Regular Transducer Expressions for Regular Transformations

Vrunda Dave
IIT Bombay
vrunda@cse.iitb.ac.in

Paul Gastin
LSV, ENS Paris-Saclay, CNRS, UMI RELAX
paul.gastin@lsv.fr

Shankara Narayanan Krishna
IIT Bombay
krishnas@cse.iitb.ac.in

Abstract

Functional MSO transductions, deterministic two-way transducers, as well as streaming string transducers are all equivalent models for regular functions. In this paper, we show that every regular function, either on finite words or on infinite words, captured by a deterministic two-way transducer, can be described with a regular transducer expression (RTE). For infinite words, the transducer uses Muller acceptance and ω-regular look-ahead. RTEs are constructed from constant functions using the combinators if-then-else (deterministic choice), Hadamard product, and unambiguous versions of the Cauchy product, the 2-chained Kleene-iteration and the 2-chained omega-iteration. Our proof works for transformations of both finite and infinite words, extending the result on finite words of Alur et al. in LICS'14. In order to construct an RTE associated with a deterministic two-way Muller transducer with look-ahead, we introduce the notion of transition monoid for such two-way transducers where the look-ahead is captured by some backward deterministic Büchi automaton. Then, we use an unambiguous version of Imre Simon's famous forest factorization theorem in order to derive a "good" (ω-)regular expression for the domain of the two-way transducer. "Good" expressions are unambiguous and Kleene-plus as well as ω-iterations are only used on subexpressions corresponding to *idempotent* elements of the transition monoid. The combinator expressions are finally constructed by structural induction on the "good" (ω-)regular expression describing the domain of the transducer.

CCS Concepts • **Theory of computation → Models of computation**;

Keywords Transducers; Transition monoid; Unambiguity

ACM Reference Format:
Vrunda Dave, Paul Gastin, and Shankara Narayanan Krishna . 2018. Regular Transducer Expressions for Regular Transformations. In *LICS '18: 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, July 9–12, 2018, Oxford, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3209108.3209182

1 Introduction

One of the most fundamental results in theoretical computer science is that the class of regular languages corresponds to the class of languages recognised by finite state automata, to the class of languages definable in MSO, and to the class of languages whose syntactic monoid is finite. Regular languages are also those that can be expressed using a regular expression; this equivalence is given by Kleene's theorem. This beautiful correspondence between machines, logics and algebra in the case of regular languages paved the way to generalizations of this fundamental theory to regular transformations [14], where, it was shown that regular transformations are those which are captured by two-way transducers and by MSO transductions a la Courcelle. Much later, streaming string transducers (SSTs) were introduced [1] as a model which makes a single pass through the input string and uses a finite set of variables that range over strings from the output alphabet. In [1], the equivalence between SSTs and MSO transductions was established, thereby showing that regular transformations are those which are captured by either SSTs, two-way transducers or MSO transductions. This theory was further extended to work for infinite string transformations [4]; the restriction from MSO transductions to first-order definable transductions, and their equivalence with aperiodic SSTs and aperiodic two-way transducers has also been established over finite and infinite strings [15], [12]. Other generalizations such as [2], extend this theory to trees. Most recently, this equivalence between SSTs and logical transductions are also shown to hold good even when one works with the origin semantics [6].

Moving on, an interesting generalization pertains to the characterization of the output computed by two-way transducers or SSTs (over finite and infinite words) using regular-like expressions. For the strictly lesser expressive case of sequential one-way transducers, this regex characterization of the output is obtained as a special case of Schützenberger's famous equivalence [13] between weighted automata and regular weighted expressions. The question is much harder when one looks at two-way transducers, due to the fact that the output is generated in a one-way fashion, while the input is read in a two-way manner. The most recent result known in this direction is [5], which provides a set of combinators, analogous to the operators used in forming regular expressions. These combinators are used to form *combinator expressions* which compute the output of an additive cost register automaton (ACRA) over finite words. ACRAs are generalizations of SSTs and compute a partial function from finite words over a finite alphabet to values from a monoid $(\mathbb{D}, +, 0)$ (SSTs are ACRAs where $(\mathbb{D}, +, 0)$ is the free monoid $(\Gamma^*, ., \epsilon)$ for some finite output alphabet Γ). The combinators introduced in [5] form the basis for a declarative language DReX [3] over finite words, which can express all regular string-to-string transformations, and can also be efficiently evaluated.

Our Contributions. We generalize the result of [5]. Over finite words, we work with two-way deterministic transducers (denoted 2DFT, see Figure 1 left) while over infinite words, the model considered is a deterministic two-way transducer with regular look-ahead, equipped with the Muller acceptance condition. For example, Figure 1 right gives an ω-$\mathsf{2DMT}_{\mathsf{la}}$ (la stands for look-ahead and M in the 2DMT for Muller acceptance).

In both cases of finite words and infinite words, we come up with a set of combinators which we use to form *regular transducer*

LICS '18, July 9–12, 2018, Oxford, United Kingdom

ACM ISBN 978-1-4503-5583-4/18/07…$15.00
https://doi.org/10.1145/3209108.3209182

Figure 1. On the left, a 2DFT $\mathcal{A}$ with $[\![\mathcal{A}]\!](ba^{m_1}ba^{m_2}b\dots a^{m_k}b) = a^{m_2}b^{m_1}a^{m_3}b^{m_2}\dots a^{m_k}b^{m_{k-1}}$. On the right, an ω-2DMT$_{\text{la}}$ $\mathcal{A}'$ with $[\![\mathcal{A}']\!](u_1\#u_2\#\dots\#u_n\#v) = u_1^R u_1 \# u_2^R u_2 \# \dots \# u_n^R u_n \# v$ where $u_1, \dots, u_n \in (a+b)^*$, $v \in (a+b)^\omega$ and u^R denotes the reverse of u. The Muller acceptance set is $\{\{q_5\}\}$. The look-ahead expressions $\Sigma^*\#\Sigma^\omega$ and $(\Sigma\setminus\{\#\})^\omega$ are used to check if there is a # in the remaining suffix.

expressions (RTE) characterizing two-way transducers (2DFT/ω-2DMT$_{\text{la}}$).

The Combinators. We describe our basic combinators for RTEs. The semantics of an RTE is a partial function $f\colon \Sigma^\infty \to \Gamma^\infty$ whose domain is denoted $\mathrm{dom}(f)$.

• We first look at the case of finite words. The constant function d maps all strings in Σ^* to some fixed value d. The *unambiguous* version of union is the if-then-else combinator $K\,?\,f : g$ which checks if $w \in \Sigma^*$ is in the regular language K or not, and produces $f(w)$ if $w \in K$ and $g(w)$ otherwise. The unambiguous Cauchy product $f \boxdot g$ when applied on $w \in \Sigma^*$ produces $f(u)\cdot g(v)$ if $w = u\cdot v$ is an unambiguous decomposition of w with $u \in \mathrm{dom}(f)$ and $v \in \mathrm{dom}(g)$. The unambiguous Kleene-plus $f^{\boxplus}$ when applied to $w \in \Sigma^*$ produces $f(u_1)\cdots f(u_n)$ if $w = u_1\cdots u_n$ is an unambiguous factorization of w, with each $u_i \in \mathrm{dom}(f)$. The Hadamard product $f \odot g$ when applied to w produces $f(w)\cdot g(w)$. Finally, the unambiguous 2-chained Kleene-plus $[K, f]^{2\boxplus}$ when applied to a string w produces as output $f(u_1u_2)\cdot f(u_2u_3)\cdots f(u_{n-1}u_n)$ if w can be unambiguously written as $u_1u_2\cdots u_n$, with each $u_i \in K$, for the regular language K. We also have the reverses $f \overleftarrow{\boxdot} g$, $f^{\overleftarrow{\boxplus}}$ and $[K, f]^{\overleftarrow{2\boxplus}}$: $[f \overleftarrow{\boxdot} g](w)$ produces $g(v)\cdot f(u)$ if w is the unambiguous concatenation $u\cdot v$ with $u \in \mathrm{dom}(f)$ and $v \in \mathrm{dom}(g)$, $f^{\overleftarrow{\boxplus}}(w)$ produces $f(u_n)\cdots f(u_1)$ if w is the unambiguous concatenation $u_1\cdots u_n$ with $u_i \in \mathrm{dom}(f)$ for all i, and, $[K, f]^{\overleftarrow{2\boxplus}}(w)$ produces $f(u_{n-1}u_n)\cdots f(u_1u_2)$ if w is the unambiguous catenation $u_1\cdots u_n$ with $u_i \in K$ for all i.

• In the case of infinite words, the Cauchy product $f \boxdot g$ works on $w \in \Sigma^\omega$ if w can be written unambiguously as $u\cdot v$ with $u \in \mathrm{dom}(f)\cap\Sigma^*$ and $v \in \mathrm{dom}(g)\cap\Sigma^\omega$. Another difference is in the use of the Hadamard product: for $w \in \Sigma^\omega$, $f \odot g$ produces $f(w)\cdot g(w)$ if $f(w)$ is a finite string. Note that these are sound with respect to the concatenation semantics for infinite words. Indeed, we also have ω-iteration and two-chained ω-iteration: $f^\omega(w) = f(u_1)f(u_2)\cdots$ if $w \in \Sigma^\omega$ can be unambiguously decomposed as $w = u_1u_2\cdots$ with $u_i \in \mathrm{dom}(f)\cap\Sigma^*$ for all $i \geq 1$. Moreover, $[K, f]^{2\omega}(w) = f(u_1u_2)f(u_2u_3)\cdots$ if $w \in \Sigma^\omega$ can be unambiguously decomposed as $w = u_1u_2\cdots$ with $u_i \in K$ for all $i \geq 1$, where $K \subseteq \Sigma^*$ is regular.

• An RTE is formed using the above basic combinators. Consider the RTE $C = C_4^{\boxplus} \boxdot C_2^{\omega}$ with $C_4 = ((a+b)^*\#)\,?\,(C_3^{\overleftarrow{\boxplus}} \odot C_1^{\boxplus}) : \bot$, $C_2 = a\,?\,a : (b\,?\,b : \bot)$. $C_1 = a\,?\,a : (b\,?\,b : (\#\,?\,\#\,?\,\bot))$, $C_3 = a\,?\,a : (b\,?\,b : (\#\,?\,\epsilon : \bot))$. Then $\mathrm{dom}(C_1) = \mathrm{dom}(C_3) = (a+b+\#)$, $\mathrm{dom}(C_2) = (a+b)$, $\mathrm{dom}(C_4) = (a+b)^*\#$ and, for $u \in (a+b)^*$, $[\![C_4]\!](u\#) = u^R u\#$ where u^R denotes the reverse of u. This gives $\mathrm{dom}(C) = [(a+b)^*\#]^+(a+b)^\omega$ and when $u_i \in (a+b)^*$ and $v \in (a+b)^\omega$ we have $[\![C]\!](u_1\#u_2\#\cdots u_n\#v) = u_1^R u_1 \# u_2^R u_2 \# \cdots \# u_n^R u_n \# v$. The RTE $C' = (a+b)^\omega\,?\,C_2^\omega : C$ corresponds to the ω-2DMT$_{\text{la}}$ $\mathcal{A}'$ in Figure 1: $[\![C']\!] = [\![\mathcal{A}']\!]$.

• The combinators proposed in [5] also require unambiguity in concatenation and iteration. The *base function* L/d in [5] maps all strings in language L to the constant d, and is undefined for strings not in L. This can be written using our if-then-else $L\,?\,d : \bot$. The *conditional choice* combinator $f \triangleright g$ of [5] maps an input σ to $f(\sigma)$ if it is in $dom(f)$, and otherwise it maps it to $g(\sigma)$. This can be written in our if-then-else as $\mathrm{dom}(f)\,?\,f : g$. The *split-sum* combinator $f \oplus g$ of [5] is our Cauchy product $f \boxdot g$. The *iterated sum* Σf of [5] is our Kleene-plus $f^{\boxplus}$. The *left-split-sum* and *left-iterated sum* of [5] are counterparts of our reverse Cauchy product $f \overleftarrow{\boxdot} g$ and reverse Kleene-plus $f^{\overleftarrow{\boxplus}}$. The *sum* $f + g$ of two functions in [5] is our Hadamard product $f \odot g$. Finally, the *chained sum* $\Sigma(f, L)$ of [5] is our two-chained Kleene-plus $[L, f]^{2\boxplus}$. In our case, notations and terminologies are directly inherited from the well-established theory of weighted automata. We believe it is more natural and reflects both the parsing of the input word with usual (unambiguous) rational expressions and how the output is produced. We also extend RTEs to infinite words.

Our main result is that two-way deterministic transducers and regular transducer expressions are effectively equivalent, both for finite words (see [11]) and infinite words.

Theorem 1.1. *(1) Given an RTE (resp. ω-RTE) we can effectively construct an equivalent 2DFT (resp. an ω-2DMT$_{la}$).*

(2) Given a 2DFT (resp. an ω-2DMT$_{la}$) we can effectively construct an equivalent RTE (resp. ω-RTE).

The proof of (1) is by structural induction on the RTE. The construction of an RTE starting from a two-way deterministic transducer $\mathcal{A}$ is quite involved. It is based on the transition monoid $\mathsf{TrM}(\mathcal{A})$ of the transducer. This is a classical notion for two-way transducers over finite words, but not for two-way transducers *with look-ahead* on infinite words (to the best of our knowledge). So we introduce the notion of transition monoid for ω-2DMT$_{\text{la}}$. We handle the look-ahead with a backward deterministic Büchi automaton (BDBA), also called *complete unambiguous* or *strongly unambiguous* Büchi automata [8, 19]. The translation of $\mathcal{A}$ to an RTE is crucially guided by a "good" rational expression induced by the transition monoid of $\mathcal{A}$. These "good" expressions are obtained thanks to an unambiguous version [16] of the celebrated forest factorization

theorem due to Imre Simon [18] (see also [9]). The unambiguous forest factorization theorem implies that, given a two-way transducer $\mathcal{A}$, any input word w in the domain of $\mathcal{A}$ can be factorized unambiguously following a "good" rational expression induced by the transition monoid of $\mathcal{A}$. This unambiguous factorization then guides the construction of the RTE corresponding to $\mathcal{A}$. This algebraic backdrop facilitates a uniform treatment in the case of infinite words and finite words. As a remark, it is not a priori clear how the result of [5] extends to infinite words using the techniques therein.

Goodness of Rational Expressions. The goodness of a rational expression over alphabet Σ is defined using a morphism φ from Σ^* to a monoid $(S, ., 1_S)$. A rational expression F is good iff (i) it is unambiguous and (ii) for each subexpression E of F, the image of all strings in $L(E)$ maps to a single monoid element s_E, and (iii) for each subexpression E^+ of F, s_E is an idempotent. Note that unambiguity ensures the functionality of the output computed. The other two conditions are used to define inductively the RTEs. Good rational expressions might be useful in settings beyond two-way transducers.

Computing the RTE. As an example, we now show how one computes an RTE equivalent to the 2DFT $\mathcal{A}$ on the left of Figure 1.
1. We work with the morphism $\mathsf{Tr}\colon \Sigma^* \to \mathsf{TrM}$ which maps words $w \in \Sigma^*$ to the transition monoid TrM of $\mathcal{A}$. An element $X \in \mathsf{TrM}$ is a set consisting of triples (p, d, q), where d is a direction $\{\rightsquigarrow, \subsetneq, \rightarrow, \leftarrow\}$. Given a word $w \in \Sigma^*$, a triple $(p, \rightsquigarrow, q) \in \mathsf{Tr}(w)$ iff when starting in state p on the left most symbol of w, the run of $\mathcal{A}$ leaves w on the left in state q. The other directions $\subsetneq$ (start at the rightmost symbol of w in state p and leave w on the right in state q), $\leftarrow$ and $\rightarrow$ are similar. In general, we have $w \in \mathrm{dom}(\mathcal{A})$ iff on input $\vdash w \dashv$, starting on $\vdash$ in the initial state of $\mathcal{A}$, the run exits on the right of $\dashv$ in some final state of $\mathcal{A}$. With the automaton $\mathcal{A}$ on the left of Figure 1 we have $w \in \mathrm{dom}(\mathcal{A})$ iff $(q_0, \rightarrow, q_2) \in \mathsf{Tr}(w)$.
2. For each $X \in \mathsf{TrM}$ such that $(q_0, \rightarrow, q_2) \in X$, we find an RTE C_X whose domain is $\mathsf{Tr}^{-1}(X)$ and such that $[\![\mathcal{A}]\!](w) = [\![C_X]\!](w)$ for all $w \in \mathsf{Tr}^{-1}(X)$. The RTE corresponding to $[\![\mathcal{A}]\!]$ is the disjoint union of all these RTEs and is written using the if-then-else construct iterating over for all such elements X. For instance, if the monoid elements containing $(q_0, \rightarrow, q_2)$ are X_1, X_2, X_3 then we set $C = \mathsf{Tr}^{-1}(X_1)\,?\,C_{X_1} : (\mathsf{Tr}^{-1}(X_2)\,?\,C_{X_2} : (\mathsf{Tr}^{-1}(X_3)\,?\,C_{X_3} : \bot))$ where $\bot$ stands for a nowhere defined function, i.e., $\mathrm{dom}(\bot) = \emptyset$.
3. Consider the language $L = (ba^+)^+b \subseteq \mathrm{dom}(\mathcal{A})$. Notice that the regular expression $(ba^+)^+b$ is not "good". For instance, condition (ii) is violated since $\mathsf{Tr}(bab) \neq \mathsf{Tr}(babab)$. Indeed, we can see in Figure 2 that if we start on the right of bab in state q_3 then we exit on the left in state q_5: $(q_3, \leftarrow, q_5) \in \mathsf{Tr}(bab)$. On the other hand, if we start on the right of $babab$ in state q_3 then we exit on the right in state q_2: $(q_3, \subsetneq, q_2) \in \mathsf{Tr}(babab)$. Also, $(q_5, \rightarrow, q_1) \in \mathsf{Tr}(bab)$ while $(q_5, \rightarrow, q_2) \in \mathsf{Tr}(babab)$. It can be seen that $\mathsf{Tr}(a)$ is an idempotent, hence $\mathsf{Tr}(a^+) = \mathsf{Tr}(a)$. We deduce also $\mathsf{Tr}(ba^+b) = \mathsf{Tr}(bab)$. Finally, we have $\mathsf{Tr}((ba^+)^n b) = \mathsf{Tr}(babab)$ for all $n \geq 2$. Therefore, to obtain the RTE corresponding to L, we compute RTEs corresponding to ba^+b and $(ba^+)^+ba^+b$ satisfying conditions (i) and (ii) of "good" rational expressions.
4. While ba^+b is good ($\mathsf{Tr}(a)$ is idempotent), $(ba^+)^+ba^+b$ is not good, the reason being that $\mathsf{Tr}(ba^+)$ is not an idempotent. We can check that $\mathsf{Tr}(ba^+ba^+)$ is still not idempotent, while $\mathsf{Tr}((ba^+)^i) = \mathsf{Tr}((ba^+)^3)$[1] for all $i \geq 3$, (see Figure 2: we only need to argue for $(q_0, \rightarrow, q_3), (q_5, \rightarrow, q_3)$ and $(q_6, \rightarrow, q_3)$ in $\mathsf{Tr}((ba)^i)$, $i \geq 3$, all other entries trivially carry over). In particular, $\mathsf{Tr}((ba^+)^3)$ is an idempotent. Thus, to compute the RTE for $L = (ba^+)^+b$, we consider the RTEs corresponding to the "good" regular expressions $E_1 = ba^+b$, $E_2 = ba^+ba^+b$, $E_3 = [(ba^+)^3]^+b$, $E_4 = [(ba^+)^3]^+ba^+b$ and $E_5 = [(ba^+)^3]^+ba^+ba^+b$.
5. We define by induction, for each "good" expression E and "step" $x = (p, d, q)$ in the monoid element $X = \mathsf{Tr}(E)$ associated with E, an RTE $C_E(x)$ whose domain is E and, given a word $w \in E$, it computes $[\![C_E(x)]\!](w)$ the output of $\mathcal{A}$ when running step x on w. For instance, if $E = a$ and $x = (q_5, \leftarrow, q_5)$ the output is b so we set $C_a(q_5, \leftarrow, q_5) = (a\,?\,b : \bot)$. The if-then-else ensures that the domain is a. Similarly, we get the RTE associated with all atomic expressions and steps. For instance, $C_b(q_1, \rightarrow, q_2) = (b\,?\,\varepsilon : \bot) = C_b(q_3, \rightsquigarrow, q_4)$. For $u, v \in \Sigma^*$, we introduce the macro $u/v = u\,?\,v : \bot$. We have $\mathrm{dom}(u/v) = \{u\}$ and $[\![u/v]\!](u) = v$.

We turn to the good expression a^+. If we start on the right of a word $w \in a^+$ from state q_5 then we read the word from right to left using always the step $(q_5, \leftarrow, q_5)$. Therefore, $C_{a^+}(q_5, \leftarrow, q_5) = (C_a(q_5, \leftarrow, q_5))^{\overleftarrow{\boxplus}} = (a/b)^{\overleftarrow{\boxplus}}$. Similarly, $C_{a^+}(q_4, \leftarrow, q_4) = (a/a)^{\overleftarrow{\boxplus}}$, $C_{a^+}(q_1, \rightarrow, q_1) = (a/\varepsilon)^{\boxplus} = C_{a^+}(q_6, \rightarrow, q_6)$. Now if we start on the left of a word $w \in a^+$ from state q_2 then we first take the step $(q_2, \rightarrow, q_3)$ and then we iterate the step $(q_3, \rightarrow, q_3)$. Therefore, $C_{a^+}(q_2, \rightarrow, q_3) = a\,?\,C_a(q_2, \rightarrow, q_3) : (C_a(q_2, \rightarrow, q_3) \boxdot (C_a(q_3, \rightarrow, q_3))^{\boxplus}) = a\,?\,(a/\varepsilon) : \big((a/\varepsilon) \boxdot (a/\varepsilon)^{\boxplus}\big)$, which is equivalent to the RTE $(a/\varepsilon)^{\boxplus}$. We consider now $E = ba^+ba^+$ and the step $x = (q_0, \rightarrow, q_3)$. We have (see Figure 2) $C_E(x) = C_b(q_0, \rightarrow, q_1) \boxdot C_{a^+}(q_1, \rightarrow, q_1) \boxdot C_b(q_1, \rightarrow, q_2) \boxdot C_{a^+}(q_2, \rightarrow, q_3) = (b/\varepsilon) \boxdot (a/\varepsilon)^{\boxplus} \boxdot (b/\varepsilon) \boxdot (a/\varepsilon)^{\boxplus} \approx (ba^+ba^+\,?\,\varepsilon : \bot)$. More interesting is $y = (q_4, \subsetneq, q_1)$ since on a word $w \in E$, the run which starts on the right in state q_4 goes all the way to the left until it reads the first b in state q_5 and then moves to the right until it exits in state q_1 (see Figure 2): $C_E(y) = \big((b/\varepsilon) \overleftarrow{\boxdot} C_{a^+}(q_5, \leftarrow, q_5) \overleftarrow{\boxdot} C_b(q_4, \leftarrow, q_5) \overleftarrow{\boxdot} C_{a^+}(q_4, \leftarrow, q_4)\big) \odot \big(C_b(q_5, \rightarrow, q_6) \boxdot C_{a^+}(q_6, \rightarrow, q_6) \boxdot C_b(q_6, \rightarrow, q_1) \boxdot C_{a^+}(q_1, \rightarrow, q_1)\big)$ which is equal to the Hadamard product of $\big((b/\varepsilon) \overleftarrow{\boxdot} (a/b)^{\overleftarrow{\boxplus}} \overleftarrow{\boxdot} (b/\varepsilon) \overleftarrow{\boxdot} (a/a)^{\overleftarrow{\boxplus}}\big)$ and $\big((b/\varepsilon) \boxdot (a/\varepsilon)^{\boxplus} \boxdot (b/\varepsilon) \boxdot (a/\varepsilon)^{\boxplus}\big)$. Hence, $C_E(y) \approx (b/\varepsilon) \overleftarrow{\boxdot} (a/b)^{\overleftarrow{\boxplus}} \overleftarrow{\boxdot} (b/\varepsilon) \overleftarrow{\boxdot} (a/a)^{\overleftarrow{\boxplus}}$. The leftmost (b/ε)

[1] $\mathsf{Tr}((ba^+)^3) = \{(q_0, \rightarrow, q_3), (q_1, \rightsquigarrow, q_5), (q_1, \subsetneq, q_1), (q_2, \rightsquigarrow, q_4), (q_2, \subsetneq, q_3), (q_3, \rightsquigarrow, q_4), (q_3, \subsetneq, q_3), (q_4, \rightsquigarrow, q_5), (q_4, \subsetneq, q_1), (q_5, \rightarrow, q_3), (q_5, \subsetneq, q_6), (q_6, \rightarrow, q_3), (q_6, \subsetneq, q_6)\}$

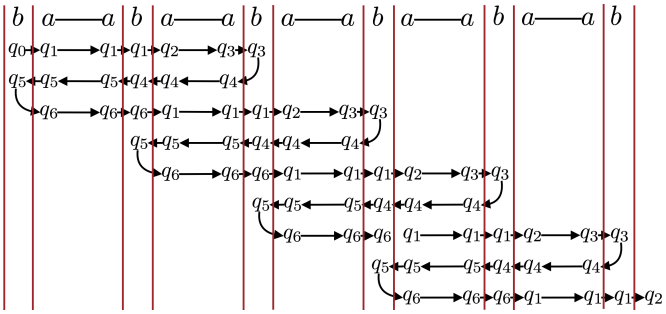


Figure 2. Run of $\mathcal{A}$ on an input word in $(ba^+)^+b$.

in the above RTEs is used to make sure that the input word belongs to $E = ba^+ba^+$. Composing these steps on the right with b, we obtain $C_2 = C_{E_2}(q_0, \rightarrow, q_2)$ which describes the behaviour of $\mathcal{A}$ on $E_2 = ba^+ba^+b \subseteq \mathrm{dom}(\mathcal{A})$: $C_2 = \big(C_E(x) \boxdot C_b(q_3, \circlearrowright, q_4)\big) \odot \big(C_E(y) \boxdot C_b(q_1, \rightarrow, q_2)\big) = \big(C_E(x) \boxdot (b/\varepsilon)\big) \odot \big(C_E(y) \boxdot (b/\varepsilon)\big) \approx \big((b/\varepsilon) \overleftarrow{\boxdot} (a/b)^{\overleftarrow{\boxplus}} \overleftarrow{\boxdot} (b/\varepsilon) \overleftarrow{\boxdot} (a/a)^{\overleftarrow{\boxplus}}\big) \boxdot (b/\varepsilon)$. Therefore, $[\![C_2]\!](ba^{m_1}ba^{m_2}b) = a^{m_2}b^{m_1} = [\![\mathcal{A}]\!](ba^{m_1}ba^{m_2}b)$.

Full proofs of all the results can be found in [11].

2 Two-way transducers over ω-words

We consider regular functions on infinite words. We restrict our attention to two way transducers as the model for computing regular functions. Given a finite alphabet Σ, let Σ^ω denote the set of infinite words over Σ, and let $\Sigma^\infty = \Sigma^* \cup \Sigma^\omega$ be the set of all finite or infinite words over Σ.

We fix a finite input alphabet Σ and a finite output alphabet Γ. Let $\vdash$ be a left end marker symbol not in Σ and let $\Sigma_\vdash = \Sigma \cup \{\vdash\}$. The input word is presented as $\vdash w$ where $w \in \Sigma^\omega$.

Let $\mathcal{R}$ be a finite set of *look-ahead* ω-regular languages. For the ω-regular languages in $\mathcal{R}$, we may use any finite descriptions such as ω-regular expressions or automata. A deterministic two-way transducer (ω-2DMT$_{\mathsf{la}}$) over ω-words is a tuple $\mathcal{A} = (Q, \Sigma, \Gamma, q_0, \delta, \mathcal{F}, \mathcal{R})$, where Q is a finite set of states, $q_0 \in Q$ is a unique initial state, and $\delta\colon Q \times \Sigma_\vdash \times \mathcal{R} \mapsto Q \times \Gamma^* \times \{-1, +1\}$ is the partial transition function. We request that for every pair $(q, a) \in Q \times \Sigma_\vdash$, the subset $\mathcal{R}(q, a)$ of languages $R \in \mathcal{R}$ such that $\delta(q, a, R)$ is defined forms a partition of Σ^ω. This ensures that $\mathcal{A}$ is complete and behaves deterministically. The set $\mathcal{F} \subseteq 2^Q$ specifies the Muller acceptance condition. As in the finite case, the reading head cannot move left while on $\vdash$. A configuration is represented by $w'qaw''$ where $w'a \in \vdash\Sigma^*$, $w'' \in \Sigma^\omega$ and q is the current state, scanning letter a. From configuration $w'qaw''$, let R be the unique ω-regular language in $\mathcal{R}(q, a)$ such that $w'' \in R$, the automaton outputs γ and moves to

$$\begin{cases} w'aq'w'' & \text{if } \delta(q, a, R) = (q', \gamma, +1) \\ w'_1q'baw'' & \text{if } \delta(q, a, R) = (q', \gamma, -1) \text{ and } w' = w'_1b\,. \end{cases}$$

The output $\gamma \in \Gamma^*$ is appended at the end of the output produced so far. A run ρ of $\mathcal{A}$ on $w \in \Sigma^\omega$ is a sequence of transitions starting from the initial configuration $q_0 \vdash w$ where the reading head is on $\vdash$:

$$q_0 \vdash w \xrightarrow{\gamma_1} w'_1q_1w''_1 \xrightarrow{\gamma_2} w'_2q_2w''_2 \xrightarrow{\gamma_3} w'_3q_3w''_3 \xrightarrow{\gamma_4} w'_4q_4w''_4 \cdots$$

We say that ρ reads the whole word w if $\sup\{|w'_n| \mid n > 0\} = \infty$. The set of states visited by ρ infinitely often is denoted $\mathrm{inf}(\rho) \subseteq Q$. The word w is accepted by $\mathcal{A}$, i.e., $w \in \mathrm{dom}(\mathcal{A})$ if ρ reads the whole word w and $\mathrm{inf}(\rho) \in \mathcal{F}$. In this case, we let $[\![\mathcal{A}]\!](w) = \gamma_1\gamma_2\gamma_3\gamma_4 \cdots$ be the output produced by ρ.

The notation ω-2DMT$_{\mathsf{la}}$ signifies the use of the look-ahead (la) using the ω-regular languages in $\mathcal{R}$. It must be noted that without look-ahead, the expressive power of two-way transducers over infinite words is lesser than regular transformations over infinite words [4]. A classical example of this is given in Example 2.1, where the look-ahead is necessary to obtain the required transformation.

Example 2.1. The ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}'$ over $\Sigma = \{a, b, \#\}$ on the right of Figure 1 defines the transformation $[\![\mathcal{A}']\!](u_1\#u_2\# \cdots \#u_n\#v) = u_1^Ru_1\#u_2^Ru_2\# \cdots \#u_n^Ru_n\#v$ where $u_1, \ldots, u_n \in (a+b)^*$, $v \in (a+b)^\omega$ and u^R denotes the reverse of u. The Muller acceptance set is $\{\{q_5\}\}$. From state q_1 reading $\vdash$, or state q_4 reading $\#$, $\mathcal{A}'$ uses the look ahead partition $\mathcal{R}(q_1, \vdash) = \mathcal{R}(q_4, \#) = \{\Sigma^*\#\Sigma^\omega, (\Sigma \setminus \{\#\})^\omega\}$, which indicates the presence or absence of a # in the remaining suffix of the word being read. For all other transitions, the look-ahead langage is Σ^ω, hence it is omitted. Also, to keep the picture light, the automaton is not complete, i.e., we have omitted the transitions going to a sink state. It can be seen that any maximal string u between two consecutive occurrences of # is replaced with u^Ru; the infinite suffix over $\{a, b\}^\omega$ is then reproduced as it is.

Remark 2.2. *Equivalently, two-way transducers over ω-words can be defined using look-behind and look-ahead automata [4]. We believe that using ω-regular languages for the look-ahead constraints is more convenient, resulting in more readable transducers. As explained in Section 6, these look-ahead languages can be replaced with automata.*

3 Regular Transducer Expressions

We define regular transducer expressions for both finite and infinite words. Let Σ and Γ be finite input and output alphabets and let $\perp$ stand for undefined. We define the output domain as $\mathbb{D} = \Gamma^\infty \cup \{\perp\}$, with the usual concatenation of a finite word on the left with a finite or infinite word on the right. Here, $\perp$ acts as zero and the unit is the empty word $1_\mathbb{D} = \varepsilon$.

The syntax of *Regular Transducer Expressions* (RTEs and ω-RTEs) from Σ^∞ to $\mathbb{D}$ is defined by:

$E ::= d \mid K\,?\,E : E \mid E \odot E \mid E \boxdot E \mid E^{\boxplus} \mid E^{\overleftarrow{\boxplus}} \mid [K, E]^{2\boxplus} \mid [K, E]^{\overleftarrow{2\boxplus}}$
$C ::= L\,?\,C : C \mid C \odot C \mid E \boxdot C \mid E^\omega \mid [K, E]^{2\omega}$

where $d \in \Gamma^* \cup \{\perp\}$, $K \subseteq \Sigma^*$ ranges over *regular* languages of *finite words* and $L \subseteq \Sigma^\omega$ ranges over ω-*regular* languages of *infinite words*. Here, E is an RTE over finite words with semantics $[\![E]\!]\colon \Sigma^* \to \Gamma^* \cup \{\perp\}$, whereas C is an ω-RTE over infinite words with semantics $[\![C]\!]\colon \Sigma^\omega \to \mathbb{D}$. The semantics is defined inductively.

Constants. For $d \in \Gamma^* \cup \{\perp\}$, we let $[\![d]\!]$ be the constant map defined by $[\![d]\!](w) = d$ for all $w \in \Sigma^*$. We have $\mathrm{dom}([\![d]\!]) = \Sigma^*$ if $d \neq \perp$ and $\mathrm{dom}([\![\perp]\!]) = \emptyset$.

Given regular languages $K \subseteq \Sigma^*, L \subseteq \Sigma^\infty$, and functions $f\colon \Sigma^* \to \Gamma^* \cup \{\perp\}$, $g, h\colon \Sigma^\infty \to \mathbb{D}$, we define

If then else. We have $\mathrm{dom}(L\,?\,g : h) = (\mathrm{dom}(g) \cap L) \cup (\mathrm{dom}(h) \setminus L)$. Moreover, $(L\,?\,g : h)(w)$ is defined as $g(w)$ for $w \in \mathrm{dom}(g) \cap L$, and $h(w)$ for $w \in \mathrm{dom}(h) \setminus L$.

Hadamard product. We have $\mathrm{dom}(g \odot h) = g^{-1}(\Gamma^*) \cap \mathrm{dom}(h)$. Moreover, $(g \odot h)(w) = g(w) \cdot h(w)$ for $w \in \mathrm{dom}(g) \cap \mathrm{dom}(h)$ with $g(w) \in \Gamma^*$.

Unambiguous Cauchy product. If $w \in \Sigma^\infty$ admits a unique factorization $w = u \cdot v$ with $u \in \mathrm{dom}(f)$ and $v \in \mathrm{dom}(g)$ then we set $(f \boxdot g)(w) = f(u) \cdot g(v)$. Otherwise, $(f \boxdot g)(w) = \perp$.

Unambiguous Kleene-plus and its reverse. If $w \in \Sigma^*$ admits a unique factorization $w = u_1u_2 \cdots u_n$ with $n \geq 1$ and $u_i \in \mathrm{dom}(f)$ for all $1 \leq i \leq n$ then we set $f^{\boxplus}(w) = f(u_1)f(u_2) \cdots f(u_n)$ and $f^{\overleftarrow{\boxplus}}(w) = f(u_n) \cdots f(u_2)f(u_1)$. Otherwise, we set $f^{\boxplus}(w) = \perp = f^{\overleftarrow{\boxplus}}(w)$. We have $\mathrm{dom}(f^{\boxplus}) = \mathrm{dom}(f^{\overleftarrow{\boxplus}}) \subseteq \mathrm{dom}(f)^+$. Notice that $\mathrm{dom}(f^{\boxplus}) = \emptyset$ when $\varepsilon \in \mathrm{dom}(f)$.

Unambiguous 2-chained Kleene-plus and its reverse. If $w \in \Sigma^*$ admits a unique factorization $w = u_1u_2 \cdots u_n$ with $n \geq 1$ and $u_i \in K$ for all $1 \leq i \leq n$ then we set $[K, f]^{2\boxplus}(w) = f(u_1u_2)f(u_2u_3) \cdots f(u_{n-1}u_n)$ and $[K, f]^{\overleftarrow{2\boxplus}}(w) = f(u_{n-1}u_n) \cdots f(u_2u_3)f(u_1u_2)$ (if $n = 1$, the empty product gives the unit of $\mathbb{D}$: $[K, f]^{2\boxplus}(w) = 1_\mathbb{D} = [K, f]^{\overleftarrow{2\boxplus}}(w)$). Otherwise, we set $[K, f]^{2\boxplus}(w) = \perp = [K, f]^{\overleftarrow{2\boxplus}}(w)$.

Again, we have $\mathrm{dom}([K,f]^{2\boxplus}) = \mathrm{dom}([K,f]^{\overleftarrow{2\boxplus}}) \subseteq K^+$ and we have equality when K^+ is unambiguous and $K^2 \subseteq \mathrm{dom}(f)$.

Unambiguous ω-iteration. If $w \in \Sigma^\omega$ admits a unique infinite factorization $w = u_1u_2u_3\cdots$ with $u_i \in \mathrm{dom}(f)$ for all $i \geq 1$ then we set $f^\omega(w) = f(u_1)f(u_2)f(u_3)\cdots \in \Gamma^\infty$. Otherwise, $f^\omega(w) = \bot$.

Unambiguous 2-chained ω-iteration. If $w \in \Sigma^\omega$ admits a unique factorization $w = u_1u_2u_3\cdots$ with $u_i \in K$ for all $i \geq 1$ and if moreover $u_iu_{i+1} \in \mathrm{dom}(f)$ for all $i \geq 1$ then we set $[K,f]^{2\omega}(w) = f(u_1u_2)f(u_2u_3)f(u_3u_4)\cdots$. Otherwise, we set $[K,f]^{2\omega}(w) = \bot$.

Remark 3.1. *Let $C_\varepsilon = (\Sigma\,?\,\varepsilon : \bot)^\omega$. We have $\mathrm{dom}(C_\varepsilon) = \Sigma^\omega$ and $[\![C_\varepsilon]\!](w) = \varepsilon$ for all $w \in \Sigma^\omega$. Now, for $\gamma \in \Gamma^+$, let $C_\gamma = (\Sigma\,?\,\gamma : \bot) \boxdot C_\varepsilon$. We have $\mathrm{dom}(C_\gamma) = \Sigma^\omega$ and $[\![C_\gamma]\!](w) = \gamma$ for all $w \in \Sigma^\omega$. Therefore, we can freely use constants $\gamma \in \Gamma^*$ when defining ω-RTEs.*

Remark 3.2. *We can express the ω-iteration with the 2-chained ω-iteration: $f^\omega = [\mathrm{dom}(f), f \boxdot (\mathrm{dom}(f)\,?\,\varepsilon : \bot)]^{2\omega}$.*

Example 3.3. We now give the ω-RTE for the transformation given in Example 2.1. Let $E_1 = a\,?\,a : (b\,?\,b : (\#\,?\,\# : \bot))$, $E_2 = a\,?\,a : (b\,?\,b : \bot)$ and $E_3 = a\,?\,a : (b\,?\,b : (\#\,?\,\varepsilon : \bot))$. Then $\mathrm{dom}(E_1) = \mathrm{dom}(E_3) = (a+b+\#)$ and $\mathrm{dom}(E_2) = (a+b)$. Also, we let $E_4 = ((a+b)^*\#)\,?\,({E_3}^{\overleftarrow{\boxplus}} \odot {E_1}^{\boxplus}) : \bot$. We have $\mathrm{dom}(E_4) = (a+b)^*\#$ and, for $u \in (a+b)^*$, $[\![E_4]\!](u\#) = u^Ru\#$ where u^R denotes the reverse of u. Next, let $C_1 = {E_4}^{\boxplus} \boxdot E_2^\omega$. Then, $\mathrm{dom}(C_1) = [(a+b)^*\#]^+(a+b)^\omega$, and $[\![C_1]\!](u_1\#u_2\#\cdots u_n\#v) = u_1^Ru_1\#u_2^Ru_2\#\cdots\#u_n^Ru_n\#v$ when $u_i \in (a+b)^*$ and $v \in (a+b)^\omega$. Finally, let $C = (a+b)^\omega\,?\,E_2^\omega : C_1$. We have $\mathrm{dom}(C) = [(a+b)^*\#]^*(a+b)^\omega$ and $[\![C]\!] = [\![\mathcal{A}']\!]$ where $\mathcal{A}'$ is the transducer on the right of Figure 1.

Theorem 3.4. *ω-2DMT$_{la}$ and ω-RTEs define the same class of functions. More precisely, (1) given an ω-RTE C, we can construct an ω-2DMT$_{la}$ $\mathcal{A}$ such that $[\![\mathcal{A}]\!] = [\![C]\!]$, and (2) given an ω-2DMT$_{la}$ $\mathcal{A}$, we can construct an ω-RTE C such that $[\![\mathcal{A}]\!] = [\![C]\!]$,*

The proof of (1) is given in the next section, while the proof of (2) will be given in Section 8 after some preparatory work on backward deterministic Büchi automata (Section 5) which are used to remove the look-ahead of ω-2DMT$_{\mathsf{la}}$ (Section 6), and the notion of transition monoid for ω-2DMT$_{\mathsf{la}}$ (Section 7) used in the unambiguous forest factorization theorem extended to infinite words (Theorem 8.1).

4 ω-RTE to ω-2DMT$_{\mathsf{la}}$

In this section, we prove one direction of Theorem 3.4: given an ω-RTE C, we can construct an ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}$ such that $[\![\mathcal{A}]\!] = [\![C]\!]$. The proof is by structural induction and follows immediately from

Lemma 4.1. *Let $K \subseteq \Sigma^*$ be regular and $L \subseteq \Sigma^\omega$ be ω-regular. Let f be an RTE with $[\![f]\!] = [\![M_f]\!]$ for some 2DFT M_f. Let g, h be ω-RTEs with $[\![g]\!] = [\![M_g]\!]$ and $[\![h]\!] = [\![M_h]\!]$ for ω-2DMT$_{la}$ M_g and M_h respectively. Then, one can construct*

1. *an ω-2DMT$_{la}$ $\mathcal{A}$ such that $[\![L\,?\,g : h]\!] = [\![\mathcal{A}]\!]$,*
2. *an ω-2DMT$_{la}$ $\mathcal{A}$ such that $[\![\mathcal{A}]\!] = [\![g \odot h]\!]$,*
3. *an ω-2DMT$_{la}$ $\mathcal{A}$ such that $[\![\mathcal{A}]\!] = [\![g \boxdot h]\!]$,*
4. *an ω-2DMT$_{la}$ $\mathcal{A}$ such that $[\![\mathcal{A}]\!] = [\![f^\omega]\!]$,*
5. *an ω-2DMT$_{la}$ $\mathcal{A}$ such that $[\![\mathcal{A}]\!] = [\![[K,f]^{2\omega}]\!]$.*

Proof. Throughout the proof, we let $M_g = (Q_g, \Sigma, \Gamma, s_g, \delta_g\mathcal{F}_g, \mathcal{R}_g)$ and $M_h = (Q_h, \Sigma, \Gamma, s_h, \delta_h, \mathcal{F}_h, \mathcal{R}_h)$ be the be the ω-2DMT$_{\mathsf{la}}$ such that $[\![M_g]\!] = [\![g]\!]$ and $[\![M_h]\!] = [\![h]\!]$.

(1) **If then else.** The set of states of $\mathcal{A}$ is $Q_\mathcal{A} = \{q_0\} \cup Q_g \cup Q_h$ with $q_0 \notin Q_g \cup Q_h$. In state q_0, we have the transitions $\delta_\mathcal{A}(q_0, (\vdash, R \cap L)) = (q, \gamma, +1)$ if $\delta_g(s_g, (\vdash, R)) = (q, \gamma, +1)$ and $\delta_\mathcal{A}(q_0, (\vdash, R' \setminus L)) = (q', \gamma', +1)$ if $\delta_h(s_h, (\vdash, R')) = (q', \gamma', +1)$. This invokes M_g (M_h) iff the input w is in L (not in L). The Muller set $\mathcal{F}$ is simply a union $\mathcal{F}_g \cup \mathcal{F}_h$ of the respective Muller sets of M_g and M_h. It is clear that $[\![\mathcal{A}]\!]$ coincides with $[\![M_g]\!]$ iff the input string is in L, and otherwise, $[\![\mathcal{A}]\!]$ coincides with $[\![M_h]\!]$.

(2) **Hadamard product.** We create a look ahead which indicates the position where we can stop reading the input word w for the transducer M_g. The look ahead should satisfy two conditions: (a) we cannot visit any position to the left of the current position in the remaining run of M_g on w, (b) the output produced by running M_g on the suffix should be ε.

To accommodate these conditions, we create look ahead automata A_q for each state $q \in Q_g$ and let $L_q = \mathrm{dom}(A_q)$. The structure of A_q is the same as M_g except that we

- add a new initial state ι_q and set $\delta_q(\iota_q, \vdash, \Sigma^\omega) = (q, \varepsilon, +1)$,
- remove transitions from M_g where the output is $\gamma \neq \varepsilon$,
- remove transitions from M_g where the input symbol is $\vdash$.

We explain the construction of the ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}$ such that $[\![g \odot h]\!] = [\![\mathcal{A}]\!]$. The set of states of $\mathcal{A}$ are $Q_\mathcal{A} = Q_g \cup Q_h \cup \{\mathrm{reset}\}$. Backward transitions in $\mathcal{A}$ and M_g are the same: $\delta_\mathcal{A}(q,a,R) = (q', \gamma, -1)$ iff $\delta_g(q,a,R) = (q', \gamma, -1)$. Forward transitions of M_g are divided into two depending on the look ahead. If we have $\delta_g(q,a,R) = (q', \gamma, +1)$ in M_g for an $a \in \Sigma_\vdash$, then $\delta_\mathcal{A}(q,a,R \setminus L_{q'}) = (q', \gamma, +1)$ and $\delta_\mathcal{A}(q,a,R \cap L_{q'}) = (\mathrm{reset}, \gamma, +1)$. From the reset state, we go to the left until $\vdash$ is reached and then start running M_h. So, $\delta_\mathcal{A}(\mathrm{reset}, a, \Sigma^\omega) = (\mathrm{reset}, \varepsilon, -1)$ for all $a \in \Sigma$ and $\delta_\mathcal{A}(\mathrm{reset}, \vdash, R) = (q'', \gamma, +1)$ if $\delta_h(s_h, \vdash, R) = (q'', \gamma, +1)$. The accepting set is the same as the Muller accepting set $\mathcal{F}_h$ of M_h.

(3) **Cauchy product.** From the transducers M_f and M_g, we can construct a DFA $\mathcal{D}_f = (Q_f, \Sigma, \delta_f, s_f, F_f)$ that accepts $\mathrm{dom}(M_f)$ and a deterministic Muller automaton (DMA) $\mathcal{D}_g = (Q_g, \Sigma, \delta_g, s_g, \mathcal{F}_g)$ that accepts $\mathrm{dom}(M_g)$.

Now, the set L of words w having at least two factorizations $w = u_1v_1 = u_2v_2$ with $u_1, u_2 \in \mathrm{dom}(f)$, $v_1, v_2 \in \mathrm{dom}(g)$ and $u_1 \neq u_2$ is ω-regular. This is easy since L can be written as $L = \bigcup_{p \in F_f, q \in Q_g} L_p \cdot M_{p,q} \cdot R_q$ where

- $L_p \subseteq \Sigma^*$ is the regular set of words which admit a run in $\mathcal{D}_f$ from its initial state to state p,
- $M_{p,q} \subseteq \Sigma^*$ is the regular set of words which admit a run in $\mathcal{D}_f$ from state p to some final state in $\mathcal{D}_f$, and also admit a run in $\mathcal{D}_g$ from the initial state to some state q in $\mathcal{D}_g$,
- $R_q \subseteq \Sigma^\omega$ is the ω-regular set of words which (i) admit an *accepting* run from state q in $\mathcal{D}_g$ and also (ii) admit an *accepting* run in $\mathcal{D}_g$ from its initial state s_g.

Therefore, $\mathrm{dom}(f \boxdot g) = (\mathrm{dom}(f) \cdot \mathrm{dom}(g)) \setminus L$ is ω-regular.

First we construct an ω-1DMT$_{\mathsf{la}}$ $\mathcal{D}$ such that $\mathrm{dom}(\mathcal{D}) = \mathrm{dom}(f \boxdot g)$ and on an input word $w = uv$ with $u \in \mathrm{dom}(f)$ and $v \in \mathrm{dom}(g)$, it produces the output $u\#v$ where $\# \notin \Sigma$ is a new symbol. From its initial state while reading $\vdash$, $\mathcal{D}$ uses the look-ahead to check whether the input word w is in $\mathrm{dom}(f \boxdot g)$ or not. If yes, it moves right and enters the initial state of $\mathcal{D}_f$. If not, it goes to a sink state and rejects. While running $\mathcal{D}_f$, $\mathcal{D}$ copies each input letter to output. Upon reaching a final state of $\mathcal{D}_f$, we use the look-ahead $\mathrm{dom}(g)$ to see whether we should continue running $\mathcal{D}_f$ or we should switch to $\mathcal{D}_g$. Formally, if $\delta_f(q,a) = q' \in F_f$ the corresponding transitions of $\mathcal{D}$ are $\delta_\mathcal{D}(q,a,\mathrm{dom}(g)) = (s_g, a\#, +1)$ and $\delta_\mathcal{D}(q,a,\Sigma^\omega \setminus \mathrm{dom}(g)) = (q', a, +1)$.

While running $\mathcal{D}_g$, $\mathcal{D}$ copies each input letter to output. Accepting sets of $\mathcal{D}$ are the accepting sets of the DMA $\mathcal{D}_g$. Thus, $\mathcal{D}$ produces an output $u\#v$ for an input string $w = uv$ which is in $\mathrm{dom}(f \boxdot g)$ such that $u \in \mathrm{dom}(f)$ and $v \in \mathrm{dom}(g)$.

Next we construct an ω-2DMT$_{\mathsf{la}}$ $\mathcal{T}$ which takes input words of the form $u\#v$ with $u \in \Sigma^*$ and $v \in \Sigma^\omega$, runs M_f on u and M_g on v. To do so, u is traversed in either direction depending on M_f and the symbol # is interpreted as right end marker $\dashv$ for M_f. While simulating a transition of M_f moving right of $\dashv$, producing the output γ and reaching state q, there are two possibilities. If q is not a final state of M_f then $\mathcal{T}$ moves to the right of #, goes to some sink state and rejects. If q is a final state of M_f, then $\mathcal{T}$ stays on # producing the output γ and goes to the initial state of M_g. Then, $\mathcal{T}$ runs M_g on v interpreting # as $\vdash$. The Muller accepting set of $\mathcal{T}$ is same as M_g.

We construct an ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}$ as the composition of $\mathcal{D}$ and $\mathcal{T}$. Regular transformations are definable by ω-2DMT$_{\mathsf{la}}$ [4] and are closed under composition [10]. Thus the composition of an ω-1DMT$_{\mathsf{la}}$ and an ω-2DMT$_{\mathsf{la}}$ is an ω-2DMT$_{\mathsf{la}}$. We deduce that $\mathcal{A}$ is an ω-2DMT$_{\mathsf{la}}$. Moreover $[\![\mathcal{A}]\!] = [\![f \boxdot g]\!]$.

(4) ω-**iteration.** By the Remark 3.2, this is a derived operator and hence the result follows from the next case.

(5) **2-chained** ω-**iteration.** First we show that the set of words w in Σ^ω having an unambiguous decomposition $w = u_1u_2\cdots$ with $u_i \in K$ for each i is ω-regular. As in case (3) above, the language L of words w having at least two factorizations $w = u_1v_1 = u_2v_2$ with $u_1, u_2 \in K$, $v_1, v_2 \in K^\omega$ and $u_1 \neq u_2$ is ω-regular. Hence, $L' = K^* \cdot L$ is ω-regular and contains all words in Σ^ω having several factorizations as products of words in K. We deduce that $\Sigma^\omega \setminus L'$ is ω-regular.

As in case (3) above, we construct an ω-1DMT$_{\mathsf{la}}$ $\mathcal{D}$ which takes as input w and outputs $u_1\#u_2\#\cdots$ iff there is an unambiguous decomposition of w as $u_1u_2\cdots$ with each $u_i \in K$. We then construct an ω-2DMT $\mathcal{D}'$ that takes as input words of the form $u_1\#u_2\#\cdots$ with each $u_i \in \Sigma^*$ and produces $u_1u_2\#u_2u_3\#\cdots$.

Next we construct an ω-2DMT $\mathcal{T}$ that takes as input words of the form $w_1\#w_2\#\cdots$ with each $w_i \in \Sigma^*$ and runs M_f on each w_i from left to right. The transducer $\mathcal{T}$ interprets # as $\vdash$ (resp. $\dashv$) when it is reached from the right (resp. from left). While simulating a transition of M_f moving right of $\dashv$, we proceed as in case (3) above, except that $\mathcal{T}$ goes to the initial state of M_f instead.

The ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}$ is obtained as the composition of $\mathcal{D}$, $\mathcal{D}'$ and $\mathcal{T}$. The output produced by $\mathcal{A}$ is $[\![M_f]\!](u_1u_2)[\![M_f]\!](u_2u_3)\cdots$. □

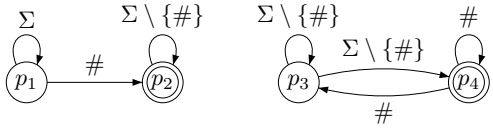
5 Backward deterministic Büchi automata

A Büchi automaton over the input alphabet Σ is a tuple $\mathcal{B} = (P, \Sigma, \Delta, \mathsf{Fin})$ where P is a finite set of states, $\mathsf{Fin} \subseteq P$ is the set of final (accepting) states, and $\Delta \subseteq P \times \Sigma \times P$ is the transition relation. A run of $\mathcal{B}$ over an infinite word $w = a_1a_2a_3\cdots$ is a sequence $\rho = p_0, a_1, p_1, a_2, p_2, \ldots$ such that $(p_{i-1}, a_i, p_i) \in \Delta$ for all $i \geq 1$. The run is final (accepting) if $\mathrm{inf}(\rho) \cap \mathsf{Fin} \neq \emptyset$ where $\mathrm{inf}(\rho)$ is the set of states visited infinitely often by ρ.

The Büchi automaton $\mathcal{B}$ is *complete unambiguous* (CUBA) [8] also called *backward deterministic* (BDBA) in [19] if for all infinite words $w \in \Sigma^\omega$, there is *exactly one* run ρ of $\mathcal{B}$ over w which is final, this run is denoted $\mathcal{B}(w)$. The fact that we request at least/most one final run on w explains why the automaton is called complete/unambiguous. Wlog, we may assume that all states of $\mathcal{B}$ are *useful*, i.e., for all $p \in P$ there exists some $w \in \Sigma^\omega$ such that $\mathcal{B}(w)$ starts from state p. In that case, it is easy to check that the transition relation is *backward deterministic and complete*: for all $(p, a) \in P \times \Sigma$ there is exactly one state p' such that $(p', a, p) \in \Delta$. We write $p' \xleftarrow{a} p$ and state p' is denoted $\Delta^{-1}(p, a)$. In other words, the inverse of the transition relation $\Delta^{-1} : P \times \Sigma \to P$ is a total function.

For each state $p \in P$, we let $\mathcal{L}(\mathcal{B}, p)$ be the set of infinite words $w \in \Sigma^\omega$ such that $\mathcal{B}(w)$ starts from p. For every subset $I \subseteq P$ of initial states, the language $\mathcal{L}(\mathcal{B}, I) = \bigcup_{p\in I} \mathcal{L}(\mathcal{B}, p)$ is ω-regular.

Example 5.1. For instance, the automaton $\mathcal{B}$ below is a BDBA. Morover, we have $\mathcal{L}(\mathcal{B}, p_2) = (\Sigma \setminus \{\#\})^\omega$, $\mathcal{L}(\mathcal{B}, p_4) = (\#\Sigma^*)^\omega$, and $\mathcal{L}(\mathcal{B}, \{p_1, p_3, p_4\}) = \Sigma^*\#\Sigma^\omega$.



Deterministic Büchi automata (DBA) are strictly weaker than non-deterministic Büchi automata (NBA) but backward determinism keeps the full expressive power.

Theorem 5.2 (Carton & Michel [8])**.** *A language $L \subseteq \Sigma^\omega$ is ω-regular iff $L = \mathcal{L}(\mathcal{B}, I)$ for some* BDBA $\mathcal{B}$ *and initial set I.*

The proof in [8] is constructive, starting with an NBA with m states, they construct an equivalent BDBA with $(3m)^m$ states.

A crucial fact on BDBA is that they are easily closed under boolean operations. In particular, the complement, which is quite difficult for NBAs, becomes trivial with BDBAs: $\mathcal{L}(\mathcal{B}, P \setminus I) = \Sigma^\omega \setminus \mathcal{L}(\mathcal{B}, I)$. For intersection and union, we simply use the classical cartesian product of two automata $\mathcal{B}_1$ and $\mathcal{B}_2$. This clearly preserves the backward determinism. For intersection, we use a generalized Büchi acceptance condition, i.e., a conjunction of Büchi acceptance conditions. For BDBAs, generalized and classical Büchi acceptance conditions are equivalent [8]. We obtain immediately

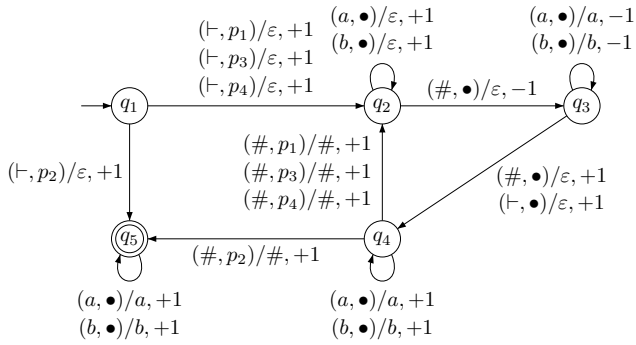
Corollary 5.3. *Let $\mathcal{R}$ be a finite family of ω-regular languages. There is a* BDBA $\mathcal{B}$ *and a tuple of initial sets $(I_R)_{R\in\mathcal{R}}$ such that $R = \mathcal{L}(\mathcal{B}, I_R)$ for all $R \in \mathcal{R}$.*

6 Replace the look-ahead with BDBA

Let $\mathcal{A} = (Q, \Sigma, \Gamma, q_0, \delta, \mathcal{F}, \mathcal{R})$ be an ω-2DMT$_{\mathsf{la}}$. By Corollary 5.3 there is a BDBA $\mathcal{B} = (P, \Sigma, \Delta, \mathsf{Fin})$ and a tuple $(I_R)_{R\in\mathcal{R}}$ of initial sets for the family $\mathcal{R}$ of ω-regular languages used as look-ahead by the automaton $\mathcal{A}$. For every pair $(q, a) \in Q \times \Sigma_\vdash$, the subset $\mathcal{R}(q, a)$ of languages $R \in \mathcal{R}$ such that $\delta(q, a, R)$ is defined forms a partition of Σ^ω. We deduce that $(I_R)_{R\in\mathcal{R}(q,a)}$ is a partition of P.

We generalize to two-way transducers on infinite words the notion of bimachine introduced in [17] for one-way transducers on finite words and studied in [7] for one-way transducers on infinite words. We construct an ω-2DMT $\widetilde{\mathcal{A}} = (Q, \widetilde{\Sigma}, \Gamma, q_0, \widetilde{\delta}, \mathcal{F})$ *without look-ahead* over the extended alphabet $\widetilde{\Sigma} = \Sigma \times P$ which is equivalent to $\mathcal{A}$ in some sense made precise below. Intuitively, in a pair $(a, p) \in \widetilde{\Sigma_\vdash}$, the state p of $\mathcal{B}$ gives the look-ahead information required by $\mathcal{A}$. Formally, the deterministic transition function $\widetilde{\delta} : Q \times \widetilde{\Sigma_\vdash} \to Q \times \Gamma^* \times \{-1, +1\}$ is defined for $q \in Q$ and $(a, p) \in \widetilde{\Sigma_\vdash}$ by $\widetilde{\delta}(q, (a, p)) = \delta(q, a, R)$ for the unique $R \in \mathcal{R}(q, a)$ such that $p \in I_R$.

Example 6.1. For instance, the automaton $\widetilde{\mathcal{A}}$ constructed from the automaton on the right of Figure 1 and the BDBA $\mathcal{B}$ of Example 5.1 is depicted bellow ($\bullet$ stands for an arbitrary state of $\mathcal{B}$).



Let $w = a_1a_2a_3\cdots \in \Sigma^\omega$ and $\mathcal{B}(w) = p_0, a_1, p_1, a_2, p_2, \ldots$ be the unique final run of $\mathcal{B}$ on w. Let $\widetilde{\vdash w} = (\vdash, p_0)(a_1, p_1)(a_2, p_2)\cdots \in \widetilde{\Sigma_\vdash}^\omega$. We easily check by induction that the unique run of $\mathcal{A}$ on w

$$q_0 \vdash w \xrightarrow{\gamma_1} w_1' q_1 w_1'' \xrightarrow{\gamma_2} w_2' q_2 w_2'' \xrightarrow{\gamma_3} w_3' q_3 w_3'' \xrightarrow{\gamma_4} w_4' q_4 w_4'' \cdots$$

corresponds to the unique run of $\widetilde{\mathcal{A}}$ on $\widetilde{\vdash w}$

$$q_0 \widetilde{\vdash w} \xrightarrow{\gamma_1} \widetilde{w_1'} q_1 \widetilde{w_1''} \xrightarrow{\gamma_2} \widetilde{w_2'} q_2 \widetilde{w_2''} \xrightarrow{\gamma_3} \widetilde{w_3'} q_3 \widetilde{w_3''} \xrightarrow{\gamma_4} \widetilde{w_4'} q_4 \widetilde{w_4''} \cdots$$

where for all $i > 0$ we have $\widetilde{\vdash w} = \widetilde{w_i'}\widetilde{w_i''}$ and $|w_i'| = |\widetilde{w_i'}|$. Indeed, assume that in a configuration $w'qaw''$ with $\vdash w = w'aw''$ the transducer $\mathcal{A}$ takes the transition $q \xrightarrow{(a,R)} (q', \gamma, +1)$ and reaches configuration $w'aq'w''$. Then, $w'' \in R$ and the corresponding configuration $\widetilde{w'}q(a,p)\widetilde{w''}$ with $\widetilde{\vdash w} = \widetilde{w'}(a,p)\widetilde{w''}$ and $|w'| = |\widetilde{w'}|$ is such that $p \in I_R$. Therefore, the transducer $\widetilde{\mathcal{A}}$ takes the transition $q \xrightarrow{(a,p)} (q', \gamma, +1)$ and reaches configuration $\widetilde{w'}(a,p)q'\widetilde{w''}$. The proof is similar for backward transitions. We have shown that $\mathcal{A}$ and $\widetilde{\mathcal{A}}$ are equivalent in the following sense:

Lemma 6.2. *For all words $w \in \Sigma^\omega$, the transducer $\mathcal{A}$ starting from $\vdash w$ accepts iff the transducer $\widetilde{\mathcal{A}}$ starting from $\widetilde{\vdash w}$ accepts, and in this case they compute the same output in Γ^∞.*

7 Transition monoid of an ω-2DMT$_{\text{la}}$

We use the notations of the previous sections, in particular for the ω-2DMT$_{\text{la}}$ $\mathcal{A}$, the BDBA $\mathcal{B}$ and the corresponding ω-2DMT $\widetilde{\mathcal{A}}$. As in the case of 2NFAs over finite words, we will define a congruence on Σ^+ such that two words $u, v \in \Sigma^+$ are equivalent iff they behave the same in the ω-2DMT$_{\text{la}}$ $\mathcal{A}$, when placed in an arbitrary right context $w \in \Sigma^\omega$. The right context w is abstracted with the first state p of the unique final run $\mathcal{B}(w)$.

The ω-2DMT $\widetilde{\mathcal{A}}$ does not use look-ahead, hence, we may use for $\widetilde{\mathcal{A}}$ the classical notion of transition monoid. Actually, in order to handle the Muller acceptance condition of $\widetilde{\mathcal{A}}$, we need a slight extension of this classical transition monoid. More precisely, the abstraction of a finite word $\widetilde{u} \in \widetilde{\Sigma}^+$ will be the set $\widetilde{\mathsf{Tr}}(\widetilde{u})$ of tuples (q, d, X, q') with $q, q' \in Q$, $X \subseteq Q$ and $d \in \{\rightarrow, \rightsquigarrow, \subsetneq, \leftarrow\}$ such that the unique run of $\widetilde{\mathcal{A}}$ on $\widetilde{u}$ starting in state q on the left of $\widetilde{u}$ if $d \in \{\rightarrow, \rightsquigarrow\}$ (resp. on the right if $d \in \{\subsetneq, \leftarrow\}$) exits in state q' on the left of $\widetilde{u}$ if $d \in \{\rightsquigarrow, \leftarrow\}$ (resp. on the right if $d \in \{\rightarrow, \subsetneq\}$) and visits the set of states X *while in* $\widetilde{u}$ (i.e., including q but not q' unless q' is also visited before the run exits $\widetilde{u}$).

For instance, with the automaton $\widetilde{\mathcal{A}}$ of Example 6.1, we have $(q_4, \rightarrow, \{q_2, q_3, q_4\}, q_5) \in \widetilde{\mathsf{Tr}}(\widetilde{u})$ when the word $\widetilde{u}$ belongs to $((a,p_1) + (b,p_1))^*(\#, p_1)((a,p_1) + (b,p_1))^*(\#, p_2)$.

We denote by $\widetilde{\mathsf{TrM}} = \{\widetilde{\mathsf{Tr}}(\widetilde{u}) \mid \widetilde{u} \in \widetilde{\Sigma}^+\} \cup \{\mathbf{1}_{\widetilde{\mathsf{TrM}}}\}$ the transition monoid of $\widetilde{\mathcal{A}}$ with unit $\mathbf{1}_{\widetilde{\mathsf{TrM}}}$. The classical product is extended by taking the union of the sets X occurring in a sequence of steps. For instance, if we have steps $(q_0, \rightarrow, X_1, q_1)$, $(q_2, \subsetneq, X_3, q_3)$, $\ldots$, $(q_{i-1}, \subsetneq, X_i, q_i)$ in $\widetilde{\mathsf{Tr}}(\widetilde{u})$ and $(q_1, \rightsquigarrow, X_2, q_2)$, $(q_3, \rightsquigarrow, X_4, q_4)$, $\ldots$, $(q_i, \rightarrow, X_{i+1}, q_{i+1})$ in $\widetilde{\mathsf{Tr}}(\widetilde{v})$ then there is a step $(q_0, \rightarrow, X_1 \cup \cdots \cup X_{i+1}, q_{i+1})$ in $\widetilde{\mathsf{Tr}}(\widetilde{u}\cdot\widetilde{v}) = \widetilde{\mathsf{Tr}}(\widetilde{u}) \cdot \widetilde{\mathsf{Tr}}(\widetilde{v})$. We denote by $\widetilde{\mathsf{Tr}}\colon \widetilde{\Sigma}^* \to \widetilde{\mathsf{TrM}}$ the canonical morphism.

Let $u = a_1 \cdots a_n \in \Sigma^+$ be a finite word of length $n > 0$ and let $p \in P$. We define the sequence of states $p_0, p_1, \ldots, p_n$ by $p_n = p$ and for all $0 \le i < n$ we have $p_i \xleftarrow{a_{i+1}} p_{i+1}$ in $\mathcal{B}$. Notice that for all infinite words $w \in \mathcal{L}(\mathcal{B}, p)$, the unique run $\mathcal{B}(uw)$ starts with $p_0, a_1, p_1, \ldots, a_n, p_n$. Define $\widetilde{u^p} = (a_1, p_1)(a_2, p_2)\cdots(a_n, p_n) \in \widetilde{\Sigma}^+$.

We are now ready to define the finite abstraction $\mathsf{Tr}(u)$ of a finite word $u \in \Sigma^+$ with respect to the pair $(\mathcal{A}, \mathcal{B})$: we let $\mathsf{Tr}(u) = (r^p, b^p, s^p)_{p\in P}$ where for each $p \in P$, $s^p = \widetilde{\mathsf{Tr}}(\widetilde{u^p}) \in \widetilde{\mathsf{TrM}}$ is the abstraction of $\widetilde{u^p}$ with respect to $\widetilde{\mathcal{A}}$, $r^p \in P$ is the unique state of $\mathcal{B}$ such that $r^p \xleftarrow{u} p$, $b^p = 1$ if the word $\widetilde{u^p}$ contains a final state of $\mathcal{B}$ and $b^p = 0$ otherwise.

The transition monoid of $(\mathcal{A}, \mathcal{B})$ is the set $\mathsf{TrM} = \{\mathsf{Tr}(u) \mid u \in \Sigma^+\} \cup \{\mathbf{1}_{\mathsf{TrM}}\}$ where $\mathbf{1}_{\mathsf{TrM}}$ is the unit. The product of $\sigma_1 = (r_1^p, b_1^p, s_1^p)_{p\in P}$ and $\sigma = (r^p, b^p, s^p)_{p\in P}$ is defined to be $\sigma_1 \cdot \sigma = (r_1^{r^p}, b_1^{r^p} \vee b^p, s_1^{r^p} \cdot s^p)_{p\in P}$. We can check that this product is associative, so that $(\mathsf{TrM}, \cdot, \mathbf{1}_{\mathsf{TrM}})$ is a monoid. Moreover, let $u, v \in \Sigma^+$ be such that $\mathsf{Tr}(u) = \sigma_1$ and $\mathsf{Tr}(v) = \sigma$. For each $p \in P$, we can check that $\widetilde{uv}^p = \widetilde{u}^{r^p} \cdot \widetilde{v}^p$. We deduce easily that $\mathsf{Tr}(uv) = \sigma_1 \cdot \sigma = \mathsf{Tr}(u) \cdot \mathsf{Tr}(v)$. Therefore, $\mathsf{Tr}\colon \Sigma^* \to \mathsf{TrM}$ is a morphism.

8 ω-2DMT$_{\text{la}}$ to ω-RTE

We prove that from an ω-2DMT$_{\text{la}}$ $\mathcal{A}$ we can construct an equivalent ω-RTE. We use the fact that any word in the domain of $\mathcal{A}$ can be factorized unambiguously into a good rational expression. For rational expressions we use the syntax: $F ::= \emptyset \mid \varepsilon \mid a \mid F \cup F \mid F \cdot F \mid F^+$ where $a \in \Sigma$. An expression is ε-free if it does not use ε.

Let $(S, \cdot, \mathbf{1}_S)$ be a *finite* monoid and $\varphi\colon \Sigma^* \to S$ be a morphism. We say that a rational expression F is *φ-good* (or simply *good* when φ is clear from the context) when (1) the rational expression F is unambiguous, (2) for each subexpression E of F we have $\varphi(\mathcal{L}(E)) = \{s_E\}$ is a singleton set, and (3) for each subexpression E^+ of F we have $s_E \cdot s_E = s_E$ is an idempotent. Notice that $\emptyset$ cannot be used in a good expression since it does not satisfy the second condition.

Theorem 8.1 (Unambiguous Forest Factorization [16])**.** *Let $\varphi\colon \Sigma^* \to S$ be a morphism to a finite monoid $(S, \cdot, \mathbf{1}_S)$. There is an* unambiguous *rational expression $G = \bigcup_{k=1}^m F_k \cdot G_k^\omega$ over Σ such that $\mathcal{L}(G) = \Sigma^\omega$ and for all $1 \le k \le m$, F_k and G_k are ε-free φ-good rational expressions and s_{G_k} is an idempotent, where $\varphi(G_k) = \{s_{G_k}\}$.*

Theorem 8.1 can be seen as an unambiguous version of Imre Simon's forest factorization theorem [18]. Its proof follows the same lines of the recent proofs of Simon's theorem, see e.g. [9].

We will apply this theorem to the morphism $\mathsf{Tr}\colon \Sigma^* \to \mathsf{TrM}$ defined in Section 7. We use the unambiguous expression $G = \bigcup_{k=1}^m F_k \cdot G_k^\omega$ as a *guide* when constructing ω-RTEs corresponding to the ω-2DMT$_{\text{la}}$ $\mathcal{A}$. The first condition of good expressions ensures an

unambiguous parsing of the input word, and therefore functionality of the transformation. As will be clear from the lemma below and its proof, the second and third conditions are essential to compute the RTEs, especially when dealing with the Kleene-plus.

Lemma 8.2. *Let G be an ε-free* Tr*-good rational expression and let* $\mathsf{Tr}(G) = \sigma_G = (r_G^p, b_G^p, s_G^p)_{p\in P}$ *be the corresponding element of the transition monoid* TrM *of* $(\mathcal{A}, \mathcal{B})$*. For each state $p \in P$, we can construct a map $C_G^p \colon s_G^p \to$ RTE such that for each step $x = (q, d, X, q') \in s_G^p$ the following invariants hold:*

- (J_1) $\mathsf{dom}(C_G^p(x)) = \mathcal{L}(G)$,
- (J_2) *for each $u \in \mathcal{L}(G)$, $[\![C_G^p(x)]\!](u)$ is the output produced by $\widetilde{\mathcal{A}}$ when running step x on $\widetilde{u}^p$ (i.e., running $\widetilde{\mathcal{A}}$ on $\widetilde{u}^p$ from q to q' following direction d).*

Proof. The proof is by induction on the rational expression. For each subexpression E of G we let $\mathsf{Tr}(E) = \sigma_E = (r_E^p, b_E^p, s_E^p)_{p\in P}$ be the corresponding element of the transition monoid TrM of $(\mathcal{A}, \mathcal{B})$. We start with atomic regular expressions. Since G is ε-free and $\emptyset$-free, we do not need to consider $E = \varepsilon$ or $E = \emptyset$. The interesting cases are concatenation and Kleene-plus.

Atomic. Assume that $E = a \in \Sigma$ is an atomic subexpression. Notice that $\widetilde{a}^p = (a, p)$ for all $p \in P$. Since the ω-2DMT $\widetilde{\mathcal{A}}$ is deterministic and complete, for each state $q \in Q$ we have

- either $\widetilde{\delta}(q, (a, p)) = (q', \gamma, 1)$ and we let $C_a^p((q, \to, \{q\}, q')) = C_a^p((q, \subsetneq, \{q\}, q')) = a \,?\, \gamma : \bot$,
- or $\widetilde{\delta}(q, (a, p)) = (q', \gamma, -1)$ and we let $C_a^p((q, \supsetneq, \{q\}, q')) = C_a^p((q, \leftarrow, \{q\}, q')) = a \,?\, \gamma : \bot$.

Clearly, invariants (J_1) and (J_2) hold for all $x \in s_E^p$.

Union. Assume that $E = E_1 \cup E_2$. Since E is good, we deduce that $\sigma_E = \sigma_{E_1} = \sigma_{E_2}$. For each $p \in P$ and $x \in s_E^p$ we define $C_E^p(x) = E_1 \,?\, C_{E_1}^p(x) : C_{E_2}^p(x)$. Since E is unambiguous we have $\mathcal{L}(E_1) \cap \mathcal{L}(E_2) = \emptyset$. We can prove easily that invariants (J_1) and (J_2) hold for all $x \in s_E^p$.

Concatenation. Assume that $E = E_1 \cdot E_2$ is a concatenation. Since E is good, we deduce that $\sigma_E = \sigma_{E_1} \cdot \sigma_{E_2}$. Let $p \in P$ and $p_1 = r_{E_2}^p$. We have $s_E^p = s_{E_1}^{p_1} \cdot s_{E_2}^p$. Let $x \in s_E^p$.

If $x = (q, \to, X, q')$ then, by definition of the product in $\widetilde{\mathsf{TrM}}$, there is a unique sequence of steps $x_1 = (q, \to, X_1, q_1)$, $x_2 = (q_1, \supsetneq, X_2, q_2)$, $x_3 = (q_2, \subsetneq, X_3, q_3)$, $x_4 = (q_3, \supsetneq, X_4, q_4)$, …, $x_i = (q_{i-1}, \subsetneq, X_i, q_i)$, $x_{i+1} = (q_i, \to, X_{i+1}, q')$ with $i \geq 1$, $x_1, x_3, \ldots, x_i \in s_{E_1}^{p_1}$ and $x_2, x_4, \ldots, x_{i+1} \in s_{E_2}^p$ and $X = X_1 \cup \cdots \cup X_{i+1}$ (Figure 3 top left). We define $C_E^p(x) = (C_{E_1}^{p_1}(x_1) \boxdot C_{E_2}^p(x_2)) \odot (C_{E_1}^{p_1}(x_3) \boxdot C_{E_2}^p(x_4)) \odot \cdots \odot (C_{E_1}^{p_1}(x_i) \boxdot C_{E_2}^p(x_{i+1}))$. Note that when $i = 1$, we have $C_E^p(x) = C_{E_1}^{p_1}(x_1) \boxdot C_{E_2}^p(x_2)$ with $x_2 = (q_1, \to, X_2, q')$.

The concatenation $\mathcal{L}(E) = \mathcal{L}(E_1) \cdot \mathcal{L}(E_2)$ is unambiguous. We obtain $\mathsf{dom}(C_{E_1}^{p_1}(y) \boxdot C_{E_2}^p(z)) = \mathcal{L}(E)$ for all $y \in s_{E_1}^{p_1}$ and $z \in s_{E_2}^p$, using (J_1) for E_1 and E_2. We deduce $\mathsf{dom}(C_E(x)) = \mathcal{L}(E)$ and (J_1) holds for E and $x = (q, \to, X, q')$.

Now, let $u \in \mathcal{L}(E)$ and let $u = u_1 u_2$ be its unique factorization with $u_1 \in \mathcal{L}(E_1)$ and $u_2 \in \mathcal{L}(E_2)$. We have $\widetilde{u_1 u_2}^p = \widetilde{u_1}^{p_1} \cdot \widetilde{u_2}^p$. Hence, the step $x = (q, \to, X, q')$ performed by $\widetilde{\mathcal{A}}$ on $\widetilde{u}^p$ is actually the concatenation of steps x_1 on $\widetilde{u_1}^{p_1}$, followed by x_2 on $\widetilde{u_2}^p$, followed by x_3 on $\widetilde{u_1}^{p_1}$, followed by x_4 on $\widetilde{u_2}^p$, …, until x_{i+1} on $\widetilde{u_2}^p$. Using (J_2) for E_1 and E_2, we deduce that the output produced by $\widetilde{\mathcal{A}}$ while making step x on $\widetilde{u}^p$ is $[\![C_{E_1}^{p_1}(x_1)]\!](u_1) \cdot [\![C_{E_2}^p(x_2)]\!](u_2) \cdots [\![C_{E_1}^{p_1}(x_i)]\!](u_1) \cdot [\![C_{E_2}^p(x_{i+1})]\!](u_2) = [\![C_E^p(x)]\!](u)$. Therefore, (J_2) holds for E and step $x = (q, \to, X, q')$. The proof is obtained mutatis mutandis for the other cases $x = (q, \supsetneq, X, q')$ or $x = (q, \subsetneq, X, q')$ or $x = (q, \leftarrow, X, q')$.

Kleene-plus. Assume that $E = F^+$. Since E is good, we deduce that $\sigma_E = \sigma_F = \sigma = (r^p, b^p, s^p)_{p\in P}$ is an idempotent of the transition monoid TrM. Notice that for all $p \in P$, since σ is an idempotent, we have $r^{r^p} = r^p$.

We first define C_E^p for states $p \in P$ with $p = r^p$. Let $x \in s^p$.

- If $x = (q, \supsetneq, X, q')$. Since F^+ is unambiguous, a word $u \in \mathcal{L}(F^+)$ admits a unique factorization $u = u_1 \cdots u_n$ with $n \geq 1$ and $u_i \in \mathcal{L}(F)$. Now, $\mathsf{Tr}(u_i) = \sigma$ for all $1 \leq i \leq n$ and since $p = r^p$ we deduce that $\widetilde{u}^p = \widetilde{u_1}^p \widetilde{u_2}^p \cdots \widetilde{u_n}^p$. Since $x = (q, \supsetneq, X, q') \in s^p$, the unique run ρ of $\widetilde{\mathcal{A}}$ starting in state q on the left of $\widetilde{u_1}^p$ exits on the left in state q'. Therefore, the unique run of $\widetilde{\mathcal{A}}$ starting in state q on the left of $\widetilde{u}^p$ only visits $\widetilde{u_1}^p$ and is actually ρ itself. Hence, we set $C_E^p(x) = C_F^p(x) \boxdot (F^* \,?\, \varepsilon : \bot)$ and we can easily check that $(\mathsf{J}_1$–$\mathsf{J}_2)$ are satisfied.
- For $x = (q, \subsetneq, X, q')$ we set $C_E^p(x) = (F^* \,?\, \varepsilon : \bot) \boxdot C_F^p(x)$.
- If $x = (q, \to, X, q')$. Since σ is an idempotent, we have $x \in s^p \cdot s^p$. We distinguish two cases depending on whether the step $y \in s^p$ starting in state q' from the left goes to the right or goes back to the left.

  First, if $y = (q', \to, X_2, q_2) \in s^p$ goes to the right. Since s^p is an idempotent, following x in $s^p \cdot s^p$ is the same as following x in (the first) s^p an then y in (the second) s^p. Therefore, we must have $q_2 = q'$ and $X_2 \subseteq X$. In this case, we set $C_E^p(x) = F \,?\, C_F^p(x) : \left(C_F^p(x) \boxdot (C_F^p(y))^{\boxplus}\right)$.

  Second, if $y = (q', \supsetneq, X_2, q_2) \in s^p$ goes to the left. Since s^p is an idempotent, there exists a unique sequence of steps in s^p: $x_1 = x$, $x_2 = y$, $x_3 = (q_2, \subsetneq, X_3, q_3)$, $x_4 = (q_3, \supsetneq, X_4, q_4)$, …, $x_i = (q_{i-1}, \subsetneq, X_i, q_i)$, $x_{i+1} = (q_i, \to, X_{i+1}, q')$ with $i \geq 3$ (see Figure 3 bottom left). Let $C_E^p(x) = \left(C_F^p(x) \boxdot (F^* \,?\, \varepsilon : \bot)\right) \odot [F, C']^{2^{\boxplus}}$, $C' = \left((F \,?\, \varepsilon : \bot) \boxdot C_F^p(x_2)\right) \odot (C_F^p(x_3) \boxdot C_F^p(x_4)) \odot \cdots \odot (C_F^p(x_i) \boxdot C_F^p(x_{i+1}))$. The proof of correctness, i.e., that $(\mathsf{J}_1$–$\mathsf{J}_2)$ are satisfied for E, can be found in [11].
- If $x = (q, \leftarrow, X, q')$, the proof is the same, using the backward unambiguous (2-chained) Kleene-plus $C^{\overleftarrow{\boxplus}}$ and $[K, C]^{2^{\overleftarrow{\boxplus}}}$.

Now, we consider $p \in P$ with $r^p \neq p$. We let $p' = r^p$. We have already noticed that since σ is idempotent we have $r^{p'} = p'$. Consider a word $u \in \mathcal{L}(F^+)$. Since F^+ is unambiguous, u admits a unique factorization $u = u_1 \cdots u_{n-1} u_n$ with $n \geq 1$ and $u_i \in \mathcal{L}(F)$. Now, $\mathsf{Tr}(u_i) = \sigma$ for all $1 \leq i \leq n$. Using $r^p = p'$ and $r^{p'} = p'$ we deduce that $\widetilde{u}^p = \widetilde{u_1}^{p'} \cdots \widetilde{u_{n-1}}^{p'} \widetilde{u_n}^p$. So when $n > 1$, the expression C_E^p that we need to compute is like the concatenation of $C_E^{p'}$ on the first $n-1$ factors with C_F^p on the last factor. Since $r^{p'} = p'$ we have already seen how to compute $C_E^{p'}$. We also know how to handle concatenation. So it should be no surprise that we can compute C_E^p when $p \neq r^p$. We define now formally $C_E^p(x)$ for $x \in s^p$.

- If $x = (q, \supsetneq, X, q') \in s^p$. There are two cases depending on whether the step $y \in s^{p'}$ starting in state q from the left goes back to the left or goes to the right.

  If it goes back to the left, then $y = (q, \supsetneq, X, q') = x$ since $s^p = s^{p'} \cdot s^p$ (recall that σ is idempotent) and we define $C_E^p(x) = F \,?\, C_F^p(x) \;:\; (C_F^{p'}(x) \boxdot (F^+ \,?\, \varepsilon : \bot))$. If it goes to the right, then

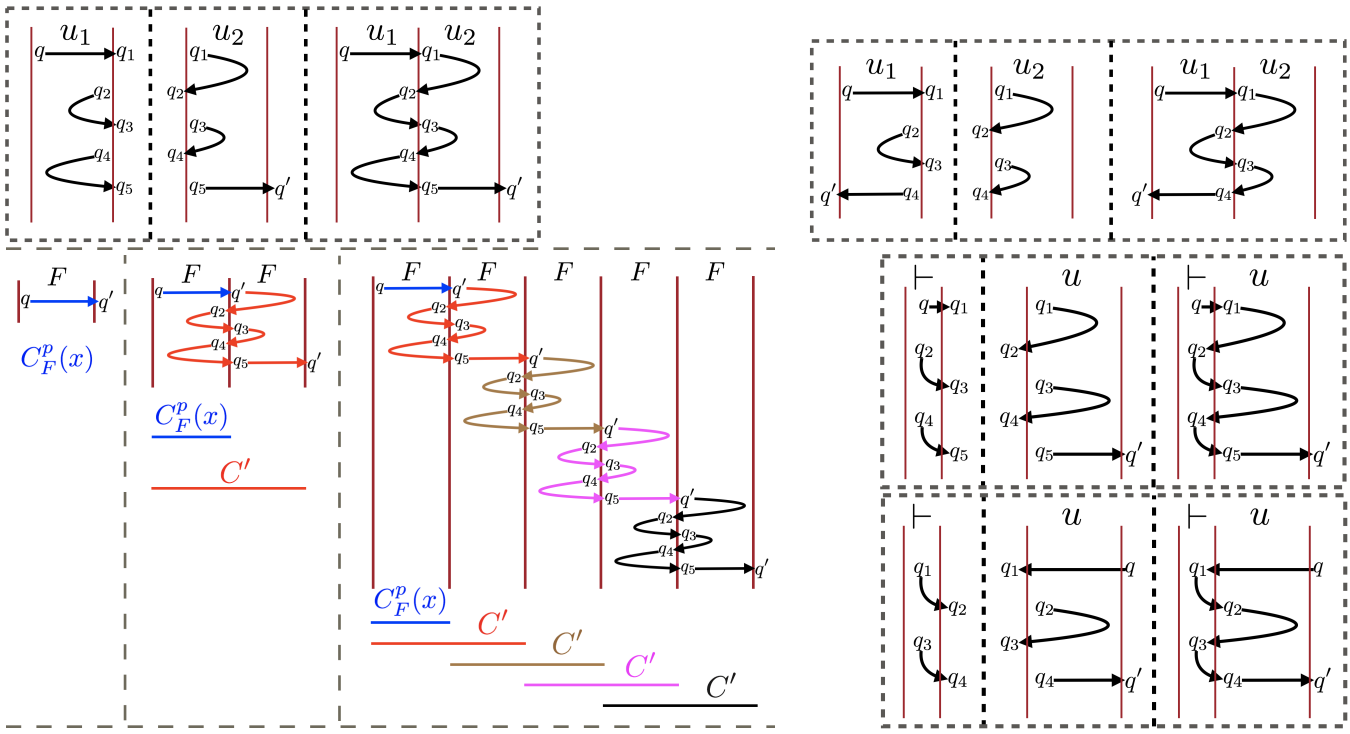


Figure 3

$y = (q, \rightarrow, X_1, q_1)$ and there exists a unique sequence of steps: $x_1 = y$, $x_2 = (q_1, \rightsquigarrow, X_2, q_2)$, $x_3 = (q_2, \leadsto, X_3, q_3)$, $x_4 = (q_3, \rightsquigarrow, X_4, q_4)$, $\ldots$, $x_i = (q_{i-1}, \leftarrow, X_i, q')$ with $i \geq 3$, $x_1, x_3, \ldots, x_i \in s^{p'}$ and $x_2, \ldots, x_{i-1} \in s^p$ (see Figure 3 top right). Notice that $X = X_1 \cup \cdots \cup X_i$. We define $C_E^p(x) = F\,?\,C_F^p(x) : C'$ where $C' = (C_E^{p'}(x_1) \boxdot C_F^p(x_2)) \odot \cdots \odot (C_E^{p'}(x_{i-2}) \boxdot C_F^p(x_{i-1})) \odot \big(C_E^{p'}(x_i) \boxdot (F\,?\,\varepsilon : \bot)\big)$. We can check that ($\mathsf{J}_1$–$\mathsf{J}_2$) are satisfied for (E, p, x).

- If $x = (q, \leftarrow, X, q') \in s^p$. There are two cases depending on whether the step $y \in s^{p'}$ starting in state q' from the right goes to the left or goes back to the right.

  If it goes to the left, then $y = (q', \leftarrow, X', q')$ with $X' \subseteq X$ and we define $C_E^p(x) = F\,?\,C_F^p(x) : (C_E^{p'}(y) \overleftarrow{\boxdot} C_F^p(x))$.

  If it goes back to the right, then $y = (q', \leadsto, X_2, q_2)$ and there is a unique sequence of steps: $x_1 = x$, $x_2 = y$, $x_3 = (q_2, \rightsquigarrow, X_3, q_3)$, $x_4 = (q_3, \leadsto, X_4, q_4)$, $\ldots$, $x_i = (q_{i-1}, \rightsquigarrow, X_i, q_i)$ $x_{i+1} = (q_i, \leftarrow, X_{i+1}, q')$ with $i \geq 3$, $x_1, x_3, \ldots, x_i \in s^p$ and $x_2, \ldots, x_{i+1} \in s^{p'}$. Note that $X_2 \cup \cdots \cup X_{i+1} \subseteq X$. Define $C_E^p(x) = F\,?\,C_F^p(x) : C'$ where $C' = (C_E^{p'}(x_2) \overleftarrow{\boxdot} C_F^p(x_1)) \odot \cdots \odot (C_E^{p'}(x_{i-1}) \overleftarrow{\boxdot} C_F^p(x_{i-2})) \odot (C_E^{p'}(x_{i+1}) \overleftarrow{\boxdot} C_F^p(x_i))$. We can check that ($\mathsf{J}_1$–$\mathsf{J}_2$) are satisfied for (E, p, x).
- The cases $x = (q, \rightarrow, X, q') \in s^p$ and $x = (q, \leadsto, X, q') \in s^p$ can be handled similarly. □

We now define RTEs corresponding to the left part of the computation of the ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}$, i.e., on some input $\vdash u$ consisting of the left end-marker and some finite word $u \in \Sigma^+$ (see Figure 3 middle and bottom right). As before, the look-ahead is determined by the state of the BDBA $\mathcal{B}$. Proof of Lemma 8.3 can be found in [11].

Lemma 8.3. *Let F be an ε-free* Tr*-good rational expression. For each state $p \in P$ and $q \in Q$, there is a unique state $q' \in Q$ and an RTE $C_{\vdash F}^p(q, \rightarrow, q')$ (resp. $C_{\vdash F}^p(q, \leadsto, q')$) such that the following invariants hold: (i) $\mathcal{L}(F) = \mathrm{dom}(C_{\vdash F}^p(q, \rightarrow, q'))$ (resp. $\mathcal{L}(F) = \mathrm{dom}(C_{\vdash F}^p(q, \leadsto, q'))$), (ii) for each $u \in \mathcal{L}(F)$, $[\![C_{\vdash F}^p(q, \rightarrow, q')]\!](u)$ (resp. $[\![C_{\vdash F}^p(q, \leadsto, q')]\!](u)$) is the output produced by $\widetilde{\mathcal{A}}$ on $\widetilde{\vdash u^p}$ when starting on the left (resp. right) in state q until it exists on the right in state q'.*

Lemma 8.4. *Let $F \cdot G^\omega$ be an unambiguous rational expression such that F and G are ε-free* Tr*-good rational expresions and* Tr$(G) = \sigma = (r^p, b^p, s^p)_{p \in P}$ *is an idempotent in the transition monoid* TrM *of $(\mathcal{A}, \mathcal{B})$. We construct an ω-RTE C_{FG^ω} s.t.* dom$(C_{FG^\omega}) = \mathcal{L}(FG^\omega) \cap$ dom$(\mathcal{A})$ *and $[\![C_{FG^\omega}]\!](w) = [\![\mathcal{A}]\!](w)$ for all $w \in$ dom(C_{FG^ω}).*

Proof. We first show that there exists one and only one state $p \in P$ such that $r^p = p$ and $b^p = 1$. For the existence, consider a word $w = u_1 u_2 u_3 \cdots \in \mathcal{L}(FG^\omega)$ with $u_1 \in \mathcal{L}(F)$ and $u_n \in \mathcal{L}(G)$ for all $n \geq 2$. By definition of BDBA there is a unique final run of $\mathcal{B}$ over w: $p_0, u_1, p_1, u_2, p_2, \ldots$. Let us show first that $p_n = p_1$ for all $n \geq 1$. Since σ is idempotent, we have Tr$(u_2 \cdots u_{n+1})$ = Tr(u_{n+1}). Since $p_1 \xleftarrow{u_2 \cdots u_{n+1}} p_{n+1}$ and $p_n \xleftarrow{u_{n+1}} p_{n+1}$, we deduce that $p_1 = r^{p_{n+1}} = p_n$. This implies $p_1 = r^{p_2} = r^{p_1}$. Let $p = p_1$ so that $p = r^p$ and the final run of $\mathcal{B}$ on w is $p_0, u_1, p, u_2, p, \ldots$. Now, for all $n \geq 2$ we have Tr$(u_n) = \sigma$ and we deduce that $p \xleftarrow{u_n} p$ visits a final state from Fin iff $b^p = 1$. Since the run is accepting, we deduce that indeed $b^p = 1$. To prove the unicity, let $p \in P$ with $p = r^p$, $b^p = 1$ and $v \in \mathcal{L}(G)$. We have $p \xleftarrow{v} p$ and this run visits a final state from Fin. Therefore, $p, v, p, v, p, v, p, \ldots$ is a final run of $\mathcal{B}$ on v^ω. Since $\mathcal{B}$ is BDBA, there is a unique final run of $\mathcal{B}$ on v^ω, proving unicity of p.

We apply Lemma 8.3. Denote by $s'_{\vdash F}$ the set of triples $(q, d, q') \in Q \times \{\rightarrow, \leadsto\} \times Q$ such that the RTE $C_{\vdash F}^p(q, d, q')$ is defined.

Starting from the initial state q_0 of $\mathcal{A}$, there exists a unique sequence of steps $x'_1 = (q_0, \to, q'_1)$, $x'_2 = (q'_1, \curvearrowright, X'_2, q'_2)$, $x'_3 = (q'_2, \subsetarrow, q'_3)$, $x'_4 = (q'_3, \curvearrowright, X'_4, q'_4)$, …, $x'_i = (q'_{i-1}, \subsetarrow, q'_i)$, $x'_{i+1} = (q'_i, \to, X'_{i+1}, q)$ with $i \geq 1$, $x'_1, x'_3, \ldots, x'_i \in s'_{\vdash F}$ and $x'_2, x'_4, \ldots, x'_{i+1} \in s^p$. We define

$$C_1 = \big(C^p_{\vdash F}(x'_1) \boxdot C^p_G(x'_2)\big) \odot \big(C^p_{\vdash F}(x'_3) \boxdot C^p_G(x'_4)\big) \odot \cdots \odot \big(C^p_{\vdash F}(x'_i) \boxdot C^p_G(x'_{i+1})\big),$$

$$C_2 = C_1 \boxdot (G^\omega \,?\, \varepsilon : \bot).$$

We have $\mathrm{dom}(C_1) = FG$ and $\widetilde{\vdash u_1 u_2}^p = \widetilde{\vdash u_1}^p \widetilde{u_2}^p$ for all $u_1 \in F$ and $u_2 \in G$. Moreover, $[\![C_1]\!](u_1u_2)$ is the output produced by $\widetilde{\mathcal{A}}$ on $\widetilde{\vdash u_1 u_2}^p$ when starting on the left in the initial state q_0 until it exists on the right in state q. Now, C_2 is an ω-RTE with $\mathrm{dom}(C_2) = FG^\omega$ and for all $w = u_1u_2u_3 \ldots \in FG^\omega$ with $u_1 \in F$ and $u_n \in G$ for all $n > 1$, we have $[\![C_2]\!](w) = [\![C_1]\!](u_1u_2) \in \Gamma^*$.

Now, we distinguish two cases. First, assume that there is a step $x = (q, \to, X, q') \in s^p$. Since σ is idempotent, so is s^p, and since $x'_{i+1} = (q'_i, \to, X'_{i+1}, q) \in s^p$ we deduce that $q' = q$. Therefore, the unique run of $\widetilde{\mathcal{A}}$ on $\widetilde{\vdash w} = \widetilde{\vdash u_1}^p \widetilde{u_2}^p \widetilde{u_3}^p \cdots$ follows the steps $x'_1 x'_2 \cdots x'_i x'_{i+1} x x x \cdots$. Hence, the set of states visited infinitely often along this run is X and the run is accepting iff $X \in \mathcal{F}$ is a Muller set. Therefore, if $X \notin \mathcal{F}$ we have $FG^\omega \cap \mathrm{dom}(\mathcal{A}) = \emptyset$ and we set $C_{FG^\omega} = \bot$. Now, if $X \in \mathcal{F}$ we have $FG^\omega \subseteq \mathrm{dom}(\mathcal{A})$ and we set $C_{FG^\omega} = C_2 \odot \big((FG \,?\, \varepsilon : \bot) \boxdot C^p_G(x)^\omega\big)$. We have $\mathrm{dom}(C_{FG^\omega}) = FG^\omega$ and for $w = u_1u_2u_3 \ldots \in FG^\omega$ with $u_1 \in F$ and $u_n \in G$ for all $n > 1$, we have $[\![C_{FG^\omega}]\!](w) = [\![C_1]\!](u_1u_2)[\![C^p_G(x)]\!](u_3)[\![C^p_G(x)]\!](u_4) \cdots$. By (J_2), we know that for all $n \geq 3$, $[\![C^p_G(x)]\!](u_n)$ is the output produced by $\widetilde{\mathcal{A}}$ when running step $x = (q, \to, X, q)$ on $\widetilde{u_n}^p$. We deduce that $[\![C_{FG^\omega}]\!](w) = [\![\widetilde{\mathcal{A}}]\!](\widetilde{\vdash w}) = [\![\mathcal{A}]\!](w)$ as desired.

The second case is when the unique step $x_1 = (q, \curvearrowright, X_1, q_1)$ in s^p which starts from the left in state q exits on the left. Since s^p is idempotent and $x'_{i+1} = (q'_i, \to, X'_{i+1}, q) \in s^p$, by definition of the product $s^p \cdot s^p$, there is a unique sequence of steps $x_2 = (q_1, \subsetarrow, X_2, q_2)$, $x_3 = (q_2, \curvearrowright, X_3, q_3)$, …, $x_j = (q_{j-1}, \subsetarrow, X_j, q_j)$, $x_{j+1} = (q_j, \to, X_{j+1}, q)$ in s^p with $j \geq 2$. Therefore, for all $w = u_1u_2u_3 \ldots \in FG^\omega$ with $u_1 \in F$ and $u_n \in G$ for all $n > 1$, the unique run of $\widetilde{\mathcal{A}}$ on $\widetilde{\vdash w} = \widetilde{\vdash u_1}^p \widetilde{u_2}^p \widetilde{u_3}^p \cdots$ follows the steps $x'_1 x'_2 \cdots x'_i x'_{i+1} (x_1 x_2 \cdots x_j x_{j+1})^\omega$. Hence, the set of states visited infinitely often along this run is $X = X_1 \cup X_2 \cup \cdots \cup X_{j+1}$. We deduce that the run is accepting iff $X \in \mathcal{F}$. Therefore, if $X \notin \mathcal{F}$ we have $FG^\omega \cap \mathrm{dom}(\mathcal{A}) = \emptyset$ and we set $C_{FG^\omega} = \bot$. Now, if $X \in \mathcal{F}$ we have $FG^\omega \subseteq \mathrm{dom}(\mathcal{A})$ and we set

$$C_3 = \big((G \,?\, \varepsilon : \bot) \boxdot C^p_G(x_1)\big) \odot \big(C^p_G(x_2) \boxdot C^p_G(x_3)\big) \odot \cdots \odot \big(C^p_G(x_j) \boxdot C^p_G(x_{j+1})\big),$$

$$C_{FG^\omega} = C_2 \odot \big((F \,?\, \varepsilon : \bot) \boxdot [G, C_3]^{2\omega}\big).$$

We have $\mathrm{dom}(C_{FG^\omega}) = FG^\omega$ and for all $w = u_1u_2u_3 \ldots \in FG^\omega$ with $u_1 \in F$ and $u_n \in G$ for all $n > 1$, we have

$$[\![C_{FG^\omega}]\!](w) = [\![C_1]\!](u_1u_2)[\![C_3]\!](u_2u_3)[\![C_3]\!](u_3u_4) \cdots.$$

Using (J_2), we can check that this is the output produced by $\widetilde{\mathcal{A}}$ when running on $\widetilde{\vdash w}$. Therefore, $[\![C_{FG^\omega}]\!](w) = [\![\widetilde{\mathcal{A}}]\!](\widetilde{\vdash w}) = [\![\mathcal{A}]\!](w)$. □

We prove that ω-2DMT$_{\mathsf{la}}$ are no more expressive than ω-RTEs.

Proof of Theorem 3.4 (2). We use the notations of the previous sections, in particular for the ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}$, the BDBA $\mathcal{B}$. We apply Theorem 8.1 to the canonical morphism Tr from Σ^* to the transition monoid TrM of $(\mathcal{A}, \mathcal{B})$. We obtain an *unambiguous* rational expression $G = \bigcup_{k=1}^m F_k \cdot G_k^\omega$ over Σ such that $\mathcal{L}(G) = \Sigma^\omega$ and for all $1 \leq k \leq m$ the expressions F_k and G_k are ε-free Tr-good rational expressions and σ_{G_k} is an idempotent, where $\mathrm{Tr}(G_k) = \{\sigma_{G_k}\}$. For each $1 \leq k \leq m$, let $C_k = C_{F_k G_k^\omega}$ be the ω-RTE given by Lemma 8.4. We define the final ω-RTE as

$$C = F_1 G_1^\omega \,?\, C_1 : (F_2 G_2^\omega \,?\, C_2 : \cdots (F_{m-1} G_{m-1}^\omega \,?\, C_{m-1} : C_m)).$$

From Lemma 8.4, we can easily check that $\mathrm{dom}(C) = \mathrm{dom}(\mathcal{A})$ and $[\![C]\!](w) = [\![\mathcal{A}]\!](w)$ for all $w \in \mathrm{dom}(C)$. □

9 Conclusion

The main contribution of the paper is to give a characterisation of regular string transductions using some combinators, giving rise to regular transducer expressions (RTE). Our proof uniformly works well for finite and infinite string transformations. RTE are a succint specification mechanism for regular transformations just like regular expressions are for regular languages. It is worthwhile to consider extensions of our technique to regular tree transformations, or in other settings where more involved primitives such as sorting or counting are needed. For readability and convenience we decided to keep some redundancy in the current set of combinators for RTEs, see e.g., Remark 3.2. Finding a minimal set of combinators achieving expressive completeness, as well as computing complexity measures for the conversion between RTEs and two-way transducers are open.

References

[1] Rajeev Alur and Pavol Cerný. 2010. Expressiveness of streaming string transducers. In *FSTTCS 2010*. 1–12.

[2] Rajeev Alur and Loris D'Antoni. 2017. Streaming Tree Transducers. *J. ACM* 64, 5 (2017), 31:1–31:55.

[3] Rajeev Alur, Loris D'Antoni, and Mukund Raghothaman. 2015. DReX: A Declarative Language for Efficiently Evaluating Regular String Transformations. In *POPL 2015*. 125–137.

[4] Rajeev Alur, Emmanuel Filiot, and Ashutosh Trivedi. 2012. Regular Transformations of Infinite Strings. In *LICS 2012*. 65–74.

[5] Rajeev Alur, Adam Freilich, and Mukund Raghothaman. 2014. Regular combinators for string transformations. In *CSL-LICS 2014*. 9:1–9:10.

[6] Mikołaj Bojańczyk, Laure Daviaud, Bruno Guillon, and Vincent Penelle. 2017. Which Classes of Origin Graphs Are Generated by Transducers. In *ICALP 2017*. 114:1–114:13.

[7] Olivier Carton. 2010. Right-Sequential Functions on Infinite Words. In *CSR 2010*. 96–106.

[8] Olivier Carton and Max Michel. 2003. Unambiguous Büchi automata. *Theoret. Comp. Sci.* 297, 1-3 (Mar 2003), 37–81.

[9] Thomas Colcombet. 2010. Factorization forests for infinite words and applications to countable scattered linear orderings. *TCS* 411, 4-5 (Jan 2010), 751–764.

[10] B. Courcelle. 1997. The Expression of Graph Properties and Graph Transformations in Monadic Second-order Logic. In *Handbook of Graph Grammars and Computing by Graph Transformation*. World Scientific Publishing, 313–400.

[11] Vrunda Dave, Paul Gastin, and Krishna Shankara Narayanan. 2018. Regular Transducer Expressions for Regular Transformations. *CoRR* abs/1802.02094 (2018). http://arxiv.org/abs/1802.02094

[12] Vrunda Dave, Shankara Narayanan Krishna, and Ashutosh Trivedi. 2016. FO-Definable Transformations of Infinite Strings. In *FSTTCS 2016*. 12:1–12:14.

[13] Manfred Droste, Werner Kuich, and Heiko Vogler. 2009. *Handbook of Weighted Automata*. Springer Publishing Company.

[14] Joost Engelfriet and Hendrik Jan Hoogeboom. 2001. MSO definable string transductions and two-way finite-state transducers. *ACM Transactions on Computational Logic* 2, 2 (Apr 2001), 216–254.

[15] Emmanuel Filiot, Shankara Narayanan Krishna, and Ashutosh Trivedi. 2014. First-order Definable String Transformations. In *FSTTCS 2014*. 147–159.

[16] Paul Gastin and Shankara Narayanan Krishna. 2018. Unambiguous forest factorization. (2018). Unpublished.

[17] M.P. Schützenberger. 1961. A remark on finite transducers. *Information and Control* 4, 2-3 (Sep 1961), 185–196.

[18] Imre Simon. 1990. Factorization forests of finite height. *Theoretical Computer Science* 72, 1 (Apr 1990), 65–94.

[19] Thomas Wilke. 2017. Backward Deterministic Büchi Automata on Infinite Words. In *FSTTCS 2017*. 6:6–6:9.