f(ux), f(uy)) \mid ux, uy \in \mathrm{dom}(f)\}$. The following definition is taken from [21, 3].

▸ **Definition 19** (delay congruence)**.** The *delay congruence* of f is defined by setting $x \overset{\Delta}{\approx}_f y$ for $x, y \in A^\omega$ if (1) for all $u \in A^*$, $ux \in \mathrm{dom}(f) \Leftrightarrow uy \in \mathrm{dom}(f)$, and (2) $|\mathsf{del}_f(x,y)| < \infty$.

▸ **Example 20.** Let us illustrate the above definition on f_{blocks} (recall Example 3). We consider $x = u_1\#\dots\#u_n\#v$ and $y = u'_1\#\dots\#u'_n\#v'$ where v and v' are infinite words not containing $\#$. Note that $x \overset{\Delta}{\approx}_{f_{\mathrm{blocks}}} y$ if and only if u_1, u'_1 are either both empty, or end with the same letter. Indeed, if the latter holds then $\mathsf{del}(f(ux), f(uy)) = \mathsf{del}(f(x), f(y))$. Conversely, if both u_1, u'_1 are non-empty but end with different letters, then for any u without $\#$, $\mathsf{del}(f_{\mathrm{blocks}}(ux), f_{\mathrm{blocks}}(uy)) = (f(ux), f(uy))$. If $u_1 = \epsilon$ and u, u'_1 end with different letters, then again, $\mathsf{del}(f_{\mathrm{blocks}}(ux), f_{\mathrm{blocks}}(uy)) = (f(ux), f(uy))$. There are two more classes with respect to $\overset{\Delta}{\approx}_{f_{\mathrm{blocks}}}$, one for infinitely many $\#$, and one for no $\#$.

The look-ahead $\overset{\Delta}{\approx}_{f_{\mathrm{blocks}}}$ provides enough information to transform the blocks deterministically (we only need the last letter before the next $\#$), but not enough information to produce the output after the last $\#$ deterministically.

The following proposition shows that the delay congruence, when used as a look-ahead (see the definition of $f_{\mathcal{R}}$ page 11), transforms any rational function into a quasi-sequential one.

▸ **Proposition 21.** Let f be a transduction and let $\mathcal{R}$ be a right automaton recognizing $\mathrm{dom}(f)$. Then $f_{\mathcal{R}}$ is quasi-sequential iff $\approx_{\mathcal{R}} \sqsubseteq \overset{\Delta}{\approx}_f$. In particular, if f is aperiodic then $\overset{\Delta}{\approx}_f$ is aperiodic.

The delay congruence is minimal, *i.e.* coarsest, among right congruences of bimachines realizing a function, and we show in appendix that it can be computed in PTime from a bimachine.

▸ **Proposition 22.** Given a transducer $\mathcal{T}$ (resp. a bimachine $\mathcal{B}$) with underlying automaton $\mathcal{A}$ (resp. right automaton $\mathcal{R}$) realizing a function f, we have that $\approx_{\mathcal{A}}$ (resp. $\approx_{\mathcal{R}}$) is finer than $\overset{\Delta}{\approx}_f$.

Canonical machine for quasi-sequential functions As noted in [2], the class of quasi-sequential functions, or equivalently, the class of functions satisfying the WTP, is strictly larger than the class of sequential functions. The last left congruence that we define now will be fine enough to make a quasi-sequential function sequential. By taking the intersection between this congruence and the left delay congruence we will obtain a congruence that is fine enough to make any rational function sequential. However, it should be noted that this look-ahead is not minimal, in the sense that it is not necessarily coarser than any look-ahead that is fine enough to realize the function.

▸ **Definition 23** (Ultimate congruence)**.** We define the *ultimate congruence* of a rational function f by setting $x \overset{\smile}{\approx}_f y$ for $x, y \in A^\omega$ if the following conditions hold:

- For all $u \in A^*$, $ux \in \mathrm{dom}(f) \Leftrightarrow uy \in \mathrm{dom}(f)$
- If $ux \in \mathrm{dom}(f)$ then $\widehat{f}(u) = \overline{f}(ux) \Leftrightarrow \widehat{f}(u) = \overline{f}(uy)$ Moreover, if $\widehat{f}(u) = \overline{f}(ux)$ then $f(ux) = f(uy)$.

Observe that $\widehat{f}(u) \preceq \overline{f}(ux)$ for every $ux \in \mathrm{dom}(f)$. So the intuition behind $\widehat{f}(u) = \overline{f}(ux)$ is that no finite look-ahead on x can help to output $f(ux)$ deterministically after u. And the intuition behind $f(ux) = f(uy)$ is that the missing outputs $\widehat{f}(u)^{-1}f(ux)$ and $\widehat{f}(u)^{-1}f(uy)$

have to be equal, which is equivalent to $f(ux) = f(uy)$. Now, for a given class of $\breve{\approx}_f$ as look-ahead, a left automaton would know the missing output and start producing it. We show in the appendix (Lemma 42) that $\breve{\approx}_f$ is a left congruence.

▸ **Example 24.** Recall the function f_{blocks} defined in Example 3. $\widehat{f_{\text{blocks}}}$ maps every block to its output and stops at the last #. Hence $\widehat{f_{\text{blocks}}}(u) = \overline{f_{\text{blocks}}}(ux)$ if and only if x does not contain #. When $\widehat{f_{\text{blocks}}}(u) = \overline{f_{\text{blocks}}}(ux)$, we have $f(ux) = f(uy)$ if and only if x and y both contain an infinite number of a's, or none of them does. The congruence classes of $\breve{\approx}_{f_{\text{blocks}}}$ are thus: a) words x with an infinite number of # (yielding ux outside the domain), b) words x with a finite (non-zero) number of #, c) words without #, with an infinite number of a's, d) words without #, with a finite number of a's. This is precisely the information lacking in the look-ahead provided by $\hat{\approx}_{f_{\text{blocks}}}$ (see Example 20) to obtain a look-ahead allowing a sequential processing of the input.

▸ **Proposition 25.** For a quasi-sequential transduction f, the ultimate congruence $\breve{\approx}_f$ has finite index. If f is given as a bimachine, $\breve{\approx}_f$ can be computed in 2-ExpTime. Furthermore, if f is aperiodic then $\breve{\approx}_f$ is aperiodic.

Let $\mathcal{R}$ be a right automaton recognizing $\breve{\approx}_f$. We define the bimachine $\mathcal{U}_f^{\mathcal{R}} = (\mathcal{A}_f, \mathcal{R}, i_f, o_{\mathcal{R}})$ with $\mathcal{A}_f$ and i_f (as in Section 2), and for $o_{\mathcal{R}}$ we take:

$$o_{\mathcal{R}}([u], a, [x]_{\mathcal{R}}) = \begin{bmatrix} \widehat{f}(u)^{-1}\widehat{f}(ua) & \text{if } \widehat{f}(ua) \prec \overline{f}(uax) \\ \beta & \text{if } \widehat{f}(u) = \overline{f}(uax) \text{ and } \widehat{f}(u)^{-1}f(uax) = \alpha\beta^\omega \\ \alpha & \text{if } \widehat{f}(u) \prec \widehat{f}(ua) = \overline{f}(uax) \text{ and } \widehat{f}(u)^{-1}f(uax) = \alpha\beta^\omega \end{bmatrix}$$

The following lemma states that $\mathcal{U}_f^{\mathcal{R}}$ realizes f.

▸ **Lemma 26.** *Let f be a quasi-sequential transduction, and let $\mathcal{R}$ be a right automaton recognizing the ultimate congruence $\breve{\approx}_f$, then $\mathcal{U}_f^{\mathcal{R}}$ realizes f.*

Let $\mathcal{R}$ be the canonical right automaton of $\breve{\approx}_f$. By the previous lemma, there exists a bimachine with $\mathcal{R}$ as right automaton realizing f. By minimizing its left automaton with respect to $\mathcal{R}$, we obtain a canonical bimachine for f.

▸ **Corollary 27.** *Let f be a quasi-sequential transduction, and let $\mathcal{R}$ be the canonical right automaton of the ultimate congruence $\breve{\approx}_f$, then $\mathcal{B}_f^{\mathcal{R}}$ realizes f (and is finite).*

Canonical bimachine We finally show that by composing the information given by the delay and the ultimate congruences, we obtain a canonical bimachine for any rational function. Let us make clear what we mean by composition. Let $\mathcal{R}_1 = (Q_1, \Delta_1, I_1, F_1)$ be a right automaton and let $\mathcal{R}_2 = (Q_1, \Delta_2, I_2, F_2)$ be a right automaton over $A \times Q_1$. The automaton $\mathcal{R}_1 \bowtie \mathcal{R}_2$ is defined as $(Q_1 \times Q_2, \Delta_{\{1,2\}}, I_1 \times I_2, F_1 \times F_2)$ with $F_1 \times F_2 = \{P_1 \times P_2 \mid P_1 \in F_1,\ P_2 \in F_2\}$ and $\Delta_{\{1,2\}} = \{((s_1, s_2), a, (r_1, r_2) \mid (s_1, a, r_1) \in \Delta_1,\ (s_2, (a, r_1), r_2) \in \Delta_2\}$, which is a right automaton.

▸ **Lemma 28.** *Let $\mathcal{R}_1 = (Q_1, \Delta, I, F)$ be a right automaton and let $\mathcal{R}_2$ be a right automaton over $A \times Q_1$. Then $[\![\ell(\mathcal{R}_2)]\!] \circ [\![\ell(\mathcal{R}_1)]\!] = [\![\ell(\mathcal{R}_1 \bowtie \mathcal{R}_2)]\!]$ (up to the isomorphism between $(A \times Q_1) \times Q_2$ and $A \times (Q_1 \times Q_2)$).*

We can now state our main result. In our construction we focused on clarity and compositionality and we obtain a several-fold exponential complexity. At the cost of greater technicality, one should obtain a tighter result.

▸ **Theorem 29** (Canonical Bimachine). *Let f be a transduction given by a bimachine, let $\mathcal{R}_1$ be the canonical automaton of the delay congruence $\overset{\triangle}{\approx}_f$, and let $\mathcal{R}_2$ be the canonical automaton of the ultimate congruence $\overset{\smile}{\approx}_{(f_{\mathcal{R}_1})}$. Then the bimachine $\mathcal{B}_f^{\mathcal{R}_1 \bowtie \mathcal{R}_2}$ realizes f. Furthermore if f is aperiodic then $\mathcal{B}_f^{\mathcal{R}_1 \bowtie \mathcal{R}_2}$ is aperiodic.*

Proof. Let f be a transduction, let $\mathcal{R}_1$ be the canonical automaton of the delay congruence $\overset{\triangle}{\approx}_f$ and let $\mathcal{R}_2$ be the canonical automaton of the ultimate congruence $\overset{\smile}{\approx}_{(f_{\mathcal{R}_1})}$. Since $\mathcal{R}_1$ recognizes $\overset{\triangle}{\approx}_f$, we know according to Proposition 22 that $f_{\mathcal{R}_1}$ is quasi-sequential. Hence since $\mathcal{R}_2$ is finer than $\overset{\smile}{\approx}_{(f_{\mathcal{R}_1})}$, we know from Cor. 27 that the bimachine $\mathcal{B}_{f_{\mathcal{R}_1}}^{\mathcal{R}_2}$ realizes f. From Proposition 17 we obtain that $(f_{\mathcal{R}_1})_{\mathcal{R}_2}$, the function obtained by composing the labelings $\ell(\mathcal{R}_2)$ and $\ell(\mathcal{R}_1)$, is left-sequential. We use Lemma 28 to obtain that $f_{\mathcal{R}_1 \bowtie \mathcal{R}_2}$ is left-sequential and thus, again by Proposition 17 we know there is a bimachine with $\mathcal{R}_1 \bowtie \mathcal{R}_2$ as right automaton which realizes f. In particular, $\mathcal{B}_f^{\mathcal{R}_1 \bowtie \mathcal{R}_2}$ realizes f.

If we assume that f is aperiodic, we obtain from Proposition 22 that $\mathcal{R}_1$ is aperiodic and from Proposition 17 that $f_{\mathcal{R}_1}$ is aperiodic. Hence from Proposition 25 we have that $\mathcal{R}_2$ is aperiodic. Again from Proposition 17, we have that $(f_{\mathcal{R}_1})_{\mathcal{R}_2} = f_{\mathcal{R}_1 \bowtie \mathcal{R}_2}$ is aperiodic. A third time from Proposition 17 we have that $\mathcal{B}_f^{\mathcal{R}_1 \bowtie \mathcal{R}_2}$ is aperiodic. ◂

Note that the right automaton constructed in Proposition 1 is actually a right *Büchi* automaton. So our result would still hold for bimachines with Büchi right automata.

5 First-Order Definability Problem

In this section, we show that given a transducer $\mathcal{T}$ realizing a transduction $[\![\mathcal{T}]\!] : A^\omega \to B^\infty$, one can decide whether $[\![\mathcal{T}]\!]$ is first-order definable (FO-definable). First, let us recall the notion of FO-definability for word languages. Any word $w \in A^\infty$ is seen as a structure of domain $\{1, \dots, |w|\}$ linearly ordered by $\preceq$ and with unary predicates $a(x)$, for all $a \in A$. By FO we denote the first-order logic over these predicates, and by MSO the extension of FO with quantification over sets and membership tests $x \in X$ (see for instance [23] for a detailed definition). We write $w \models \phi$ if some word w satisfies a formula ϕ, and $\phi(x_1, \dots, x_n)$ any formula ϕ with n free first-order variables $x_1, \dots, x_n$. Interpreted over words in A^ω (resp. A^∞), any sentence ϕ defines a language $[\![\phi]\!] \subseteq A^\omega$ (resp. $[\![\phi]\!] \subseteq A^\infty$) defined as the set of words satisfying ϕ. *E.g.* the sentence $\phi_0 = \forall x, y \cdot a(x) \wedge b(y) \to x \preceq y$, interpreted on A^ω, defines the language $a^\omega \cup a^*b^\omega$. Interpreted on A^∞, it defines the language $a^\omega \cup a^*b^\omega \cup a^*b^*$. A language L is said to be FO-definable (resp. MSO-definable) if $L = [\![\phi]\!]$ for some sentence $\phi \in$ FO (resp. $\phi \in$ MSO).

Definability of transductions An MSO-transducer is a tuple $\mathcal{F} = (A, B, \phi_{dom}, V, \mu)$ where ϕ_{dom} is an MSO-sentence, V is a finite subset of B^* and μ a function mapping any word $v \in V$ to some MSO-formula (over alphabet A) denoted $\phi_v(x)$, with one-free variable. An *FO-transducer* is an MSO-transducer which uses only FO-formulas. Any MSO-transducer defines a transduction denoted $[\![\mathcal{F}]\!] \subseteq A^\omega \times B^\infty$ such that $(u, v) \in [\![\mathcal{F}]\!]$ if $u \models \phi_{dom}$ and there exists $(v_i)_{i \geqslant 1}$ such that $v = v_1 v_2 v_3 \dots$ and for all $i \geqslant 1$, $v_i \in V$ and $u \models \phi_{v_i}(i)$. We say that $f : A^\omega \to B^\infty$ is MSO- (resp. FO-) definable if there exists some MSO- (resp. FO-) transducer $\mathcal{F}$ such that $[\![\mathcal{F}]\!] = f$.

For example the functional transduction which erases all a's of any input ω-word over $\{a, b\}$ is defined by $\phi_{dom} = \top$ and the two formulas $\phi_\epsilon(x) = a(x)$ and $\phi_b(x) = b(x)$. The functional transduction mapping any word of the form $a^n b^\omega$ to $a^{\lfloor n/2 \rfloor} b^\omega$ is not FO-definable, even though its domain is. Intuitively, the formula $\phi_a(x)$ would have to decide whether

x is an odd or even position, which is a typical non FO-definable property. It is one of the goal of this paper to automatically verify that such a property is indeed not FO. It is however MSO-definable with $\phi_{dom} = \phi_0 \wedge \exists x \cdot b(x)$, where ϕ_0 has been defined before, and the three formulas $\phi_\epsilon(x) = a(x) \wedge odd(x)$, $\phi_a(x) = a(x) \wedge even(x)$ (properties which are MSO-definable) and $\phi_b(x) = b(x)$.

As a remark, Courcelle has defined in the context of graph transductions the notion of MSO-transducers [9], which can also be restricted to FO-transducers. Cast to infinite words, Courcelle's formalism is strictly more expressive than rational functions, as they allow to mirror factors of the input word for instance. Restricted to the so called *order-preserving* Courcelle transducers [4, 12], they are equivalent to our MSO- and FO-transducers, however with a more complicated definition. This equivalence can be seen, for finite words, in the proof of Theorem 4 in [12]. The same proof works for infinite words as well.

We first exhibit a correspondence between logics and transducers, the proof of which is similar to the finite case [12], but requires some additional results on aperiodic automata on ω-words.

▸ **Theorem 30** (Logic-transducer correspondences). *Let $f : A^\omega \to B^\infty$. Then:*
- *f is MSO-definable if and only if it is realizable by some transducer.*
- *f is FO-definable if and only if it is realizable by some aperiodic transducer.*

We obtain the following decidability result (in elementary complexity if the input is a transducer).

▸ **Theorem 31.** *It is decidable whether a rational function $f : A^\omega \to B^\infty$, given as a transducer or equivalently as an MSO-transducer, is definable in FO.*

Proof. By Theorem 30, it suffices to show that f is aperiodic, *i.e.* definable by some aperiodic transducer. By Theorem 16, one can construct a bimachine which is aperiodic if and only if f is. So, it suffices to construct this bimachine and to test its aperiodicity, *i.e.*, whether its left and right automata are both aperiodic, a property which is decidable [10]. ◂

References

1. André Arnold. A syntactic congruence for rational ω-languages. *Theoretical Computer Science*, 39(2–3):333–335, August 1985. Note.
2. Marie-Pierre Béal and Olivier Carton. Determinization of transducers over infinite words: The general case. *Theory Comput. Syst.*, 37(4):483–502, 2004. Available from: `https://doi.org/10.1007/s00224-003-1014-9`.
3. Adrien Boiret, Aurélien Lemay, and Joachim Niehren. Learning rational functions. In *Developments in Language Theory - 16th International Conference, DLT 2012, Taipei, Taiwan, August 14-17, 2012. Proceedings*, pages 273–283, 2012. Available from: `https://doi.org/10.1007/978-3-642-31653-1_25`.
4. Mikolaj Bojanczyk. Transducers with origin information. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part II*, volume 8573 of *Lecture Notes in Computer Science*, pages 26–37. Springer, 2014.
5. Olivier Carton. Right-sequential functions on infinite words. In *Computer Science - Theory and Applications, 5th International Computer Science Symposium in Russia, CSR 2010, Kazan, Russia, June 16-20, 2010. Proceedings*, pages 96–106, 2010. Available from: `https://doi.org/10.1007/978-3-642-13182-0_9`.

6 Olivier Carton and Max Michel. Unambiguous büchi automata. *Theor. Comput. Sci.*, 297(1-3):37–81, 2003. Available from: https://doi.org/10.1016/S0304-3975(02)00618-7.

7 Olivier Carton, Dominique Perrin, and Jean-Eric Pin. Automata and semigroups recognizing infinite words. In *Logic and Automata: History and Perspectives [in Honor of Wolfgang Thomas]*., pages 133–168, 2008.

8 Christian Choffrut. Minimizing subsequential transducers: a survey. *Theor. Comput. Sci.*, 292(1):131–143, 2003. Available from: https://doi.org/10.1016/S0304-3975(01)00219-5.

9 Bruno Courcelle. Monadic second-order definable graph transductions: A survey. *Theor. Comput. Sci.*, 126(1):53–75, 1994.

10 Volker Diekert and Paul Gastin. First-order definable languages. In Jörg Flum, Erich Grädel, and Thomas Wilke, editors, *Logic and Automata*, volume 2 of *Texts in Logic and Games*, pages 261–306. Amsterdam University Press, 2008.

11 Calvin C. Elgot and Jorge E. Mezei. On relations defined by generalized finite automata. *IBM Journal of Research and Development*, 9(1):47–68, 1965. Available from: https://doi.org/10.1147/rd.91.0047.

12 Emmanuel Filiot. Logic-automata connections for transformations. In *Indian Conference on Logic and Its Applications (ICLA)*, volume 8923 of *Lecture Notes in Computer Science*, pages 30–57. Springer, 2015.

13 Emmanuel Filiot, Olivier Gauwin, and Nathan Lhote. Aperiodicity of rational functions is PSPACE-complete. In *36th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2016, December 13-15, 2016, Chennai, India*, pages 13:1–13:15, 2016. Available from: https://doi.org/10.4230/LIPIcs.FSTTCS.2016.13.

14 Emmanuel Filiot, Olivier Gauwin, and Nathan Lhote. First-order definability of rational transductions: An algebraic approach. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS*, pages 387–396. ACM, 2016.

15 Emmanuel Filiot, Olivier Gauwin, and Nathan Lhote. Logical and algebraic characterizations of rational transductions. *CoRR*, abs/1705.03726, 2017. Available from: http://arxiv.org/abs/1705.03726.

16 Françoise Gire. Two decidability problems for infinite words. *Inf. Process. Lett.*, 22(3):135–140, 1986. Available from: https://doi.org/10.1016/0020-0190(86)90058-X.

17 Leonid Libkin. *Elements of Finite Model Theory*. Springer-Verlag, Berlin, 2004.

18 Robert McNaughton and Seymour Papert. *Counter-free automata*. M.I.T. Press, 1971.

19 D. Perrin. Recent results on automata and infinite words. In M. P. Chytil and V. Koubek, editors, *Proceedings of the 11th Symposium on Mathematical Foundations of Computer Science*, volume 176 of *LNCS*, pages 134–148, Praha, Czechoslovakia, September 1984. Springer.

20 Christophe Prieur. How to decide continuity of rational functions on infinite words. *Theor. Comput. Sci.*, 276(1-2):445–447, 2002.

21 Christophe Reutenauer and Marcel Paul Schützenberger. Minimization of rational word functions. *SIAM J. Comput.*, 20(4):669–685, 1991. Available from: https://doi.org/10.1137/0220042.

22 Marcel-Paul Schützenberger. On finite monoids having only trivial subgroups. *Information and Control*, 8(2):190–194, 1965.

23 W. Thomas. Languages, automata and logic. In A. Salomaa and G. Rozenberg, editors, *Handbook of Formal Languages*, volume 3, Beyond Words. Springer, Berlin, 1997.

24 Wolfgang Thomas. Star-free regular sets of omega-sequences. *Information and Control*, 42(2):148–156, August 1979.

25 Wolfgang Thomas. A combinatorial approach to the theory of omega-automata. *Information and Control*, 48(3):261–283, 1981.

26 Thomas Wilke. Past, present, and infinite future. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, pages 95:1–95:14, 2016. Available from: `https://doi.org/10.4230/LIPIcs.ICALP.2016.95`.

A Proofs for Section 1

The *syntactic congruence* of an ω-language L is defined for $u, v \in A^+$ by $u \equiv_L v$ if $\forall r, s \in A^*, t \in A^+$: 1) $rust^\omega \in L \Leftrightarrow rvst^\omega \in L$ and 2) $r(us)^\omega \in L \Leftrightarrow r(vs)^\omega \in L$.

Proof of Proposition 1. Let $\approx$ be a left congruence. The construction of [6] starts with a congruence that is finer than the syntactic congruence of the ω-regular language L, and produces a right automaton of exponential size with Büchi acceptance condition, recognizing L.

For our purpose it suffices to start with the coarsest congruence that refines the syntactic congruence of any of the languages $[x]_\approx$, call it $\equiv$. This means that the automaton we obtain from $\equiv$ applying the construction of [6] can recognize any congruence class of $\approx$, and hence recognizes $\approx$.

Let us now discuss aperiodicity. A language is aperiodic if and only if its syntactic congruence is aperiodic. We just give the main arguments why the construction of [6] preserves aperiodicity and refer the reader to [6] for a thorough understanding (and we also use the same notations). The states in the constructions are pairs of the form $([s, e], (s_1, \ldots, s_n))$, where $[s, e]$ denotes a conjugacy class of linked pairs and $(s_1, \ldots, s_n)$ denotes a chain of R-classes. Since we start from an aperiodic semigroup S (given by the classes of the syntactic congruence), the left action of the semigroup on conjugacy classes is obviously aperiodic. More precisely, for n large enough we have $u^{n+1} \cdot [s, e] = [\left[u_S^{n+1}\right] s, e] = [[u^n]_S s, e]$. Similarly, using Proposition 69 in the article and since u^{n+1} and u^n are in the same R-class (for n large enough) we have for any infinite word w that $\widehat{\varphi}(u^{n+1}w) = \widehat{\varphi}(u^n w)$. Finally it is clear that the construction of Lemma 17 in [6] which goes from a right Büchi automaton with a final set of *transitions* to a right Büchi automaton with a final set of *states*, preserves aperiodicity. ◂

B Proofs for Section 2

B.1 Syntactic congruence

Proof of Proposition 6. Let $u \sim_f v$ and let $a \in A$ we want to show that $ua \sim_f va$. If $\widehat{f}(u)$ and $\widehat{f}(v)$ are both infinite with the same ultimate period then it is also the case for $\widehat{f}(ua)$ and $\widehat{f}(va)$. Otherwise, for any x such that $ux, vx \in \mathrm{dom}(f)$, we have $\widehat{f}(u)^{-1} f(ux) = \widehat{f}(v)^{-1} f(vx)$ and we denote this word by $g(x)$. Note that $f(ux) = \widehat{f}(u) g(x)$. If $\bigwedge_x g(ax)$ is infinite then we have both $\widehat{f}(ua)$ and $\widehat{f}(va)$ infinite with the same ultimate period. Otherwise:

$$\begin{aligned}
\widehat{f}(ua)^{-1} f(uax) &= \left(\bigwedge_y f(uay)\right)^{-1} f(uax) \\
&= \left(\bigwedge_y \widehat{f}(u) g(ay)\right)^{-1} f(uax) \\
&= \left(\widehat{f}(u) \bigwedge_y g(ay)\right)^{-1} f(uax) \\
&= \left(\bigwedge_y g(ay)\right)^{-1} \widehat{f}(u)^{-1} f(uax) \\
&= \left(\bigwedge_y g(ay)\right)^{-1} \widehat{f}(v)^{-1} f(vax) \\
&= \widehat{f}(va)^{-1} f(vax)
\end{aligned}$$

◂

$\mathcal{T}_f$ is well-defined. We have left to show that the outputs are well-defined. If $u \sim_f v$ and $\widehat{f}(u) = \alpha\beta^\omega$ with $\beta \neq \epsilon$, then $\widehat{f}(v) = \alpha'\beta^\omega$ since both words are in normal form, and thus the

output is uniquely defined. Otherwise it suffices to show that $\widehat{f}(u)^{-1}\widehat{f}(ua) = \widehat{f}(v)^{-1}\widehat{f}(va)$ to have well-defined outputs.

$$\begin{array}{rcl} \widehat{f}(u)^{-1}\widehat{f}(ua) & = & \widehat{f}(u)^{-1}\bigwedge_x f(uax) \\ & = & \widehat{f}(u)^{-1}\bigwedge_x \widehat{f}(u)g(ax) \\ & = & \widehat{f}(u)^{-1}\widehat{f}(u)\bigwedge_x g(ax) \\ & = & \bigwedge_x g(ax) \\ & = & \widehat{f}(v)^{-1}\widehat{f}(va) \end{array}$$

◂

Proof of Proposition 7. We denote by g the sequential function realized by $\mathcal{T}_f$. Let us first show that $\mathrm{dom}(g) = \overline{\mathrm{dom}(f)}$. Let $x \in A^\omega$, since $\mathcal{A}_f$ has no acceptance condition then $x \in \mathrm{dom}(g)$ if and only if there is a run of $\mathcal{A}_f$ over it. Let $x \in \overline{\mathrm{dom}(f)}$, then for any integer n, there is an ω-word y_n such that $x(:n)y_n \in \mathrm{dom}(f)$, hence by definition of Δ_f there is an infinite run of $\mathcal{A}_f$ over x. Let us now assume $x \notin \overline{\mathrm{dom}}(f)$, then there is an integer n such that for all $y \in A^\omega$, $x(:n)x(n)y \notin \mathrm{dom}(f)$, hence there is no run of $\mathcal{A}_f$ over x. Hence $\mathrm{dom}(g) = \overline{\mathrm{dom}(f)}$.

Let x be a word in $\overline{\mathrm{dom}(f)}$. Let us first assume that for any integer n, $\widehat{f}(x(:n))$ is finite. Then in $\mathcal{T}_f$, we have by definition that $[\epsilon] \xrightarrow{x(:n)|\alpha_n} [x(:n)]$ with $i_f([\epsilon])\alpha_n = \widehat{f}(x(:n))$. Thus $g(x) = \lim_n \widehat{f}(x(:n)) = \overline{f}(x)$.

Now let us assume that for some integer $k \geqslant 0$ we have $\widehat{f}(x(:k)) = \alpha\beta^\omega$ with $\beta \in B^+$, and let us also assume without loss of generality that k is the smallest of such integers. Then we must have $\overline{f}(x) = \widehat{f}(x(:k)) = \alpha\beta^\omega$. Furthermore, by definition we have in $\mathcal{T}_f$, $[\epsilon] \xrightarrow{x(:k)|\gamma} [x(:k)]$ with $i([\epsilon])\gamma = \alpha\beta^l$ for some integer l, and for $n \geqslant k$ we have $[x(:n)] \xrightarrow{x(n+1)|\beta} [x(:n+1)]$. Hence $g(x) = \lim_{n \geqslant k} \alpha\beta^{n-k+l} = \alpha\beta^\omega = \overline{f}(x)$. ◂

B.2 Sequential transductions

We start with two properties of sequential functions.

▸ **Proposition 32.** For any sequential transduction f, it holds that $\overline{f}_{|\mathrm{dom}(f)} = f$.

Proof. Let $\mathcal{T}$ be a sequential transducer realizing f and let $x \in \mathrm{dom}(f)$. If $f(x)$ is finite, then there exists an integer k such that $q_0 \xrightarrow{x(:k)|f(x)} p$. Since f is sequential, for any $n \geqslant k$ we have $\widehat{f}(x(:n)) = f(x)$ which means that $\overline{f}(x) = f(x)$. Otherwise, if $f(x)$ is infinite, then there must be an increasing sequence of indexes $n_1, n_2, \ldots$ such that $q_0 \xrightarrow{x(:n_k)|\alpha_k} p_k$ with $\alpha_k \prec \alpha_{k+1}$. By sequentiality of f, $i(q_0)\alpha_k \preceq \widehat{f}(x(:n_k))$ and since $f(x) = \lim_k i(q_0)\alpha_k$ is infinite, we have that $\overline{f}(x) = \lim_k \widehat{f}(x(:n_k))$ is also infinite and the two words are equal. ◂

▸ **Proposition 33.** For any sequential transducer with underlying deterministic automaton $\mathcal{A}$ realizing a transduction f, we have $\sim_{\mathcal{A}} \sqsubseteq \sim_f$.

Proof. Let $\mathcal{T} = (\mathcal{A}, i, o)$ be a sequential transducer realizing a function f, hence $\mathcal{A} = (Q, \Delta, q_0, F)$ is deterministic. Let $u \sim_{\mathcal{A}} v$, we want to show that $u \sim_f v$. Let $x \in A^\omega$, since u and v reach the same state from q_0 we have that $ux \in \mathrm{dom}(f) \Leftrightarrow vx \in \mathrm{dom}(f)$. Let us assume that $ux \in \overline{\mathrm{dom}(f)}$ for some word x, then we know that for any $w \prec x$ there exists z such that $uwz \in \mathrm{dom}(f)$. However we have by assumption $uwz \in \mathrm{dom}(f) \Leftrightarrow vwz \in \mathrm{dom}(f)$, hence $vx \in \overline{\mathrm{dom}(f)}$.

Let $q \in Q$, we define $\mathcal{T}_q$ by $\mathcal{T}_q = (\mathcal{A}_q, i, o)$ with $\mathcal{A}_q = (Q, \Delta, q, F)$ and we denote the corresponding sequential function by f_q. Let q be the state reached from q_0 by reading u (or v), and let γ be the longest common prefix of all outputs along final runs starting at q, *i.e.* $\gamma = \widehat{f}_q(\epsilon)$. If γ is infinite then $\widehat{f}(u)$ and $\widehat{f}(v)$ have the same ultimate period, which is also the one of γ. If γ is finite, then let $q_0 \xrightarrow{u|\alpha} q$ and $q_0 \xrightarrow{v|\beta} q$ denote the initial runs of $\mathcal{A}$ over u and v, respectively. We thus have $\widehat{f}(u) = i(q_0)\alpha\gamma$ and $\widehat{f}(v) = i(q_0)\beta\gamma$. Let x be such that $ux, vx \in \mathrm{dom}(f)$. Since $\mathcal{A}$ is deterministic, we have:

$$\begin{aligned} \widehat{f}(u)^{-1} f(ux) &= (i(q_0)\alpha\gamma)^{-1}\, i(q_0)\alpha f_q(x) \\ &= \gamma^{-1} f_q(x) \\ &= \widehat{f}(v)^{-1} f(vx) \end{aligned}$$

Thus we obtain $u \sim_f v$ which concludes the proof. ◂

Proof of Proposition 8. To compute $\mathcal{T}_f$ we need to compute the longest common prefix function $\widehat{f}$ and to determine the classes of the syntactic congruence $\sim_f$. For the classes of $\sim_f$ we can use Proposition 33: for every state p of $\mathcal{T}$ we fix some (minimal-length) representative word u_p leading from the initial state to p. Then we check whether $u_p \sim_f u_q$, for every pair of states p, q of $\mathcal{T}$ (we explain below how this can be done). Doing so, we have determined the classes $c_1, \ldots, c_n$ of $\sim_f$ together with a representative $u_1, \ldots, u_n$ for each of them. To compute the initial state, it suffices to test which i satisfies $u_i \sim_f \epsilon$. To compute the transitions, given a state $[u_j]$ and a letter a, it suffices to determine which i satisfies $u_i \sim_f u_j a$ to get the next state. The outputs are computed with the function $\widehat{f}$. We now explain how to test in polynomial time (in $|\mathcal{T}|$) whether two words u, v satisfy $u \sim_f v$, and how to compute $\widehat{f}(u)$.

Let $\mathcal{T} = (\mathcal{A}, i, o)$ with $\mathcal{A} = (Q, \Delta, q_0, F)$ deterministic, and let us consider a finite word u over A. We want first to check whether $\widehat{f}(u)$ is ultimately periodic and compute it, if this is the case. If q denotes the state reached by u in $\mathcal{T}$, then we ask whether all paths from q are labeled by the same output word. Equivalently, it suffices to check whether there exist two accepting paths starting in q, with different output words. The check can be done by a product construction where we ignore the input and just monitor the delay between the two outputs, keeping it bounded by the maximal length of output words in $\mathcal{T}$. So the question boils down to emptiness check of an automaton of size $|Q|^2 \times |o|$. If all paths from q have the same output word, then we can compute this word in polynomial time by selecting an accepting path from q.

Let us now consider the case where $\widehat{f}(u)$ is finite, and show that we can compute it in polynomial time as well. To compute $\widehat{f}(u)$ we can view $\mathcal{T}$ as a graph where we remove the inputs, keeping only the output words labeling transitions. First we argue that the length of $\widehat{f}(u)$ is polynomial, by showing that we can always find two paths from q of polynomial length, with different outputs. We start with two arbitrary finite paths π_0, π_1 from q, with different outputs, such that their lengths are minimal. So the mismatch between the outputs of π_0, π_1 is in the last transitions. If one of the paths is loop-free, the claim is shown. Otherwise, each of π_0, π_1 contains some loops. If there are two loops with overall label of same length, one in π_0, the other in π_1, then we get a contradiction to the minimality of π_0, π_1, by cutting these loops. If all loops of π_0 have overall labels of length different from those of π_1, then we fix some loop ℓ_i in π_i with overall label of length k_i. We remove all loops from π_i except ℓ_i, and get a new path π'_i by iterating ℓ_i k_{1-i} times. Thus, π'_0, π'_1 are of (the same) polynomial length and have different outputs, as treated in the previous case.

Once we know that the length of $\widehat{f}(u)$ is bounded by a polynomial we can compute it by a.k.a. subset construction on the above graph, that is executed only a polynomial number of

steps. Roughly, we follow from q all possible paths, and store only their delays, bounded by the maximal output length of $\mathcal{T}$.

We show now how to check for two words u, v over A whether $u \sim_f v$.

The first condition of $\sim_f$ asks that $ux \in \overline{\mathrm{dom}(f)}$ if and only if $vx \in \overline{\mathrm{dom}(f)}$. Since $\mathcal{T}$ is trim, $ux \in \overline{\mathrm{dom}(f)}$ is equivalent to saying that ux labels a path of $\mathcal{A}$. Hence, if u and v lead to some states p and q respectively in $\mathcal{A}$, testing the former condition amounts to check whether $L_p = L_q$, where L_p (resp. L_q) denotes the set of ω-words for which there exists a path from p (resp. q). Since $\mathcal{A}$ is deterministic, the latter can be checked in polynomial time.

For the second condition we may assume that $\widehat{f}(u)$, $\widehat{f}(v)$ are both finite (and already computed). A standard product construction of $\mathcal{A}$ with itself allows to check whether some x exists s.t. $\widehat{f}(u)^{-1} f(ux) = \widehat{f}(v)^{-1} f(vx)$. ◂

Proof of Theorem 9. If f is sequential then Propositions 33 and 32 yield the claim. Conversely, assume that $\mathcal{T}_f$ is finite. From Proposition 7 we know that $\mathcal{T}_f$ realizes $\overline{f}$. Since $\mathcal{A}_f$ is deterministic we also know that $\overline{f}$ is sequential. Let $\mathcal{A}$ be a deterministic automaton recognizing $\mathrm{dom}(f)$. By taking the product of $\mathcal{A}_f$ and $\mathcal{A}$ we obtain a deterministic transducer realizing $\overline{f}_{|\mathrm{dom}(f)} = f$. ◂

B.3 Quasi-sequential transductions

We now recall an algorithm described in [2] (referring to it as *subset construction with delays*) and argue that it terminates on transducers satisfying WTP with a sequential transducer computing $\overline{f}$. We will need this construction in order to define the canonical bimachine and show that it preserves aperiodicity in Section 4.

Let $\mathcal{T} = (\mathcal{A}, i, o)$ with $\mathcal{A} = (Q, \Delta, I, F)$ be a transducer which satisfies the WTP.

For any state $q \in Q$ we denote by $\beta_q \in B^\infty$ the longest common prefix of the outputs over final runs starting in q and by C the set of constant states. Recall that Proposition 8 shows how to compute β_q and C is PTime. We thus assume that $\mathcal{T}$ is in earliest form, meaning that outputs are generated as soon as possible, according to $\widehat{f}$.

We define now the transducer $\mathcal{S} = (\mathcal{D}, i', o')$. States of $\mathcal{D}$ are sets of pairs (q, x), with q state of $\mathcal{A}$ and x a regular word.

Let $\alpha\beta^\omega = \bigwedge_{q \in I} i(q)\beta_q$. The initial state of $\mathcal{D}$ is $I_0 = \{(q, \alpha^{-1} i(q)\beta_q) \mid q \in C \cap I\} \cup \{(q, \alpha^{-1} i(q)) \mid q \in I \setminus C\}$, and $i'(I_0) = \alpha$. Given an already constructed state P and a letter $a \in A$, we define:

$$R = \left\{(q, w) \mid (p, w) \in P,\ p \in C \text{ and } p \xrightarrow{a} q\right\} \cup \left\{(q, uv) \mid (p, u) \in P,\ p, q \notin C \text{ and } p \xrightarrow{a|v} q\right\}$$
$$\cup \left\{(q, uv\beta_q) \mid (p, u) \in P,\ p \notin C,\ q \in C \text{ and } p \xrightarrow{a|v} q\right\}$$

Let now $\alpha\beta^\omega$ be the longest common prefix of all words appearing in R. If $\alpha = \epsilon$ we define $v = \beta$, otherwise we define $v = \alpha$. Note that if $\beta \neq \epsilon$ then this means that all the words appearing in R are equal to $\alpha\beta^\omega$. We define a new state $P' = \{(q, v^{-1} w) \mid (q, w) \in R\}$ and add the transition $P \xrightarrow{a|v}_{\mathcal{S}} P'$. Finally we keep only the accessible part of $\mathcal{S}$.

▸ Remark. The transducer $\mathcal{S}$ is (almost) sequential, in the sense that its underlying automaton is deterministic. However it may be infinite.

▸ **Proposition 34** ([2])**.** Let $\mathcal{T}$ be a transducer satisfying the WTP. Then the transducer obtained by subset construction with delays is finite (of exponential size).

Proof. The authors of [2] have a slightly different model of transducers where a run has to produce an infinite word in order to be accepting. However this difference doesn't affect the results and their proofs can be applied almost unchanged. In particular, for a transducer satisfying the WTP, it is shown in Lemma 12 that the difference of outputs between two runs over the same input is polynomial (with a small catch for constant states). As for automata the exponential blow-up is unavoidable. ◂

▸ **Proposition 35.** Let $\mathcal{T}$ be a transducer realizing a function f, and let $\mathcal{S}$ be the transducer obtained by the subset construction with delays. Then $\mathcal{S}$ realizes $\overline{f}$.

Proof. Let $\mathcal{T} = (\mathcal{A}, i, o)$ with $\mathcal{A} = (Q, \Delta, I, F)$ be a transducer realizing a function f. Let $\mathcal{S} = (\mathcal{D}, i', o')$ with $\mathcal{D} = (S, \delta, \{I_0\})$ be the sequential transducer obtained by subset construction with delays, and let g denote the function realized by $\mathcal{S}$.

First let us note that, by construction, for any initial run $I_0 \xrightarrow{u|v} P$ of $\mathcal{S}$, we have that $i'(I_0)v \preceq \widehat{f}(u)$ since the output along a run over u is a prefix of all the possible outputs along runs over an infinite word beginning with u in $\mathcal{T}$. Let x be a word in $\overline{\mathrm{dom}(f)}$, let u be a strict prefix of x and let $P = \{(p_1, w_1), \ldots, (p_n, w_n)\}$ be the state reached in $\mathcal{S}$ after reading u. Let us assume that $\widehat{f}(u)$ is a finite word. By construction we have $\bigwedge_{1 \leqslant i \leqslant n} w_i = \epsilon$, since $\widehat{f}(u)$ is finite, and there are two cases: either $w_i = \epsilon$ for some i, or there are w_i, w_j non-empty such that $w_i \wedge w_j = \epsilon$. In the latter case we have $\widehat{f}(u) = i'(I_0)v$. In the first case, since $\mathcal{T}$ is in earliest form and p_i is non-constant, there are two words y, z which have a final run from p_i, and whose respective outputs α, β have no common prefix. Hence $\widehat{f}(u) \preceq (i'(I_0)v\alpha \wedge i'(I_0)v\beta) = i'(I_0)v$. Thus if for any strict prefix u of x, $\widehat{f}(u) = i'(I_0)v$ is a finite word, we have $\overline{f}(x) = \lim_{u \prec x} \widehat{f}(u) = \lim_{u \prec x} v = g(x)$. Otherwise let us now assume that for some prefix u, $\widehat{f}(u)$ is an infinite word of the form $\alpha\beta^\omega$. All the w_i's must thus be equal to β^ω and $i'(I_0)v = \alpha\beta^k$ for some k. By definition of $\mathcal{S}$, the output of $\mathcal{S}$ when reading $x(|u|+1)$ from P is equal to β and it is the same for all the following transitions over x. We obtain $g(x) = \alpha\beta^\omega = \overline{f}(x)$. ◂

Proof of Theorem 12. According to Proposition 34, (1) implies that $\mathcal{S}$ is finite, which proves (2).

Let $u \sim_{\mathcal{S}} v$, we want to show that $u \sim_f v$ which will prove that (2) implies (3). Let $I_0 \xrightarrow{u|\alpha} R$ and $I_0 \xrightarrow{v|\beta} R$ denote the initial runs of $\mathcal{S}$ over u and v, respectively. If R is a constant state which produces an infinite word, then $\widehat{f}(u)$ and $\widehat{f}(v)$ are infinite words with the same ultimate period hence $u \sim_f v$. If R contains only one pair (p, w), then we have $w = \epsilon$ by construction. Furthermore since $\mathcal{T}$ is in earliest form, we have $i'(I_0)\alpha = \widehat{f}(u)$ and $i'(I_0)\beta = \widehat{f}(v)$. Otherwise there must be two pairs $(p_1, w_1), (p_2, w_2) \in R$ such that $\mathsf{del}(w_1, w_2) = \epsilon$ and in that case we also have $i'(I_0)\alpha = \widehat{f}(u)$ and $i'(I_0)\beta = \widehat{f}(v)$. Let x be a word such that $(p, w) \in R$ and x has a final run in $\mathcal{T}$ $p \xrightarrow{x|\gamma} F$. Then we have $f(ux) = i'(I_0)\alpha w\gamma$ (with the convention that $y\epsilon = y$ even for infinite words). Thus $\widehat{f}(u)^{-1} f(ux) = w\gamma = \widehat{f}(v)^{-1} f(vx)$. Hence $u \sim_f v$.

We now show that (3) implies (1). Let us now assume that $\sim_f$ has finite index. Let us assume, towards a contradiction, that $\mathcal{T}$ does not satisfy the WTP. Let us consider two runs such that $p_1 \xrightarrow{u|u_1} q_1 \xrightarrow{v|v_1} q_1$ and $p_2 \xrightarrow{u|u_2} q_2 \xrightarrow{v|v_2} q_2$. By taking $u' = uv^k$ and $v' = v^l$ for some integers k, l we have an initial run of $\mathcal{T}_f$: $q_0 \xrightarrow{u'|u_3} q \xrightarrow{v'|v_3} q$. We can assume without loss of generality that $u = u'$ and $v = v'$.

Let us first consider the case where q_1, q_2 are non-constant and $\mathsf{del}(i(p_1)u_1, i(p_2)u_2) \neq \mathsf{del}(i(p_1)u_1v_1, i(p_2)u_2v_2)$. If $|v_1| = |v_2|$ then there must be a mismatch between $i(p_1)u_1v_1$

and $i(p_2)u_2v_2$, *i.e.* a position k such that $i(p_1)u_1v_1(k) \neq i(p_2)u_2v_2(k)$. Thus we have for $n \geqslant 1$, that $|\widehat{f}(uv^n)| \leqslant |i(p_1)u_1v_1|$, hence there must be an integer N such that for all $n \geqslant N$, $\widehat{f}(uv^n) = \widehat{f}(uv^N)$. Furthermore, since q_2 is non-constant, by the pre-processing step there is a word x which produces y from q_2 such that $v_2 \wedge y = \epsilon$ and we have for $m \neq n$ that $i(p_2)u_2v_2^m y \neq i(p_2)u_2v_2^n y$. Hence for each $n \geqslant N$ we have that $\widehat{f}(uv^n)^{-1}f(uv^n x) = \widehat{f}(uv^N)^{-1}i(p_2)u_2v_2^n y$ takes a different value, which contradicts the fact that $\sim_f$ has finite index.

If $|v_1| \neq |v_2|$, we assume $|v_1| < |v_2|$ without loss of generality. Since q_1 is non-constant, we have $\widehat{f}(uv^n) \preceq i(p_1)u_1v_1^n$ for any $n \geqslant 0$ which means that $|v_3| \leqslant |v_1| < |v_2|$. Hence for any n we have a distinct word $\widehat{f}(uv^n)^{-1}i(p_2)u_2v_2^n = \left(i(q_0)u_3v_3^n\right)^{-1} i(p_2)u_2v_2^n$. Again, we use the fact that q_2 is constant and choose a word x producing y from q_2 such that $v_2 \wedge y = \epsilon$. Hence for each n we have that $\widehat{f}(uv^n)^{-1}f(uv^n x) = \left(i(q_0)u_3v_3^n\right)^{-1} i(p_2)u_2v_2^n y$ takes a different value, again leading to a contradiction.

Now we consider the case where q_1 is non-constant and q_2 is constant, we denote by y_2^ω the word produced from q_2 (x_2 is empty thanks to the preprocessing step) and we assume we have both $v_1 \neq \epsilon$ and $i(p_1)u_1v_1^\omega \neq i(p_2)u_2y_2^\omega$. Let k be such that $i(p_1)u_1v_1^k$ is not a prefix of $i(p_2)u_2y_2^\omega$, we thus have for any n, $|\widehat{f}(uv^n)| \leqslant |i(p_1)u_1v_1^k|$ and in particular there exists an integer N such that for all $n \geqslant N$, $\widehat{f}(uv^n) = \widehat{f}(uv^N)$. We choose a word x which produces y from q_1 and such that $v_1 \wedge y = \epsilon$. We obtain that for each $n \geqslant N$, $\widehat{f}(uv^n)^{-1}f(uv^n x) = \widehat{f}(uv^N)^{-1}i(p_1)u_1v_1^n y$ takes a different value, which is in contradiction with the fact that $\sim_f$ has finite index.

For proving that the subset construction with delays preserves aperiodicity, we rely on [14] where it is shown that the subset construction with delays preserves aperiodicity for transducers over finite words. The proof is almost the same when dealing with infinite words. Let us give the basic ideas to adapt the proof to infinite delays. The main idea of the original proof is to show that if the starting transducer is aperiodic then $\mathcal{S}$ must be counter-free, and thus aperiodic. We assume that we have a counter of minimal size k and show that k must be 1. A counter means that there is a run in $\mathcal{S}$ of the form $R_0 \xrightarrow{u} R_1 \cdots R_{k-1} \xrightarrow{u} R_0$. In the original proof, we have by aperiodicity that the R_j's contain the same states and then we show that the delays cannot change which means that all the R_j's must be equal. If we add infinite delays in the R_j's, the same idea works. First let us notice that all the delays in a given state R_j must be prefixes of each other, otherwise the length of the outputs from R_j must be bounded, which automatically means that the delays cannot change. This means that two infinite delays in a given state must be equal. Now we have three cases, either none of the states have infinite delays, then the original proof works. Or all states have an infinite delay, then by construction the output must be the period of the delay. Finally if some of the states have finite delays and some have infinite delays, then we can show in the same way that all finite delays are the same and thus, all infinite delays have an arbitrarily large common prefix, which concludes the proof. ◂

C Proofs for Section 3

C.1 Bimachines over infinite words

▸ **Proposition 36.** Given a bimachine with left and right automata $\mathcal{L}$ and $\mathcal{R}$, we can obtain an equivalent unambiguous transducer with underlying automaton $\mathcal{L} \times \mathcal{R}$. In particular any transduction realized by an aperiodic bimachine is aperiodic.

Proof of Proposition 36. Let $\mathcal{B} = (\mathcal{L}, \mathcal{R}, i, o)$ be a bimachine with $\mathcal{L} = (Q_\mathcal{L}, \Delta_\mathcal{L}, \{l_0\})$

and $\mathcal{R} = (Q_\mathcal{R}, \Delta_\mathcal{R}, I, F)$ as left and right automata. We define $\mathcal{T} = (\mathcal{A}, o', i')$ with $\mathcal{A} = (Q_\mathcal{L} \times Q_\mathcal{R}, \Delta, \{l_0\} \times I, F')$ with:

- $\Delta = \{((l, r'), a, (l', r)) \mid (l, a, l') \in \Delta_\mathcal{L},\ (r', a, r) \in \Delta_\mathcal{R}\}$
- $o'((l, r'), a, (l', r)) = o(l, a, r)$
- $i'(l_0, r) = i(r)$
- $F' = \bigcup_{P_\mathcal{L} \subseteq Q_\mathcal{L}, P_\mathcal{R} \in F} P_\mathcal{L} \times P_\mathcal{R}$

Let us assume that $\mathcal{B}$ is aperiodic. Since the underlying automaton of $\mathcal{T}$ is the product of two aperiodic automata, $\mathcal{T}$ is also aperiodic. ◂

C.2 Left minimization

Proof that $\mathcal{B}_f^\mathcal{R}$ is well-defined. Let us show that $\mathcal{B}_f^\mathcal{R}$ is well-defined, meaning that 1) $\sim_f^\mathcal{R}$ is indeed a right congruence and 2) the output functions do not depend on the choice of representatives. Let $u \sim_f^\mathcal{R} v$ and let $a \in A$ and let us assume that there exists x such that $uax \in \mathrm{dom}(f)$, we want to show that $ua \sim_f^\mathcal{R} va$. If $\widehat{f}_{ax}(u)$ and $\widehat{f}_{ax}(v)$ are both infinite with the same ultimate period then it is also the case for $\widehat{f}_x(ua)$ and $\widehat{f}_x(va)$. For any such x we have $\widehat{f}_x(u)^{-1} f(ux) = \widehat{f}_x(v)^{-1} f(vx)$, and let $g(x)$ denote this word. Note that $f(ux) = \widehat{f}_x(u)g(x)$. If $\bigwedge_{y \approx_\mathcal{R} x} g(ay)$ is infinite then we have both $\widehat{f}_x(ua)$ and $\widehat{f}_x(va)$ infinite with the same ultimate period. Otherwise:

$$\begin{aligned}
\widehat{f}_x(ua)^{-1} f(uax) &= \left(\bigwedge_{y \approx_\mathcal{R} x} f(uay)\right)^{-1} f(uax) \\
&= \left(\bigwedge_{y \approx_\mathcal{R} x} \widehat{f}_{ax}(u) g(ay)\right)^{-1} f(uax) \\
&= \left(\widehat{f}_{ax}(u) \bigwedge_{y \approx_\mathcal{R} x} g(ay)\right)^{-1} f(uax) \\
&= \left(\bigwedge_{y \approx_\mathcal{R} x} g(ay)\right)^{-1} \widehat{f}_{ax}(u)^{-1} f(uax) \\
&= \left(\bigwedge_{y \approx_\mathcal{R} x} g(ay)\right)^{-1} \widehat{f}_{ax}(v)^{-1} f(vax) \\
&= \widehat{f}_x(va)^{-1} f(vax)
\end{aligned}$$

Now we have left to show that the outputs are well-defined. If $u \sim_f^\mathcal{R} v$ and $\widehat{f}_x(u) = \alpha\beta^\omega$ with $\beta \in \mathcal{B}^+$ then $\widehat{f}_x(v) = \alpha'\beta^\omega$ since both are in normal form, thus the output is uniquely defined. Otherwise it suffices to show that $\widehat{f}_{ax}(u)^{-1}\widehat{f}_x(ua) = \widehat{f}_{ax}(v)^{-1}\widehat{f}_x(va)$ have well-defined outputs.

$$\begin{aligned}
\widehat{f}_{ax}(u)^{-1}\widehat{f}_x(ua) &= \widehat{f}_{ax}(u)^{-1} \bigwedge_{y \approx_\mathcal{R} x} f(uay) \\
&= \widehat{f}_{ax}(u)^{-1} \bigwedge_{y \approx_\mathcal{R} x} \widehat{f}_{ax}(u) g(ay) \\
&= \widehat{f}_{ax}(u)^{-1} \widehat{f}_{ax}(u) \bigwedge_{y \approx_\mathcal{R} x} g(ay) \\
&= \bigwedge_{y \approx_\mathcal{R} x} g(ay) \\
&= \widehat{f}_{ax}(v)^{-1} \widehat{f}_x(va)
\end{aligned}$$

◂

▸ **Proposition 37.** For any transduction f and right automaton $\mathcal{R}$ recognizing $\mathrm{dom}(f)$, the transducer $\mathcal{B}_f^\mathcal{R}$ realizes $\overline{f}^\mathcal{R}$.

Proof. Let f be a transduction, let $\mathcal{R}$ be a right automaton recognizing $\mathrm{dom}(f)$ and let x be a word in $\mathrm{dom}(f)$. We denote by g transduction realized by $\mathcal{B}_f^\mathcal{R}$. First let us assume that for any integer n, $\widehat{f}_{x(n+1:)}(x(:n))$ is finite. Then in $\mathcal{L}_f$, we have by definition that $o([\epsilon], x(:n), [x(n+1:)]^\mathcal{R}) = \widehat{f}_{x(n+1:)}(x(:n))$. Thus $g(x) = \lim_n \widehat{f}_{x(n+1:)}(x(:n)) = \overline{f}^\mathcal{R}(x)$.

Now let us assume that for some integer $k \geqslant 0$ we have $\widehat{f}_{x(k+1:)}(x(:k)) = \alpha\beta^\omega$ with $\beta \neq \epsilon$, and let us choose k minimal. By definition, we have in $\mathcal{B}_f^{\mathcal{R}}$ that $o([\epsilon], x(:k), [x(k+1:)]^{\mathcal{R}}) = \gamma$ with $i([x]^{\mathcal{R}})\gamma = \alpha\beta^l$ for some integer l. We also have for $n \geqslant k$, $o([x(:n)], x(n+1), [x(n+2:)]^{\mathcal{R}}) = \beta$, hence $i([\epsilon])o([\epsilon], x) = \widehat{f}_{x(k+1:)}(x(:k)) = \alpha\beta^\omega$. ◂

▸ **Proposition 38.** If there exists a bimachine with right automaton $\mathcal{R}$ realizing a transduction f, then $\overline{f}^{\mathcal{R}} = f$.

Proof. Let f be a transduction realized by a bimachine $\mathcal{B} = (\mathcal{L}, \mathcal{R}, i, o)$ with automata $\mathcal{L} = (Q_{\mathcal{L}}, \Delta_{\mathcal{L}}, \{l_0\})$ and $\mathcal{R} = (Q_{\mathcal{R}}, \Delta_{\mathcal{R}}, I, F)$, and let $x \in \mathrm{dom}(f)$. If $f(x)$ is finite, then there exists an integer k such that $o(l_0, x(:k), [x(k+1:)]^{\mathcal{R}}) = f(x)$. Abusing notations, we write $[u]$ for the state of $\mathcal{L}$ reached by reading u from the initial state. Hence for any $n \geqslant k$ we have $\widehat{f}_{x(n+1:)}(x(:n)) = f(x)$ which means that $\overline{f}^{\mathcal{R}}(x) = f(x)$. Otherwise, if $f(x)$ is infinite, then there must be an increasing sequence of indices $n_1, n_2, \ldots$ such that $o(l_0, x(:n_i), [x(n_i+1:)]^{\mathcal{R}}) = \alpha_i$ with α_i a strict prefix of α_{i+1}. In particular, $\alpha_i \preceq \widehat{f}_{x(n_i+1:)}(x(:n_i))$ and since $f(x) = \lim_i \alpha_i$ is infinite, we have that $\overline{f}^{\mathcal{R}}(x) = \lim_i \widehat{f}_{x(n_i+1:)}(x(:n_i))$ is also infinite and the two words are equal. ◂

Together with Proposition 38, the next proposition shows that $\mathcal{B}_f^{\mathcal{R}}$ is the bimachine with smallest left automaton under all bimachines with right automaton $\mathcal{R}$ realizing $f^{\mathcal{R}}$ (if any).

▸ **Proposition 39.** Let f be a transduction. For any bimachine $\mathcal{B} = (\mathcal{L}, \mathcal{R}, i, o)$ realizing f we have $\sim_{\mathcal{L}} \sqsubseteq \sim_f^{\mathcal{R}}$.

Proof. Let $\mathcal{L} = (Q_{\mathcal{L}}, \Delta_{\mathcal{L}}, \{l_0\})$ and $\mathcal{R} = (Q_{\mathcal{R}}, \Delta_{\mathcal{R}}, I, F)$. We suppose that $u, v \in A^*$ reach the same state in $\mathcal{L}$, and we want to show that $u \sim_f^{\mathcal{R}} v$. First, we observe that $ux \in \mathrm{dom}(f) \Leftrightarrow vx \in \mathrm{dom}(f)$ for all $x \in A^\omega$. Let $x \in A^\omega$ with $ux \in \mathrm{dom}(f)$, and $\alpha\beta^\omega = \widehat{f}_x(u)$ in normal form. If $\beta \neq \epsilon$ then $\widehat{f}_x(u)$ and $\widehat{f}_x(v)$ have the same ultimate period, which is β. If $\beta = \epsilon$ then define μ as the output of the bimachine on ux after reading u. Similarly, define ν as the output of the bimachine on ux after reading v. Observe that $\widehat{f}_x(u) = \mu\alpha$ and $\widehat{f}_x(v) = \nu\alpha$, hence with γ denoting the output of the bimachine on ux after reading u (or on vx after reading v):

$$\widehat{f}_x(u)^{-1} f(ux) = (\mu\alpha)^{-1} \mu\gamma = \alpha^{-1}\gamma = \widehat{f}_x(v)^{-1} f(vx)$$

Thus we obtain $u \sim_f^{\mathcal{R}} v$ which concludes the proof. ◂

▸ **Proposition 40.** There is a PTime algorithm that computes for a given bimachine $\mathcal{B} = (\mathcal{L}, \mathcal{R}, i, o)$ the bimachine $\mathcal{B}_f^{\mathcal{R}}$.

Proof of Proposition 40. The proof goes along the lines of Proposition 8. By Proposition 39 we check whether $u \sim_f^{\mathcal{R}} v$ for suitably chosen $u \not\sim_{\mathcal{L}} v$. For the domain condition it suffices to find some $x \in A^\omega$ such that $ux \in [\![\mathcal{R}]\!]$ and $vx \notin [\![R]\!]$. This can be verified in PTime, by searching over all initial/non-initial pairs of states of $\mathcal{R}$. For the second condition we compute for every state r of the right automaton $\mathcal{R}$ the function $\widehat{f}_x$, with x accepted from state r. We proceed as in the proof of Proposition 8, but this time we are interested only in paths of $\mathcal{L} \times \mathcal{R}$ that correspond to uy, with $y \approx_{\mathcal{R}} x$, so y accepted from r as well. ◂

C.3 From transducers to bimachines

Proof of Theorem 15. Let $\mathcal{T} = (\mathcal{A}, i, o)$ with $\mathcal{A} = (Q, \Delta, I, F)$ be a transducer realizing a function f, and $\mathcal{R}$ such that $\approx_{\mathcal{R}} \sqsubseteq \approx_{\mathcal{A}}$. We show first that $\sim_{\mathcal{A}} \sqsubseteq \sim_f^{\mathcal{R}}$. Let $u \sim_{\mathcal{A}} v$, since $\mathcal{A}$ recognizes $\mathrm{dom}(f)$ and is trim, we have for any $x \in A^\omega$, $ux \in \overline{\mathrm{dom}(f)} \Leftrightarrow vx \in \overline{\mathrm{dom}(f)}$.

Let x be a word such that $ux \in \mathrm{dom}(f)$, and let $p_1, \ldots, p_n$ be the states of $\mathcal{A}$ which can be reached from I by reading u or v and from which x has a final run. We write $q_i \xrightarrow{u|\alpha_i} p_i$, $q_i' \xrightarrow{v|\beta_i} p_i$, corresponding to initial runs with $q_i, q_i' \in I$ for $1 \leqslant i \leqslant n$. Let $\gamma_i = \bigwedge \left\{ \gamma \mid p_i \xrightarrow{y|\gamma} P_i \text{ with } P_i \in F,\ y \approx_{\mathcal{R}} x \right\}$. For any word $y \approx_{\mathcal{R}} x$ we have also $y \approx_{\mathcal{A}} x$, and in particular, $uy \in \mathrm{dom}(f)$. By definition, $\widehat{f}_x(u) = i(q_i)\alpha_i\gamma_i'$, and $\widehat{f}_x(v) = i(q_i')\beta_i\gamma_i'$, for some $\gamma_i' \prec \gamma_i$. If $\widehat{f}_x(u)$ is infinite, then so are all γ_i and $\widehat{f}_x(v)$. Moreover, $\widehat{f}_x(u)$ and $\widehat{f}_x(v)$ have the same ultimate period in this case.

Otherwise, some γ_i is finite, say γ_1. Let $p_1 \xrightarrow{x|\gamma_1\nu} P$ with $P \in F$ denote a final run of x from p_1. Then we have $f(ux) = i(q_1)\alpha_1\gamma_1\nu$ and $f(vx) = i(q_1')\beta_1\gamma_1\nu$, thus:

$$\begin{aligned} \widehat{f}_x(u)^{-1} f(ux) &= (i(q_1)\alpha_1\gamma_1')^{-1} i(q_1)\alpha_1\gamma_1\nu = (\gamma_1')^{-1}\gamma_1\nu \\ &= (i(q_1')\beta_1\gamma_1')^{-1} i(q_1')\beta_1\gamma_1\nu = \widehat{f}_x(v)^{-1} f(vx) \end{aligned}$$

This means that $u \sim_f^{\mathcal{R}} v$.

It remains to show that $\mathcal{B}_f^{\mathcal{R}}$ realizes f. Let g be the transduction realized by $\mathcal{B}_f^{\mathcal{R}}$. First, the domain of g equals the language of $\mathcal{R}$, thus $\mathrm{dom}(f)$ because of $\approx_{\mathcal{R}} \sqsubseteq \approx_{\mathcal{A}}$.

Let $x \in \mathrm{dom}(f)$, we want to show that $g(x) = f(x)$. First we assume that $\widehat{f}_{x(n+1:)}(x(:n))$ is finite, for every n. Then by definition of $\mathcal{B}_f^{\mathcal{R}}$, we get that $g(x)$ is the limit of $(\widehat{f}_{x(n+1:)}(x(:n))_n$, so $g(x) = f(x)$ since $f(x)$ is the limit of prefixes of $\widehat{f}_{x(n+1:)}(x(:n)$. The other case is where $\widehat{f}_{x(n+1:)}(x(:n)) = \alpha\beta^\omega$ for some n, α and $\beta \neq \epsilon$. It can be checked that $o([\epsilon], x(:n), [x(n+1:])^{\mathcal{R}}) = \alpha\beta^m$ for some m, and for all $n' \geqslant n$, $o([x(:n'-1)], x(n'), [x(n'+1:])^{\mathcal{R}}) = \beta$, which shows the claim.

Let f be an aperiodic transduction and let $\mathcal{T}$ be an aperiodic transducer realizing it with underlying automaton $\mathcal{A}$. According to Proposition 1 the canonical automaton $\mathcal{R}$ of $\approx_{\mathcal{A}}$ is aperiodic and since $\equiv_{\mathcal{A}} \sqsubseteq \sim_{\mathcal{A}} \sqsubseteq \sim_f^{\mathcal{R}}$, the left automaton of $\mathcal{B}_f^{\mathcal{R}}$ is also aperiodic. ◂

C.4 Labelings and bimachines

Proof of Proposition 17. Let $\mathcal{B} = (\mathcal{L}, \mathcal{R}, i, o)$ be a bimachine realizing f with automata $\mathcal{L} = (Q_{\mathcal{L}}, \Delta_{\mathcal{L}}, \{l_0\})$ and $\mathcal{R} = (Q_{\mathcal{R}}, \Delta_{\mathcal{R}}, I, F_{\mathcal{R}})$. We define $\mathcal{T} = (\mathcal{A}, i', o')$ and $\mathcal{A} = (Q, \Delta, \{q_0\}, F)$ realizing $f_{\mathcal{R}} : (A \times Q_{\mathcal{R}})^\omega \to B^\infty$:

- $Q = Q_{\mathcal{L}} \times Q_{\mathcal{R}} \uplus \{q_0\}$
- $\Delta = \left\{ (q_0, (a, r), (l, r)) \mid l_0 \xrightarrow{a}_{\mathcal{L}} l,\ r' \xrightarrow{a}_{\mathcal{R}} r,\ r' \in I \right\}$
- $\quad \uplus \left\{ ((l, r'), (a, r), (l', r)) \mid l \xrightarrow{a}_{\mathcal{L}} l',\ r' \xrightarrow{a}_{\mathcal{R}} r \right\}$
- $F = \{P \subseteq Q \mid \pi_2(P) \in F_{\mathcal{R}}\}$, with π_2 being the projection over the second component.
- $i'(q_0) = \epsilon$
- $o'(q_0, (a, r), (l, r)) = i(r')o(l, a, r)$ with $r' \xrightarrow{a}_{\mathcal{R}} r$.

  $o'((l, r'), (a, r), (l', r)) = o(l, a, r)$.

Clearly, $\mathcal{T}$ realizes $f_{\mathcal{R}}$ and is left-sequential.

Let $\mathcal{T} = (\mathcal{A}, i, o)$ with $\mathcal{A} = (Q, \Delta, I)$ be a left-sequential transducer realizing $f_{\mathcal{R}}$ over the alphabet $A \times Q_{\mathcal{R}}$. We define $\mathcal{T}' = (\mathcal{A}', i, o')$ with $\mathcal{A}' = (Q, \Delta', I)$, as the transducer obtained by projecting the input alphabet on A. The new transducer $\mathcal{T}'$ realizes f and is unambiguous, otherwise there would be a word with two different labelings accepted by $\mathcal{A}$.

Let $\mathcal{D}$ be the left automaton obtained from the subset construction of $\mathcal{A}' \times \mathcal{R}$. We define the bimachine $\mathcal{B} = (\mathcal{D}, \mathcal{R}, i', o')$ by:

$$o'(P, a, r) = o(q, (a, r), q') \text{ for } (q, r') \in P \text{ such that } (r', a, r) \in \Delta$$

The output is well-defined since $\mathcal{A}$ is deterministic and the state q such that $(q, r') \in P$ and $(r', a, r) \in \Delta$ is unique: if there were two states q_1, q_2 with $(q_i, r') \in P$ for $i = 1, 2$, then there would be a word uy such that u reaches in $\mathcal{A}'$ both q_is and y has a final run from both q_is, which contradicts the unambiguity of $\mathcal{A}'$.

Let us assume in the first construction above that $\mathcal{B}_f^{\mathcal{R}}$ is aperiodic and we want to show that the left automaton $\mathcal{A}$ obtained is aperiodic which would imply that $f_{\mathcal{R}}$ is aperiodic. Let u be a word over $A \times Q_{\mathcal{R}}$, let $(p, [x]) \xrightarrow{u^n} (q, [y])$ denote a run over u^n, if n is large enough, we know that $p \xrightarrow{\pi_1(u)^{n+1}} q$. We also have $[\pi_1(u)^n y] = [x]$. By aperiodicity of $\mathcal{R}$, we must have $\left[\pi_1(u)^{n+1} y\right] = [x]$ and since $u(|u|) = (a, [y])$, it means that $[y] = [uy] = [x]$. Hence we obtain that $(p, [x]) \xrightarrow{u^{n+1}} (q, [y])$, which means that $f_{\mathcal{R}}$ is aperiodic. Now we assume that $f_{\mathcal{R}}$ is aperiodic, we want to show that the automaton $\mathcal{D}$ is aperiodic, and since $\sim_{\mathcal{D}} \sqsubseteq \sim_f^{\mathcal{R}}$, from Proposition 39 it implies that $\sim_f^{\mathcal{R}}$ is aperiodic as well. First we know that is $f_{\mathcal{R}}$ is sequential and aperiodic, then it is realized by a sequential aperiodic transducer. Indeed, given an aperiodic transducer realizing $f_{\mathcal{R}}$ we can obtain *via* subset construction with delays a sequential transducer realizing it which is still aperiodic according to Theorem 12. Hence we assume that $\mathcal{T}$ is aperiodic. Let us show that $\mathcal{A}' \times \mathcal{R}$ is again aperiodic. Let u be a word over A and let $(p, [x]) \xrightarrow{u^n} (q, [y])$ denote a run over u. Since $\mathcal{R}$ is aperiodic, we have for n large enough that $[x] = [u^n y] = [uy] = [y]$. Hence there is a unique labeling of u, $\tilde{u}$ which is consistent with the run. Hence there is a run of $\mathcal{A}$ $p \xrightarrow{\tilde{u}^n} q$. Since $\mathcal{A}$ is aperiodic, for n large enough, we have a run $p \xrightarrow{\tilde{u}^{n+1}} q$. By projection, we have a run $p \xrightarrow{u^{n+1}} q$ of $\mathcal{A}'$. Thus we obtain that $(p, [x]) \xrightarrow{u^n} (q, [y])$ in $\mathcal{A}' \times \mathcal{R}$. Finally since the subset construction preserves aperiodicity, we have that $\mathcal{D}$ is aperiodic. ◂

We define the *labeling function* associated with a left automaton $\mathcal{L} = (Q, \Delta, \{q_0\})$ by the left transducer $\ell(\mathcal{L}) = (\mathcal{L}, i, o)$, with $i(q_0) = \epsilon$ and $o(p, a, q) = (a, p)$. The labeling function associated with a left automaton labels each position with the information about the prefix. For a transduction f let $f_{\mathcal{L}} = f \circ [\![\ell(\mathcal{L})]\!]^{-1}$, which is again a function.

Proof of Corollary 18. Let f be a rational transduction, according to Theorem 16 there exists $\mathcal{B} = (\mathcal{L}, \mathcal{R}, i, o)$ with $\mathcal{L} = (Q_{\mathcal{L}}, \Delta_{\mathcal{L}}, \{l_0\})$ and $\mathcal{R} = (Q_{\mathcal{R}}, \Delta_{\mathcal{R}}, I, F)$, a bimachine realizing f. By definition of $\ell(\mathcal{R})$, $f = f_{\mathcal{R}} \circ [\![\ell(\mathcal{R})]\!]$ furthermore $\ell(\mathcal{R})$ is letter-to-letter and right-sequential, and according to Proposition 17, $f_{\mathcal{R}}$ is left-sequential, concluding the proof of the first direction. Symmetrically, $\ell(\mathcal{L})$ is letter-to-letter and left-sequential. We only have to exhibit a right-sequential transducer realizing $f_{\mathcal{L}}$. This is done in the same way as in the first (easy) part of the proof of Proposition 17. Let $\mathcal{T} = (\mathcal{R}', i, o')$ with $\mathcal{R}' = (Q_{\mathcal{R}}, \Delta'_{\mathcal{R}}, I, F)$ over the alphabets $A \times Q_{\mathcal{L}}$ and B.

- $\Delta' = \{(r, (a, p), s) \mid (r, a, s) \in \delta_{\mathcal{R}}\}$
- $o'(r, (a, p), s) = o(p, a, s)$

$\mathcal{R}'$ is a right-automaton since $\mathcal{R}$ is, and by construction $\mathcal{T}$ realizes $f_{\mathcal{L}}$. ◂

D Proofs for Section 4

D.1 Minimal look-ahead

Proof of Proposition 21. Let $\mathcal{T}$ be a transducer realizing $f_{\mathcal{R}}$ and let us assume that $\mathcal{T}$ satisfies the WTP. Let $x \approx_{\mathcal{R}} y$, and let u be such that $ux \in \mathrm{dom}(f)$. We want to show that $x \overset{\Delta}{\approx}_f y$, for this we will show that $\mathsf{del}(f(ux), f(uy))$ is bounded by a value which does not depend on u. Since $x \approx_{\mathcal{R}} y$ we can write $u'x' = [\![\ell(\mathcal{R})]\!](ux)$ and $u'y' = [\![\ell(\mathcal{R})]\!](uy)$ with $|u'| = |u|$. Let k be the number of states of $\mathcal{T}$. We consider two runs of $\mathcal{T}$ over $u'x'$ and $u'y'$, respectively, and we factorize them as $p_1 \xrightarrow{u_1|\alpha_1} p_2 \xrightarrow{u_2|\alpha_2} p_2 \xrightarrow{u_3|\alpha_3} p_3 \xrightarrow{x'|\alpha_4} F$ and $q_1 \xrightarrow{u_1|\beta_1} q_2 \xrightarrow{u_2|\beta_2} q_2 \xrightarrow{u_3|\beta_3} q_3 \xrightarrow{y'|\beta_4} F$ such that $u' = u_1u_2u_3$, and $|u_1|, |u_3| \leqslant k^2$.

We get $\mathsf{del}(f(ux), f(uy)) = \mathsf{del}(f_{\mathcal{R}}(u'x'), f_{\mathcal{R}}(u'y')) = \mathsf{del}(i(p_1)\alpha_1\alpha_2\alpha_3\alpha_4, i(q_1)\beta_1\beta_2\beta_3\beta_4)$. If p_2 and q_2 are both non-constant, we use the WTP and obtain:

$$\mathsf{del}(i(p_1)\alpha_1\alpha_2\alpha_3\alpha_4, i(q_1)\beta_1\beta_2\beta_3\beta_4) = \mathsf{del}(i(p_1)\alpha_1\alpha_3\alpha_4, i(q_1)\beta_1\beta_3\beta_4)$$

Assume now that p_2 is not constant and q_2 is constant. The output of q_2 is a regular word γ, with $\gamma = \beta_2^\omega$ if $\beta_2 \neq \epsilon$, so in particular $\gamma = \beta_2\gamma$. Thus, $\mathsf{del}(f(ux), f(uy)) = \mathsf{del}(i(p_1)\alpha_1\alpha_2\alpha_3\alpha_4, i(q_1)\beta_1\gamma)$. If $\alpha_2 = \epsilon$ then $\mathsf{del}(f(ux), f(uy)) = \mathsf{del}(i(p_1)\alpha_1\alpha_3\alpha_4, i(q_1)\beta_1\gamma)$. Otherwise, by the WTP we know that $i(p_1)\alpha_1\alpha_2^\omega = i(p_2)\beta_1\beta_2\gamma$. It can be checked that either $\mathsf{del}(f(ux), f(uy)) = \mathsf{del}(\alpha_3\alpha_4, \gamma')$, with γ' a suffix of γ, or $\mathsf{del}(f(ux), f(uy)) = \mathsf{del}(\alpha_3\alpha_4, \beta_1\gamma)$, depending on the lengths of $i(p_1)\alpha_1\alpha_2$, $i(p_2)\beta_1\beta_2$.

If p_2 and q_2 are both constant, then their output words are both regular, say γ and γ'. We have $i(p_1)\alpha_1\alpha_2\alpha_3\alpha_4 = i(p_1)\alpha_1\gamma$ and $i(q_1)\beta_1\beta_2\beta_3\beta_4 = i(p_2)\beta_1\gamma'$, so $\mathsf{del}(f(ux), f(uy)) = \mathsf{del}(i(p_1)\alpha_1\gamma, i(p_2)\beta_1\gamma')$, and the delay depends only on $i(p_1)\alpha_1$, $i(p_2)\beta_1$ and the normal forms of γ, γ'. In all cases the delay $\mathsf{del}(f(ux), f(uy))$ is independent of α_2, β_2.

Let us assume now that $\approx_{\mathcal{R}} \sqsubseteq \overset{\Delta}{\approx}_f$ and show that $\mathcal{T}$ satisfies WTP. For this we consider two initial runs of $\mathcal{T}$ over the same finite word: $p_1 \xrightarrow{u|\alpha_1} q_1 \xrightarrow{v|\beta_1} q_1$ and $p_2 \xrightarrow{u|\alpha_2} q_2 \xrightarrow{v|\beta_2} q_2$. Up to taking $u' = uv$, we can assume that the last letter of u and v is the same. Let (a, r) be the last letter of v, and let us consider two words x_1, x_2 which have final runs from q_1 and q_2, respectively with $q_1 \xrightarrow{x_1|\gamma_1} F$ and $q_2 \xrightarrow{x_2|\gamma_2} F$. Let π be the projection $(A \times Q_{\mathcal{R}})^\infty \to A^\infty$ such that $\pi \circ \ell(\mathcal{R}) = \mathrm{id}$. By definition of $\ell(\mathcal{R})$ we must have $\pi(x_1) \approx_{\mathcal{R}} \pi(x_2)$ and, since $\mathcal{R}$ recognizes $\mathrm{dom}(f)$, both $\pi(x_1), \pi(x_2)$ have a final run of $\mathcal{R}$ from r. For q_1, q_2 non-constant, we assume towards a contradiction that $\mathsf{del}(i(p_1)\alpha_1, i(p_2)\alpha_2) \neq \mathsf{del}(i(p_1)\alpha_1\beta_1, i(p_2)\alpha_2\beta_2)$. Since q_1 and q_2 are non-constant, we can choose x_1 and x_2 such that $\gamma_1 \neq \beta_1^\omega$ and $\gamma_2 \neq \beta_2^\omega$. Thus for infinitely many $m \neq n$, we have $\mathsf{del}(i(p_1)\alpha_1\beta_1^m\gamma_1, i(p_2)\alpha_2\beta_2^m\gamma_2) \neq \mathsf{del}(i(p_1)\alpha_1\beta_1^n\gamma_1, i(p_2)\alpha_2\beta_2^n\gamma_2)$, which means that $\mathsf{del}(f_{\mathcal{R}}(uv^m x_1), f_{\mathcal{R}}(uv^m x_2)) \neq \mathsf{del}(f_{\mathcal{R}}(uv^n x_1), f_{\mathcal{R}}(uv^n x_2))$. Hence, $\pi(x_1), \pi(x_2)$ are not equivalent with respect to $\overset{\Delta}{\approx}_f$, hence by assumption, $\pi(x_1) \not\approx_{\mathcal{R}} \pi(x_2)$ which is a contradiction.

Similarly, let us we assume that q_1 is not constant, q_2 is constant producing γ, $v_1 \neq \epsilon$ and $i(p_1)\alpha_1\beta_1^\omega \neq i(p_2)\alpha_2\beta_2\gamma$. Again, by choosing x_1 such that $\gamma_1 \neq \beta_1^\omega$, we obtain an infinite number of delays and $\pi(x_1) \not\approx_{\mathcal{R}} \pi(x_2)$.

If f is aperiodic, it means that there is an aperiodic bimachine realizing it, with an aperiodic right automaton $\mathcal{R}$. According to Proposition 17 we have that $f_{\mathcal{R}}$ is aperiodic, and hence $\approx_{\mathcal{R}} \sqsubseteq \overset{\Delta}{\approx}_f$. Since $\approx_{\mathcal{R}}$ is aperiodic (recognized by $\mathcal{R}$) we have that $\overset{\Delta}{\approx}$ is also aperiodic in particular. ◂

Proof of Proposition 22. We show the result for a transducer, the proof being very similar

in the case of a bimachine. Let $\mathcal{T} = (\mathcal{A}, i, o)$ with $\mathcal{A} = (Q, \Delta, I, F)$ be a transducer realizing a function f. Let $x \approx_{\mathcal{A}} y$, and let us consider the states $p_1, \ldots, p_n$ from which x, y have final runs in $\mathcal{A}$, denoted by $p_i \xrightarrow{x|\alpha_i} P_i$ and $p_i \xrightarrow{y|\beta_i} P'_i$, respectively. Choose some i and some initial run $q_i \xrightarrow{u|\gamma_i} p_i$ of $\mathcal{T}$ over u, and let $\gamma'_i = i(q_i)\gamma_i$. Let $\delta = f(ux) \wedge f(uy)$, and $\delta_i = \alpha_i \wedge \beta_i$, then $\delta = \gamma'_i\delta_i$. Clearly, $(\delta^{-1}f(ux), \delta^{-1}f(uy)) = (\delta_i^{-1}\alpha_i, \delta_i^{-1}\beta_i)$ does not depend on u. Thus $|\mathsf{del}_f(x, y)| \leqslant n$, showing that $x \overset{\Delta}{\approx}_f y$. ◂

▸ **Proposition 41.** Given a bimachine realizing a function f, the delay congruence $\overset{\Delta}{\approx}_f$ can be computed in PTime.

Proof. Let $\mathcal{B} = (\mathcal{L}, \mathcal{R}, o, i)$ be a bimachine. We know from Proposition 22, that the right automaton of the bimachine, $\mathcal{R}$, satisfies $\approx_{\mathcal{R}} \sqsubseteq \overset{\Delta}{\approx}_f$. In order to compute $\overset{\Delta}{\approx}_f$, we only need to decide, given two words x, y such that $x \not\approx_{\mathcal{R}} y$ if $x \overset{\Delta}{\approx}_f y$. We want to decide if the set of delays between $f(ux)$ and $f(uy)$, for all $u \in A^*$ is finite. Let us fix a state of $\mathcal{L}$, $[u]^{\mathcal{L}}$, and one can easily see that the set of delays is finite if and only if it is finite for all possible states of $\mathcal{L}$. From the state $[u]^{\mathcal{L}}$, we can define the output of x and y denoted by α, β, respectively. We define the function $g_x(v)$, for $v \sim_{\mathcal{L}} u$, by the output of the bimachine due to v upon reading vx, *i.e.* $g_x(v) = i([vx]^{\mathcal{R}})o([\epsilon]^{\mathcal{L}}, v, [x]^{\mathcal{R}})$ and similarly we define g_y, such that $f(vx) = g_x(v)\alpha$ and $f(vy) = g_y(v)\beta$. Let us assume that $|\{\mathsf{del}(f(vx), f(vy)) \mid v \sim_{\mathcal{L}} u\}| = |\{\mathsf{del}(g_x(v), g_y(v)) \mid v \sim_{\mathcal{L}} u\}|$. In that case we have reduced the problem of deciding a finite set of delays for a function over infinite words to functions over finite words. Indeed we can obtain a bimachine over finite words realizing g_x (and g_y) by taking a bimachine where the left automaton is $\mathcal{L}$ with final state $[u]^{\mathcal{L}}$ and the right automaton is $\mathcal{R}$ with (unique) final state $[x]^{\mathcal{R}}$. We refer the reader to [14] for an article dealing with bimachines over finite words and we use the result of [21] (Proposition 1) which gives a PTime algorithm to decide if two functions are so-called *adjacent*, *i.e.* have a finite set of delays. We only have to figure out when it is true that $|\{\mathsf{del}(f(vx), f(vy)) \mid v \sim_{\mathcal{L}} u\}| = |\{\mathsf{del}(g_x(v), g_y(v)) \mid v \sim_{\mathcal{L}} u\}|$ and what to do when it is not the case. The property is necessarily verified if none of the two words α, β is periodic. When one of the two words α, β (for instance α) is of the form γ^ω, $\gamma \neq \epsilon$, we can transform the bimachine realizing g_x such that γ is never a suffix of the image of a word by guessing at any point that the output will be of the form γ^n, outputting ϵ and checking that the output is indeed a power of γ. We obtain a new function g'_x such that for any v, $g_x(v) = g'_x(v)\gamma^n$ with $n \in \mathbb{N}$ and γ not a suffix of $g'_x(v)$, while still having $f(vx) = g'(v)\alpha$. We also have to modify g_y such that, as long as no mismatch has been found and if g'_x has stopped producing outputs, then all outputs of the form γ^n can be removed. Similarly, if β is periodic of the form δ^ω then we can modify the bimachine realizing g_y to obtain a transducer such that δ is not a suffix of the outputs. For these modified functions, we have indeed that $|\{\mathsf{del}(f(vx), f(vy)) \mid v \sim_{\mathcal{L}} u\}| = |\{\mathsf{del}(g'_x(v), g'_y(v)) \mid v \sim_{\mathcal{L}} u\}|$, and we can again use the result of [21]. ◂

D.2 Canonical machine for quasi-sequential functions

▸ **Lemma 42.** $\breve{\approx}_f$ *is a left congruence.*

Proof. Let $x \breve{\approx}_f y$ and let $a \in A$. We have of course $ax \in \mathrm{dom}(f) \Leftrightarrow ay \in \mathrm{dom}(f)$ and for all u such that $uax \in \mathrm{dom}(f)$ we have $\widehat{f}(ua) = \overline{f}(uax) \Leftrightarrow \widehat{f}(ua) = \overline{f}(uay)$. Either $\widehat{f}(u) = \widehat{f}(ua)$ and we have indeed $\widehat{f}(u) = \overline{f}(uax) \Leftrightarrow \widehat{f}(u) = \overline{f}(uay)$, or $\widehat{f}(u) \prec \widehat{f}(ua)$ and we have $\widehat{f}(u) \neq \overline{f}(uax)$ and $\widehat{f}(u) \neq \overline{f}(uay)$ which means that condition 1) is satisfied. Let us now

assume $\widehat{f}(u) = \overline{f}(uax)$, then in particular $\widehat{f}(ua) = \overline{f}(uax)$ and we have $f(uax) = f(uay)$, which means that condition 2) is satisfied as well, hence $ax \breve{\approx}_f ay$. ◂

Proof that $\mathcal{U}_f^{\mathcal{R}}$ is well-defined. Let us show that the output function is well-defined, meaning that it does not depend on the representatives of the congruence classes. Let $a \in \mathcal{A}$, $u \sim_f v$ and $x \sim_{\mathcal{R}} y$. Let us first show that $o_{\mathcal{R}}([u], a, [x]_{\mathcal{R}}) = o_{\mathcal{R}}([v], a, [x]_{\mathcal{R}})$. By definition, if $u \sim_f v$ then $\widehat{f}(u)^{-1} f(uax) = \widehat{f}(v)^{-1} f(vax)$ and as we have seen in Section 2, $\widehat{f}(u)^{-1}\widehat{f}(ua) = \widehat{f}(v)^{-1}\widehat{f}(va)$ and $\widehat{f}(u)^{-1}\overline{f}(uax) = \widehat{f}(v)^{-1}\overline{f}(vax)$. Then it is routine to check that $o_{\mathcal{R}}([u], a, [x]_{\mathcal{R}}) = o_{\mathcal{R}}([v], a, [x]_{\mathcal{R}})$ in all cases. Let us now show that $\widehat{f}(u)^{-1} f(uax) = \widehat{f}(u)^{-1} f(uay)$. Since $\mathcal{R}$ recognizes $\breve{\approx}_f$, we have $x \breve{\approx}_f y$, hence $\widehat{f}(u) = \overline{f}(uax) \Leftrightarrow \widehat{f}(u) = \overline{f}(uay)$, and if $\widehat{f}(u) = \overline{f}(uax)$, then $f(ux) = f(uy)$, which means that $\widehat{f}(u)^{-1} f(uax) = \widehat{f}(u)^{-1} f(uay)$. In all cases we have $\widehat{f}(u)^{-1} f(uax) = \widehat{f}(u)^{-1} f(uay)$. ◂

Proof of Lemma 26. Let $x \in \mathrm{dom}(f)$ and let g denote the function realized by $\mathcal{U}_f^R$. Let us first assume that for any prefix u of x we have $\widehat{f}(u) \prec \overline{f}(x)$. Then since we have that $\lim_u \widehat{f}(u) = \overline{f}(x)$ it means that $\overline{f}(x)$ is infinite and thus $\overline{f}(x) = f(x)$. In that case $\mathcal{U}_f^{\mathcal{R}}$ behaves just like $\mathcal{T}_f$ and we have $g(x) = \overline{f}(x) = f(x)$. Now let us assume that at some point $\widehat{f}(u) \prec \widehat{f}(ua) = \overline{f}(uay)$ for $uay = x$. The output of $\mathcal{T}_f$ over uay after having read ua is equal to $\widehat{f}(u)\alpha$ such that $f(uay) = \widehat{f}(u)\alpha\beta^\omega$ (in normal form). The output of any letter after ua will be β and in the end we obtain $g(x) = f(x)$. ◂

Proof of Corollary 27. By Lemma 26, we know that there exists a bimachine realizing f with $\mathcal{R}$ as right automaton. Hence Props. 37 and 38 imply that $\mathcal{B}_f^{\mathcal{R}}$ realizes f. Furthermore, from Proposition 39 we know that $\sim_f \sqsubseteq \sim_f^{\mathcal{R}}$. ◂

Proof of Proposition 25. From a bimachine $\mathcal{B}$ with automata $\mathcal{L}$ and $\mathcal{R}$ we define a left congruence $\approx$ and show that it is finer than $\breve{\approx}_f$. Let $\mathcal{S}$ be the transducer obtained by subset construction with delays from the transducer with underlying automaton $\mathcal{L} \times \mathcal{R}$ (Proposition 36). Given two words, x, y we let $x \approx y$ if 1) for any state (p, P) of $\mathcal{R} \times \mathcal{S}$, the runs over x, y from (p, P) have the same set of states visited infinitely often, and 2) for any state P of $\mathcal{S}$, the run of $\mathcal{S}$ over x from P produces ϵ if and only if the run of $\mathcal{S}$ from R over y produces ϵ.

Now we show that $\approx$ is finer than $\breve{\approx}_f$, which shows that the index of $\breve{\approx}_f$ is doubly exponential. Let $x \approx y$, and let u be a finite word with an initial run over $\mathcal{S}$: $I \xrightarrow{u|\alpha} P$ and by construction we have $i(I)\alpha = \widehat{f}(u)$. Since by Proposition 35, $\mathcal{S}$ realizes $\overline{f}$ and 2) we know that $\widehat{f}(u) = \overline{f}(ux)$ if and only if $\widehat{f}(u) = \overline{f}(uy)$. Now assume $\widehat{f}(u) = \overline{f}(ux)$, then from 1) there is (p, P') a state appearing infinitely often in the runs r, s of x and y respectively from P. Let i, j be such that $r(i) = s(j) = (p, P')$, in particular we have $x(i{:}) \approx_{\mathcal{A}} y(j{:})$. Then let $(q, w) \in P'$ such that both $x(i{:})$ and $y(j{:})$ have a final run from q in $\mathcal{A}$. Then we have $f(ux) = i(I)\alpha w = f(uy)$ hence $x \breve{\approx}_f y$.

Now we only have to decide when two words x, y are equivalent for $\breve{\approx}_f$. For each word x and each state R of $\mathcal{S}$, we can associate a value $\mathrm{rest}(x, R)$ which is either a word w, if x produces ϵ from R and the missing output is w or $\perp$ otherwise. Since $\mathcal{S}$ realizes $\overline{f}$ we have that $x \breve{\approx}_f y$ if and only if $\forall u \in A^*,\ \forall R$ state of $\mathcal{S}$, $\mathrm{rest}(ux, R) = \mathrm{rest}(uy, R)$.

We now show that from an aperiodic bimachine, we obtain an aperiodic congruence. We only need to show that the congruence $\approx$ computed is aperiodic. From Theorem 12 we know that $\mathcal{S}$ is aperiodic. From this we know that for any R, S states of $\mathcal{S}$, the languages $L_{R,S} \subseteq A^*$ of words which can go from R, S and $L_S \subseteq A^\omega$ of words which have a run from S are aperiodic. Then since aperiodic languages are closed under concatenation, if we have a

transition $R \xrightarrow{a|w} T$ (with $w \neq \epsilon$), we can define the language of words which have a run from R and which don't go though the transition by $L_R \cap (L_{R,S} \cdot a \cdot L_S)^C$, which is aperiodic since aperiodic languages are also closed under complement. Then if we fix a state R of $\mathcal{S}$ as initial state we can define the set of words which have a run from R and which don't go through any producing transition which is aperiodic since aperiodic languages are closed under union. Now we can fix a state and any accepting set as Muller condition for the automaton $\mathcal{R} \times \mathcal{S}$ and get an aperiodic language. Hence any class of $\approx$ is aperiodic which means that $\breve{\approx}$ is also aperiodic. ◂

E Proofs for Section 5

Our goal is to prove the transducer-logic correspondence of Theorem 30. We start by a result on languages. It is known that a language $L \subseteq A^\omega$ is FO-definable if and only if it is recognized by some aperiodic non-deterministic Büchi automaton [10]. With simple arguments and by using a result by Thomas [25], it is possible to lift this result to aperiodic deterministic Muller automata:

▸ **Theorem 43** ([25]). *A language $L \subseteq A^\omega$ is FO-definable if and only if it is recognizable by some aperiodic and deterministic Muller automaton.*

Proof. Thomas has shown that L is FO-definable if and only if it is recognizable by some counter-free and deterministic Rabin automaton [25]. Rabin automata are a particular case of Muller automata, and therefore L is FO-definable if and only if it is recognizable by some counter-free and deterministic Muller automaton. Counter-freeness means that for some m, $u^m \in L_{q,q}$ (the set of words for which there exists a run from state q to q) if and only if $u \in L_{q,q}$, for all states q and finite words u. It is not difficult to prove that any counter-free automaton is aperiodic, and conversely any deterministic and aperiodic automaton is counter-free, see for instance Lemma 11.6 in [10]. Hence, any deterministic Muller automaton is counter-free if and only if it is aperiodic, and we get the desired result. ◂

▸ **Proposition 44.** A language $L \subseteq A^\omega$ is FO-definable if and only if it is recognizable by some aperiodic and non-deterministic Muller automaton.

Proof. Direction $\Rightarrow$ is a consequence of Theorem 43. For the other direction, from an aperiodic and non-deterministic Muller automaton $\mathcal{A} = (Q, \Delta, I, F)$ we construct an equivalent FO-formula. For all $P \subseteq Q$, we let $L_P \subseteq A^*$ be the language of finite words such that there exists a run of $\mathcal{A}$ which visits at least once the states of P and only those ones. The language L_P can be recognized by some aperiodic finite automaton A_P with set of states $Q' = \{(q, P') \mid q \in Q, P' \subseteq P\}$, initial states $I' = \{(q, P) \mid q \in P\}$ and accepting states $F' = \{(q, \varnothing) \mid q \in Q\}$. Its transitions are $(p, P') \xrightarrow{a} (q, P' \backslash \{p\})$ if there exists a transitions $p \xrightarrow{a}_{\mathcal{A}} q$ and $q \in P'$. The automaton A_P is easily seen to be aperiodic, since $\mathcal{A}$ is aperiodic and the second component of the states of $\mathcal{A}_P$ is monotonic (for inclusion) along the runs. Hence, L_P is FO-definable by some formula ϕ_P. Then, $L(\mathcal{A})$ is FO-definable by the formula

$$\bigvee_{P \in F} \exists x_0 \forall x \geq x_0 \exists y \geq x \ \phi_P(x, y)$$

where $\phi_P(x, y)$ is the formula ϕ_P where the quantifiers have been restricted to range between x and y, *i.e.*, quantifiers Qz are replaced by $Q(x \leqslant z \leqslant y)$ (x, y can be assumed to have no occurrence in ϕ_P without loss of generality). This concludes the proof. ◂

▸ **Proposition 45.** Let $f : A^\omega \to B^\infty$. If f is realizable by some transducer $\mathcal{T}$, then it is realizable by some unambiguous transducer $\mathcal{T}'$ such that additionally, if $\mathcal{T}$ is aperiodic, so is $\mathcal{T}'$.

Proof. If f is realizable by some aperiodic transducer $\mathcal{T}$, then by Theorem 16 it is realizable by some aperiodic bimachine $\mathcal{B}$. Since the construction of a transducer from a bimachine (Proposition 36) is done by a standard product construction of its left and right automata and aperiodicity is preserved under automata product, we get the result. ◂

Proof of Theorem 30. We first show the equivalence between transducers and MSO-definability, and then analyze our back and forth constructions to show the equivalence between FO-definability and aperiodic transducers.

(1) Let $\mathcal{F} = (A, B, \phi_{dom}, V, \mu)$ be some MSO-transducer such that $[\![\mathcal{F}]\!] = f$. Let construct some transducer $\mathcal{T}$ equivalent to $\mathcal{F}$, *i.e.*, such that $[\![\mathcal{T}]\!] = [\![\mathcal{F}]\!]$. Let $V = \{v_1, \dots, v_n\}$ and for all $i \in \{1, \dots, n\}$, let $\phi_i(x) = \phi_{v_i}(x)$. Given two ω-words u, v over Σ and Γ respectively, we define $u \otimes v$ as the ω-word over $\Sigma \times \Gamma$ defined by $(u \otimes v)(i) = (u(i), v(i))$ for all $i \geqslant 1$.

We now define the language of ω-words $L_{\mathcal{F}} \subseteq (A \times \{0,1\}^n)^\omega$ as the set of ω-words $u \otimes u_1 \otimes \cdots \otimes u_n$ such that $u \models \phi_{dom}$ and such that for all i, $u_i \in \{0,1\}^\omega$ and for all $j \geqslant 1$, $u_i(j) = 1$ if and only if $u \models \phi_i(j)$. Let us show that $L_{\mathcal{F}}$ is MSO-definable. For any MSO-formula ϕ over A^ω, let ϕ^+ be the MSO-formula over $(A \times \{0,1\}^n)^\omega$ obtained by replacing in ϕ any atom $a(x)$, $a \in A$, by $\bigvee_{\overline{b} \in \{0,1\}^n} (a, \overline{b})(x)$. Then, $L_{\mathcal{F}}$ is definable by the MSO-formula

$$\phi_{\mathcal{F}} \equiv \phi^+_{dom} \wedge \bigwedge_{i=1}^{n} \forall x \cdot \Big(\bigvee_{(a,\overline{b}) \in A \times \{0,1\}^n \text{ s.t. } b_i = 1} (a, \overline{b})(x) \Big) \leftrightarrow \phi_i^+(x)$$

By Büchi's Theorem, $L_{\mathcal{F}}$ is definable by a deterministic Muller automaton $\mathcal{A} = (Q, \Delta, I, F)$ over $A \times \{0,1\}^n$. The transducer $\mathcal{T}$ is obtained from $\mathcal{A}$ by projecting it on A and by selecting, using non-determinism, some word v_i to output when reading $a \in A$, whenever there exists a transition of $\mathcal{A}$ on some $(a, \overline{b})$ with $\overline{b}_i = 1$. More precisely, the underlying automaton of $\mathcal{T}$ is $\mathcal{B} = (Q', \Delta', I', F')$ where:

- $Q' = Q \times \{1, \dots, n\}$,
- $I' = I \times \{1, \dots, n\}$,
- $\Delta' = \{((p,i), a, (q,j)) \mid \exists \overline{b} \in \{0,1\}^n \cdot \overline{b}_i = 1 \wedge (p, (a, \overline{b}), q) \in \Delta\}$,
- F' is the set of subsets $P \subseteq Q'$ such that the projection of P on Q is in F.

Finally, the output function of $\mathcal{T}$ is defined by $o((p,i), a, (q,j)) = v_i$.

Conversely, let $\mathcal{T} = (\mathcal{A}, o)$ be some functional transducer realizing f. By Proposition 45, it can be assumed to be unambiguous. We turn $\mathcal{T}$ into an MSO-transducer $\mathcal{F} = (A, B, \phi_{dom}, V, \mu)$ such that $[\![\mathcal{T}]\!] = [\![\mathcal{F}]\!]$. Again by Büchi's theorem, the domain of $\mathcal{T}$, which is regular, is MSO-definable by some formula ϕ_{dom}. We let $V = \mathrm{CoDom}(o)$ and for all $v \in V$, we define the language L_v of words $u \in (A \times \{0,1\})^\omega$ such that

1. $\pi_A(u) \in \mathrm{dom}(\mathcal{T})$
2. u contains exactly one position, denoted i_0, labeled in $A \times \{1\}$,
3. the (unique) run $r = q_0 q_1 \dots$ of $\mathcal{A}$ on $\pi_A(u)$ satisfies $o(q_{i_0 - 1}, \pi_A(u(i_0)), q_{i_0}) = v$.

The language L_v is definable by a Muller automaton, obtained as a product of an automaton which accepts all words in $(A \times \{0,1\})^\omega$ containing exactly one 1, and a Muller automaton which simulates $\mathcal{A}$ on the projection $\pi_A(u)$ and checks, when reading a position labeled 1, that the transitions t of $\mathcal{A}$ applied satisfies $o(t) = v$. Hence, by Büchi's theorem, L_v is MSO-definable by some formula ψ_v. We let $\mu(v) = \phi_v(x)$ where $\phi_v(x)$ is defined by applying the following transformations to ψ_v:

1. first rename any occurrence of x in ψ_v by some variable x'
2. for all variables y and $a \in A$, replace in ψ_v any atom of the form $(a,1)(y)$ by $a(x) \wedge y = x$ and any atom $(a,0)(y)$ by $a(y) \wedge x \neq y$.

For all words $u \in A^\omega$ and $i \geqslant 1$, we have $u \models \phi_v(i)$ if and only if $u \otimes (0^{i-1}10^\omega) \models \psi_v$ if and only if $u \otimes (0^{i-1}10^\omega) \in L_v$ if and only if $u \in \mathrm{dom}(\mathcal{T})$ and the unique run of $\mathcal{A}$ on u produces v when reading position i.

(2) Let us now prove statement (2) and consider first the $\Rightarrow$ implication, *i.e.*, assume $\mathcal{F}$ is some FO-transducer. Note that the formula $\phi_{\mathcal{F}}$ is in this case an FO-formula. By Theorem 43, this implies that $\mathcal{A}$ can be assumed to be aperiodic. There exists $m \geqslant 0$ such that for all states p, q and all $\alpha \in (A \times \{0,1\}^n)^*$, $p \xrightarrow{\alpha^m}_{\mathcal{A}} q$ if and only if $p \xrightarrow{\alpha^{m+1}}_{\mathcal{A}} q$. Let us show that $\mathcal{B}$ is aperiodic. We also assume that $\mathcal{B}$ is trim, otherwise we trim it.

Let $u \in A^*$ and $s = (p,i)$, $t = (q,j)$ be two states of $\mathcal{B}$ and $\ell \geqslant 0$ such that $s \xrightarrow{u^\ell}_{\mathcal{B}} t$. We show that $s \xrightarrow{u^{\ell+1}}_{\mathcal{B}} t$ if ℓ is large enough. Since $\mathcal{B}$ is trim, there exists $u_0 \in A^*$ and $u_1 \in A^\omega$ such that $u_1 u^\ell u_2 \in L(\mathcal{B})$. By definition of $\mathcal{B}$, we have in particular that $u_1 u^\ell u_2 \models \phi_{dom}$. Now, there exists a unique annotation $v \in (\{0,1\}^n)^*$ of $u_1 u^\ell u_2$ such that $u_1 u^\ell u_2 \otimes v \in L_{\mathcal{F}}$, because $L_{\mathcal{F}}$ consists only of the words whose projection on A satisfies ϕ_{dom} and where every position x has been extended with a tuple of bits $(b_1, \ldots, b_n)$ indicating which of the FO-formulas $\phi_i(x)$ hold at position x or not. By taking ℓ large enough, since the formulas $\phi_i(x)$ are first-order, it is possible to decompose u^ℓ into $u^{\ell_1+\ell_2+\ell_3}$ such that ℓ_2 is as large as we want, and any factor u in the factor u^{ℓ_2} receives the same annotation. This is because the FO-formulas $\phi_i(x)$ are not able to distinguish between the kth and $(k+1)$th occurrence of u in the word $u_1 u^\ell u_2$, for all $k \in \{\ell_1, \ldots, \ell_1 + \ell_2 - 1\}$. More precisely, there exist ℓ_1, ℓ_2, ℓ_3 such that $\ell_2 \geqslant m$ and for all $k \in \{\ell_1, \ldots, \ell_1 + \ell_2 - 1\}$, for all $j \in \{|u_1| + k|u| + 1, \ldots, |u_1| + (k+1)|u|\}$, for all $i \in \{1, \ldots, n\}$, $u_1 u^{\ell_1+\ell_2+\ell_3} u_2 \models \phi_i(j)$ if and only if $u_1 u^{\ell_1+\ell_2+\ell_3} u_2 \models \phi_i(j + |u|)$. This can be shown using Ehrenfeucht-Fraissé games, see for instance [17].

Hence, there are words $w_1, w_1', w_2, w_2', w_3' \in (\{0,1\}^n)^*$ and $w_3 \in (\{0,1\}^n)^\omega$ such that

$$u' = (u_1 \otimes w_1)(u^{\ell_1} \otimes w_1')(u \otimes w_2)^{\ell_2}(u^{\ell_3} \otimes w_3')(u_3 \otimes w_3) \in L_{\mathcal{F}}.$$

and moreover, the run of $\mathcal{A}$ on u' can be decomposed into:

$$q_0 \xrightarrow{u_1 \otimes w_1}_{\mathcal{A}} p \xrightarrow{u^{\ell_1} \otimes w_1'}_{\mathcal{A}} p_1 \xrightarrow{(u \otimes w_2)^{\ell_2}}_{\mathcal{A}} p_2 \xrightarrow{u^{\ell_3} \otimes w_3'}_{\mathcal{A}} q \xrightarrow{u_3 \otimes w_3}_{\mathcal{A}}$$

for some initial state q_0 and states p_1, p_2 of $\mathcal{A}$. By aperiodicity of $\mathcal{A}$ and since $\ell_2 \geqslant m$, we obtain the following run of $\mathcal{A}$:

$$q_0 \xrightarrow{u_1 \otimes w_1}_{\mathcal{A}} p \xrightarrow{u^{\ell_1} \otimes w_1'}_{\mathcal{A}} p_1 \xrightarrow{(u \otimes w_2)^{\ell_2+1}}_{\mathcal{A}} p_2 \xrightarrow{u^{\ell_3} \otimes w_3'}_{\mathcal{A}} q \xrightarrow{u_3 \otimes w_3}_{\mathcal{A}}$$

In particular, we have $p \xrightarrow{u^{\ell+1} \otimes (w_1' w_2^{\ell_2+1} w_3')}_{\mathcal{A}} q$. Moreover, the ith bit of the first letter of w_1' is 1, because of the existence of an accepting run of $\mathcal{B}$ on $u_0 u^\ell u_1$ of the form $q_0' \xrightarrow{u_0} (p,i) \xrightarrow{u^\ell} (q,j) \xrightarrow{u_1}$. By definition of $\mathcal{B}$, for all $j' \in \{1, \ldots, n\}$, we therefore obtain a run of the form $(p,i) \xrightarrow{u^{\ell+1}}_{\mathcal{B}} (q,j')$, and in particular it is true for $j' = j$, concluding the first direction of the proof. The other direction (showing that $(p,i) \xrightarrow{u^{\ell+1}}_{\mathcal{B}} (q,j)$ implies $(p,i) \xrightarrow{u^\ell}_{\mathcal{B}} (q,j)$) is completely similar.

Now, we prove the left implication $\Leftarrow$ of statement (2). We again inspect the proof of statement (1), but this time the left implication. Let $\mathcal{T} = (\mathcal{A}, o)$ be some aperiodic transducer realizing f. By Proposition 45, $\mathcal{T}$ can be assumed to be unambiguous and aperiodic. The domain of $\mathcal{T}$ is defined by its underlying aperiodic and unambiguous Muller automaton $\mathcal{A}$.

By Proposition 44, the language of $\mathcal{A}$ is FO-definable by some formula ϕ_{dom}. To conclude the proof, it suffices to remark that the Muller automaton which accepts L_v is aperiodic whenever $\mathcal{A}$ is aperiodic, as the product of two aperiodic automata. This shows that L_v, for all $v \in V$, is FO-definable by some formula ψ_v. Then, the transformation applied on ψ_v to obtain $\phi_v(x)$ preserves the fact of being first-order. ◂