Regular Transducer Expressions for Regular Transformations

Vrunda Dave, Paul Gastin, Shankara Narayanan

Regular Transducer Expressions for Regular Transformations[1]

Vrunda Dave[a], Paul Gastin[b], Shankara Narayanan Krishna[a]

[a]*Dept of CSE, IIT Bombay, India*
vrunda,krishnas@cse.iitb.ac.in
[b]*LSV, ENS Paris-Saclay & CNRS, Université Paris-Saclay, France*
paul.gastin@ens-paris-saclay.fr

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 two-way 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.

The construction of an RTE associated with a deterministic two-way transducer is guided by a regular expression which is "good" wrt. its transition monoid. "Good" expressions are unambiguous, ensuring the functionality of the output computed. Moreover, in "good" expressions, iterations (Kleene-plus or omega) are restricted to subexpressions corresponding to *idempotent* elements of the transition monoid. "Good" expressions can be obtained with an unambiguous version of Imre Simon's famous forest factorization theorem.

To handle infinite words, we introduce the notion of transition monoids for deterministic two-way Muller transducers with look-ahead, where the look-ahead is captured by some backward deterministic Büchi automaton.

This paper is an extended version of [15] presented at LICS'18.

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 regular expressions; 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

[1]This work has been partially supported by IRL RELAX

fundamental theory to regular transformations [19], 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 [20], [17]. Other generalizations such as [2], extend this theory to trees. More recently, this equivalence between SSTs and logical transductions is also shown to hold good even when one works with the origin semantics [8].

Moving on, a natural problem 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 [18] 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. Recently, [5] proposed a set of combinators, analogous to the operators used in forming regular expressions, to form *combinator expressions* and proved their equivalence with SSTs.

Our Contributions. We generalize the result of [5] from finite to infinite words, and we propose a completely different proof technique based on transition monoids and on Simon's forest factorization theorem.

Over finite words, we work with two-way deterministic transducers (denoted 2DFT) while over infinite words, the model considered is deterministic two-way transducers with regular look-ahead, equipped with the Muller acceptance condition. Figure 1 gives an ω-$\mathsf{2DMT_{la}}$ (la stands for look-ahead and M in the $\mathsf{2DMT}$ 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 expressions* (RTE) characterizing two-way transducer ($\mathrm{2DFT}/\omega$-$\mathsf{2DMT_{la}}$).

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

The constant function $d \in \Gamma^*$ maps all strings in Σ^∞ to some fixed finite output word d.

Given a string $w \in \Sigma^\infty$, the if-then-else combinator $K \,?\, f : g$ checks if w is in the regular language K or not, and produces $f(w)$ if $w \in K$ and $g(w)$ otherwise.

The Hadamard product $f \odot g$ when applied to w produces $f(w) \cdot g(w)$, provided $f(w)$ is finite, otherwise it is undefined.

The unambiguous Cauchy product $f \boxdot g$ when applied on $w \in \Sigma^\infty$ produces $f(u)\cdot g(v)$ if $w = u\cdot v$ is an unambiguous decomposition of w with $u \in \mathsf{dom}(f)\cap\Sigma^*$ and $v \in \mathsf{dom}(g)$.

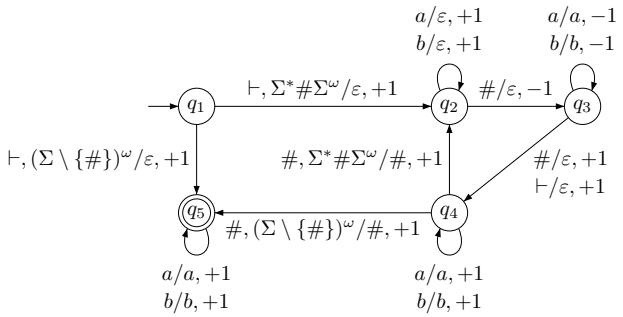


Figure 1: An ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}'$ with $[\![\mathcal{A}']\!](u_1\#u_2\#\ldots\#u_n\#v) = u_1^R u_1\#u_2^R u_2\#\ldots\#u_n^R u_n\#v$ where $u_1, \ldots, u_n \in (a+b)^*$, $v \in (a+b)^\omega$ and u^R denotes the reverse of u. The Muller acceptance set is $\{\{q_5\}\}$. The look-ahead expressions $\Sigma^*\#\Sigma^\omega$ and $(\Sigma\setminus\{\#\})^\omega$ are used to check if there is a $\#$ in the remaining suffix of the input word.

The unambiguous Kleene-plus $f^{\boxplus}$ 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 $u_i \in \mathsf{dom}(f)$.

The unambiguous 2-chained Kleene-plus $[K, f]^{2\boxplus}$ when applied to a string $w \in \Sigma^*$ 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 $f \overleftarrow{\odot} g$, $f^{\overleftarrow{\boxplus}}$ and $[K, f]^{\overleftarrow{2\boxplus}}$, which parse the input from left to right as before, but produce the output from right to left. For instance, with the notation above, $f^{\overleftarrow{\boxplus}}(w)$ produces $f(u_n)\cdots f(u_1)$.

The unambiguous ω-iteration produces $f^\omega(w) = f(u_1)f(u_2)\cdots$ if $w \in \Sigma^\omega$ can be unambiguously decomposed as $w = u_1u_2\cdots$ with each $u_i \in \mathsf{dom}(f) \cap \Sigma^*$.

Finally, the unambiguous two-chained ω-iteration produces $[K, f]^{2\omega}(w) = f(u_1u_2)f(u_2u_3)\cdots$ 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.

Example 1. *Consider the RTE* $C = {C_4}^{\boxplus} \odot C_2^\omega$ *with*

$$C_1 = a\,?\,a : (b\,?\,b : (\#\,?\,\# : \bot))$$
$$C_2 = a\,?\,a : (b\,?\,b : \bot)$$
$$C_3 = a\,?\,a : (b\,?\,b : (\#\,?\,\varepsilon : \bot))$$
$$C_4 = ((a+b)^*\#)\,?\,({C_3}^{\overleftarrow{\boxplus}} \odot {C_1}^{\boxplus}) : \bot$$

Then $\mathsf{dom}(C_1) = \mathsf{dom}(C_3) = (a+b+\#)$ *and* $\mathsf{dom}(C_2) = (a+b)$. *Next, we see that* $\mathsf{dom}(C_4) = (a+b)^*\#$ *and, for* $u \in (a+b)^*$, $[\![C_4]\!](u\#) = u^R u\#$ *where* u^R *denotes the reverse of* u. *This gives* $\mathsf{dom}(C) = ((a+b)^*\#)^+(a+b)^\omega$ *with* $[\![C]\!](u_1\#u_2\#\cdots u_n\#v) = u_1^R u_1\#u_2^R u_2\#\cdots\#u_n^R u_n\#v$ *when* $u_i \in (a+b)^*$ *and* $v \in (a+b)^\omega$. *The RTE* $C' = (a+b)^\omega\,?\,C_2^\omega : C$ *corresponds to the* ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}'$ *in Figure 1; that is,* $[\![C']\!] = [\![\mathcal{A}']\!]$.

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

Theorem 2.

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

(2) Given a 2DFT (resp. an ω-2DMT$_{\mathsf{la}}$) we can effectively construct an equivalent RTE (resp. ω-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$_{\mathsf{la}}$. We handle the look-ahead with a backward deterministic Büchi automaton (BDBA), also called *complete unambiguous* or *strongly unambiguous* Büchi automata [10, 24]. The translation of $\mathcal{A}$ to an RTE is crucially guided by a "good" (ω-)regular expression induced by the transition monoid of $\mathcal{A}$. The good (ω-)regular expression facilitates a uniform treatment of finite and infinite words. As a remark, it is not a priori clear how the result of [5] extends to infinite words using the techniques therein.

A regular expression F over alphabet Σ is *good* wrt. a morphism φ from Σ^* to a monoid $(S, ., 1_S)$ if (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. Note that (ii) implies that for each subexpression E^+ of F, s_E is an idempotent. These *good* expressions are obtained thanks to an unambiguous version [21] of the celebrated forest factorization theorem due to Imre Simon [23]. Good rational expressions might be useful in settings beyond two-way transducers.

See [16, Appendix A.2] for a practical example using transducers.

Related Work. We briefly discuss two recent papers which are closely related to this paper. As mentioned above, we generalized the result of [5] from finite to infinite words. Actually, [5] works with copyless cost register automata (CCRA) over finite words. CCRA 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 correspond to CCRAs where the output monoid is the free monoid $(\Gamma^*, \cdot, \varepsilon)$ 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.

The proof in [5] is rather simple in the case of commutative output monoids, and quite non-trivial in the other case. The output generated in a CCRA is stored in registers, and it is important to keep track of the flow of the content between registers on each input word. To this end, [5] uses *shapes*, which are bi-partite graphs over the set of registers. An edge from register X to register Y in a shape implies that register X flows into register Y after reading the input word. The expression representing $[\![A]\!]$ for a CCRA A is obtained by "summarizing" sets of paths having some fixed shape S, and then combining the summaries appropriately: this includes concatenation of shapes, as well as iteration. While concatenation of shapes is easy, the iteration of shapes is handled via a "normalization" which ensures that the iterated shapes are idempotent.

Very recently, [6] proposed an alternative proof for the result of [5] over finite words. The proof of [6] has some similarities with the one we proposed in our extended abstract which appeared in [15]. Instead of using the transition monoid of a two-way automaton which fully describes how a word w acts on states (starting on the left/right of w in state p, the run exists on the left/right of w in state q), they define a flow automaton based on Shepherdson construction [22].

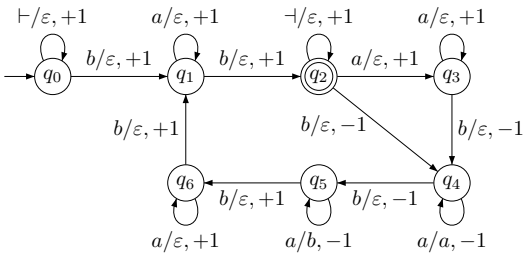


Figure 2: A 2DFT $\mathcal{A}$ with $[\![\mathcal{A}]\!](ba^{m_1}ba^{m_2}b\dots a^{m_k}b) = a^{m_2}b^{m_1}a^{m_3}b^{m_2}\dots a^{m_k}b^{m_{k-1}}$.

Then, they use the state elimination technique of Brzozowski and McCluskey to obtain flows labelled with function expressions. Their technique for handling concatenation is similar to ours. The main difference is in the way loops are handled. We use the unambiguous version of Simon's theorem so that Kleene-plus only occurs on idempotents, whereas [6] defines *simple* loops for which they give a direct translation, and then shows how to reduce arbitrary loops to simple ones.

2. Finite Words

We start with the definition of two-way automata and transducers for the case of finite words.

2.1. Two-way automata and transducers

Let Σ be a finite input alphabet and let $\vdash, \dashv$ be two special symbols not in Σ. We assume that every input string $w \in \Sigma^*$ is presented as $\vdash w \dashv$, where $\vdash, \dashv$ serve as left and right delimiters that appear nowhere else in w. We write $\Sigma_{\vdash\dashv} = \Sigma \cup \{\vdash, \dashv\}$. A two-way automaton $\mathcal{A} = (Q, \Sigma, \delta, I, F)$ has a finite set of states Q, subsets $I, F \subseteq Q$ of initial and final states and a transition relation $\delta \subseteq Q \times \Sigma_{\vdash\dashv} \times Q \times \{-1, 1\}$. The -1 represents the reading head moving to the left, while a 1 represents the reading head moving to the right. The reading head cannot move left when it is on $\vdash$. See Figure 2 for an example.

A configuration of $\mathcal{A}$ is represented by $w_1 q w_2$ where $q \in Q$ and $w_1 w_2 \in \vdash\Sigma^*\dashv$. If $w_2 = \varepsilon$ the computation has come to an end. Otherwise, the reading head of $\mathcal{A}$ is scanning the first symbol of $w_2 \neq \varepsilon$ in state q. If $w_2 = aw_2'$ and if $(q, a, q', -1) \in \delta$ (hence $a \neq \vdash$), then there is a transition from the configuration $w_1' bqaw_2'$ to $w_1' q' baw_2'$. Likewise, if $(q, a, q', 1) \in \delta$, we obtain a transition from $w_1 qaw_2'$ to $w_1 aq' w_2'$. A run of $\mathcal{A}$ is a sequence of transitions; it is accepting if it starts in a configuration $p\vdash w\dashv$ with $p \in I$ and ends in a configuration $\vdash w \dashv q$ with $q \in F$. The language $\mathcal{L}(\mathcal{A})$ or domain $\mathsf{dom}(\mathcal{A})$ of $\mathcal{A}$ is the set of all words $w \in \Sigma^*$ which have an accepting run in $\mathcal{A}$.

To extend the definition of a two-way automaton $\mathcal{A}$ into a two-way transducer, $(Q, \Sigma, \delta, I, F)$ is extended to $(Q, \Sigma, \Gamma, \delta, I, F)$ by adding a finite output alphabet Γ and the definition of the transition relation as a *finite* subset $\delta \subseteq Q \times \Sigma_{\vdash\dashv} \times Q \times \Gamma^* \times \{-1, 1\}$. The output produced on each transition is appended to the right of the output produced so far. $\mathcal{A}$ defines a relation $[\![\mathcal{A}]\!] = \{(u, w) \mid u \in \mathcal{L}(\mathcal{A})$ and w is the output produced on an accepting run of $u\}$.

The transducer $\mathcal{A}$ is said to be functional if for each input $u \in \mathsf{dom}(\mathcal{A})$, at most one output w can be produced. In this case, for each $u \in \mathsf{dom}(\mathcal{A})$, there is exactly one $w \in \Gamma^*$ such that $(u, w) \in [\![\mathcal{A}]\!]$. We also denote this by $[\![\mathcal{A}]\!](u) = w$. We consider a special symbol $\perp \notin \Gamma$ that will stand for *undefined*. We let $[\![\mathcal{A}]\!](u) = \perp$ when $u \notin \mathsf{dom}(\mathcal{A})$. Thus, the semantics of a functional transducer $\mathcal{A}$ is a map $[\![\mathcal{A}]\!] \colon \Sigma^* \to \mathbb{D} = \Gamma^* \cup \{\perp\}$ such that $u \in \mathsf{dom}(\mathcal{A})$ iff $[\![\mathcal{A}]\!](u) \neq \perp$.

We use non-deterministic unambiguous two-way transducers (2NUFT) in some proofs. A two-way transducer is unambiguous if each string $u \in \Sigma^*$ has at most one accepting run. Clearly, 2NUFTs are functional. A deterministic two-way transducer (2DFT) is one having a single initial state and where, from each state, on each symbol $a \in \Sigma_{\vdash\dashv}$, at most one transition is enabled. In that case, the transition relation is a partial function $\delta \colon Q \times \Sigma_{\vdash\dashv} \to Q \times \Gamma^* \times \{-1, 1\}$. 2DFTs are by definition unambiguous. It is known [11] that 2DFTs are equivalent to 2NUFTs.

A 1DFT (1NUFT) represents a deterministic (non-deterministic unambiguous) transducer where the reading head only moves to the right.

Example 3. *Figure 2 shows a two-way transducer $\mathcal{A}$ with $\mathsf{dom}(\mathcal{A}) = (ba^*)^+b$, $[\![\mathcal{A}]\!](ba^{m_1}b) = \varepsilon$ and $[\![\mathcal{A}]\!](ba^{m_1}ba^{m_2}b \cdots a^{m_k}b) = a^{m_2}b^{m_1}a^{m_3}\ b^{m_2} \cdots a^{m_k}b^{m_{k-1}}$ for $k \geq 2$ and $m_i \in \mathbb{N}$ for $1 \leq i \leq k$.*

2.2. Regular Transducer Expressions

Let Σ and Γ be finite input and output alphabets. Recall that $\perp \notin \Gamma$ is a special symbol that stands for *undefined*. We define the output monoid as $\mathbb{D} = \Gamma^* \cup \{\perp\}$ with the usual concatenation on words, $\perp$ acting as a zero: $d \cdot \perp = \perp \cdot d = \perp$ for all $d \in \mathbb{D}$. The unit is the empty word $\mathbf{1}_{\mathbb{D}} = \varepsilon$.

We define *Regular Transducer Expressions* (RTE) from Σ^* to $\mathbb{D}$ using some basic combinators. The syntax of RTE is defined with the following grammar:

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

where $d \in \mathbb{D}$ ranges over output values, and $K \subseteq \Sigma^*$ ranges over regular languages of *finite words*. The semantics of an RTE C is a function $[\![C]\!] \colon \Sigma^* \to \mathbb{D}$ defined inductively following the syntax of the expression, starting from constant functions. Since $\perp$ stands for *undefined*, we define the *domain* of a function $f \colon \Sigma^* \to \mathbb{D}$ by $\mathsf{dom}(f) = f^{-1}(\mathbb{D} \setminus \{\perp\}) = \Sigma^* \setminus f^{-1}(\perp)$.

Constants. For $d \in \mathbb{D}$, we let $[\![d]\!]$ be the constant map defined by $[\![d]\!](w) = d$ for all $w \in \Sigma^*$.

We have $\mathsf{dom}([\![d]\!]) = \Sigma^*$ if $d \neq \perp$ and $\mathsf{dom}([\![\perp]\!]) = \emptyset$.

Each regular combinator defined above allows to combine functions from Σ^* to $\mathbb{D}$. For functions $f, g \colon \Sigma^* \to \mathbb{D}$, $w \in \Sigma^*$ and a regular language $K \subseteq \Sigma^*$, we define the following combinators.

If then else. $(K \,?\, f : g)(w)$ is defined as $f(w)$ for $w \in K$, and $g(w)$ for $w \notin K$.

We have $\mathsf{dom}(K \,?\, f : g) = (\mathsf{dom}(f) \cap K) \cup (\mathsf{dom}(g) \setminus K)$.

Hadamard product. $(f \odot g)(w) = f(w) \cdot g(w)$ (recall that $(\mathbb{D}, \cdot, \mathbf{1}_{\mathbb{D}})$ is a monoid).

We have $\mathsf{dom}(f \odot g) = \mathsf{dom}(f) \cap \mathsf{dom}(g)$.

Unambiguous Cauchy product and its reverse. If w admits a unique factorization $w = u \cdot v$ with $u \in \mathsf{dom}(f)$ and $v \in \mathsf{dom}(g)$ then we set $(f \odot g)(w) = f(u) \cdot g(v)$ and $(f \overleftarrow{\odot} g)(w) = g(v) \cdot f(u)$. Otherwise, we set $(f \odot g)(w) = \bot = (f \overleftarrow{\odot} g)(w)$.

We have $\mathsf{dom}(f \odot g) = \mathsf{dom}(f \overleftarrow{\odot} g) \subseteq \mathsf{dom}(f) \cdot \mathsf{dom}(g)$ and the inclusion is strict if the concatenation of $\mathsf{dom}(f)$ and $\mathsf{dom}(g)$ is ambiguous.

Unambiguous Kleene-plus and its reverse. If w admits a unique factorization $w = u_1 \cdot u_2 \cdots u_n$ with $n \geq 1$ and $u_i \in \mathsf{dom}(f)$ for all $1 \leq i \leq n$ then we set $f^{\boxplus}(w) = f(u_1) \cdot f(u_2) \cdots f(u_n)$ and $f^{\overleftarrow{\boxplus}}(w) = f(u_n) \cdots f(u_2) \cdot f(u_1)$. Otherwise, we set $f^{\boxplus}(w) = \bot = f^{\overleftarrow{\boxplus}}(w)$.

We have $\mathsf{dom}(f^{\boxplus}) = \mathsf{dom}(f^{\overleftarrow{\boxplus}}) \subseteq \mathsf{dom}(f)^+$ and the inclusion is strict if the Kleene iteration $\mathsf{dom}(f)^+$ of $\mathsf{dom}(f)$ is ambiguous. Notice that $\mathsf{dom}(f^{\boxplus}) = \emptyset$ when $\varepsilon \in \mathsf{dom}(f)$.

Unambiguous 2-chained Kleene-plus and its reverse. If w admits a unique factorization $w = u_1 \cdot u_2 \cdots u_n$ with $n \geq 1$ and $u_i \in K$ for all $1 \leq i \leq n$ then we set $[K, f]^{2\boxplus}(w) = f(u_1u_2) \cdot f(u_2u_3) \cdots f(u_{n-1}u_n)$ and $[K, f]^{\overleftarrow{2\boxplus}}(w) = f(u_{n-1}u_n) \cdots f(u_2u_3) \cdot f(u_1u_2)$ (if $n = 1$, the empty product gives the unit of $\mathbb{D}$: $[K, f]^{2\boxplus}(w) = \mathbf{1}_{\mathbb{D}} = [K, f]^{\overleftarrow{2\boxplus}}(w)$). Otherwise, we set $[K, f]^{2\boxplus}(w) = \bot = [K, f]^{\overleftarrow{2\boxplus}}(w)$.

Again, we have $\mathsf{dom}([K, f]^{2\boxplus}) = \mathsf{dom}([K, f]^{\overleftarrow{2\boxplus}}) \subseteq K^+$ and the inclusion is strict if the Kleene iteration K^+ of K is ambiguous. Notice that, even if $w \in K^+$ admits a unique factorization $w = u_1 \cdot u_2 \cdots u_n$ with $u_i \in K$ for all $1 \leq i \leq n$, w is not necessarily in the domain of $[K, f]^{2\boxplus}$ or $[K, f]^{\overleftarrow{2\boxplus}}$. For w to be in this domain, it is further required that $u_1u_2, u_2u_3, \ldots, u_{n-1}u_n \in \mathsf{dom}(f)$. Notice that we have $\mathsf{dom}([K, f]^{2\boxplus}) = \mathsf{dom}([K, f]^{\overleftarrow{2\boxplus}}) = K^+$ when K^+ is unambiguous and $K^2 \subseteq \mathsf{dom}(f)$.

Lemma 4. *The domain of an RTE C is a regular language $\mathsf{dom}(C) \subseteq \Sigma^*$.*

Example 5. *Consider the RTEs*

$$C_1 = ([(a+b)^+\#] \,?\, \varepsilon : \bot) \boxdot ((a+b)^+ \,?\, \mathsf{copy} : \bot)$$
$$C_2 = \#$$
$$C_3 = ((a+b)^+ \,?\, \mathsf{copy} : \bot) \boxdot ([\#(a+b)^+] \,?\, \varepsilon : \bot)$$

where $\mathsf{copy} = (a \,?\, a : (b \,?\, b : \bot))^{\boxplus}$.
Then, $\mathsf{dom}(\llbracket C_2 \rrbracket) = \Sigma^*$, $\mathsf{dom}(\llbracket \mathsf{copy} \rrbracket) = (a+b)^+$ *and* $\mathsf{dom}(\llbracket C_1 \rrbracket) = \mathsf{dom}(\llbracket C_3 \rrbracket) = (a+b)^+\#(a+b)^+$. *Moreover,* $\llbracket C_1 \odot C_2 \odot C_3 \rrbracket(u\#v) = v\#u$ *for all* $u, v \in (a+b)^+$.

Example 6. *Consider the RTEs*

$$C_a = (b \,?\, \varepsilon : \bot) \boxdot (a \,?\, a : \bot)^{\boxplus}$$
$$C_b = (b \,?\, \varepsilon : \bot) \boxdot (a \,?\, b : \bot)^{\boxplus}$$

We have $\mathsf{dom}(\llbracket C_a \rrbracket) = ba^+ = \mathsf{dom}(\llbracket C_b \rrbracket)$ *and* $\llbracket C_a \rrbracket(ba^n) = a^n$ *and* $\llbracket C_b \rrbracket(ba^n) = b^n$.
We deduce that $\mathsf{dom}(\llbracket C_b \overleftarrow{\boxdot} C_a \rrbracket) = ba^+ba^+$ *and* $\llbracket C_b \overleftarrow{\boxdot} C_a \rrbracket(ba^nba^m) = a^mb^n$.

Consider the expression

$$C = [ba^+, C_b \overleftarrow{\boxdot} C_a]^{2\boxplus} \boxdot (b \,?\, \varepsilon : \bot)\,.$$

Then, $\mathsf{dom}([\![C]\!]) = (ba^+)^+b$*, and* $[\![C]\!](ba^mb) = \varepsilon$ *and for* $k \geq 2$ *we have* $[\![C]\!](ba^{m_1}ba^{m_2}b \cdots a^{m_k}b) = a^{m_2}b^{m_1}a^{m_3}b^{m_2} \cdots a^{m_k}\ b^{m_{k-1}}$*.*

Theorem 7. *2DFTs and RTEs define the same class of functions. More precisely,*

1. *given an RTE* C*, we can construct a 2DFT* $\mathcal{A}$ *such that* $[\![\mathcal{A}]\!] = [\![C]\!]$*,*
2. *given a 2DFT* $\mathcal{A}$*, we can construct an 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 2.6 after some preliminaries in Section 2.5 on transition monoids for 2DFTs and the unambiguous forest factorization theorem.

Remark 8. *Notice that the reverse Cauchy product is redundant, it can be expressed with the Hadamard product and the Cauchy product:*

$$f \overleftarrow{\boxdot} g = ((\mathsf{dom}(f) \,?\, \varepsilon : \bot) \boxdot g) \odot (f \boxdot (\mathsf{dom}(g) \,?\, \varepsilon : \bot))\,.$$

The unambiguous Kleene-plus is also redundant, it can be expressed with the unambiguous 2-chained Kleene-plus:

$$f^{\boxplus} = [\mathsf{dom}(f), f \boxdot (\mathsf{dom}(f) \,?\, \varepsilon : \bot)]^{2\boxplus} \odot ((\mathsf{dom}(f)^* \,?\, \varepsilon : \bot) \boxdot f)\,.$$

Remark 9. *We can extend the 2-chained Kleene-plus to* k*-chained Kleene-plus for any* $k \geq 3$*. It is defined as follows: If* w *admits a unique factorization* $w = u_1u_2 \ldots u_n$*, with* $n \geq 1$ *and* $u_i \in K$ *for all* $1 \leq i \leq n$*, then* $[K, f]^{k\boxplus}(w) = f(u_1u_2 \ldots u_k)f(u_2u_3 \ldots u_{k+1}) \ldots f(u_{n-k+1}u_{n-k+2} \ldots u_n)$*. Otherwise, we set* $[K, f]^{k\boxplus}(w) = \bot$*. Notice that if* $n < k$*, we have an empty product which gives the unit of* $\mathbb{D}$*:* $[K, f]^{k\boxplus}(w) = \mathbf{1}_{\mathbb{D}}$*. In [16], we have shown that adding the* k*-plus combinator (or its reverse) does not increase the expressive power of RTEs.*

Remark 10. *The combinator expressions proposed in [5] are equivalent to our RTEs on finite words (see below). Our terminology and notation are all inspired from weighted automata literature. We prefer to stick to these classical notions since they are well-established and we believe they are more natural for string to string transducers.*

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 syntax as $\mathsf{dom}(f) \,?\, f : g$. The* split-sum *combinator $f \oplus g$ of [5] is the classical Cauchy product $f \boxdot g$. The* iterated sum *Σf of [5] is the Kleene-plus $f^{\boxplus}$. The* left-split-sum *and* left-iterated sum *of [5] correspond to 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 the classical Hadamard product $f \odot g$. Finally, the* chained sum *$\Sigma(f, L)$ of [5] is our two-chained Kleene-plus $[L, f]^{2\boxplus}$.*

2.3. RTE to 2DFT

In this section, we prove Theorem 7(1), i.e., we show that given an RTE C, we can construct a 2DFT $\mathcal{A}$ such that $[\![\mathcal{A}]\!] = [\![C]\!]$. We do this by structural induction on RTEs, starting with constant functions, and then later showing that 2DFTs are closed under all the combinators used in RTEs.

Constant functions: We start with the constant function $d \in \mathbb{D}$ for which it is easy to construct a 2DFT $\mathcal{A}$ such that $[\![d]\!] = [\![\mathcal{A}]\!]$. For $d = \bot$, we take $\mathcal{A}$ such that $\mathsf{dom}(\mathcal{A}) = \emptyset$ (for instance we use a single state and an empty transition function). Assume now that $d \in \Gamma^*$. The 2DFT scans the word up to the right end marker, outputs d and stops. Formally, we let $\mathcal{A} = (\{q\}, \Sigma, \Gamma, \delta, q, \{q\})$ s.t. $\delta(q, a) = (q, \varepsilon, +1)$ for all $a \in \Sigma \cup \{\vdash\}$ and $\delta(q, \dashv) = (q, d, +1)$. Clearly, $[\![\mathcal{A}]\!](w) = d$ for all $w \in \Sigma^*$.

The inductive steps follow directly from:

Lemma 11. *Let $K \subseteq \Sigma^*$ be regular, and let f and g be RTEs with $[\![f]\!] = [\![M_f]\!]$ and $[\![g]\!] = [\![M_g]\!]$ for 2DFTs M_f and M_g respectively. Then, one can construct*

1. *a 2DFT $\mathcal{A}$ such that $[\![K \,?\, f : g]\!] = [\![\mathcal{A}]\!]$.*
2. *a 2DFT $\mathcal{A}$ such that $[\![\mathcal{A}]\!] = [\![f \odot g]\!]$.*
3. *2DFTs $\mathcal{A}$, $\mathcal{B}$ such that $[\![\mathcal{A}]\!] = [\![f \boxdot g]\!]$ and $[\![\mathcal{B}]\!] = [\![f \overleftarrow{\boxdot} g]\!]$.*
4. *2DFTs $\mathcal{A}$, $\mathcal{B}$ such that $[\![\mathcal{A}]\!] = [\![f^{\boxplus}]\!]$ and $[\![\mathcal{B}]\!] = [\![f^{\overleftarrow{\boxplus}}]\!]$.*
5. *2DFTs $\mathcal{A}$, $\mathcal{B}$ such that $[\![\mathcal{A}]\!] = [\![[K, f]^{2\boxplus}]\!]$ and $[\![\mathcal{B}]\!] = [\![[K, f]^{\overleftarrow{2\boxplus}}]\!]$.*

Proof. (1) **If then else.** Let $\mathcal{B}$ be a complete DFA that accepts the regular language K. The idea of the proof is to construct a 2DFT $\mathcal{A}$ which first runs $\mathcal{B}$ on the input w until the end marker $\dashv$ is reached in some state q of $\mathcal{B}$. Then, $w \in K$ iff $q \in F$ is some accepting state of $\mathcal{B}$. The automaton $\mathcal{A}$ moves left all the way to $\vdash$, and starts running either M_f or M_g depending on whether $q \in F$ or not. Since $\mathcal{B}$ is complete, it is clear that $\mathsf{dom}(\mathcal{A}) = \mathsf{dom}(K \,?\, f : g)$ and the output of $\mathcal{A}$ coincides with $[\![M_f]\!]$ iff the input is in K, and otherwise coincides with $[\![M_g]\!]$.

(2) **Hadamard product.** Given an input w, the constructed 2DFT $\mathcal{A}$ first runs M_f. Instead of executing a transition $p \xrightarrow{\dashv/\gamma,+1} q$ with q a final state of M_f, it executes $p \xrightarrow{\dashv/\gamma,-1} \mathsf{reset}$ where reset is a new state. While in the reset state, it moves all the way back to $\vdash$ and it starts running M_g by executing $\mathsf{reset} \xrightarrow{\vdash/\gamma',+1} q'$ if $\delta_g(q_0, \vdash) = (q', \gamma', +1)$ where δ_g is the transition function of M_g and q_0 is the initial state of M_g. The final states of $\mathcal{A}$ are those of M_g, and its initial state is the initial state of M_f. Clearly, $\mathsf{dom}(\mathcal{A}) = \mathsf{dom}(M_f) \cap \mathsf{dom}(M_g)$ and the output of $\mathcal{A}$ is the concatenation of the outputs of M_f and M_g.

(3) **Cauchy product.** The domain of a 2DFT is a regular language, accepted by the 2DFA obtained by ignoring the outputs. Since 2DFAs are effectively equivalent to (1)DFAs, we can construct from M_f and M_g two DFAs $\mathcal{C}_f = (Q_f, \Sigma, \delta_f, s_f, F_f)$ and $\mathcal{C}_g = (Q_g, \Sigma, \delta_g, s_g, F_g)$ such that $\mathcal{L}(\mathcal{C}_f) = \mathsf{dom}(f)$ and $\mathcal{L}(\mathcal{C}_g) = \mathsf{dom}(g)$.

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

- L_p is the set of words which admit a run in $\mathcal{C}_f$ from its initial state to the final state $p \in F_f$,
- $M_{p,q}$ is the set of words which admit a run in $\mathcal{C}_f$ from state p to some final state in F_f, and also admit a run in $\mathcal{C}_g$ from its initial state to state $q \in Q_g$,
- R_q is the set of words which admit a run in $\mathcal{C}_g$ from state q to some final state in F_g, and also admit a run in $\mathcal{C}_g$ from its initial state to some final state in F_g.

Therefore, we have $\mathsf{dom}(f \boxdot g) = \mathsf{dom}(f \overleftarrow{\boxdot} g) = (\mathsf{dom}(f) \cdot \mathsf{dom}(g)) \setminus K$ is a regular language and we construct a complete DFA $\mathcal{C} = (Q, \Sigma, \delta, q_0, F)$ which accepts this language.

1. From $\mathcal{C}_f, \mathcal{C}_g$ and $\mathcal{C}$ we construct a 1NUFT $\mathcal{D}$ such that $\mathsf{dom}(\mathcal{D}) = \mathsf{dom}(f \boxdot g)$ and on an input word $w = u \cdot v$ with $u \in \mathsf{dom}(f)$ and $v \in \mathsf{dom}(g)$ it produces the output $u\#v$ where $\# \notin \Sigma$ is a new symbol. On an input word $w \in \Sigma^*$, the transducer $\mathcal{D}$ runs a copy of $\mathcal{C}$. Simultaneously, $\mathcal{D}$ runs a copy of $\mathcal{C}_f$ on some prefix u of w, copying each input letter to the output. Whenever $\mathcal{C}_f$ is in a final state after reading u, the transducer $\mathcal{D}$ may non-deterministically decide to stop running $\mathcal{C}_f$, to output $\#$, and to start running C_g on the corresponding suffix v of w ($w = u \cdot v$) while copying again each input letter to the output. The transducer $\mathcal{D}$ accepts if $\mathcal{C}$ accepts w and $\mathcal{C}_g$ accepts v. Then, we have $u \in \mathcal{L}(\mathcal{C}_f) = \mathsf{dom}(f)$, $v \in \mathcal{L}(\mathcal{C}_g) = \mathsf{dom}(g)$ and $w = u \cdot v \in \mathcal{L}(\mathcal{C}) = \mathsf{dom}(f \boxdot g)$. The output produced by $\mathcal{D}$ is $u\#v$. The only non-deterministic choice in an accepting run of $\mathcal{D}$ is unambiguous since a word $w \in \mathcal{L}(\mathcal{C}) = \mathsf{dom}(f \boxdot g)$ has a unique factorization $w = u \cdot v$ with $u \in \mathsf{dom}(f)$ and $v \in \mathsf{dom}(g)$.

2. We construct a 2DFT $\mathcal{T}$ which takes as input words of the form $u\#v$ with $u, v \in \Sigma^*$, runs M_f on u and then M_g on v. To do so, u is traversed in either direction depending on M_f, and the symbol $\#$ is interpreted as the right end marker $\dashv$. We explain how $\mathcal{T}$ simulates a transition of M_f moving to the right of $\dashv$, producing some output γ and going to a state q. If q is not final, then $\mathcal{T}$ moves to the right of $\#$ and then all the way to the end and rejects. If q is final, then $\mathcal{T}$ stays on $\#$ (simulated by moving right and then back left), producing the output γ, but goes to the initial state of M_g instead. $\mathcal{T}$ then runs M_g on v, interpreting $\#$ as $\vdash$. When M_g moves to the right of $\dashv$, $\mathcal{T}$ does the same and accepts iff M_g accepts.

3. In a similar manner, we construct a 2DFT $\mathcal{T}'$ which takes as input strings of the form $u\#v$, first runs M_g on v and then runs M_f on u. Assume that M_g wants to move to the right of $\dashv$ going to state q. If q is not final then $\mathcal{T}'$ also moves to the right of $\dashv$ and rejects. Otherwise, $\mathcal{T}'$ traverses back to $\vdash$ and runs M_f on u. When M_f wants to move to the right of $\#$ going to some state q and producing γ, $\mathcal{T}'$ moves also to the right of $\#$ producing γ and then all the way right producing ε. After moving to the right of $\dashv$, it accepts if q is a final state of M_f and rejects otherwise.

We construct a 2NUFT $\mathcal{A}'$ as the composition of $\mathcal{D}$ and $\mathcal{T}$. The composition of a 1NUFT and a 2DFT is a 2NUFT [11], hence $\mathcal{A}'$ is a 2NUFT. Moreover,

$[\![\mathcal{A}']\!] = [\![f \boxdot g]\!]$. Using the equivalence of 2NUFT and 2DFT, we can convert $\mathcal{A}'$ into an equivalent 2DFT $\mathcal{A}$. In a similar way, to obtain $[\![f \overleftarrow{\boxdot} g]\!]$, the 2NUFT $\mathcal{B}'$ is obtained as a composition of $\mathcal{D}$ and $\mathcal{T}'$ and is then converted to an equivalent 2DFT $\mathcal{B}$.

(4) **Kleene-plus.** The proof is similar to case (3). First, we show that $\mathsf{dom}(f^{\boxplus})$ is regular. Notice that if $\varepsilon \in \mathsf{dom}(f)$ then $\mathsf{dom}(f^{\boxplus}) = \emptyset$, hence we assume below that $\varepsilon \notin \mathsf{dom}(f)$. As in case (3), the language K of words w having at least two factorizations $w = u_1v_1 = u_2v_2$ with $u_1, u_2 \in \mathsf{dom}(f)$, $v_1, v_2 \in \mathsf{dom}(f)^*$ and $u_1 \neq u_2$ is regular. Hence, $K' = \mathsf{dom}(f)^* \cdot K$ is regular and contains all words in $\mathsf{dom}(f)^+$ having several factorizations as products of words in $\mathsf{dom}(f)$. We deduce that $\mathsf{dom}(f^{\boxplus}) = \mathsf{dom}(f)^+ \setminus K'$ is regular and we can construct a complete DFA $\mathcal{C}$ recognizing this domain.

As in case (3), from $\mathcal{C}_f$ and $\mathcal{C}$, we construct a 1NUFT $\mathcal{D}$ which takes as input w and outputs $u_1\#u_2\#\cdots\#u_n$ iff there is an unambiguous decomposition of w as $u_1u_2\cdots u_n$, with each $u_i \in \mathsf{dom}(f)$. We then construct a 2DFT $\mathcal{T}$ that takes as input words of the form $u_1\#u_2\#\cdots\#u_n$ with each $u_i \in \Sigma^*$ and runs M_f on each u_i from left to right, i.e., starting with u_1 and ending with u_n. The transducer $\mathcal{T}$ interprets $\#$ as $\vdash$ (resp. $\dashv$) when it is reached from the right (resp. left). The simulation by $\mathcal{T}$ reading $\#$ of a transition of M_f moving to the right of $\dashv$ is as in case (3), except that $\mathcal{T}$ goes to the initial state of M_f.

The 2NUFT $\mathcal{A}'$ is then obtained as the composition of $\mathcal{D}$ with the 2DFT $\mathcal{T}$. Finally, a 2DFT $\mathcal{A}$ equivalent to the 2NUFT $\mathcal{A}'$ is constructed. Likewise, $\mathcal{B}$ is obtained using the composition of $\mathcal{D}$ with a 2DFT $\mathcal{T}'$ that runs M_f on each factor u_i from right to left.

(5) **2-chained Kleene-plus.** As in case (4), we construct the 1NUFT $\mathcal{D}$ which takes as input w and outputs $u_1\#u_2\#\cdots\#u_n$ iff there is an unambiguous decomposition of w as $u_1u_2\cdots u_n$, with each $u_i \in K$. We then construct a 2DFT $\mathcal{D}'$ that takes as input words of the form $u_1\#u_2\#\cdots\#u_n$ with each $u_i \in \Sigma^*$ and produces $u_1u_2\#u_2u_3\#\cdots\#u_{n-1}u_n$. The 2NUFT $\mathcal{A}'$ is then obtained as the composition of $\mathcal{D}'$ with the 2DFT $\mathcal{T}$ constructed for case (4). Finally, a 2DFT $\mathcal{A}$ equivalent to the 2NUFT $\mathcal{A}'$ is constructed. The output produced by $\mathcal{A}$ is thus $[\![M_f]\!](u_1u_2) \cdot [\![M_f]\!](u_2u_3) \cdots [\![M_f]\!](u_{n-1}u_n)$. We proceed similarly for $\mathcal{B}$. □

2.4. Unambiguous forest factorization

In Section 2.6, we prove that, given a 2DFT $\mathcal{A}$, we can obtain an RTE C such that $[\![\mathcal{A}]\!] = [\![C]\!]$. We use the fact that any $w \in \Sigma^*$ in the domain of $\mathcal{A}$ can be factorized unambiguously into a good rational expression. The unambiguous factorization of words in Σ^* guides the construction of the combinator expression for $[\![\mathcal{A}]\!](w)$ over Γ in an inductive way.

For rational expressions over Σ we will use the following syntax:

$$F ::= \emptyset \mid \varepsilon \mid a \mid F \cup F \mid F \cdot F \mid F^+$$

where $a \in \Sigma$. For reasons that will be clear below, we prefer to use the Kleene-plus instead of the Kleene-star, hence we also add ε explicitly in the syntax. An expression is said to be 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.

Notice that $\emptyset$ cannot be used in a good expression since it does not satisfy the second condition. Also, the second condition implies that for each subexpression E^+ of F we have $s_E \cdot s_E = s_E$ is an idempotent.

Theorem 12 (Unambiguous Forest Factorization [21])**.** *For each $s \in S$, there is an ε-free* good *rational expression F_s such that $\mathcal{L}(F_s) = \varphi^{-1}(s) \setminus \{\varepsilon\} \subseteq \Sigma^+$. Therefore, $G = \varepsilon \cup \bigcup_{s \in S} F_s$ is an unambiguous rational expression over Σ such that $\mathcal{L}(G) = \Sigma^*$.*

Theorem 12 can be seen as an unambiguous version of Imre Simon's forest factorization theorem [23]. Its proof, which can be found in [21], follows the same lines of the recent proofs of Simon's theorem, see e.g. [12, 13]. For the sake of completeness, we summarize the proof idea and contributions in [21] here. Given a semigroup morphism $\varphi\colon \Sigma^+ \to S$, [21] constructs a universal, unambiguous automaton $\mathcal{A}$, which is "good" wrt φ in the following sense: (1) $\mathcal{A}$ is unambiguous and accepts all words in $\Sigma^* \cup \Sigma^\omega$, (2) $\mathcal{A}$ has a unique initial state i which has no incoming transitions to it, as well as a unique final state f with no outgoing transitions from it, (3) the states of $\mathcal{A}$ are totally ordered as $Q \setminus \{i, f\} < f < i$, where Q is the set of states of $\mathcal{A}$, (4) for each state q, the set of words that have a run originating at q and ending at q, visiting only states lower than q in the ordering are mapped to a unique idempotent $e_q \in S$. These properties of $\mathcal{A}$ ensure that, for any word $w \in \Sigma^* \cup \Sigma^\omega$, the unique accepting run of w produces a Ramsey split in the sense of [13], with the height of the split being bounded by the number of states of $\mathcal{A}$. The construction of $\mathcal{A}$ proceeds according to the local divisor technique, which uses a lexicographic induction on $(|S|, |\varphi(\Sigma)|)$. While the base cases (i) when S is a group, and (ii) $|\varphi(\Sigma)| = 1$ are easy, the inductive cases are non trivial. The inductive cases follow by identifying an element $c \in S$ for which $Sc \subsetneq S$ or $cS \subsetneq S$, and the details are in [21].

The forest factorization theorem can be derived easily from the construction of $\mathcal{A}$ as follows : consider a morphism $\varphi : \Sigma^+ \to S$, and define a monotone bijection $h : (Q, <) \to (\{1, 2, \ldots, |Q|\}, <)$. For any word $w = a_1 a_2 \cdots \in \Sigma^\infty$, consider the unique accepting run $q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} \ldots$ of w in $\mathcal{A}$. Define a split σ of w as $\sigma(i) = h(q_i)$ for all positions $i \geq 0$ in w. Two positions $i < j$ are σ-equivalent iff $q_i = q_j$ and $q_k \leq q_i$ for all $i \leq k \leq j$. We obtain this way, $w(i, j] = a_{i+1} \ldots a_j$ as a word whose run originates and ends in q_i, while visiting only states whose orderings are lower. Thus, $\varphi(w(i, j]) = e_{q_i}$ is the unique idempotent associated to q_i, resulting in σ being a Ramsey split. Thus, we obtain a Ramsey split using the construction of $\mathcal{A}$, s.t. the height of the factorization tree is bounded by the number of states of $\mathcal{A}$.

The second implication arising from the construction of $\mathcal{A}$ is that we obtain the good expressions used in this paper, by a state elimination of $\mathcal{A}$, using the ordering on its states.

In the rest of the section, we assume Theorem 12, and use it in obtaining an RTE corresponding to $\mathcal{A}$. For the purposes of this paper, we work with the transition monoid of the two-way transducer.

2.5. Transition monoid of 2NFAs

Consider a two-way possibly non-deterministic automaton (2NFA) $\mathcal{A}$. Let TrM be the transition monoid of $\mathcal{A}$ which is obtained by quotienting the free

monoid $(\Sigma^*, \cdot, \varepsilon)$ by a congruence which equates words behaving alike in the underlying automaton. Transition monoids for two way automata were defined in [9] for finite words and later extended to infinite words [17]. We recall the definition.

In a one way automaton, the canonical morphism $\mathsf{Tr}\colon \Sigma^* \to \mathsf{TrM}$ is such that $\mathsf{Tr}(w)$ consists of the set of pairs (p, q) such that there is a run from state p to state q reading w. In the case of two-way automaton, we also consider the starting side (left/right) and ending side (left/right) of the reading head while going from state p to q. This is represented with a *direction* d amongst "left-left" ($\curvearrowright$), "left-right" ($\rightarrow$), "right-left"($\leftarrow$) and "right-right"($\subsetneq$). Hence, an element of TrM is a set X of tuples (p, d, q) with $p, q \in Q$ states of $\mathcal{A}$ and $d \in \{\rightarrow, \curvearrowright, \subsetneq, \leftarrow\}$. The canonical morphism $\mathsf{Tr}\colon \Sigma^* \to \mathsf{TrM}$ is such that $\mathsf{Tr}(w)$ is the set of triples (p, d, q) which are compatible with w. For instance, $(p, \rightarrow, q) \in \mathsf{Tr}(w)$ iff $\mathcal{A}$ has a run starting in state p on the left most symbol of w and which exits w on its right and in state q. Likewise, $(p, \subsetneq, q) \in \mathsf{Tr}(w)$ iff $\mathcal{A}$ has a run starting in state p on the right most symbol of w and which exits w on its right and in state q. The explanation is similar for other directions.

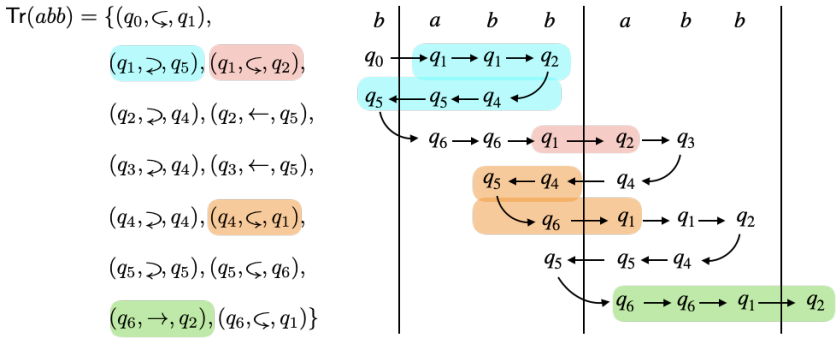


Figure 3: Illustrations of subset of $\mathsf{Tr}(abb)$

Consider the 2DFT $\mathcal{A}$ of Figure 2 and its underlying input 2DFA $\mathcal{B}$. The run for word *babbabb* starting from state q_0 is shown in Figure 3. In the transition monoid of $\mathcal{B}$, we have

$$\begin{aligned}\mathsf{Tr}(abb) = \{&(q_0, \subsetneq, q_1), (q_1, \curvearrowright, q_5), (q_1, \subsetneq, q_2), (q_2, \curvearrowright, q_4), (q_2, \leftarrow, q_5),\\ &(q_3, \curvearrowright, q_4), (q_3, \leftarrow, q_5), (q_4, \curvearrowright, q_4), (q_4, \subsetneq, q_1),\\ &(q_5, \curvearrowright, q_5), (q_5, \subsetneq, q_6), (q_6, \rightarrow, q_2), (q_6, \subsetneq, q_1)\}\,.\end{aligned}$$

Some of these triples are highlighted in Figure 3.

It is well-known that TrM is a monoid and that Tr is a morphism, see for instance [7]. The left-right and right-right relations were already used by Shepherdson to prove the equivalence between two-way and one-way automata [22]. These relations define a right-congruence. We obtain a congruence by considering also the right-left and left-left relations. The quotient of the free monoid by this congruence is the transition monoid of the 2NFA.

Let $(p, d, q) \in \mathsf{Tr}(w)$. If $w = a \in \Sigma$, then we know that reading a in state p, $\mathcal{A}$ may move in direction d and enter state q. If $w = w_1 \cdot w_2$ for $w_1, w_2 \in \Sigma^+$, then we can possibly decompose (p, d, q) into several "steps" depending on the behaviour of $\mathcal{A}$ on w starting in state p. As an example, see Figure 4, where we

decompose $(p, \to, q) \in \mathsf{Tr}(w)$. We show only those elements of $\mathsf{Tr}(w_1)$ and $\mathsf{Tr}(w_2)$ which help in the decomposition; the pictorial depiction is visually intuitive.

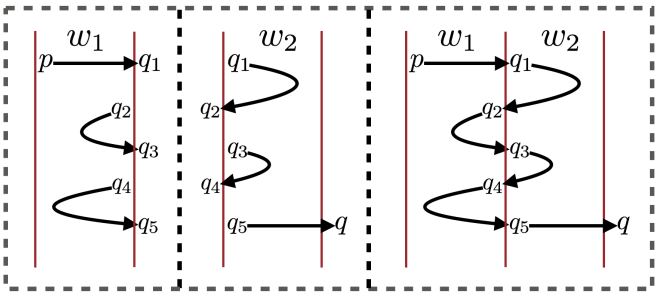
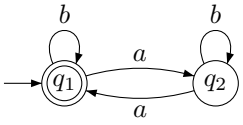


Figure 4: The first and second pictures are illustrations of subsets of $\mathsf{Tr}(w_1)$ and $\mathsf{Tr}(w_2)$ respectively. $(p, \to, q_1)$, $(q_2, \subsetneq, q_3)$, $(q_4, \subsetneq, q_5) \in \mathsf{Tr}(w_1)$ while $(q_1, \supsetneq, q_2), (q_3, \supsetneq, q_4), (q_5, \to, q) \in \mathsf{Tr}(w_2)$. The third picture shows that $(p, \to, q) \in \mathsf{Tr}(w_1 \cdot w_2)$: $(p, \to, q)$ consists of "steps" $(p, \to, q_1), (q_1, \supsetneq, q_2), (q_2, \subsetneq, q_3), (q_3, \supsetneq, q_4), (q_4, \subsetneq, q_5), (q_5, \to, q)$ alternately from $\mathsf{Tr}(w_1)$ and $\mathsf{Tr}(w_2)$.

Example 13. *Let $\Sigma = \{a, b\}$ and let $\mathcal{A}$ be the following 1DFT:*



Let TrM be the transition monoid of $\mathcal{A}$ and let $\mathsf{Tr}\colon \Sigma^* \to \mathsf{TrM}$ be the canonical morphism. The expression $F = a^+(ba)^+$ is not Tr-good: one of the reasons why F is not Tr-good is that the subexpression a^+ is such that $\mathsf{Tr}(a)$ is not an idempotent since $\mathsf{Tr}(a) = \{(q_1, q_2), (q_2, q_1)\}$ and $\mathsf{Tr}(a^2) = \{(q_1, q_1), (q_2, q_2)\}$, thus $\mathsf{Tr}(a^2) \neq \mathsf{Tr}(a)$. We have omitted the direction in the tuples as the underlying automaton is one way. Similarly, the subexpression $(ba)^+$ is also not Tr-good. The expression $F' = aba \cup aaba \cup a(aa)^+ba \cup a(baba)^+ \cup a(aa)^+(baba)^+$ is not Tr-good, even though each of the expressions $aba, aaba, a(aa)^+ba, a(baba)^+$ and $a(aa)^+(baba)^+$ are Tr-good. F' is not Tr-good since $\mathsf{Tr}(\mathcal{L}(F'))$ is not a singleton. The expression $F'' = aba \cup (aa)^+ \cup a(aa)^+ba$ is Tr-good.

2.6. 2DFT to RTE

In Appendix A, we give a practical example showing how to compute an RTE equivalent to the transducer $\mathcal{A}$ of Figure 2.

Consider a deterministic and complete two-way transducer $\mathcal{A}$. Let TrM be the transition monoid of the underlying input automaton. We can apply the unambiguous factorization theorem to the morphism $\mathsf{Tr}\colon \Sigma^* \to \mathsf{TrM}$ in order to obtain, for each $s \in \mathsf{TrM}$, an ε-free good rational expression F_s for $\mathsf{Tr}^{-1}(s) \setminus \{\varepsilon\}$. We use the unambiguous expression $G = \varepsilon \cup \bigcup_{s \in \mathsf{TrM}} F_s$ as a *guide* when constructing RTEs corresponding to the 2DFT $\mathcal{A}$.

Lemma 14. *Let F be an ε-free Tr-good rational expression and let $\mathsf{Tr}(F) = s_F$ be the corresponding element of the transition monoid TrM of $\mathcal{A}$. We can construct a map $C_F\colon s_F \to \mathsf{RTE}$ such that for each step $x = (p, d, q) \in s_F$ the following invariants hold:*

- *(I_1) $\mathsf{dom}(C_F(x)) = \mathcal{L}(F)$,*
- *(I_2) for each $u \in \mathcal{L}(F)$, $[\![C_F(x)]\!](u)$ is the output produced by $\mathcal{A}$ when running step x on u (i.e., running $\mathcal{A}$ on u from p to q following direction d).*

Proof. The proof is by structural induction on the rational expression. For each subexpression E of F we let $\mathsf{Tr}(E) = s_E$ be the corresponding element of the transition monoid TrM of $\mathcal{A}$. We start with atomic regular expressions. Since F is ε-free and $\emptyset$-free, we do not need to consider $E = \varepsilon$ or $E = \emptyset$.

atomic Assume that $E = a \in \Sigma$ is an atomic subexpression. Since the 2DFT $\mathcal{A}$ is deterministic and complete, for each state $p \in Q$ we have

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

Clearly, invariants (I_1) and (I_2) hold for all $x \in \mathsf{Tr}(a) = s_E$.

union Assume that $E = E_1 \cup E_2$. Since the expression is good, we deduce that $s_E = s_{E_1} = s_{E_2}$. For each $x \in s_E$ we define $C_E(x) = E_1 \,?\, C_{E_1}(x) : C_{E_2}(x)$. Since E is unambiguous we have $\mathcal{L}(E_1) \cap \mathcal{L}(E_2) = \emptyset$. Using (I_1) for E_1 and E_2, we deduce that

$$\begin{aligned}\mathsf{dom}(C_E(x)) &= (\mathcal{L}(E_1) \cap \mathsf{dom}(C_{E_1}(x))) \cup (\mathsf{dom}(C_{E_2}(x)) \setminus \mathcal{L}(E_1)) \\ &= \mathcal{L}(E_1) \cup \mathcal{L}(E_2) = \mathcal{L}(E)\,.\end{aligned}$$

Therefore, (I_1) holds for E. Now, for each $u \in \mathcal{L}(E)$, either $u \in \mathcal{L}(E_1)$ and $[\![C_E(x)]\!](u) = [\![C_{E_1}(x)]\!](u)$ or $u \in \mathcal{L}(E_2)$ and $[\![C_E(x)]\!](u) = [\![C_{E_2}(x)]\!](u)$. In both cases, applying (I_2) for E_1 or E_2, we deduce that $[\![C_E(x)]\!](u)$ is the output produced by $\mathcal{A}$ when running step x on u.

concatenation Assume that $E = E_1 \cdot E_2$ is a concatenation. Since the expression is good, we deduce that $s_E = s_{E_1} \cdot s_{E_2}$. Let $x \in s_E$.

- If $x = (p, \to, q)$ then, by definition of the product in the transition monoid TrM, there is a unique sequence of steps $x_1 = (p, \to, q_1)$, $x_2 = (q_1, \supsetneq, q_2)$, $x_3 = (q_2, \subsetneq, q_3)$, $x_4 = (q_3, \supsetneq, q_4)$, $\ldots$, $x_i = (q_{i-1}, \subsetneq, q_i)$, $x_{i+1} = (q_i, \to, q)$ with $i \geq 1$, $x_1, x_3, \ldots, x_i \in s_{E_1}$ and $x_2, x_4, \ldots, x_{i+1} \in s_{E_2}$ (see Figure 4). We define

$$\begin{aligned}C_E(x) = (C_{E_1}(x_1) \boxdot C_{E_2}(x_2)) \odot (C_{E_1}(x_3) \boxdot C_{E_2}(x_4)) \odot \cdots \odot \\ (C_{E_1}(x_i) \boxdot C_{E_2}(x_{i+1}))\,.\end{aligned}$$

  Notice that when $i = 1$ we simply have $C_E(x) = C_{E_1}(x_1) \boxdot C_{E_2}(x_2)$ with $x_2 = (q_1, \to, q)$.
  The concatenation $\mathcal{L}(E) = \mathcal{L}(E_1) \cdot \mathcal{L}(E_2)$ is unambiguous. Therefore, for all $y \in s_{E_1}$ and $z \in s_{E_2}$, using (I_1) for E_1 and E_2, we obtain $\mathsf{dom}(C_{E_1}(y) \boxdot C_{E_2}(z)) = \mathcal{L}(E)$. We deduce that $\mathsf{dom}(C_E(x)) = \mathcal{L}(E)$ and (I_1) holds for E.
  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 $u_2 \in \mathcal{L}(E_2)$. The step $x = (p, \to, q)$ performed by $\mathcal{A}$ on u is actually the concatenation of steps x_1 on u_1, followed by x_2 on u_2, followed by x_3 on u_1, followed by x_4 on u_2, $\ldots$, until x_{i+1} on u_2. Using (I_2) for E_1 and E_2, we deduce that the output produced by $\mathcal{A}$ while running step x on u is $[\![C_{E_1}(x_1)]\!](u_1) \cdot [\![C_{E_2}(x_2)]\!](u_2) \cdots [\![C_{E_1}(x_i)]\!](u_1) \cdot [\![C_{E_2}(x_{i+1})]\!](u_2) = [\![C_E(x)]\!](u)$.

- If $x = (p, \curvearrowright, q)$ then, following the definition of the product in the transition monoid TrM, we distinguish two cases.
  Either $x \in s_{E_1}$ and we let $C_E(x) = C_{E_1}(x) \boxdot (E_2 \,?\, \varepsilon : \bot)$. Since $\mathsf{dom}(E_2 \,?\, \varepsilon : \bot) = \mathcal{L}(E_2)$, we deduce as above that $\mathsf{dom}(C_E(x)) = \mathcal{L}(E)$. Moreover, let $u \in \mathcal{L}(E)$ and $u = u_1 u_2$ be its unique factorization with $u_1 \in \mathcal{L}(E_1)$ and $u_2 \in \mathcal{L}(E_2)$. The step $x = (p, \curvearrowright, q)$ performed by $\mathcal{A}$ on u reduces to the step x on u_1. Using (I_2) for E_1, we deduce that the output produced by $\mathcal{A}$ while making step x on u is $[\![C_{E_1}(x)]\!](u_1) = [\![C_E(x)]\!](u)$.

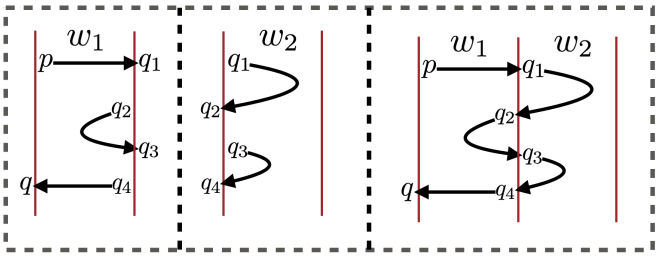


Figure 5: Let $w = w_1 \cdot w_2 \in \mathcal{L}(E)$ with $w_1 \in \mathcal{L}(E_1)$, $w_2 \in \mathcal{L}(E_2)$. We have $(p, \rightarrow, q_1), (q_2, \subsetneq, q_3), (q_4, \leftarrow, q) \in \mathsf{Tr}(w_1)$ and $(q_1, \curvearrowright, q_2), (q_3, \curvearrowright, q_4) \in \mathsf{Tr}(w_2)$. Then $(p, \curvearrowright, q)$ is composed of "steps" $(p, \rightarrow, q_1), (q_1, \curvearrowright, q_2), (q_2, \subsetneq, q_3), (q_3, \curvearrowright, q_4), (q_4, \leftarrow, q)$ alternately from $\mathsf{Tr}(w_1)$ and $\mathsf{Tr}(w_2)$.

  Or there is a unique sequence of steps (see Figure 5) $x_1 = (p, \rightarrow, q_1)$, $x_2 = (q_1, \curvearrowright, q_2)$, $x_3 = (q_2, \subsetneq, q_3)$, $x_4 = (q_3, \curvearrowright, q_4)$, $\ldots$, $x_i = (q_{i-1}, \leftarrow, q)$ with $i \geq 3$, $x_1, x_3, \ldots, x_i \in s_{E_1}$ and $x_2, x_4, \ldots, x_{i-1} \in s_{E_2}$. We define

$$C_E(x) = (C_{E_1}(x_1) \boxdot C_{E_2}(x_2)) \odot (C_{E_1}(x_3) \boxdot C_{E_2}(x_4)) \odot \cdots \odot (C_{E_1}(x_i) \boxdot (E_2 \,?\, \varepsilon : \bot)) \,.$$

  As for the first item, we can prove that invariants (I_1) and (I_2) are satisfied for E.
- The cases $x = (p, \leftarrow, q)$ or $x = (p, \subsetneq, q)$ are handled symmetrically. For instance, when $x = (p, \leftarrow, q)$, the unique sequence of steps is $x_1 = (p, \leftarrow, q_1)$, $x_2 = (q_1, \subsetneq, q_2)$, $x_3 = (q_2, \curvearrowright, q_3)$, $x_4 = (q_3, \subsetneq, q_4)$, $\ldots$, $x_i = (q_{i-1}, \curvearrowright, q_i)$, $x_{i+1} = (q_i, \leftarrow, q)$ with $i \geq 1$, $x_1, x_3, \ldots, x_i \in s_{E_2}$ and $x_2, x_4, \ldots, x_{i+1} \in s_{E_1}$ (see Figure 6). We define

$$C_E(x) = ((E_1 \,?\, \varepsilon : \bot) \boxdot C_{E_2}(x_1)) \odot (C_{E_1}(x_2) \boxdot C_{E_2}(x_3)) \odot \cdots \odot (C_{E_1}(x_{i-1}) \boxdot C_{E_2}(x_i)) \odot (C_{E_1}(x_{i+1}) \boxdot (E_2 \,?\, \varepsilon : \bot)) \,.$$

Kleene-plus Assume that $E = F^+$. Since the expression is good, we deduce that $s_E = s_F = s$ is an idempotent of the transition monoid TrM. Let $x \in s$.

- If $x = (p, \curvearrowright, q)$. Since F^+ is unambiguous, a word $u \in \mathcal{L}(F^+)$ admits a unique factorization $u = u_1 u_2 \cdots u_n$ with $n \geq 1$ and $u_i \in \mathcal{L}(F)$. Now, $\mathsf{Tr}(u_1) = s_E$ and since $x = (p, \curvearrowright, q) \in s_E$ the unique run ρ of $\mathcal{A}$ starting in state p on the left of u_1 exits on the left in state q. Therefore, the unique run of $\mathcal{A}$ starting in state p on the left of $u = u_1 u_2 \cdots u_n$ only visits u_1 and is actually ρ itself. Therefore, we

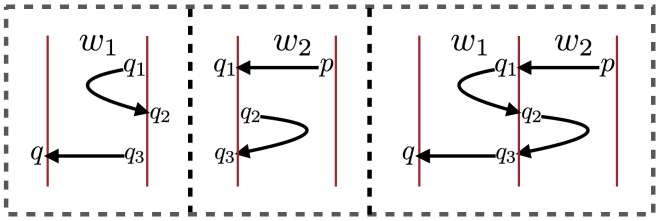


Figure 6: Let $w = w_1 \cdot w_2 \in \mathcal{L}(E)$ with $w_1 \in \mathcal{L}(E_1)$, $w_2 \in \mathcal{L}(E_2)$. We have $(p, \leftarrow, q_1), (q_2, \curvearrowright, q_3) \in \mathsf{Tr}(w_2)$ and $(q_1, \subsetneq, q_2), (q_3, \leftarrow, q) \in \mathsf{Tr}(w_1)$. Then $(p, \leftarrow, q) \in \mathsf{Tr}(w)$ is composed of "steps" $(p, \leftarrow, q_1), (q_1, \subsetneq, q_2), (q_2, \curvearrowright, q_3), (q_3, \leftarrow, q)$ alternately from $\mathsf{Tr}(w_2)$ and $\mathsf{Tr}(w_1)$.

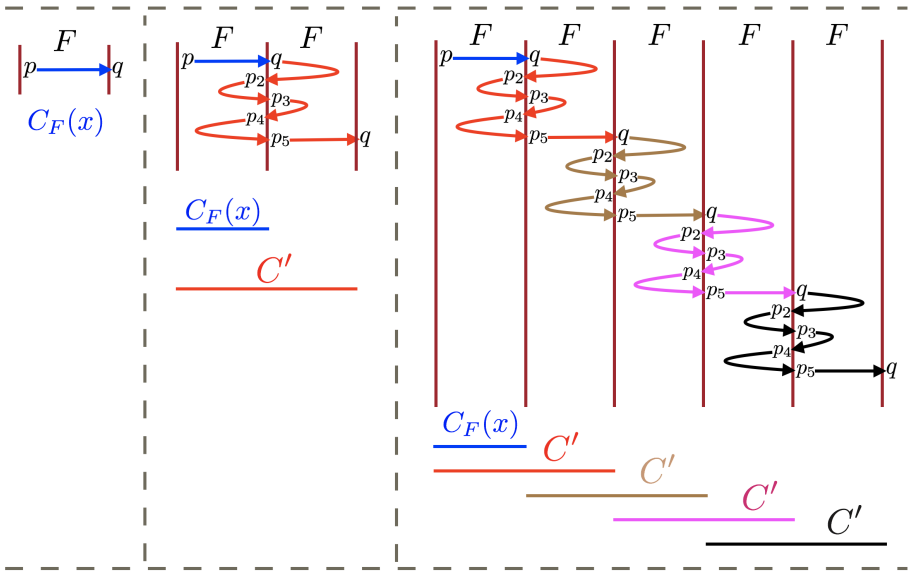


Figure 7: In the Kleene-plus $E = F^+$, a step $x = (p, \rightarrow, q) \in s_E$ on some $u = u_1 u_2 \cdots u_n$ with $u_\ell \in \mathcal{L}(F)$ is obtained by composing the following steps in s_F: $x_1 = x$, $x_2 = (q, \curvearrowright, p_2)$, $x_3 = (p_2, \subsetneq, p_3)$, $x_4 = (p_3, \curvearrowright, p_4)$, $x_5 = (p_4, \subsetneq, p_5)$, $x_6 = (p_5, \rightarrow, q)$.

set $C_E(x) = C_F(x) \boxdot (F^* \,?\, \varepsilon : \bot)$ and we can easily check that (I_1–I_2) are satisfied.

- Similarly for $x = (p, \subsetneq, q)$ we set $C_E(x) = (F^* \,?\, \varepsilon : \bot) \boxdot C_F(x)$.
- If $x = (p, \rightarrow, q)$. Recall that s is an idempotent, hence $x \in s^2$. We distinguish two cases.
  Assume first that $y = (q, \rightarrow, q) \in s$. Let $u = u_1 u_2 \cdots u_n$ be a word with $u_i \in \mathcal{L}(F)$ for $1 \leq i \leq n$. When reading u starting in state p on the left, the transducer will use step x on u_1 and then step y on each u_i with $2 \leq i \leq n$. Therefore, we set

$$C_E(x) = F \,?\, C_F(x) : \left(C_F(x) \boxdot (C_F(y))^{\boxplus}\right).$$

  Otherwise, there exists a unique sequence of steps in s: $x_1 = x$, $x_2 = (q, \curvearrowright, p_2)$, $x_3 = (p_2, \subsetneq, p_3)$, $x_4 = (p_3, \curvearrowright, p_4)$, $\ldots$, $x_i = (p_{i-1}, \subsetneq, p_i)$,

$x_{i+1} = (p_i, \to, q)$ with $i \geq 3$ (see Figure 7). We define

$$C_E(x) = \big(C_F(x) \boxdot (F^* \,?\, \varepsilon : \bot)\big) \odot [F, C']^{2^{\boxplus}}$$
$$C' = \big((F \,?\, \varepsilon : \bot) \boxdot C_F(x_2)\big) \odot (C_F(x_3) \boxdot C_F(x_4)) \odot \cdots \odot (C_F(x_i) \boxdot C_F(x_{i+1}))$$

Since the expression is good, the Kleene-plus $E = F^+$ is unambiguous. We have $\mathsf{dom}(C_F(x_j)) = \mathcal{L}(F)$ for $1 \leq j \leq i+1$ by (I_1). Also $\mathsf{dom}(F^* \,?\, \varepsilon : \bot) = \mathcal{L}(F^*)$. Since F^+ is unambiguous, the concatenation $F \cdot F^*$ is also unambiguous and we get $\mathsf{dom}(C_F(x) \boxdot (F^* \,?\, \varepsilon : \bot)) = \mathcal{L}(F) \cdot \mathcal{L}(F^*) = \mathcal{L}(E)$. Also, the product $F \cdot F$ is unambiguous and we deduce that $\mathsf{dom}(C_F(x_j) \boxdot C_F(x_{j+1})) = \mathcal{L}(F)^2$ for $1 \leq j \leq i$ and $\mathsf{dom}((F \,?\, \varepsilon : \bot) \boxdot C_F(x_2)) = \mathcal{L}(F)^2$. Therefore, $\mathsf{dom}(C') = \mathcal{L}(F)^2$ and using once again that F^+ is unambiguous, we deduce that $\mathsf{dom}([F, C']^{2^{\boxplus}}) = \mathcal{L}(F^+) = \mathcal{L}(E)$. We deduce that $\mathsf{dom}(C_E(x)) = \mathcal{L}(E)$ and (I_1) holds for E.

Let now $u \in \mathcal{L}(F^+) = \mathsf{dom}(C_E(x))$. We have to show that the output $\gamma \in \mathbb{D}$ produced by $\mathcal{A}$ when running step x on u is $[\![C_E(x)]\!](u)$. There is a unique factorization $u = u_1 u_2 \cdots u_n$ with $n \geq 1$ and $u_\ell \in \mathcal{L}(F)$ for $1 \leq \ell \leq n$.

Assume first that $n = 1$ (see Figure 7 left). By definition, we have $[\![[F, C']^{2^{\boxplus}}]\!](u) = \varepsilon$ and $[\![C_F(x) \boxdot (F^* \,?\, \varepsilon : \bot)]\!](u) = [\![C_F(x)]\!](u)$ which, by induction, is the output γ produced by $\mathcal{A}$ running step x on u. Therefore, $[\![C_E(x)]\!](u) = \gamma \cdot \varepsilon = \gamma$.

Assume now that $n \geq 2$ (see Figure 7 middle for $n = 2$ and right for $n = 5$). For $1 \leq \ell \leq n$ and $1 \leq j \leq i+1$, we denote $\gamma_j^\ell = [\![C_F(x_j)]\!](u_\ell)$ the output produced by $\mathcal{A}$ when running step x_j on u_ℓ. We can check (see Figure 7) that the output γ produced by $\mathcal{A}$ when running x on $u = u_1 u_2 \cdots u_n$ is

$$\gamma = \gamma_1^1(\gamma_2^2\gamma_3^1\gamma_4^2 \cdots \gamma_i^1\gamma_{i+1}^2)(\gamma_2^3\gamma_3^2\gamma_4^3 \cdots \gamma_i^2\gamma_{i+1}^3) \cdots (\gamma_2^n\gamma_3^{n-1}\gamma_4^n \cdots \gamma_i^{n-1}\gamma_{i+1}^n)\,.$$

We have $[\![C']\!](u_\ell u_{\ell+1}) = \gamma_2^{\ell+1}\gamma_3^\ell\gamma_4^{\ell+1} \cdots \gamma_i^\ell\gamma_{i+1}^{\ell+1}$ for $1 \leq \ell < n$. Therefore, we obtain $\gamma = \gamma_1^1 [\![C']\!](u_1u_2)[\![C']\!](u_2u_3) \cdots [\![C']\!](u_{n-1}u_n)$. Since $[\![C_F(x) \boxdot (F^* \,?\, \varepsilon : \bot)]\!](u) = \gamma_1^1$ we deduce that $\gamma = [\![C_E(x)]\!](u)$.

- The case of $x = (p, \leftarrow, q)$ can be handled similarly. □

Lemma 14 is the main ingredient in the construction of an RTE equivalent to a 2DFT.

Proof of Theorem 7 (2). First, we let $C_\varepsilon = [\![\mathcal{A}]\!](\varepsilon) \in \Gamma^* \cup \{\bot\}$. Then, we will define for each $s \in \mathsf{TrM}$, an RTE C_s such that $\mathsf{dom}(C_s) = \mathsf{dom}(\mathcal{A}) \cap (\mathsf{Tr}^{-1}(s) \setminus \{\varepsilon\})$ and $[\![C_s]\!](u) = [\![\mathcal{A}]\!](u)$ for all $u \in \mathsf{dom}(C_s)$. Assuming an arbitrary enumeration $s_1, s_2, \ldots, s_m$ of TrM, we define the final RTE as

$$C_{\mathcal{A}} = \varepsilon \,?\, C_\varepsilon : (\mathsf{Tr}^{-1}(s_1) \,?\, C_{s_1} : (\mathsf{Tr}^{-1}(s_2) \,?\, C_{s_2} : \cdots (\mathsf{Tr}^{-1}(s_{m-1}) \,?\, C_{s_{m-1}} : C_{s_m})))\,.$$

It remains to define the RTE C_s for $s \in \mathsf{TrM}$. We first define RTEs for steps in the 2DFT $\mathcal{A}$ on some input $\vdash u$ with $u \in \mathsf{Tr}^{-1}(s) \setminus \{\varepsilon\}$. Such a step must exit on the right since there are no transitions of $\mathcal{A}$ going left when reading $\vdash$. So either the step $(q_0, \to, q)$ starts on the left in the initial state q_0 and exits on the

right in some state q. Or the step $(p,\subsetneq,q)$ starts on the right in some state p and exits on the right in some state q. See Figure 8.

Let $s_{\vdash}$ be the set of steps $(p,\rightarrow,q),(p,\subsetneq,q)$ such that there is a transition $\delta(p,\vdash) = (q,\gamma_p,+1)$ in $\mathcal{A}$. From the initial state q_0 of $\mathcal{A}$, there is a unique sequence of steps $x_1 = (q_0,\rightarrow,q_1)$, $x_2 = (q_1,\supsetneq,q_2)$, $x_3 = (q_2,\subsetneq,q_3)$, $x_4 = (q_3,\supsetneq,q_4)$, $\ldots$, $x_i = (q_{i-1},\subsetneq,q_i)$, $x_{i+1} = (q_i,\rightarrow,q)$ with $i \geq 1$, $x_1, x_3, \ldots, x_i \in s_{\vdash}$ and $x_2, x_4, \ldots, x_{i+1} \in s$ (see Figure 8 left). We define

$$C_{\vdash F_s}((q_0,\rightarrow,q)) = \gamma_{q_0} \odot C_{F_s}(x_2) \odot \gamma_{q_2} \odot C_{F_s}(x_4) \odot \cdots \odot \gamma_{q_{i-1}} \odot C_{F_s}(x_{i+1})\,.$$

Notice that when $i = 1$ we simply have $C_{\vdash F_s}((q_0,\rightarrow,q)) = \gamma_{q_0} \odot C_{F_s}((q_1,\rightarrow,q))$. Since $\mathsf{dom}(C_{F_s}(x_i)) = \mathcal{L}(F_s) = \mathsf{Tr}^{-1}(s)\setminus\{\varepsilon\}$ for $i = 2,4,\ldots,i+1$, we deduce that $\mathsf{dom}(C_{\vdash F_s}((q_0,\rightarrow,q))) = \mathsf{Tr}^{-1}(s)\setminus\{\varepsilon\}$. Moreover, for each $u \in \mathsf{Tr}^{-1}(s)\setminus\{\varepsilon\}$, the output produced by $\mathcal{A}$ performing step $(q_0,\rightarrow,q)$ on $\vdash u$ is $[\![C_{\vdash F_s}((q_0,\rightarrow,q))]\!](u)$.

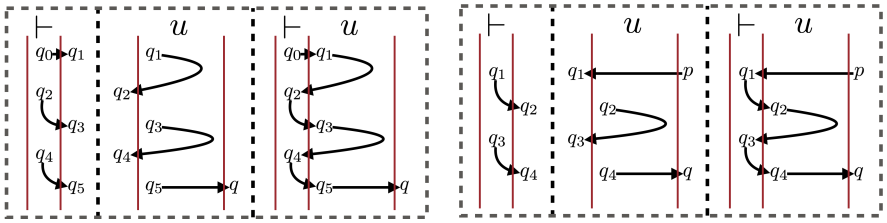


Figure 8: (Left) Given steps of s, a step $(q_0,\rightarrow,q)$ of $\vdash u$ for some $u \in F_s$, is obtained by composing the following steps alternatively from $s_{\vdash}$ and s: $x_1 = (q_0,\rightarrow,q_1)$, $x_2 = (q_1,\supsetneq,q_2)$, $x_3 = (q_2,\subsetneq,q_3)$, $x_4 = (q_3,\supsetneq,q_4)$, $x_5 = (q_4,\subsetneq,q_5)$, $x_6 = (q_5,\rightarrow,q)$. (Right) A step $(p,\subsetneq,q)$ of $\vdash u$ for some $u \in F_s$, is obtained by composing the following steps alternatively from s and $s_{\vdash}$: $x_1 = (p,\leftarrow,q_1)$, $x_2 = (q_1,\subsetneq,q_2)$, $x_3 = (q_2,\supsetneq,q_3)$, $x_4 = (q_3,\subsetneq,q_4)$, $x_5 = (q_4,\rightarrow,q)$.

Let p be a state of $\mathcal{A}$. Either there is a step $(p,\subsetneq,q) \in s$ and we let $C_{\vdash F_s}((p,\subsetneq,q)) = C_{F_s}((p,\subsetneq,q))$. Or, there is a unique sequence of steps $x_1 = (p,\leftarrow,q_1)$, $x_2 = (q_1,\subsetneq,q_2)$, $x_3 = (q_2,\supsetneq,q_3)$, $x_4 = (q_3,\subsetneq,q_4)$, $\ldots$, $x_i = (q_{i-1},\rightarrow,q)$ with $i \geq 3$, $x_1, x_3, \ldots, x_i \in s$ and $x_2, x_4, \ldots, x_{i-1} \in s_{\vdash}$ (see Figure 8 right). We define

$$C_{\vdash F_s}((p,\subsetneq,q)) = C_{F_s}(x_1) \odot \gamma_{q_1} \odot C_{F_s}(x_3) \odot \gamma_{q_3} \odot \cdots \odot \gamma_{q_{i-2}} \odot C_{F_s}(x_i)\,.$$

As above, we have $\mathsf{dom}(C_{\vdash F_s}((p,\subsetneq,q))) = \mathsf{Tr}^{-1}(s)\setminus\{\varepsilon\}$. Moreover, for each $u \in \mathsf{Tr}^{-1}(s)\setminus\{\varepsilon\}$, the output produced by $\mathcal{A}$ performing step $(p,\subsetneq,q)$ on $\vdash u$ is $[\![C_{\vdash F_s}((p,\subsetneq,q))]\!](u)$.

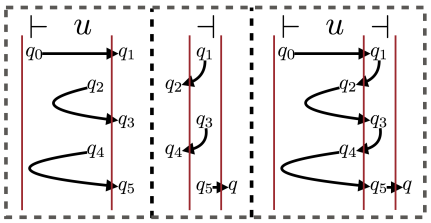


Figure 9: On input $\vdash u \dashv$, a step $x = (q_0,\rightarrow,q)$ is obtained by composing the following steps alternatively from steps of $\vdash u$ and $s_{\dashv}$: $x_1 = (q_0,\rightarrow,q_1)$, $x_2 = (q_1,\supsetneq,q_2)$, $x_3 = (q_2,\subsetneq,q_3)$, $x_4 = (q_3,\supsetneq,q_4)$, $x_5 = (q_4,\subsetneq,q_5)$ and $x_6 = (q_5,\rightarrow,q)$.

Similarly, let $s_{\dashv}$ be the set of steps $(p, \curvearrowright, q)$ such that there is a transition $\delta(p, \dashv) = (q, \gamma_p, -1)$ in $\mathcal{A}$ or steps $(p, \rightarrow, q)$ such that there is a transition $\delta(p, \dashv) = (q, \gamma_p, +1)$ in $\mathcal{A}$. From the initial state q_0 of $\mathcal{A}$, there is a unique sequence of steps $x_1 = (q_0, \rightarrow, q_1)$, $x_2 = (q_1, \curvearrowright, q_2)$, $x_3 = (q_2, \subsetneq, q_3)$, $x_4 = (q_3, \curvearrowright, q_4)$, $\ldots$, $x_i = (q_{i-1}, \subsetneq, q_i)$, $x_{i+1} = (q_i, \rightarrow, q)$ with $i \geq 1$, and $x_1, x_3, \ldots, x_i$ are steps where $C_{\vdash F_s}$ is defined and $x_2, x_4, \ldots, x_{i+1} \in s_{\dashv}$ (see Figure 9).

Notice that this sequence of steps corresponds to an accepting run iff $q \in F$ is an accepting state of $\mathcal{A}$. Therefore, either $q \notin F$ and $\mathsf{dom}(\mathcal{A}) \cap (\mathsf{Tr}^{-1}(s) \setminus \{\varepsilon\}) = \emptyset$ so we set $C_s = \bot$. Or, $q \in F$ and $\mathsf{Tr}^{-1}(s) \setminus \{\varepsilon\} \subseteq \mathsf{dom}(\mathcal{A})$ so we define

$$C_s = C_{\vdash F_s}(x_1) \odot \gamma_{q_1} \odot C_{\vdash F_s}(x_3) \odot \gamma_{q_3} \odot \cdots \odot C_{\vdash F_s}(x_i) \odot \gamma_{q_i} \,.$$

We have $\mathsf{dom}(C_s) = \mathsf{Tr}^{-1}(s) \setminus \{\varepsilon\}$ and for all $u \in \mathsf{dom}(C_s)$ we have $[\![C_s]\!](u) = [\![\mathcal{A}]\!](u)$. □

3. Infinite Words

In this section, we start looking at regular functions on infinite words. As in Section 2, 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.

3.1. Two-way transducers over ω-words (ω-2DMT$_{\mathsf{la}}$)

Let Σ be a finite input alphabet and let Γ be a finite output alphabet. 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. Below, we will use *complete unambiguous Büchi automata* (CUBA) [10], also called *backward deterministic Büchi automata* [24]). A deterministic two-way transducer (ω-2DMT$_{\mathsf{la}}$) over ω-words is given by $\mathcal{A} = (Q, \Sigma, \Gamma, q_0, \delta, \mathcal{F}, \mathcal{R})$, 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

$$\begin{cases} w'aq'w'' & \text{if } \delta(q, a, R) = (q', \gamma, +1) \\ w'_1 q' baw'' & \text{if } \delta(q, a, R) = (q', \gamma, -1) \text{ and } w' = w'_1 b \,. \end{cases}$$

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 $\vdash$:

$$q_0 \vdash w \xrightarrow{\gamma_1} w'_1 q_1 w''_1 \xrightarrow{\gamma_2} w'_2 q_2 w''_2 \xrightarrow{\gamma_3} w'_3 q_3 w''_3 \xrightarrow{\gamma_4} w'_4 q_4 w''_4 \cdots$$

We say that ρ reads the whole word w if $\mathsf{sup}\{|w'_n| \mid n > 0\} = \infty$. The set of states visited by ρ infinitely often is denoted $\mathsf{inf}(\rho) \subseteq Q$. The word w is accepted by $\mathcal{A}$, i.e., $w \in \mathsf{dom}(\mathcal{A})$ if ρ reads the whole word w and $\mathsf{inf}(\rho) \in \mathcal{F}$ is a Muller set. 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-$2\mathsf{DMT}_{\mathsf{la}}$ 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 15, where the look-ahead is necessary to obtain the required transformation.

Example 15. *Figure 1 shows an ω-2DMT$_{la}$ $\mathcal{A}'$ over $\Sigma = \{a, b, \#\}$ that defines the transformation $[\![\mathcal{A}']\!](u_1\#u_2\#\cdots\#u_n\#v) = u_1^R u_1\#\ u_2^R u_2\#\cdots\#u_n^R u_n\#v$ 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 language 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 $u^R u$; the infinite suffix over $\{a, b\}^\omega$ is then reproduced as it is.*

Remark 16. *The model used here is a two-way, deterministic Muller automaton, which has for each pair (q, a) consisting of a state and symbol, a tuple of look-ahead ω-regular languages which are mutually exclusive. The model (denoted 2WST$_{la}$) used in [4] however is a two-way deterministic Muller automaton which is equipped with a look-behind automaton (a NFA) and a look-ahead automaton (a possibly non-deterministic Muller automaton). It is easy to see that the two models are equivalent, see [16] for details.*

3.2. ω-Regular Transducer Expressions (ω-RTE)

As in the case of finite words, we define regular transducer expressions for 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. Again, $\perp$ acts as zero and the unit is the empty word $1_{\mathbb{D}} = \varepsilon$.

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

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

where $K \subseteq \Sigma^+$ ranges over *regular* languages of *finite non-empty words*, $L \subseteq \Sigma^\omega$ ranges over *ω-regular* languages of *infinite words* and E is an RTE over finite words as defined in Section 2.2. The semantics $[\![E]\!] \colon \Sigma^* \to \Gamma^* \cup \{\perp\}$ of the finitary combinator expressions $E \in \mathsf{RTE}$ is unchanged (see Section 2.2). The semantics of an ω-RTE C is a function $[\![C]\!] \colon \Sigma^\omega \to \mathbb{D}$. Given a regular language $K \subseteq \Sigma^+$, an ω-regular language $L \subseteq \Sigma^\omega$, and functions $f \colon \Sigma^* \to \Gamma^* \cup \{\perp\}$, $g, h \colon \Sigma^\omega \to \mathbb{D}$, we define

If then else. We have $\mathsf{dom}(L\,?\,g : h) = (\mathsf{dom}(g) \cap L) \cup (\mathsf{dom}(h) \setminus L)$.

Moreover, $(L\,?\,g : h)(w)$ is defined as $g(w)$ for $w \in \mathsf{dom}(g) \cap L$, and $h(w)$ for $w \in \mathsf{dom}(h) \setminus L$.

Hadamard product. We have $\mathsf{dom}(g \odot h) = g^{-1}(\Gamma^*) \cap \mathsf{dom}(h)$.

Moreover, $(g \odot h)(w) = g(w) \cdot h(w)$ for $w \in \mathsf{dom}(g) \cap \mathsf{dom}(h)$ with $g(w) \in \Gamma^*$.

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

Unambiguous ω-iteration. If $w \in \Sigma^\omega$ admits a unique infinite factorization $w = u_1 u_2 u_3 \cdots$ with $u_i \in \mathsf{dom}(f)$ 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, we set $f^\omega(w) = \bot$.

Unambiguous 2-chained ω-iteration. If $w \in \Sigma^\omega$ admits a unique factorization $w = u_1 u_2 u_3 \cdots$ with $u_i \in K$ for all $i \geq 1$ and if moreover $u_i u_{i+1} \in \mathsf{dom}(f)$ for all $i \geq 1$ then we set $[K, f]^{2\omega}(w) = f(u_1 u_2) f(u_2 u_3) f(u_3 u_4) \cdots$. Otherwise, we set $[K, f]^{2\omega}(w) = \bot$.

Remark 17. *Let $C_\varepsilon = (\Sigma \,?\, \varepsilon : \bot)^\omega$. We have $\mathsf{dom}(C_\varepsilon) = \Sigma^\omega$ 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 $\mathsf{dom}(C_\gamma) = \Sigma^\omega$ 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 18. *We can express the ω-iteration with the 2-chained ω-iteration as follows: $f^\omega = [\mathsf{dom}(f), f \boxdot (\mathsf{dom}(f) \,?\, \varepsilon : \bot)]^{2\omega}$.*

Remark 19. *In a similar manner to $[K, f]^{k\boxplus}$, we can extend 2-chained ω-iteration as well to k-chained ω-iteration for any $k \geq 3$. It is defined as follows: If w admits a unique factorization $w = u_1 u_2 \ldots$, with $u_i \in K$ for all $i \geq 1$, then $[K, f]^{k\omega}(w) = f(u_1 u_2 \ldots u_k) f(u_2 u_3 \ldots u_{k+1}) \ldots$. Otherwise, we set $[K, f]^{k\omega}(w) = \bot$. In [16], we have shown that adding k-chained ω-iteration does not increase the expressive power of ω-RTEs.*

Example 20. *We now give the ω-RTE for the transformation given in Example 15. It was also sketched in Example 1. Let*

$$E_1 = a \,?\, a : (b \,?\, b : (\# \,?\, \# : \bot))$$
$$E_2 = a \,?\, a : (b \,?\, b : \bot)$$
$$E_3 = a \,?\, a : (b \,?\, b : (\# \,?\, \varepsilon : \bot)) \,.$$

Then $\mathsf{dom}(E_1) = \mathsf{dom}(E_3) = (a + b + \#)$ and $\mathsf{dom}(E_2) = (a + b)$. Let

$$E_4 = ((a + b)^* \#) \,?\, ({E_3}^{\overleftarrow{\boxplus}} \odot {E_1}^{\boxplus}) : \bot \,.$$

We have $\mathsf{dom}(E_4) = (a + b)^* \#$ and, for $u \in (a + b)^*$, $[\![E_4]\!](u\#) = u^R u \#$ where u^R denotes the reverse of u. Next, let

$$C_1 = {E_4}^{\boxplus} \boxdot E_2^\omega \,.$$

Then, $\mathsf{dom}(C_1) = [(a + b)^* \#]^+ (a + b)^\omega$, and

$$[\![C_1]\!](u_1 \# u_2 \# \cdots u_n \# v) = u_1^R u_1 \# u_2^R u_2 \# \cdots \# u_n^R u_n \# v$$

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 $\mathsf{dom}(C) = [(a+b)^* \#]^* (a+b)^\omega$ and $[\![C]\!] = [\![\mathcal{A}']\!]$ where $\mathcal{A}'$ is the transducer of Figure 1.

The main theorem connecting ω-2DMT$_{\mathsf{la}}$ and ω-RTE is as follows.

Theorem 21. *ω-2DMT$_{\mathsf{la}}$ and ω-RTEs define the same class of functions. More precisely,*

1. *given an ω-RTE C, we can construct an ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}$ such that $[\![\mathcal{A}]\!] = [\![C]\!]$.*
2. *given an ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}$, 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 3.7 after some preparatory work on backward deterministic Büchi automata (Section 3.4) which are used to remove the look-ahead of ω-2DMT$_{\mathsf{la}}$ (Section 3.5), and the notion of transition monoid for ω-2DMT$_{\mathsf{la}}$ (Section 3.6) used in the unambiguous forest factorization theorem extended to infinite words (Theorem 28).

3.3. ω-RTE to ω-2DMT$_{\mathsf{la}}$

In this section, we prove one direction of Theorem 21: 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 22. *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$_{\mathsf{la}}$ M_g and M_h respectively. Then, one can construct*

1. *an ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}$ such that $[\![L \,?\, g : h]\!] = [\![\mathcal{A}]\!]$,*
2. *an ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}$ such that $[\![\mathcal{A}]\!] = [\![g \odot h]\!]$,*
3. *an ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}$ such that $[\![\mathcal{A}]\!] = [\![f \boxdot g]\!]$,*
4. *an ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}$ such that $[\![\mathcal{A}]\!] = [\![f^\omega]\!]$,*
5. *an ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}$ 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 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.** Recall that for a word w to be in $\mathsf{dom}(g \odot h)$ we should have $w \in \mathsf{dom}(g) \cap \mathsf{dom}(h)$ and also $[\![g]\!](w) \in \Gamma^*$. Hence, M_g will produce $[\![g]\!](w)$ after reading a finite prefix of w. We create a look ahead which indicates the position where the transducer M_g can stop reading the input word w so that we can reset the head to the left most position and start M_h. The look ahead should satisfy two conditions for this purpose:

- M_g will not visit any position to the left of the current position in its remaining run on w.
- The output produced by running M_g on the suffix of w should be ε.

To accommodate these two conditions, we construct for each state $q \in Q_g$, a transducer A_q and we define an ω-regular look ahead language as $L_q = \mathsf{dom}(A_q)$. The structure of A_q is the same as M_g except that we

- add a new initial state ι_q and the transition $\delta_q(\iota_q, \vdash, \Sigma^\omega) = (q, \varepsilon, +1)$,
- remove all transitions from M_g where the output is $\gamma \neq \varepsilon$,
- remove all 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 $Q_\mathcal{A} = Q_g \cup Q_h \cup \{\mathsf{reset}\}$. 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

$$\delta_\mathcal{A}(q, a, R \setminus L_{q'}) = (q', \gamma, +1) \quad \text{and} \quad \delta_\mathcal{A}(q, a, R \cap L_{q'}) = (\mathsf{reset}, \gamma, +1).$$

From the reset state, we go to the left until $\vdash$ is reached and then start running M_h. So, $\delta_\mathcal{A}(\mathsf{reset}, a, \Sigma^\omega) = (\mathsf{reset}, \varepsilon, -1)$ for all $a \in \Sigma$ and $\delta_\mathcal{A}(\mathsf{reset}, \vdash, R) = (q'', \gamma, +1)$ 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 $\mathsf{dom}(M_f)$ and a deterministic Muller automaton (DMA) $\mathcal{D}_g = (Q_g, \Sigma, \delta_g, s_g, \mathcal{F}_g)$ that accepts $\mathsf{dom}(M_g)$.

Now, the set L of words w having at least two factorizations $w = u_1v_1 = u_2v_2$ with $u_1, u_2 \in \mathsf{dom}(f)$, $v_1, v_2 \in \mathsf{dom}(g)$ 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, $\mathsf{dom}(f \boxdot g) = (\mathsf{dom}(f) \cdot \mathsf{dom}(g)) \setminus L$ is ω-regular.

First we construct an ω-1DMT$_{\mathsf{la}}$ $\mathcal{D}$ such that $\mathsf{dom}(\mathcal{D}) = \mathsf{dom}(f \boxdot g)$ and on an input word $w = uv$ with $u \in \mathsf{dom}(f)$ and $v \in \mathsf{dom}(g)$, 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 $\mathsf{dom}(f \boxdot g)$ 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 $\mathsf{dom}(g)$ 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

$$\delta_\mathcal{D}(q, a, \mathsf{dom}(g)) = (s_g, a\#, +1) \quad \text{and} \quad \delta_\mathcal{D}(q, a, \Sigma^\omega \setminus \mathsf{dom}(g)) = (q', a, +1).$$

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 $\mathsf{dom}(f \boxdot g)$ such that $u \in \mathsf{dom}(f)$ and $v \in \mathsf{dom}(g)$.

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 the 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 [14]. 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 Remark 18, 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 $w = u_1 u_2 \cdots$ 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 $w = u_1 v_1 = u_2 v_2$ 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 $u_1 u_2 \cdots$ 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 $u_1 u_2 \# u_2 u_3 \# \cdots$.

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 then obtained as the composition of $\mathcal{D}$, $\mathcal{D}'$ and $\mathcal{T}$. The output produced by $\mathcal{A}$ is thus $[\![M_f]\!](u_1 u_2)[\![M_f]\!](u_2 u_3) \cdots$. □

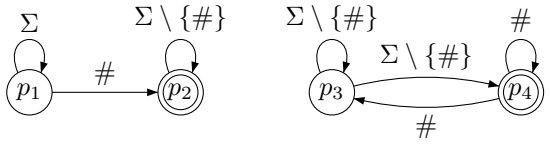
*3.4. Backward deterministic Büchi automata (*BDBA*)*

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 $w = a_1 a_2 a_3 \cdots$ 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 $\mathsf{inf}(\rho) \cap \mathsf{Fin} \neq \emptyset$ where $\mathsf{inf}(\rho)$ is the set of states visited infinitely often by ρ. This is a Büchi acceptance condition.

The Büchi automaton $\mathcal{B}$ is *backward deterministic* (BDBA) or *complete unambiguous* (CUBA) 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 $\Delta^{-1} \colon P \times \Sigma \to P$ 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. Notice that, $\Sigma^\omega = \biguplus_{p \in P} \mathcal{L}(\mathcal{B}, p)$, i.e., words in Σ^ω are partitioned according to the starting state of their unique final run. 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 23. *For instance, the automaton $\mathcal{B}$ below is a* BDBA*. Moreover, 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 24 (Carton & Michel [10])**.** *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 [10] 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 [10]. We obtain immediately

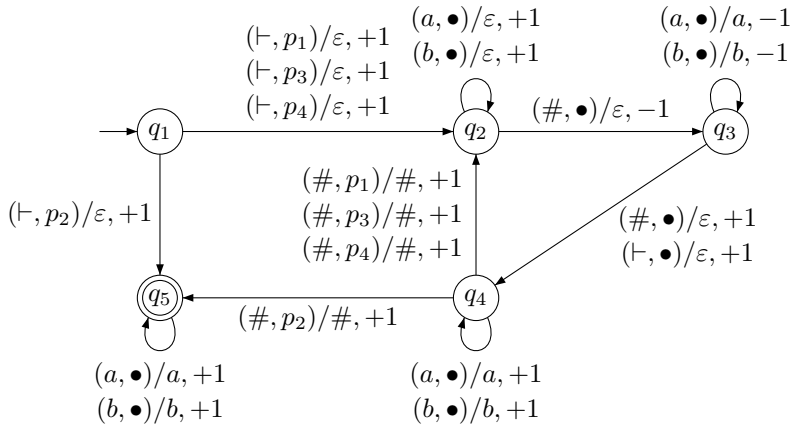
Corollary 25. *Let $\mathcal{R}$ be a finite family of ω-regular languages. There is a* BDBA *$\mathcal{B}$ and a tuple of initial sets $(I_R)_{R\in\mathcal{R}}$ such that $R = \mathcal{L}(\mathcal{B}, I_R)$ for all $R \in \mathcal{R}$.*

3.5. Replacing the look-ahead of an ω-2DMT$_{\mathsf{la}}$ with a BDBA

Let $\mathcal{A} = (Q, \Sigma, \Gamma, q_0, \delta, \mathcal{F}, \mathcal{R})$ be an ω-2DMT$_{\mathsf{la}}$. By Corollary 25 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 finite family $\mathcal{R}$ of ω-regular languages used as look-ahead by the transducer $\mathcal{A}$. Recall 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. We deduce that $(I_R)_{R\in\mathcal{R}(q,a)}$ is a partition of P.

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 $\widetilde{\delta}\colon Q \times \widetilde{\Sigma_\vdash} \to Q \times \Gamma^* \times \{-1, +1\}$ is defined as follows: for $q \in Q$ and $(a, p) \in \widetilde{\Sigma_\vdash}$ we let $\widetilde{\delta}(q, (a, p)) = \delta(q, a, R)$ for the unique $R \in \mathcal{R}(q, a)$ such that $p \in I_R$.

Example 26. *For instance, the ω-2DMT $\widetilde{\mathcal{A}}$ constructed from the ω-2DMT$_{\mathsf{la}}$ of Figure 1 and the* BDBA *$\mathcal{B}$ of Example 23 is depicted below, where $\bullet$ stands for an arbitrary state of $\mathcal{B}$.*



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

$$q_0\vdash w \xrightarrow{\gamma_1} w_1'q_1w_1'' \xrightarrow{\gamma_2} w_2'q_2w_2'' \xrightarrow{\gamma_3} w_3'q_3w_3'' \xrightarrow{\gamma_4} w_4'q_4w_4''\cdots$$

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 27. *For all words $w \in \Sigma^\omega$, the ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}$ starting from $\vdash w$ accepts iff the ω-2DMT $\widetilde{\mathcal{A}}$ starting from $\widetilde{\vdash w}$ accepts, and in this case they compute the same output in Γ^∞.*

3.6. Transition monoid of an ω-2DMT$_{\mathsf{la}}$

We use the notations of the previous sections, in particular for the ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}$, 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$_{\mathsf{la}}$ $\mathcal{A}$, 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 the transition monoid defined in Section 2.5. 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 $d \in \{\leftsquigarrow, \leftarrow\}$) exits in state q' on the left of $\widetilde{u}$ if $d \in \{\rightsquigarrow, \leftarrow\}$ (resp. on the right if $d \in \{\rightarrow, \leftsquigarrow\}$) 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 26, we have $(q_4, \rightarrow, \{q_2, q_3, q_4\}, q_5) \in \widetilde{\mathsf{Tr}}(\widetilde{u})$ when $\widetilde{u} \in ((a,p_1)+(b,p_1))^*(\#,p_1)((a,p_1)+(b,p_1))^*(\#,p_2)$.

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 of the transition monoid of a two-way automaton [7] is extended by taking the union of the sets X occurring in a sequence of steps. For instance, if we have steps $(q_0, \rightarrow, X_1, q_1)$, $(q_2, \leftsquigarrow, X_3, q_3)$, $\ldots$, $(q_{i-1}, \leftsquigarrow, X_i, q_i)$ 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 $0 \leq i < n$ 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 $p_0, a_1, p_1, \ldots, a_n, p_n$. We 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.

We define the transition monoid of $(\mathcal{A}, \mathcal{B})$ as 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.

3.7. ω-2DMT$_{\mathsf{la}}$ to ω-RTE

We prove in this section that from an ω-2DMT$_{\mathsf{la}}$ $\mathcal{A}$ we can construct an equivalent ω-RTE. The proof follows the ideas already used for finite words in Section 2.6. We will use the following generalization to infinite words of the unambiguous forest factorization Theorem 12.

Theorem 28 (Unambiguous Forest Factorization [21])**.** *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 $F_k \cdot G_k^+$ are ε-free φ-good rational expressions for all $1 \leq k \leq m$.*

We will apply this theorem to the morphism $\mathsf{Tr} \colon \Sigma^* \to \mathsf{TrM}$ defined in Section 3.6. 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$_{\mathsf{la}}$ $\mathcal{A}$.

The following lemma is similar to Lemma 14. It shows how to construct the RTEs associated with steps of elements of the transition monoid TrM.

Lemma 29. *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 $C_G^p \colon s_G^p \to \mathsf{RTE}$ 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 structural 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 construction is similar to the one given in Section 2.6. 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 $\mathcal{L}(E_1) \cap \mathcal{L}(E_2) = \emptyset$. As in Section 2.6 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 the transition monoid $\widetilde{\mathsf{TrM}}$, there is a unique sequence of steps $x_1 = (q,\to,X_1,q_1)$, $x_2 = (q_1,\supsetneq,X_2,q_2)$, $x_3 = (q_2,\subsetneq,X_3,q_3)$, $x_4 = (q_3,\supsetneq,X_4,q_4)$, $\ldots$, $x_i = (q_{i-1},\subsetneq,X_i,q_i)$, $x_{i+1} = (q_i,\to,X_{i+1},q')$ 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 $X = X_1 \cup \cdots \cup X_{i+1}$ (see Figure 10). We define

$$C_E^p(x) = (C_{E_1}^{p_1}(x_1) \boxdot C_{E_2}^p(x_2)) \odot (C_{E_1}^{p_1}(x_3) \boxdot C_{E_2}^p(x_4)) \odot \cdots \odot (C_{E_1}^{p_1}(x_i) \boxdot C_{E_2}^p(x_{i+1}))\,.$$

Notice 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. Therefore, 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 obtain $\mathsf{dom}(C_{E_1}^{p_1}(y) \boxdot C_{E_2}^p(z)) = \mathcal{L}(E)$. We deduce that $\mathsf{dom}(C_E(x)) = \mathcal{L}(E)$ and (J_1) holds for E and $x = (q,\to,X,q')$.

Now, let $u \in \mathcal{L}(E)$ and let $u = u_1u_2$ be its unique factorization with $u_1 \in \mathcal{L}(E_1)$ and $u_2 \in \mathcal{L}(E_2)$. We have $\widetilde{u_1u_2}^p = \widetilde{u_1}^{p_1} \cdot \widetilde{u_2}^p$. Hence, the step $x = (q,\to,X,q')$ performed by $\widetilde{\mathcal{A}}$ on $\widetilde{u}^p$ is actually the concatenation of

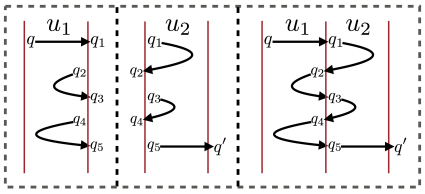


Figure 10: In the concatenation $E = E_1 \cdot E_2$, a step $x = (q, \to, X, q') \in s_E^p$ on some u_1u_2 with $u_1 \in E_1$ and $u_2 \in E_2$, is obtained by composing the following steps alternatively from $s_{E_1}^{p_1}$ and $s_{E_2}^p$ for a unique state p_1: $x_1 = (q, \to, X_1, q_1)$, $x_2 = (q_1, \curvearrowright, X_2, q_2)$, $x_3 = (q_2, \subsetarrow, X_3, q_3)$, $x_4 = (q_3, \curvearrowright, X_4, q_4)$, $x_5 = (q_4, \subsetarrow, X_5, q_5)$, $x_6 = (q_5, \to, X_6, q')$ with $X = X_1 \cup \cdots \cup X_6$.

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 $x = (q, \curvearrowright, X, q')$ or $x = (q, \subsetarrow, X, q')$ 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 $p \in P$ such that $p = r^p$. Let $x \in s^p$.

- If $x = (q, \curvearrowright, X, q')$. Since F^+ is unambiguous, a word $u \in \mathcal{L}(F^+)$ admits a unique factorization $u = u_1u_2 \cdots u_n$ 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 $x = (q, \curvearrowright, X, q') \in s^p$, 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. Therefore, we set $C_E^p(x) = C_F^p(x) \boxdot (F^* \,?\, \varepsilon : \bot)$ and we can easily check that $(\mathsf{J}_1\text{–}\mathsf{J}_2)$ are satisfied.
- Similarly for $x = (q, \subsetarrow, X, q')$ 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 and 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 $y = (q', \curvearrowright, X_2, q_2) \in s^p$ goes to the left. Since s^p is an idempotent, there exists a unique sequence of steps in s^p: $x_1 = x$, $x_2 = y$, $x_3 = (q_2, \subsetarrow, X_3, q_3)$, $x_4 = (q_3, \curvearrowright, X_4, q_4)$, ..., $x_i = (q_{i-1}, \subsetarrow$

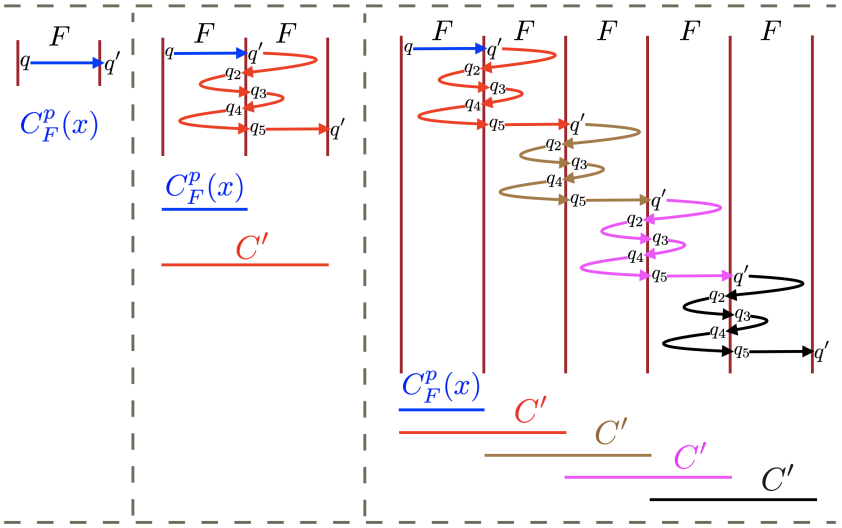


Figure 11: In the Kleene-plus $E = F^+$, a step $x = (q, \to, X, q') \in s_E^p$ on some $u = u_1 u_2 \cdots u_n$ with $u_\ell \in \mathcal{L}(F)$ is obtained by composing the following steps in s_F^p: $x_1 = x$, $x_2 = (q', \curvearrowright, X_2, q_2)$, $x_3 = (q_2, \subsetneq, X_3, q_3)$, $x_4 = (q_3, \curvearrowright, X_4, q_4)$, $x_5 = (q_4, \subsetneq, X_5, q_5)$, $x_6 = (q_5, \to, X_6, q')$ with $X = X_1 \cup \cdots \cup X_6$.

$, X_i, q_i)$, $x_{i+1} = (q_i, \to, X_{i+1}, q')$ with $i \geq 3$ (see Figure 11). We define

$$C_E^p(x) = \big(C_F^p(x) \boxdot (F^* \,?\, \varepsilon : \bot)\big) \odot [F, C']^{2\boxplus}$$
$$C' = \big((F \,?\, \varepsilon : \bot) \boxdot C_F^p(x_2)\big) \odot (C_F^p(x_3) \boxdot C_F^p(x_4)) \odot \cdots \odot (C_F^p(x_i) \boxdot C_F^p(x_{i+1}))$$

The proof of correctness, i.e., that (J_1–J_2) are satisfied for E, is as in Section 2.6.

- If $x = (q, \leftarrow, X, q')$, the proof is obtained mutatis mutandis, 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 $x = (q, \curvearrowright, X, q') \in s^p$. 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 $y = (q, \curvearrowright, X, q') = x$ since $s^p = s^{p'} \cdot s^p$

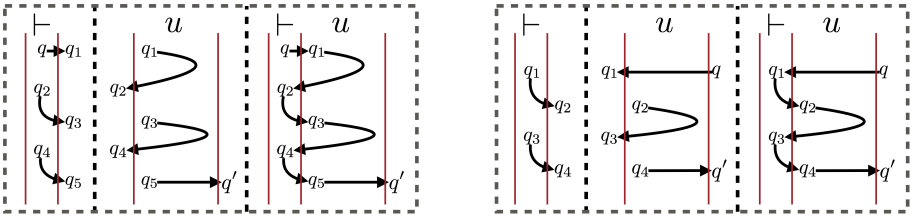


Figure 12: (Left) Given a look-ahead $p \in P$, a step $(q, \rightarrow, q')$ of $\vdash u$ for some u with $\mathsf{Tr}(u) = (r^p, b^p, s^p)_{p \in P}$, is obtained by composing the following steps alternatively from $s^p_\vdash$ and s^p: $x_1 = (q, \rightarrow, q_1)$, $x_2 = (q_1, \curvearrowright, X_2, q_2)$, $x_3 = (q_2, \subsetneq, q_3)$, $x_4 = (q_3, \curvearrowright, X_4, q_4)$, $x_5 = (q_4, \subsetneq, q_5)$, $x_6 = (q_5, \rightarrow, X_6, q')$. (Right) Similarly, a step $(q, \subsetneq, q')$ of $\vdash u$ is obtained by composing the following steps alternatively from s^p and $s^p_\vdash$: $x_1 = (q, \leftarrow, X_1, q_1)$, $x_2 = (q_1, \subsetneq, q_2)$, $x_3 = (q_2, \curvearrowright, X_3, q_3)$, $x_4 = (q_3, \subsetneq, q_4)$, $x_5 = (q_4, \rightarrow, X_5, q')$.

(recall that σ is idempotent) and we define

$$C^p_E(x) = F \,?\, C^p_F(x) : (C^{p'}_F(x) \boxdot (F^+ \,?\, \varepsilon : \bot))\,.$$

If it goes to the right, then $y = (q, \rightarrow, X_1, q_1)$ and there exists a unique sequence of steps: $x_1 = y$, $x_2 = (q_1, \curvearrowright, X_2, q_2)$, $x_3 = (q_2, \subsetneq, X_3, q_3)$, $x_4 = (q_3, \curvearrowright, X_4, q_4)$, $\ldots$, $x_i = (q_{i-1}, \leftarrow, X_i, q')$ with $i \geq 3$, $x_1, x_3, \ldots, x_i \in s^{p'}$ and $x_2, \ldots, x_{i-1} \in s^p$. Notice that $X = X_1 \cup \cdots \cup X_i$. We define $C^p_E(x) = F \,?\, C^p_F(x) : C'$ where

$$C' = (C^{p'}_E(x_1) \boxdot C^p_F(x_2)) \odot \cdots \odot (C^{p'}_E(x_{i-2}) \boxdot C^p_F(x_{i-1})) \odot \left(C^{p'}_E(x_i) \boxdot (F \,?\, \varepsilon : \bot)\right).$$

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
  $$C^p_E(x) = F \,?\, C^p_F(x) : (C^{p'}_E(y) \overleftarrow{\boxdot} C^p_F(x))\,.$$
  If it goes back to the right, then $y = (q', \subsetneq, X_2, q_2)$ and there exists a unique sequence of steps: $x_1 = x$, $x_2 = y$, $x_3 = (q_2, \curvearrowright, X_3, q_3)$, $x_4 = (q_3, \subsetneq, X_4, q_4)$, $\ldots$, $x_i = (q_{i-1}, \curvearrowright, X_i, q_i)$ $x_{i+1} = (q_i, \leftarrow, X_{i+1}, q')$ with $i \geq 3$, $x_1, x_3, \ldots, x_i \in s^p$ and $x_2, \ldots, x_{i+1} \in s^{p'}$. Notice that $X_2 \cup \cdots \cup X_{i+1} \subseteq X$. We define $C^p_E(x) = F \,?\, C^p_F(x) : C'$ where
  $$C' = (C^{p'}_E(x_2) \overleftarrow{\boxdot} C^p_F(x_1)) \odot \cdots \odot (C^{p'}_E(x_{i-1}) \overleftarrow{\boxdot} C^p_F(x_{i-2})) \odot (C^{p'}_E(x_{i+1}) \overleftarrow{\boxdot} C^p_F(x_i))\,.$$
  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 $x = (q, \subsetneq, X, q') \in s^p$ 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 $u \in \Sigma^+$. As before, the look-ahead is determined by the state of the BDBA $\mathcal{B}$.

Lemma 30. *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* RTEs *$C^p_{\vdash F}((q,\to,q'))$ (resp. $C^p_{\vdash F}((q,\subsetneq,q'))$) such that the following invariants hold:*

(K_1*)* $\mathsf{dom}(C^p_{\vdash F}((q,\to,q'))) = \mathcal{L}(F)$ *(resp.* $\mathsf{dom}(C^p_{\vdash F}((q,\subsetneq,q'))) = \mathcal{L}(F)$*),*

(K_2*) for each $u \in \mathcal{L}(F)$, $[\![C^p_{\vdash F}((q,\to,q'))]\!](u)$ (resp. $[\![C^p_{\vdash F}((q,\subsetneq,q'))]\!](u)$) 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'.*

Proof. Let $\sigma = (r^p, b^p, s^p)_{p\in P} = \mathsf{Tr}(F)$. We fix some state $p \in P$. For all words $u \in \mathcal{L}(F)$, we have $\widetilde{\vdash u}^p = (\vdash, r^p)\widetilde{u}^p$. Let $s^p_\vdash$ be the set of steps $(q,\to,q'), (q,\subsetneq,q')$ such that $\widetilde{\delta}(q,(\vdash,r^p)) = (q',\gamma^p_q,+1)$ in $\widetilde{\mathcal{A}}$.

For each $q \in Q$, there is a unique sequence of steps $x_1 = (q,\to,q_1)$, $x_2 = (q_1,\supsetneq,X_2,q_2)$, $x_3 = (q_2,\subsetneq,q_3)$, $x_4 = (q_3,\supsetneq,X_4,q_4)$, …, $x_i = (q_{i-1},\subsetneq,q_i)$, $x_{i+1} = (q_i,\to,X_{i+1},q')$ with $i \geq 1$, $x_1, x_3, \ldots, x_i \in s^p_\vdash$ and $x_2, x_4, \ldots, x_{i+1} \in s^p$ (see Figure 12 left). We define

$$C^p_{\vdash F}((q,\to,q')) = \gamma^p_q \odot C^p_F(x_2) \odot \gamma^p_{q_2} \odot C^p_F(x_4) \odot \cdots \odot \gamma^p_{q_{i-1}} \odot C^p_F(x_{i+1})\,.$$

Using Lemma 29, we can show that $\mathcal{L}(F) = \mathsf{dom}(C^p_{\vdash F}((q,\to,q')))$ and also that for each $u \in \mathcal{L}(F)$, $[\![C^p_{\vdash F}((q,\to,q'))]\!](u)$ is the output produced by $\widetilde{\mathcal{A}}$ on $\widetilde{\vdash u}^p$ when starting on the left in state q until it exists on the right in state q'.

For each $q \in Q$, there is a unique sequence of steps $x_1 = (q,\leftarrow,X_1,q_1)$, $x_2 = (q_1,\subsetneq,q_2)$, $x_3 = (q_2,\supsetneq,X_3,q_3)$, $x_4 = (q_3,\subsetneq,q_4)$, …, $x_i = (q_{i-1},\subsetneq,q_i)$, $x_{i+1} = (q_i,\to,X_{i+1},q')$ with $i \geq 2$, $x_2, x_4, \ldots, x_i \in s^p_\vdash$ and $x_1, x_3, \ldots, x_{i+1} \in s^p$ (see Figure 12 right). We define

$$C^p_{\vdash F}((q,\subsetneq,q')) = C^p_F(x_1) \odot \gamma^p_{q_1} \odot C^p_F(x_3) \odot \gamma^p_{q_3} \odot \cdots \odot \gamma^p_{q_{i-1}} \odot C^p_F(x_{i+1})\,.$$

Using Lemma 29, we can show that $\mathcal{L}(F) = \mathsf{dom}(C^p_{\vdash F}((q,\subsetneq,q')))$ and also that for each $u \in \mathcal{L}(F)$, $[\![C^p_{\vdash F}((q,\subsetneq,q'))]\!](u)$ is the output produced by $\widetilde{\mathcal{A}}$ on $\widetilde{\vdash u}^p$ when starting on the right in state q until it exists on the right in state q'. □

Lemma 31. *Let $F \cdot G^\omega$ be an unambiguous rational expression such that F and G are ε-free* Tr*-good rational expressions and $\mathsf{Tr}(G) = \sigma = (r^p, b^p, s^p)_{p\in P}$ is an idempotent in the transition monoid* TrM *of $(\mathcal{A},\mathcal{B})$. We can construct an ω-*RTE *C_{FG^ω} such that $\mathsf{dom}(C_{FG^\omega}) = \mathcal{L}(FG^\omega) \cap \mathsf{dom}(\mathcal{A})$ and for each $w \in \mathsf{dom}(C_{FG^\omega})$, $[\![C_{FG^\omega}]\!](w) = [\![\mathcal{A}]\!](w)$.*

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 $w = u_1u_2u_3\cdots \in \mathcal{L}(FG^\omega)$ 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 $\mathsf{Tr}(u_2\cdots u_{n+1}) = \mathsf{Tr}(u_{n+1})$. 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 $\mathsf{Tr}(u_n) = \sigma$ 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 $p = r^p$ and $b^p = 1$. Let $v \in \mathcal{L}(G)$. We have $p \xleftarrow{v} p$ and this subrun 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, which proves the unicity of p.

We apply Lemma 30. We denote by $s'_{\vdash F}$ the set of triples $(q,d,q') \in Q \times \{\to, \curvearrowright, \subsetarrow\} \times Q$ such that the RTE $C^p_{\vdash F}(q,d,q')$ is defined.

Starting from the initial state q_0 of $\mathcal{A}$, there exists a unique sequence of steps $x'_1 = (q_0, \to, q'_1)$, $x'_2 = (q'_1, \curvearrowright, X'_2, q'_2)$, $x'_3 = (q'_2, \subsetarrow, q'_3)$, $x'_4 = (q'_3, \curvearrowright, X'_4, q'_4)$, $\ldots$, $x'_i = (q'_{i-1}, \subsetarrow, q'_i)$, $x'_{i+1} = (q'_i, \to, X'_{i+1}, q)$ 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 $\mathsf{dom}(C_1) = FG$ 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 $\mathsf{dom}(C_2) = FG^\omega$ 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 $x'_1x'_2\cdots x'_ix'_{i+1}xxx\cdots$. 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 $FG^\omega \cap \mathsf{dom}(\mathcal{A}) = \emptyset$ and we set $C_{FG^\omega} = \bot$. Now, if $X \in \mathcal{F}$ we have $FG^\omega \subseteq \mathsf{dom}(\mathcal{A})$ and we set

$$C_{FG^\omega} = C_2 \odot \big((FG \,?\, \varepsilon : \bot) \boxdot C^p_G(x)^\omega\big)\,.$$

We have $\mathsf{dom}(C_{FG^\omega}) = FG^\omega$ 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

$$[\![C_{FG^\omega}]\!](w) = [\![C_1]\!](u_1u_2)[\![C^p_G(x)]\!](u_3)[\![C^p_G(x)]\!](u_4)\cdots\,.$$

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 $x_2 = (q_1, \subsetarrow, X_2, q_2)$, $x_3 = (q_2, \curvearrowright, X_3, q_3)$, $\ldots$, $x_j = (q_{j-1}, \subsetarrow, X_j, q_j)$, $x_{j+1} = (q_j, \to, X_{j+1}, q)$ 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 $x'_1x'_2\cdots x'_ix'_{i+1}(x_1x_2x_3\cdots x_jx_{j+1})^\omega$. 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 $X \in \mathcal{F}$ is a Muller set. Therefore, if $X \notin \mathcal{F}$ we have $FG^\omega \cap \mathsf{dom}(\mathcal{A}) = \emptyset$ and we set $C_{FG^\omega} = \bot$. Now, if $X \in \mathcal{F}$ we have $FG^\omega \subseteq \mathsf{dom}(\mathcal{A})$ and we set

$$\begin{aligned} C_3 = {} & \big((G \,?\, \varepsilon : \bot) \boxdot C^p_G(x_1)\big) \odot \big(C^p_G(x_2) \boxdot C^p_G(x_3)\big) \odot \cdots \odot \\ & \big(C^p_G(x_j) \boxdot C^p_G(x_{j+1})\big) \\ C_{FG^\omega} = {} & C_2 \odot \big((F \,?\, \varepsilon : \bot) \boxdot [G, C_3]^{2\omega}\big)\,. \end{aligned}$$

We have $\mathsf{dom}(C_{FG^\omega}) = FG^\omega$ 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 $\widetilde{\vdash w}$. We deduce that $[\![C_{FG^\omega}]\!](w) = [\![\widetilde{\mathcal{A}}]\!](\widetilde{\vdash w}) = [\![\mathcal{A}]\!](w)$ as desired. □

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

Proof of Theorem 21 (2). We use the notations of the previous sections, in particular for the ω-$2\mathsf{DMT}_{\mathsf{la}}$ $\mathcal{A}$, the BDBA $\mathcal{B}$. We apply Theorem 28 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 $\mathsf{Tr}(G_k) = \{\sigma_{G_k}\}$. For each $1 \leq k \leq m$, let $C_k = C_{F_kG_k^\omega}$ be the ω-RTE given by Lemma 31. We define the final ω-RTE as

$$C = F_1G_1^\omega \,?\, C_1 : (F_2G_2^\omega \,?\, C_2 : \cdots (F_{m-1}G_{m-1}^\omega \,?\, C_{m-1} : C_m)) .$$

From Lemma 31, we can easily check that $\mathsf{dom}(C) = \mathsf{dom}(\mathcal{A})$ and $[\![C]\!](w) = [\![\mathcal{A}]\!](w)$ for all $w \in \mathsf{dom}(C)$. □

4. 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 succinct 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. The minimality of our combinators in achieving expressive completeness, as well as computing complexity measures for the conversion between RTEs and two-way transducers are open.

[1] Rajeev Alur and Pavol Černý. Expressiveness of streaming string transducers. In Kamal Lodaya and Meena Mahajan, editors, *30th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2010)*, volume 8 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 1–12. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2010.

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

[3] Rajeev Alur, Loris D'Antoni, and Mukund Raghothaman. DReX: A declarative language for efficiently evaluating regular string transformations. In Sriram K. Rajamani and David Walker, editors, *42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages - POPL'15*, pages 125–137. ACM Press, 2015.

[4] Rajeev Alur, Emmanuel Filiot, and Ashutosh Trivedi. Regular transformations of infinite strings. In *Proceedings of the 27th Annual IEEE Symposium on Logic in Computer Science, LICS 2012, Dubrovnik, Croatia, June 25-28, 2012*, pages 65–74. IEEE Computer Society, 2012.

[5] Rajeev Alur, Adam Freilich, and Mukund Raghothaman. Regular combinators for string transformations. In Thomas A. Henzinger and Dale Miller, editors, *Joint Meeting of the 23rd EACSL Annual Conference on Computer Science Logic (CSL) and the 29th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS '14, Vienna, Austria, July 14 - 18, 2014*, pages 9:1–9:10. ACM, 2014.

[6] Nicolas Baudru and Pierre-Alain Reynier. From two-way transducers to regular function expressions. In Mizuho Hoshi and Shinnosuke Seki, editors, *22nd International Conference on Developments in Language Theory, DLT 2018*, volume 11088 of *Lecture Notes in Computer Science*, pages 96–108. Springer, 2018.

[7] Jean-Camille Birget. Concatenation of inputs in a two-way automaton. *Theoretical Computer Science*, 63(2):141–156, 1989.

[8] Mikołaj Bojańczyk, Laure Daviaud, Bruno Guillon, and Vincent Penelle. Which classes of origin graphs are generated by transducers. In *44th International Colloquium on Automata, Languages, and Programming (ICALP'17)*, volume 80 of *LIPIcs*, pages 114:1–114:13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.

[9] Olivier Carton and Luc Dartois. Aperiodic two-way transducers and fo-transductions. In *Proceedings of the 24th EACSL Annual Conference on Computer Science Logic (CSL'15)*, volume 41 of *LIPIcs*, pages 160–174. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015.

[10] Olivier Carton and Max Michel. Unambiguous Büchi automata. *Theoretical Computer Science*, 297(1-3):37–81, Mar 2003.

[11] Michal P. Chytil and Vojtěch Jákl. Serial composition of 2-way finite-state transducers and simple programs on strings. In Arto Salomaa and Magnus Steinby, editors, *4th International Colloquium on Automata, Languages and Programming (ICALP'77)*, pages 135–147. Springer Berlin Heidelberg, 1977.

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

[13] Thomas Colcombet. The factorisation forest theorem. To appear in Handbook "Automata: from Mathematics to Applications", 2013.

[14] Bruno Courcelle. The expression of graph properties and graph transformations in monadic second-order logic. In Grzegorz Rozenberg, editor, *Handbook of Graph Grammars and Computing by Graph Transformations, Volume 1: Foundations*, pages 313–400. World Scientific, 1997.

[15] Vrunda Dave, Paul Gastin, and ShankaraNarayanan Krishna. Regular Transducer Expressions for Regular Transformations. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic In Computer Science (LICS'18)*, pages 315–324, Oxford, UK, July 2018. ACM Press.

[16] Vrunda Dave, Paul Gastin, and ShankaraNarayanan Krishna. Regular transducer expressions for regular transformations. *CoRR*, abs/1802.02094, 2018.

[17] Vrunda Dave, Shankara Narayanan Krishna, and Ashutosh Trivedi. FO-definable transformations of infinite strings. In Akash Lal, S. Akshay, Saket Saurabh, and Sandeep Sen, editors, *36th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2016)*, volume 65 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 12:1–12:14. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016.

[18] Manfred Droste, Werner Kuich, and Heiko Vogler. *Handbook of Weighted Automata*. Springer Publishing Company, 1st edition, 2009.

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

[20] Emmanuel Filiot, Shankara Narayanan Krishna, and Ashutosh Trivedi. First-order Definable String Transformations. In Venkatesh Raman and S. P. Suresh, editors, *34th International Conference on Foundation of Software Technology and Theoretical Computer Science (FSTTCS 2014)*, volume 29 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 147–159. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2014.

[21] Paul Gastin and ShankaraNarayanan Krishna. Unambiguous forest factorization. *CoRR*, abs/1810.07285, 2018.

[22] J. C. Shepherdson. The reduction of two-way automata to one-way automata. *IBM Journal of Research and Development*, 3(2):198–200, 1959.

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

[24] Thomas Wilke. Backward deterministic Büchi automata on infinite words. In *37th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'17)*, volume 93 of *LIPIcs*, pages 6:1–6:9. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.

Appendix A. 2DFT to RTE: A practical example

We show in this section how one computes an RTE equivalent to the 2DFT $\mathcal{A}$ of Figure 2.

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 $\{\rightleftarrows, \subsetarrow, \rightarrow, \leftarrow\}$. Given a word $w \in \Sigma^*$, a triple $(p, \rightleftarrows, q) \in \mathsf{Tr}(w)$ 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 $\subsetarrow$ (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 $w \in \mathsf{dom}(\mathcal{A})$ 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 $\mathcal{A}$ of Figure 2 we have $w \in \mathsf{dom}(\mathcal{A})$ 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., $\mathsf{dom}(\bot) = \emptyset$.
3. Consider the language $L = (ba^+)^+b \subseteq \mathsf{dom}(\mathcal{A})$. 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 A.13 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: $(q_3, \subsetarrow, q_2) \in \mathsf{Tr}(babab)$. 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 $\mathsf{Tr}(a)^2$ is an idempotent, hence $\mathsf{Tr}(a^+) = \mathsf{Tr}(a)$. We deduce also $\mathsf{Tr}(ba^+b) = \mathsf{Tr}(bab)$[3]. Finally, we have $\mathsf{Tr}((ba^+)^n b) = \mathsf{Tr}(babab)$[4] 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 since $\mathsf{Tr}(a)$ is an idempotent, $(ba^+)^+ba^+b$ is not good, the reason being that $\mathsf{Tr}(ba^+)$ is not an idempotent. We can check that $\mathsf{Tr}(ba^+ba^+)$[5] is still not idempotent, while $\mathsf{Tr}((ba^+)^i) = \mathsf{Tr}((ba^+)^3)$ for all $i \geq 3$, (see Figure A.13: 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[6]. Thus, to compute the RTE for $L = (ba^+)^+b$, we consider the RTEs corresponding to the

[2] $\mathsf{Tr}(a) = \{(q_1, \rightarrow, q_1), (q_1, \subsetarrow, q_1), (q_2, \rightarrow, q_3), (q_2, \subsetarrow, q_3), (q_3, \rightarrow, q_3), (q_3, \subsetarrow, q_3), (q_4, \leftarrow, q_4), (q_4, \rightleftarrows, q_4), (q_5, \leftarrow, q_5), (q_5, \rightleftarrows, q_5), (q_6, \rightarrow, q_6), (q_6, \subsetarrow, q_6)\}$

[3] $\mathsf{Tr}(ba^+b) = \{(q_0, \rightarrow, q_2), (q_0, \subsetarrow, q_1), (q_1, \rightleftarrows, q_5), (q_1, \subsetarrow, q_2), (q_2, \rightleftarrows, q_4), (q_2, \leftarrow, q_5), (q_3, \rightleftarrows, q_4), (q_3, \leftarrow, q_5), (q_4, \rightleftarrows, q_5), (q_4, \subsetarrow, q_1), (q_5, \rightarrow, q_1), (q_5, \subsetarrow, q_6), (q_6, \rightarrow, q_2), (q_6, \subsetarrow, q_1)\}$

[4] $\mathsf{Tr}(ba^+ba^+b) = \{(q_0, \rightarrow, q_2), (q_0, \subsetarrow, q_1), (q_1, \rightleftarrows, q_5), (q_1, \subsetarrow, q_2), (q_2, \rightleftarrows, q_4), (q_2, \subsetarrow, q_2), (q_3, \rightleftarrows, q_4), (q_3, \subsetarrow, q_2), (q_4, \rightleftarrows, q_5), (q_4, \subsetarrow, q_1), (q_5, \rightarrow, q_2), (q_5, \subsetarrow, q_6), (q_6, \rightarrow, q_2), (q_6, \subsetarrow, q_1)\}$

[5] $\mathsf{Tr}(ba^+ba^+) = \{(q_0, \rightarrow, q_3), (q_1, \rightleftarrows, q_5), (q_1, \subsetarrow, q_1), (q_2, \rightleftarrows, q_4), (q_2, \subsetarrow, q_3), (q_3, \rightleftarrows, q_4), (q_3, \subsetarrow, q_3), (q_4, \rightleftarrows, q_5), (q_4, \subsetarrow, q_1), (q_5, \rightarrow, q_1), (q_5, \subsetarrow, q_6), (q_6, \rightarrow, q_3), (q_6, \subsetarrow, q_6)\}$

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

"good" regular expressions $E_1 = ba^+b$, $E_2 = ba^+ba^+b$, $E_3 = [(ba^+)^3]^+b$, $E_4 = [(ba^+)^3]^+ba^+b$ and $E_5 = [(ba^+)^3]^+ba^+ba^+b$.

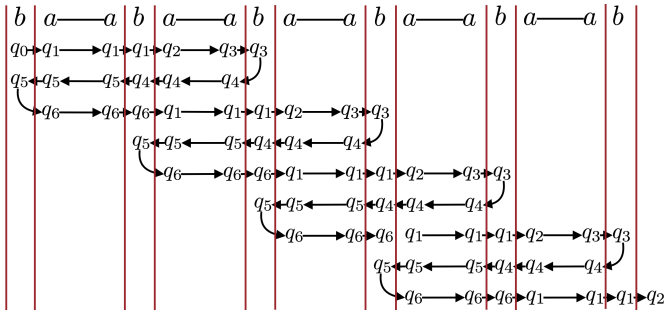


Figure A.13: Run of $\mathcal{A}$ on an input word in $(ba^+)^+b$.

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, $C_b(q_1, \rightarrow, q_2) = (b \,?\, \varepsilon : \bot) = C_b(q_3, \circlearrowright, q_4)$. For $u, v \in \Sigma^*$, we introduce the macro $u/v = u \,?\, v : \bot$. We have $\mathsf{dom}(u/v) = \{u\}$ 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, we have $C_{a^+}(q_5, \leftarrow, q_5) = (C_a(q_5, \leftarrow, q_5))^{\overleftarrow{\boxplus}} = (a/b)^{\overleftarrow{\boxplus}}$. 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, we have $C_{a^+}(q_2, \rightarrow, q_3) = a \,?\, C_a(q_2, \rightarrow, q_3) : (C_a(q_2, \rightarrow, q_3) \boxdot (C_a(q_3, \rightarrow, q_3))^{\boxplus}) = a \,?\, (a/\varepsilon) : \big((a/\varepsilon) \boxdot (a/\varepsilon)^{\boxplus}\big)$, 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 A.13)

$$\begin{aligned} C_E(x) &= C_b(q_0, \rightarrow, q_1) \boxdot C_{a^+}(q_1, \rightarrow, q_1) \boxdot C_b(q_1, \rightarrow, q_2) \boxdot C_{a^+}(q_2, \rightarrow, q_3) \\ &= (b/\varepsilon) \boxdot (a/\varepsilon)^{\boxplus} \boxdot (b/\varepsilon) \boxdot (a/\varepsilon)^{\boxplus} \\ &\approx (ba^+ba^+ \,?\, \varepsilon : \bot)\,. \end{aligned}$$

More interesting is the step $y = (q_4, \subsetneq, q_1)$ 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 A.13). Therefore, we have

$$\begin{aligned}
C_E(y) &= \big((b/\varepsilon) \overleftarrow{\boxdot} C_{a^+}(q_5,\leftarrow,q_5) \overleftarrow{\boxdot} C_b(q_4,\leftarrow,q_5) \overleftarrow{\boxdot} C_{a^+}(q_4,\leftarrow,q_4)\big) \odot \\
&\qquad \big(C_b(q_5,\rightarrow,q_6) \boxdot C_{a^+}(q_6,\rightarrow,q_6) \boxdot C_b(q_6,\rightarrow,q_1) \boxdot C_{a^+}(q_1,\rightarrow,q_1)\big) \\
&= \big((b/\varepsilon) \overleftarrow{\boxdot} (a/b)^{\overleftarrow{\boxplus}} \overleftarrow{\boxdot} (b/\varepsilon) \overleftarrow{\boxdot} (a/a)^{\overleftarrow{\boxplus}}\big) \odot \\
&\qquad \big((b/\varepsilon) \boxdot (a/\varepsilon)^{\boxplus} \boxdot (b/\varepsilon) \boxdot (a/\varepsilon)^{\boxplus}\big) \\
&\approx (b/\varepsilon) \overleftarrow{\boxdot} (a/b)^{\overleftarrow{\boxplus}} \overleftarrow{\boxdot} (b/\varepsilon) \overleftarrow{\boxdot} (a/a)^{\overleftarrow{\boxplus}}.
\end{aligned}$$

The leftmost (b/ε) in the first line is used to make sure that the input word belongs to $E = ba^+ba^+$. Composing these steps on the right with b, we obtain the RTE $C_2 = C_{E_2}(q_0,\rightarrow,q_2)$ which describes the behaviour of $\mathcal{A}$ on the subset $E_2 = ba^+ba^+b \subseteq \mathsf{dom}(\mathcal{A})$:

$$\begin{aligned}
C_2 &= \big(C_E(x) \boxdot C_b(q_3,\curvearrowright,q_4)\big) \odot \big(C_E(y) \boxdot C_b(q_1,\rightarrow,q_2)\big) \\
&= \big(C_E(x) \boxdot (b/\varepsilon)\big) \odot \big(C_E(y) \boxdot (b/\varepsilon)\big) \\
&\approx \big((b/\varepsilon) \overleftarrow{\boxdot} (a/b)^{\overleftarrow{\boxplus}} \overleftarrow{\boxdot} (b/\varepsilon) \overleftarrow{\boxdot} (a/a)^{\overleftarrow{\boxplus}}\big) \boxdot (b/\varepsilon).
\end{aligned}$$

Therefore, $[\![C_2]\!](ba^{m_1}ba^{m_2}b) = a^{m_2}b^{m_1} = [\![\mathcal{A}]\!](ba^{m_1}ba^{m_2}b)$.
The computation of the RTE $C_{E_3}(q_0,\rightarrow,q_2)$ for $E_3 = [(ba^+)^3]^+b \subseteq \mathsf{dom}(\mathcal{A})$ is shown below. This involves the use of the 2-chained Kleene-plus.

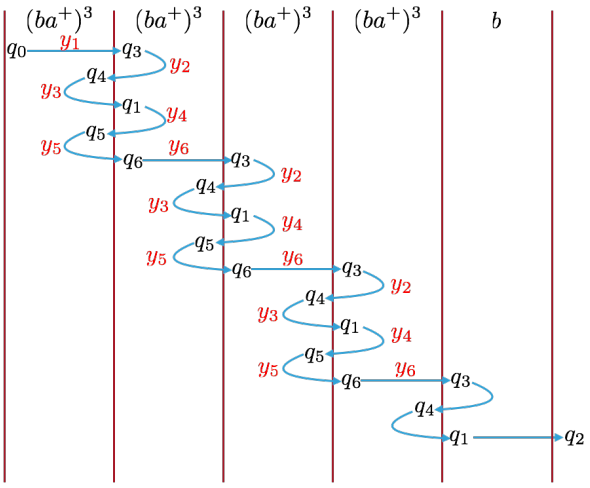


Figure A.14: run of a word in $E_3 = [(ba^+)^3]^+b$

We want to compute the RTE for the step $(q_0,\rightarrow,q_2)$ on a word $u \in E_3$. It can be decomposed as shown in Figure A.14. Unlike the case of E_2, we have to use the 2-chained Kleene plus. Let $F = (ba^+)^3$ so that $E_3 = F^+b$. We have (see Figure A.14),

$$\begin{aligned}
C_{E_3}(q_0,\rightarrow,q_2) = (C_{F^+}(q_0,\rightarrow,q_3) \boxdot C_b(q_3,\curvearrowright,q_4)) \odot \\
(C_{F^+}(q_4,\curvearrowleft,q_1) \boxdot C_b(q_1,\rightarrow,q_2)).
\end{aligned}$$

We know that $C_b(q_3,\curvearrowright,q_4) = (b/\varepsilon) = C_b(q_1,\rightarrow,q_2)$ hence it remains to compute $C_{F^+}(q_0,\rightarrow,q_3)$ and $C_{F^+}(q_4,\curvearrowleft,q_1)$. First we define RTEs associated

with atomic expressions and steps which are going to be used in constructing $C_{E_3}(q_0, \rightarrow, q_2)$. They are $C_b(q_3, \curvearrowright, q_4) = C_b(q_6, \rightarrow, q_1) = C_b(q_1, \rightarrow, q_2) = C_b(q_5, \rightarrow, q_6) = (b/\varepsilon)$ and $C_{a^+}(q_2, \rightarrow, q_3) = C_{a^+}(q_1, \rightarrow, q_1) = (a/\varepsilon)^{\boxplus}$, $C_{a^+}(q_4, \leftarrow, q_4) = (a/a)^{\overleftarrow{\boxplus}}$, $C_{a^+}(q_5, \leftarrow, q_5) = (a/b)^{\overleftarrow{\boxplus}}$. We compute RTE $C_F(x)$ for the relevant steps x in the monoid element $X = \mathsf{Tr}(F)$. F is an unambiguous catenation of $E_2 = ba^+ba^+b$ with a^+ and from Figure A.13, it can be seen that:

(a) For $y_1 = (q_0, \rightarrow, q_3)$

$$\begin{aligned} C_F(y_1) &= C_{E_2}(q_0, \rightarrow, q_2) \boxdot C_{a^+}(q_2, \rightarrow, q_3) \\ &= \big((b/\varepsilon) \overleftarrow{\boxdot} (a/b)^{\overleftarrow{\boxplus}} \overleftarrow{\boxdot} (b/\varepsilon) \overleftarrow{\boxdot} (a/a)^{\overleftarrow{\boxplus}}\big) \boxdot (b/\varepsilon) \boxdot (a/\varepsilon)^{\boxplus} \end{aligned}$$

where $C_{E_2}(q_0, \rightarrow, q_2)$ has been computed in Section 1.
For example, $[\![C_F(y_1)]\!](ba^{m_1}ba^{m_2}ba^{m_3}) = a^{m_2}b^{m_1}$.

(b) Continuing with the computation for $(ba^+)^3$ as in Figure A.14, for $y_2 = (q_3, \curvearrowright, q_4)$, we take the Cauchy product of $C_b(q_3, \curvearrowright, q_4)$ with $(a^+ba^+ba^+ \,?\, \varepsilon : \bot)$.

$$C_F(y_2) = C_b(q_3, \curvearrowright, q_4) \boxdot (a^+ba^+ba^+ \,?\, \varepsilon : \bot) \approx ((ba^+)^3 \,?\, \varepsilon : \bot)$$

$[\![C_F(y_2)]\!](ba^{m_1}ba^{m_2}ba^{m_3}) = \varepsilon$.

(c) For $y_3 = (q_4, \subsetarrow, q_1)$, we have

$$\begin{aligned} C_F(y_3) &= (ba^+ \,?\, \varepsilon : \bot) \overleftarrow{\boxdot} C_{ba^+ba^+}(y_3) \\ &= (ba^+ \,?\, \varepsilon : \bot) \overleftarrow{\boxdot} \big((b/\varepsilon) \overleftarrow{\boxdot} (a/b)^{\overleftarrow{\boxplus}} \overleftarrow{\boxdot} (b/\varepsilon) \overleftarrow{\boxdot} (a/a)^{\overleftarrow{\boxplus}}\big) \\ &\approx (ba^+b \,?\, \varepsilon : \bot) \boxdot \big((a/b)^{\overleftarrow{\boxplus}} \overleftarrow{\boxdot} (b/\varepsilon) \overleftarrow{\boxdot} (a/a)^{\overleftarrow{\boxplus}}\big) \end{aligned}$$

where $C_{ba^+ba^+}(y_3)$ is already computed in Section 1.
As an example, $[\![C_F(y_3)]\!](ba^{m_1}ba^{m_2}ba^{m_3}) = a^{m_3}b^{m_2}$.

(d) For $y_4 = (q_1, \curvearrowright, q_5)$, it is similar to the $C_E(y)$ computed for C_{E_2} in Section 1. Here we have

$$\begin{aligned} C_F(y_4) &= C_{ba^+b}(y_4) \boxdot (a^+ba^+ \,?\, \varepsilon : \bot) \\ &= \big((C_b(q_1, \rightarrow, q_2) \boxdot C_{a^+}(q_2, \rightarrow, q_3) \boxdot (b/\varepsilon)) \odot \\ &\quad (C_b(q_4, \leftarrow, q_5) \overleftarrow{\boxdot} C_{a^+}(q_4, \leftarrow, q_4) \overleftarrow{\boxdot} C_b(q_3, \curvearrowright, q_4))\big) \boxdot \\ &\quad (a^+ba^+ \,?\, \varepsilon : \bot) \\ &= (((b/\varepsilon) \boxdot (a/\varepsilon)^{\boxplus} \boxdot (b/\varepsilon)) \odot ((b/\varepsilon) \overleftarrow{\boxdot} (a/a)^{\overleftarrow{\boxplus}} \overleftarrow{\boxdot} (b/\varepsilon))) \boxdot \\ &\quad (a^+ba^+ \,?\, \varepsilon : \bot) \\ &\approx \big((b/\varepsilon) \overleftarrow{\boxdot} (a/a)^{\overleftarrow{\boxplus}} \overleftarrow{\boxdot} (b/\varepsilon)\big) \boxdot (a^+ba^+ \,?\, \varepsilon : \bot) \end{aligned}$$

As an example, $[\![C_F(y_4)]\!](ba^{m_1}ba^{m_2}ba^{m_3}) = a^{m_1}$.

(e) For $y_5 = (q_5, \subsetarrow, q_6)$, in the computation of $C_F(y_5)$ we need $C_{ba^+}(y_5)$. Thus, we compute $C_{ba^+}(y_5)$ below whose computation is similar to

$C_E(y)$ computed above.

$$C_{ba^+}(y_5) = ((b/\varepsilon) \overleftarrow{\boxdot} C_{a^+}(q_5, \leftarrow, q_5)) \odot (C_b(q_5, \rightarrow, q_6) \boxdot C_{a^+}(q_6, \rightarrow, q_6))$$
$$= ((b/\varepsilon) \overleftarrow{\boxdot} (a/b)^{\overleftarrow{\boxplus}}) \odot ((b/\varepsilon) \boxdot (a/\varepsilon)^{\boxplus}) \approx (b/\varepsilon) \overleftarrow{\boxdot} (a/b)^{\overleftarrow{\boxplus}}$$

We can compute $C_F(y_5)$ as

$$C_F(y_5) = (ba^+ba^+ \,?\, \varepsilon : \bot) \boxdot C_{ba^+}(y_5) \approx (ba^+ba^+b \,?\, \varepsilon : \bot) \overleftarrow{\boxdot} (a/b)^{\overleftarrow{\boxplus}}$$

As an example, $[\![C_F(y_5)]\!](ba^{m_1}ba^{m_2}ba^{m_3}) = b^{m_3}$.

(f) For $y_6 = (q_6, \rightarrow, q_3)$, the computation of $C_F(y_6)$ is similar to that of $C_{ba^+ba^+}(q_0, \rightarrow, q_2)$ computed above. We need $C_{ba^+ba^+}(q_6, \rightarrow, q_3)$ and $C_{ba^+ba^+b}(q_6, \rightarrow, q_2)$. We see the computation of these below.

$$\begin{aligned}
C_{ba^+ba^+}(q_6, \rightarrow, q_3) &= C_b(q_6, \rightarrow, q_1) \boxdot C_{a^+}(q_1, \rightarrow, q_1) \boxdot \\
&\quad C_b(q_1, \rightarrow, q_2) \boxdot C_{a^+}(q_2, \rightarrow, q_3) \\
&= (b/\varepsilon) \boxdot (a/\varepsilon)^{\boxplus} \boxdot (b/\varepsilon) \boxdot (a/\varepsilon)^{\boxplus} \\
&\approx (ba^+ba^+ \,?\, \varepsilon : \bot) \\
C_{ba^+ba^+b}(q_6, \rightarrow, q_2) &= (C_{ba^+ba^+}(q_6, \rightarrow, q_3) \boxdot C_b(q_3, \curvearrowright, q_4)) \odot \\
&\quad (C_{ba^+ba^+}(q_4, \curvearrowleft, q_1) \boxdot C_b(q_1, \rightarrow, q_2)) \\
&\approx (ba^+ba^+b \,?\, \varepsilon : \bot) \odot \\
&\quad (((b/\varepsilon) \overleftarrow{\boxdot} (a/b)^{\overleftarrow{\boxplus}} \overleftarrow{\boxdot} (b/\varepsilon) \overleftarrow{\boxdot} (a/a)^{\overleftarrow{\boxplus}}) \boxdot (b/\varepsilon)) \\
&\approx \left((b/\varepsilon) \overleftarrow{\boxdot} (a/b)^{\overleftarrow{\boxplus}} \overleftarrow{\boxdot} (b/\varepsilon) \overleftarrow{\boxdot} (a/a)^{\overleftarrow{\boxplus}}\right) \boxdot (b/\varepsilon)
\end{aligned}$$

Note that $C_{ba^+ba^+}(q_4, \curvearrowleft, q_1)$ has been computed in Section 1. Now we concatenate with $C_{a^+}(q_2, \rightarrow, q_3)$ needed in the computation.

$$\begin{aligned}
C_F(y_6) &= C_{ba^+ba^+b}(q_6, \rightarrow, q_2) \boxdot C_{a^+}(q_2, \rightarrow, q_3) \\
&= \left((b/\varepsilon) \overleftarrow{\boxdot} (a/b)^{\overleftarrow{\boxplus}} \overleftarrow{\boxdot} (b/\varepsilon) \overleftarrow{\boxdot} (a/a)^{\overleftarrow{\boxplus}}\right) \boxdot (b/\varepsilon) \boxdot (a/\varepsilon^{\boxplus}) \\
&\approx \left((b/\varepsilon) \overleftarrow{\boxdot} (a/b)^{\overleftarrow{\boxplus}} \overleftarrow{\boxdot} (b/\varepsilon) \overleftarrow{\boxdot} (a/a)^{\overleftarrow{\boxplus}}\right) \boxdot (ba^+ \,?\, \varepsilon : \bot)
\end{aligned}$$

As an example, $[\![C_F(y_6)]\!](ba^{m_1}ba^{m_2}ba^{m_3}) = a^{m_2}b^{m_1}$.

Now we are in a position to compute RTE $C_{F^+}(q_0, \rightarrow, q_3)$. As shown in figureA.14, it is a concatenation of step y_1 and then steps y_2, y_3, y_4, y_5 and y_6 repetitively. Consecutive pairs of $(ba^+)^3$ are needed to compute the RTE and thanks to the 2-chained Kleene plus, we can define the RTE for the same.

$$C_{F^+}(y_1) = (C_F(y_1) \boxdot (F^* \,?\, \varepsilon : \bot)) \odot [F, C']^{2\boxplus}$$
$$C' = ((F \,?\, \varepsilon : \bot) \boxdot C_F(y_2)) \odot (C_F(y_3) \boxdot C_F(y_4)) \odot (C_F(y_5) \boxdot C_F(y_6))$$

As an example,
$[\![C_{F^+}(y_1)]\!](ba^{m_1}ba^{m_2}ba^{m_3}ba^{m_4}ba^{m_5}ba^{m_6}) = a^{m_2}b^{m_1}a^{m_3}b^{m_2}a^{m_4}b^{m_3}a^{m_5}b^{m_4}$.
Finally, we compute RTE for $y = (q_0, \rightarrow, q_2)$ for the expression $E_3 = [(ba^+)^3]^+b$ by concatenating b with the above RTE.

$$C_{E_3}(y) = (C_{F^+}(q_0, \rightarrow, q_3) \boxdot C_b(q_3, \curvearrowright, q_4)) \odot (C_{F^+}(q_4, \curvearrowleft, q_1) \boxdot C_b(q_1, \rightarrow, q_2))$$

Notice that $C_{F^+}(q_4, \subsetneq, q_1) = (F^* \,?\, \varepsilon : \bot) \boxdot C_F(y_3)$.
We have already seen that $C_{E_3}(y)$ computes the output produced by a successful run on a word $w \in E_3$. Applying the RTE as above, we have, for example,

$$[\![C_{E_3}(y)]\!](ba^{m_1}ba^{m_2}ba^{m_3}ba^{m_4}ba^{m_5}ba^{m_6}b) = a^{m_2}b^{m_1}a^{m_3}b^{m_2}a^{m_4}b^{m_3}a^{m_5}b^{m_4}a^{m_6}b^{m_5}$$