



HAL
open science

Regular Transducer Expressions for Regular Transformations

Vrunda Dave, Paul Gastin, Shankara Narayanan

► **To cite this version:**

Vrunda Dave, Paul Gastin, Shankara Narayanan. Regular Transducer Expressions for Regular Transformations. *Information and Computation*, 2022, 282, pp.104655. 10.1016/j.ic.2020.104655. hal-03709884

HAL Id: hal-03709884

<https://hal.science/hal-03709884>

Submitted on 30 Jun 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Regular Transducer Expressions for Regular Transformations¹

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

¹This work has been partially supported by IRL RELAX

9 fundamental theory to regular transformations [19], where, it was shown that
10 regular transformations are those which are captured by two-way transducers and
11 by MSO transductions a la Courcelle. Much later, streaming string transducers
12 (SSTs) were introduced [1] as a model which makes a single pass through the
13 input string and uses a finite set of variables that range over strings from the
14 output alphabet. In [1], the equivalence between SSTs and MSO transductions
15 was established, thereby showing that regular transformations are those which
16 are captured by either SSTs, two-way transducers or MSO transductions. This
17 theory was further extended to work for infinite string transformations [4]; the
18 restriction from MSO transductions to first-order definable transductions, and
19 their equivalence with aperiodic SSTs and aperiodic two-way transducers has also
20 been established over finite and infinite strings [20], [17]. Other generalizations
21 such as [2], extend this theory to trees. More recently, this equivalence between
22 SSTs and logical transductions is also shown to hold good even when one works
23 with the origin semantics [8].

24 Moving on, a natural problem pertains to the characterization of the output
25 computed by two-way transducers or SSTs (over finite and infinite words) using
26 regular-like expressions. For the strictly lesser expressive case of sequential
27 one-way transducers, this regex characterization of the output is obtained as
28 a special case of Schützenberger’s famous equivalence [18] between weighted
29 automata and regular weighted expressions. The question is much harder when
30 one looks at two-way transducers, due to the fact that the output is generated
31 in a one-way fashion, while the input is read in a two-way manner. Recently,
32 [5] proposed a set of combinators, analogous to the operators used in forming
33 regular expressions, to form *combinator expressions* and proved their equivalence
34 with SSTs.

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

38 Over finite words, we work with two-way deterministic transducers (denoted
39 2DFT) while over infinite words, the model considered is deterministic two-
40 way transducers with regular look-ahead, equipped with the Muller acceptance
41 condition. Figure 1 gives an ω -2DMT_{la} (la stands for look-ahead and M in the
42 2DMT for Muller acceptance).

43 In both cases of finite words and infinite words, we come up with a set
44 of combinators which we use to form *regular transducer expressions* (RTE)
45 characterizing two-way transducer (2DFT/ ω -2DMT_{la}).

46 **The Combinators.** We describe our basic combinators that form the building
47 blocks of RTEs. The semantics of an RTE is a partial function $f: \Sigma^\infty \rightarrow \Gamma^\infty$
48 whose domain is denoted $\text{dom}(f)$.

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

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

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

55 The unambiguous Cauchy product $f \boxplus g$ when applied on $w \in \Sigma^\infty$ produces
56 $f(u) \cdot g(v)$ if $w = u \cdot v$ is an unambiguous decomposition of w with $u \in \text{dom}(f) \cap \Sigma^*$
57 and $v \in \text{dom}(g)$.

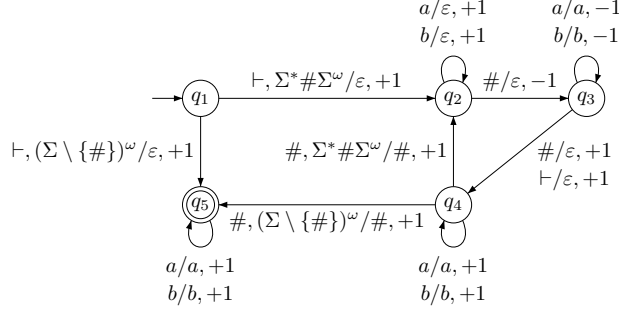


Figure 1: An ω -2DMT_{1a} \mathcal{A}' with $\llbracket \mathcal{A}' \rrbracket (u_1 \# u_2 \# \dots \# u_n \# v) = u_1^R u_1 \# u_2^R u_2 \# \dots \# u_n^R u_n \# v$ where $u_1, \dots, u_n \in (a+b)^*$, $v \in (a+b)^\omega$ and u^R denotes the reverse of u . The Muller acceptance set is $\{\{q_5\}\}$.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

58 The unambiguous Kleene-plus f^{\boxplus} applied to $w \in \Sigma^*$ produces $f(u_1) \cdots f(u_n)$
 59 if $w = u_1 \cdots u_n$ is an unambiguous factorization of w , with each $u_i \in \text{dom}(f)$.

60 The unambiguous 2-chained Kleene-plus $[K, f]^{2\boxplus}$ when applied to a string $w \in$
 61 Σ^* produces as output $f(u_1 u_2) \cdot f(u_2 u_3) \cdots f(u_{n-1} u_n)$ if w can be unambiguously
 62 written as $u_1 u_2 \cdots u_n$, with each $u_i \in K$, for the regular language K .

63 We also have the reverses $f^{\overleftarrow{\boxplus}} g$, $f^{\overleftarrow{\boxplus}}$ and $[K, f]^{2\overleftarrow{\boxplus}}$, which parse the input from
 64 left to right as before, but produce the output from right to left. For instance,
 65 with the notation above, $f^{\overleftarrow{\boxplus}}(w)$ produces $f(u_n) \cdots f(u_1)$.

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

68 Finally, the unambiguous two-chained ω -iteration produces $[K, f]^{2\omega}(w) =$
 69 $f(u_1 u_2) f(u_2 u_3) \cdots$ if $w \in \Sigma^\omega$ can be unambiguously decomposed as $w = u_1 u_2 \cdots$
 70 with $u_i \in K$ for all $i \geq 1$, where $K \subseteq \Sigma^*$ is regular.

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

$$\begin{aligned} C_1 &= a? a : (b? b : (\#? \# : \perp)) \\ C_2 &= a? a : (b? b : \perp) \\ C_3 &= a? a : (b? b : (\#? \varepsilon : \perp)) \\ C_4 &= ((a+b)^* \#)? (C_3^{\overleftarrow{\boxplus}} \odot C_1^{\boxplus}) : \perp \end{aligned}$$

71 Then $\text{dom}(C_1) = \text{dom}(C_3) = (a+b+\#)$ and $\text{dom}(C_2) = (a+b)$. Next, we
 72 see that $\text{dom}(C_4) = (a+b)^* \#$ and, for $u \in (a+b)^*$, $\llbracket C_4 \rrbracket (u\#) = u^R u \#$ where
 73 u^R denotes the reverse of u . This gives $\text{dom}(C) = ((a+b)^* \#)^+ (a+b)^\omega$ with
 74 $\llbracket C \rrbracket (u_1 \# u_2 \# \cdots \# u_n \# v) = u_1^R u_1 \# u_2^R u_2 \# \cdots \# u_n^R u_n \# v$ when $u_i \in (a+b)^*$ and
 75 $v \in (a+b)^\omega$. The RTE $C' = (a+b)^\omega ? C_2^\omega : C$ corresponds to the ω -2DMT_{1a} \mathcal{A}'
 76 in Figure 1; that is, $\llbracket C' \rrbracket = \llbracket \mathcal{A}' \rrbracket$.

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

79 **Theorem 2.**

80 (1) Given an RTE (resp. ω -RTE) we can effectively construct an equivalent
 81 2DFT (resp. an ω -2DMT_{1a}).

82 (2) Given a 2DFT (resp. an ω -2DMT_{la}) we can effectively construct an equiv-
 83 alent RTE (resp. ω -RTE).

84 The construction of an RTE starting from a two-way deterministic transducer
 85 \mathcal{A} is quite involved. It is based on the transition monoid $\text{TrM}(\mathcal{A})$ of the transducer.
 86 This is a classical notion for two-way transducers over finite words, but not
 87 for two-way transducers *with look-ahead* on infinite words (to the best of our
 88 knowledge). So we introduce the notion of transition monoid for ω -2DMT_{la}. We
 89 handle the look-ahead with a backward deterministic Büchi automaton (BDBA),
 90 also called *complete unambiguous* or *strongly unambiguous* Büchi automata
 91 [10, 24]. The translation of \mathcal{A} to an RTE is crucially guided by a “good” (ω -
 92)regular expression induced by the transition monoid of \mathcal{A} . The good (ω -)regular
 93 expression facilitates a uniform treatment of finite and infinite words. As a
 94 remark, it is not a priori clear how the result of [5] extends to infinite words
 95 using the techniques therein.

96 A regular expression F over alphabet Σ is *good* wrt. a morphism φ from Σ^*
 97 to a monoid $(S, \cdot, 1_S)$ if (i) it is unambiguous and (ii) for each subexpression
 98 E of F , the image of all strings in $L(E)$ maps to a single monoid element s_E .
 99 Note that (ii) implies that for each subexpression E^+ of F , s_E is an idempotent.
 100 These *good* expressions are obtained thanks to an unambiguous version [21]
 101 of the celebrated forest factorization theorem due to Imre Simon [23]. Good
 102 rational expressions might be useful in settings beyond two-way transducers.
 103

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

104 **Related Work.** We briefly discuss two recent papers which are closely related
 105 to this paper. As mentioned above, we generalized the result of [5] from finite to
 106 infinite words. Actually, [5] works with copyless cost register automata (CCRA)
 107 over finite words. CCRA are generalizations of SSTs and compute a partial
 108 function from finite words over a finite alphabet to values from a monoid $(\mathbb{D}, +, 0)$.
 109 SSTs correspond to CCRAs where the output monoid is the free monoid $(\Gamma^*, \cdot, \varepsilon)$
 110 for some finite output alphabet Γ . The combinators introduced in [5] form the
 111 basis for a declarative language DReX [3] over finite words, which can express
 112 all regular string-to-string transformations, and can also be efficiently evaluated.

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

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

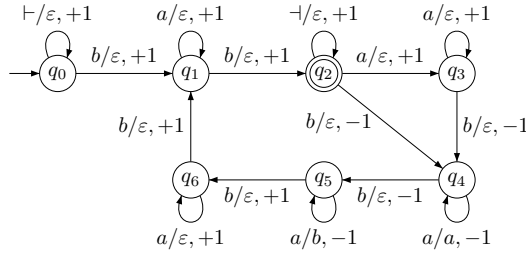


Figure 2: A 2DFT \mathcal{A} with $\llbracket \mathcal{A} \rrbracket (ba^{m_1}ba^{m_2}b \dots a^{m_k}b) = a^{m_2}b^{m_1}a^{m_3}b^{m_2} \dots a^{m_k}b^{m_{k-1}}$.

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

138 2. Finite Words

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

141 2.1. Two-way automata and transducers

142 Let Σ be a finite input alphabet and let \vdash, \dashv be two special symbols not
 143 in Σ . We assume that every input string $w \in \Sigma^*$ is presented as $\vdash w \dashv$, where
 144 \vdash, \dashv serve as left and right delimiters that appear nowhere else in w . We write
 145 $\Sigma_{\vdash \dashv} = \Sigma \cup \{\vdash, \dashv\}$. A two-way automaton $\mathcal{A} = (Q, \Sigma, \delta, I, F)$ has a finite set of
 146 states Q , subsets $I, F \subseteq Q$ of initial and final states and a transition relation
 147 $\delta \subseteq Q \times \Sigma_{\vdash \dashv} \times Q \times \{-1, 1\}$. The -1 represents the reading head moving to the
 148 left, while a 1 represents the reading head moving to the right. The reading
 149 head cannot move left when it is on \vdash . See Figure 2 for an example.

150 A configuration of \mathcal{A} is represented by w_1qw_2 where $q \in Q$ and $w_1w_2 \in \vdash\Sigma^*\dashv$.
 151 If $w_2 = \varepsilon$ the computation has come to an end. Otherwise, the reading head
 152 of \mathcal{A} is scanning the first symbol of $w_2 \neq \varepsilon$ in state q . If $w_2 = aw'_2$ and if
 153 $(q, a, q', -1) \in \delta$ (hence $a \neq \vdash$), then there is a transition from the configuration
 154 $w'_1bqaw'_2$ to $w'_1q'bw'_2$. Likewise, if $(q, a, q', 1) \in \delta$, we obtain a transition from
 155 $w_1qaw'_2$ to $w_1aq'w'_2$. A run of \mathcal{A} is a sequence of transitions; it is accepting if
 156 it starts in a configuration $p\vdash w \dashv$ with $p \in I$ and ends in a configuration $\vdash w \dashv q$
 157 with $q \in F$. The language $\mathcal{L}(\mathcal{A})$ or domain $\text{dom}(\mathcal{A})$ of \mathcal{A} is the set of all words
 158 $w \in \Sigma^*$ which have an accepting run in \mathcal{A} .

159 To extend the definition of a two-way automaton \mathcal{A} into a two-way transducer,
 160 $(Q, \Sigma, \delta, I, F)$ is extended to $(Q, \Sigma, \Gamma, \delta, I, F)$ by adding a finite output alphabet Γ
 161 and the definition of the transition relation as a *finite* subset $\delta \subseteq Q \times \Sigma_{\vdash \dashv} \times Q \times$
 162 $\Gamma^* \times \{-1, 1\}$. The output produced on each transition is appended to the right
 163 of the output produced so far. \mathcal{A} defines a relation $\llbracket \mathcal{A} \rrbracket = \{(u, w) \mid u \in \mathcal{L}(\mathcal{A})$
 164 and w is the output produced on an accepting run of $u\}$.

165 The transducer \mathcal{A} is said to be functional if for each input $u \in \text{dom}(\mathcal{A})$, at
 166 most one output w can be produced. In this case, for each $u \in \text{dom}(\mathcal{A})$, there is
 167 exactly one $w \in \Gamma^*$ such that $(u, w) \in \llbracket \mathcal{A} \rrbracket$. We also denote this by $\llbracket \mathcal{A} \rrbracket(u) = w$.
 168 We consider a special symbol $\perp \notin \Gamma$ that will stand for *undefined*. We let
 169 $\llbracket \mathcal{A} \rrbracket(u) = \perp$ when $u \notin \text{dom}(\mathcal{A})$. Thus, the semantics of a functional transducer
 170 \mathcal{A} is a map $\llbracket \mathcal{A} \rrbracket: \Sigma^* \rightarrow \mathbb{D} = \Gamma^* \cup \{\perp\}$ such that $u \in \text{dom}(\mathcal{A})$ iff $\llbracket \mathcal{A} \rrbracket(u) \neq \perp$.

171 We use non-deterministic unambiguous two-way transducers (2NUFT) in
 172 some proofs. A two-way transducer is unambiguous if each string $u \in \Sigma^*$ has
 173 at most one accepting run. Clearly, 2NUFTs are functional. A deterministic
 174 two-way transducer (2DFT) is one having a single initial state and where, from
 175 each state, on each symbol $a \in \Sigma_{\uparrow\downarrow}$, at most one transition is enabled. In that
 176 case, the transition relation is a partial function $\delta: Q \times \Sigma_{\uparrow\downarrow} \rightarrow Q \times \Gamma^* \times \{-1, 1\}$.
 177 2DFTs are by definition unambiguous. It is known [11] that 2DFTs are equivalent
 178 to 2NUFTs.

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

181 **Example 3.** Figure 2 shows a two-way transducer \mathcal{A} with $\text{dom}(\mathcal{A}) = (ba^*)^+b$,
 182 $\llbracket \mathcal{A} \rrbracket(ba^{m_1}b) = \varepsilon$ and $\llbracket \mathcal{A} \rrbracket(ba^{m_1}ba^{m_2}b \dots a^{m_k}b) = a^{m_2}b^{m_1}a^{m_3}b^{m_2} \dots a^{m_k}b^{m_{k-1}}$
 183 for $k \geq 2$ and $m_i \in \mathbb{N}$ for $1 \leq i \leq k$.

184 2.2. Regular Transducer Expressions

185 Let Σ and Γ be finite input and output alphabets. Recall that $\perp \notin \Gamma$ is
 186 a special symbol that stands for *undefined*. We define the output monoid as
 187 $\mathbb{D} = \Gamma^* \cup \{\perp\}$ with the usual concatenation on words, \perp acting as a zero:
 188 $d \cdot \perp = \perp \cdot d = \perp$ 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:

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

189 where $d \in \mathbb{D}$ ranges over output values, and $K \subseteq \Sigma^*$ ranges over regular
 190 languages of *finite words*. The semantics of an RTE C is a function $\llbracket C \rrbracket: \Sigma^* \rightarrow \mathbb{D}$
 191 defined inductively following the syntax of the expression, starting from constant
 192 functions. Since \perp stands for *undefined*, we define the *domain* of a function
 193 $f: \Sigma^* \rightarrow \mathbb{D}$ by $\text{dom}(f) = f^{-1}(\mathbb{D} \setminus \{\perp\}) = \Sigma^* \setminus f^{-1}(\perp)$.

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

196 We have $\text{dom}(\llbracket d \rrbracket) = \Sigma^*$ if $d \neq \perp$ and $\text{dom}(\llbracket \perp \rrbracket) = \emptyset$.

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

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

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

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

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

205 **Unambiguous Cauchy product and its reverse.** If w admits a unique factorization
 206 $w = u \cdot v$ with $u \in \text{dom}(f)$ and $v \in \text{dom}(g)$ then we set
 207 $(f \boxdot g)(w) = f(u) \cdot g(v)$ and $(f \overleftarrow{\boxdot} g)(w) = g(v) \cdot f(u)$. Otherwise, we
 208 set $(f \boxdot g)(w) = \perp = (f \overleftarrow{\boxdot} g)(w)$.

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

211 **Unambiguous Kleene-plus and its reverse.** If w admits a unique factorization
 212 $w = u_1 \cdot u_2 \cdots u_n$ with $n \geq 1$ and $u_i \in \text{dom}(f)$ for all $1 \leq i \leq n$ then
 213 we set $f^{\boxplus}(w) = f(u_1) \cdot f(u_2) \cdots f(u_n)$ and $f^{\overleftarrow{\boxplus}}(w) = f(u_n) \cdots f(u_2) \cdot f(u_1)$.
 214 Otherwise, we set $f^{\boxplus}(w) = \perp = f^{\overleftarrow{\boxplus}}(w)$.

215 We have $\text{dom}(f^{\boxplus}) = \text{dom}(f^{\overleftarrow{\boxplus}}) \subseteq \text{dom}(f)^+$ and the inclusion is strict
 216 if the Kleene iteration $\text{dom}(f)^+$ of $\text{dom}(f)$ is ambiguous. Notice that
 217 $\text{dom}(f^{\boxplus}) = \emptyset$ when $\varepsilon \in \text{dom}(f)$.

218 **Unambiguous 2-chained Kleene-plus and its reverse.** If w admits a unique
 219 factorization $w = u_1 \cdot u_2 \cdots u_n$ with $n \geq 1$ and $u_i \in K$ for all $1 \leq$
 220 $i \leq n$ then we set $[K, f]^{2\boxplus}(w) = f(u_1 u_2) \cdot f(u_2 u_3) \cdots f(u_{n-1} u_n)$ and
 221 $[K, f]^{\overleftarrow{2\boxplus}}(w) = f(u_{n-1} u_n) \cdots f(u_2 u_3) \cdot f(u_1 u_2)$ (if $n = 1$, the empty product
 222 gives the unit of \mathbb{D} : $[K, f]^{2\boxplus}(w) = \mathbf{1}_{\mathbb{D}} = [K, f]^{\overleftarrow{2\boxplus}}(w)$). Otherwise, we set
 223 $[K, f]^{2\boxplus}(w) = \perp = [K, f]^{\overleftarrow{2\boxplus}}(w)$.

224 Again, we have $\text{dom}([K, f]^{2\boxplus}) = \text{dom}([K, f]^{\overleftarrow{2\boxplus}}) \subseteq K^+$ and the inclusion is
 225 strict if the Kleene iteration K^+ of K is ambiguous. Notice that, even if
 226 $w \in K^+$ admits a unique factorization $w = u_1 \cdot u_2 \cdots u_n$ with $u_i \in K$ for all
 227 $1 \leq i \leq n$, w is not necessarily in the domain of $[K, f]^{2\boxplus}$ or $[K, f]^{\overleftarrow{2\boxplus}}$. For w
 228 to be in this domain, it is further required that $u_1 u_2, u_2 u_3, \dots, u_{n-1} u_n \in$
 229 $\text{dom}(f)$. Notice that we have $\text{dom}([K, f]^{2\boxplus}) = \text{dom}([K, f]^{\overleftarrow{2\boxplus}}) = K^+$ when
 230 K^+ is unambiguous and $K^2 \subseteq \text{dom}(f)$.

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

Example 5. *Consider the RTEs*

$$\begin{aligned} C_1 &= ((a+b)^+ \#) ? \varepsilon : \perp \boxdot ((a+b)^+ ? \text{copy} : \perp) \\ C_2 &= \# \\ C_3 &= ((a+b)^+ ? \text{copy} : \perp) \boxdot (\#(a+b)^+ ? \varepsilon : \perp) \end{aligned}$$

232 where $\text{copy} = (a ? a : (b ? b : \perp))^{\boxplus}$.

233 Then, $\text{dom}(\llbracket C_2 \rrbracket) = \Sigma^*$, $\text{dom}(\llbracket \text{copy} \rrbracket) = (a+b)^+$ and $\text{dom}(\llbracket C_1 \rrbracket) = \text{dom}(\llbracket C_3 \rrbracket) =$
 234 $(a+b)^+ \# (a+b)^+$. 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*

$$\begin{aligned} C_a &= (b ? \varepsilon : \perp) \boxdot (a ? a : \perp)^{\boxplus} \\ C_b &= (b ? \varepsilon : \perp) \boxdot (a ? b : \perp)^{\boxplus} \end{aligned}$$

235 We have $\text{dom}(\llbracket C_a \rrbracket) = ba^+ = \text{dom}(\llbracket C_b \rrbracket)$ and $\llbracket C_a \rrbracket(ba^n) = a^n$ and $\llbracket C_b \rrbracket(ba^n) = b^n$.

236 We deduce that $\text{dom}(\llbracket C_b \overleftarrow{\boxdot} C_a \rrbracket) = ba^+ ba^+$ and $\llbracket C_b \overleftarrow{\boxdot} C_a \rrbracket(ba^n ba^m) = a^m b^n$.

Consider the expression

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

237 Then, $\text{dom}(\llbracket C \rrbracket) = (ba^+)^+b$, and $\llbracket C \rrbracket(ba^m b) = \varepsilon$ and for $k \geq 2$ we have
 238 $\llbracket C \rrbracket(ba^{m_1} ba^{m_2} b \dots a^{m_k} b) = a^{m_2} b^{m_1} a^{m_3} b^{m_2} \dots a^{m_k} b^{m_{k-1}}$.

239 **Theorem 7.** *2DFTs and RTEs define the same class of functions. More pre-*
 240 *cisely,*

- 241 1. *given an RTE C , we can construct a 2DFT \mathcal{A} such that $\llbracket \mathcal{A} \rrbracket = \llbracket C \rrbracket$,*
- 242 2. *given a 2DFT \mathcal{A} , we can construct an RTE C such that $\llbracket \mathcal{A} \rrbracket = \llbracket C \rrbracket$.*

243 The proof of (1) is given in the next section, while the proof of (2) will be
 244 given in Section 2.6 after some preliminaries in Section 2.5 on transition monoids
 245 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:*

$$f \overleftarrow{\square} g = ((\text{dom}(f)? \varepsilon : \perp) \square g) \odot (f \square (\text{dom}(g)? \varepsilon : \perp)).$$

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

$$f^{\boxplus} = [\text{dom}(f), f \square (\text{dom}(f)? \varepsilon : \perp)]^{2\boxplus} \odot ((\text{dom}(f)^*? \varepsilon : \perp) \square f).$$

246 **Remark 9.** *We can extend the 2-chained Kleene-plus to k -chained Kleene-*
 247 *plus for any $k \geq 3$. It is defined as follows: If w admits a unique factor-*
 248 *ization $w = u_1 u_2 \dots u_n$, with $n \geq 1$ and $u_i \in K$ for all $1 \leq i \leq n$, then*
 249 *$[K, f]^{k\boxplus}(w) = f(u_1 u_2 \dots u_k) f(u_2 u_3 \dots u_{k+1}) \dots f(u_{n-k+1} u_{n-k+2} \dots u_n)$. Oth-*
 250 *erwise, we set $[K, f]^{k\boxplus}(w) = \perp$. Notice that if $n < k$, we have an empty product*
 251 *which gives the unit of \mathbb{D} : $[K, f]^{k\boxplus}(w) = \mathbf{1}_{\mathbb{D}}$. In [16], we have shown that adding*
 252 *the k -plus combinator (or its reverse) does not increase the expressive power of*
 253 *RTEs.*

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

259 *The base function L/d in [5] maps all strings in language L to the constant d ,*
 260 *and is undefined for strings not in L . This can be written using our if-then-else*
 261 *$L? d : \perp$. The conditional choice combinator $f \triangleright g$ of [5] maps an input σ to*
 262 *$f(\sigma)$ if it is in $\text{dom}(f)$, and otherwise it maps it to $g(\sigma)$. This can be written*
 263 *in our syntax as $\text{dom}(f)? f : g$. The split-sum combinator $f \oplus g$ of [5] is the*
 264 *classical Cauchy product $f \square g$. The iterated sum Σf of [5] is the Kleene-plus f^{\boxplus} .*
 265 *The left-split-sum and left-iterated sum of [5] correspond to our reverse Cauchy*
 266 *product $f \overleftarrow{\square} g$ and reverse Kleene-plus $f^{\overleftarrow{\boxplus}}$. The sum $f + g$ of two functions in*
 267 *[5] is the classical Hadamard product $f \odot g$. Finally, the chained sum $\Sigma(f, L)$ of*
 268 *[5] is our two-chained Kleene-plus $[L, f]^{2\boxplus}$.*

269 2.3. RTE to 2DFT

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

274 **Constant functions:** We start with the constant function $d \in \mathbb{D}$ for which it
 275 is easy to construct a 2DFT \mathcal{A} such that $\llbracket d \rrbracket = \llbracket \mathcal{A} \rrbracket$. For $d = \perp$, we take \mathcal{A} such
 276 that $\text{dom}(\mathcal{A}) = \emptyset$ (for instance we use a single state and an empty transition
 277 function). Assume now that $d \in \Gamma^*$. The 2DFT scans the word up to the
 278 right end marker, outputs d and stops. Formally, we let $\mathcal{A} = (\{q\}, \Sigma, \Gamma, \delta, q, \{q\})$
 279 s.t. $\delta(q, a) = (q, \varepsilon, +1)$ for all $a \in \Sigma \cup \{\vdash\}$ and $\delta(q, \dashv) = (q, d, +1)$. Clearly,
 280 $\llbracket \mathcal{A} \rrbracket(w) = d$ for all $w \in \Sigma^*$.

281 The inductive steps follow directly from:

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

- 284 1. a 2DFT \mathcal{A} such that $\llbracket K ? f : g \rrbracket = \llbracket \mathcal{A} \rrbracket$.
- 285 2. a 2DFT \mathcal{A} such that $\llbracket \mathcal{A} \rrbracket = \llbracket f \odot g \rrbracket$.
- 286 3. 2DFTs \mathcal{A}, \mathcal{B} such that $\llbracket \mathcal{A} \rrbracket = \llbracket f \boxplus g \rrbracket$ and $\llbracket \mathcal{B} \rrbracket = \llbracket f \overleftarrow{\boxplus} g \rrbracket$.
- 287 4. 2DFTs \mathcal{A}, \mathcal{B} such that $\llbracket \mathcal{A} \rrbracket = \llbracket f^{\boxplus} \rrbracket$ and $\llbracket \mathcal{B} \rrbracket = \llbracket f^{\overleftarrow{\boxplus}} \rrbracket$.
- 288 5. 2DFTs \mathcal{A}, \mathcal{B} such that $\llbracket \mathcal{A} \rrbracket = \llbracket [K, f]^{2\boxplus} \rrbracket$ and $\llbracket \mathcal{B} \rrbracket = \llbracket [K, f]^{\overleftarrow{2\boxplus}} \rrbracket$.

289 *Proof.* (1) **If then else.** Let \mathcal{B} be a complete DFA that accepts the regular
 290 language K . The idea of the proof is to construct a 2DFT \mathcal{A} which first runs \mathcal{B}
 291 on the input w until the end marker \dashv is reached in some state q of \mathcal{B} . Then,
 292 $w \in K$ iff $q \in F$ is some accepting state of \mathcal{B} . The automaton \mathcal{A} moves left all
 293 the way to \vdash , and starts running either M_f or M_g depending on whether $q \in F$
 294 or not. Since \mathcal{B} is complete, it is clear that $\text{dom}(\mathcal{A}) = \text{dom}(K ? f : g)$ and the
 295 output of \mathcal{A} coincides with $\llbracket M_f \rrbracket$ iff the input is in K , and otherwise coincides
 296 with $\llbracket M_g \rrbracket$.

297 (2) **Hadamard product.** Given an input w , the constructed 2DFT \mathcal{A} first
 298 runs M_f . Instead of executing a transition $p \xrightarrow{\dashv/\gamma, +1} q$ with q a final state of
 299 M_f , it executes $p \xrightarrow{\dashv/\gamma, -1} \text{reset}$ where reset is a new state. While in the reset
 300 state, it moves all the way back to \vdash and it starts running M_g by executing
 301 $\text{reset} \xrightarrow{\vdash/\gamma', +1} q'$ if $\delta_g(q_0, \vdash) = (q', \gamma', +1)$ where δ_g is the transition function of
 302 M_g and q_0 is the initial state of M_g . The final states of \mathcal{A} are those of M_g , and
 303 its initial state is the initial state of M_f . Clearly, $\text{dom}(\mathcal{A}) = \text{dom}(M_f) \cap \text{dom}(M_g)$
 304 and the output of \mathcal{A} is the concatenation of the outputs of M_f and M_g .

305 (3) **Cauchy product.** The domain of a 2DFT is a regular language, accepted
 306 by the 2DFA obtained by ignoring the outputs. Since 2DFAs are effectively
 307 equivalent to (1)DFAs, we can construct from M_f and M_g two DFAs $\mathcal{C}_f =$
 308 $(Q_f, \Sigma, \delta_f, s_f, F_f)$ and $\mathcal{C}_g = (Q_g, \Sigma, \delta_g, s_g, F_g)$ such that $\mathcal{L}(\mathcal{C}_f) = \text{dom}(f)$ and
 309 $\mathcal{L}(\mathcal{C}_g) = \text{dom}(g)$.

310 Now, the set K of words w having at least two factorizations $w = u_1 v_1 = u_2 v_2$
 311 with $u_1, u_2 \in \text{dom}(f)$, $v_1, v_2 \in \text{dom}(g)$ and $u_1 \neq u_2$ is also regular. This is easy
 312 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

- 313 • L_p is the set of words which admit a run in \mathcal{C}_f from its initial state to the
314 final state $p \in F_f$,
- 315 • $M_{p,q}$ is the set of words which admit a run in \mathcal{C}_f from state p to some
316 final state in F_f , and also admit a run in \mathcal{C}_g from its initial state to state
317 $q \in Q_g$,
- 318 • R_q is the set of words which admit a run in \mathcal{C}_g from state q to some final
319 state in F_g , and also admit a run in \mathcal{C}_g from its initial state to some final
320 state in F_g .

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

- 324 1. From $\mathcal{C}_f, \mathcal{C}_g$ and \mathcal{C} we construct a 1NUFT \mathcal{D} such that $\text{dom}(\mathcal{D}) = \text{dom}(f \boxplus g)$
325 and on an input word $w = u \cdot v$ with $u \in \text{dom}(f)$ and $v \in \text{dom}(g)$ it produces
326 the output $u\#v$ where $\# \notin \Sigma$ is a new symbol. On an input word $w \in \Sigma^*$,
327 the transducer \mathcal{D} runs a copy of \mathcal{C} . Simultaneously, \mathcal{D} runs a copy of \mathcal{C}_f on
328 some prefix u of w , copying each input letter to the output. Whenever \mathcal{C}_f is
329 in a final state after reading u , the transducer \mathcal{D} may non-deterministically
330 decide to stop running \mathcal{C}_f , to output $\#$, and to start running \mathcal{C}_g on the
331 corresponding suffix v of w ($w = u \cdot v$) while copying again each input
332 letter to the output. The transducer \mathcal{D} accepts if \mathcal{C} accepts w and \mathcal{C}_g
333 accepts v . Then, we have $u \in \mathcal{L}(\mathcal{C}_f) = \text{dom}(f)$, $v \in \mathcal{L}(\mathcal{C}_g) = \text{dom}(g)$ and
334 $w = u \cdot v \in \mathcal{L}(\mathcal{C}) = \text{dom}(f \boxplus g)$. The output produced by \mathcal{D} is $u\#v$. The
335 only non-deterministic choice in an accepting run of \mathcal{D} is unambiguous
336 since a word $w \in \mathcal{L}(\mathcal{C}) = \text{dom}(f \boxplus g)$ has a unique factorization $w = u \cdot v$
337 with $u \in \text{dom}(f)$ and $v \in \text{dom}(g)$.
- 338 2. We construct a 2DFT \mathcal{T} which takes as input words of the form $u\#v$ with
339 $u, v \in \Sigma^*$, runs M_f on u and then M_g on v . To do so, u is traversed in
340 either direction depending on M_f , and the symbol $\#$ is interpreted as
341 the right end marker \dashv . We explain how \mathcal{T} simulates a transition of M_f
342 moving to the right of \dashv , producing some output γ and going to a state q .
343 If q is not final, then \mathcal{T} moves to the right of $\#$ and then all the way to
344 the end and rejects. If q is final, then \mathcal{T} stays on $\#$ (simulated by moving
345 right and then back left), producing the output γ , but goes to the initial
346 state of M_g instead. \mathcal{T} then runs M_g on v , interpreting $\#$ as \vdash . When M_g
347 moves to the right of \dashv , \mathcal{T} does the same and accepts iff M_g accepts.
- 348 3. In a similar manner, we construct a 2DFT \mathcal{T}' which takes as input strings
349 of the form $u\#v$, first runs M_g on v and then runs M_f on u . Assume that
350 M_g wants to move to the right of \dashv going to state q . If q is not final then
351 \mathcal{T}' also moves to the right of \dashv and rejects. Otherwise, \mathcal{T}' traverses back to
352 \vdash and runs M_f on u . When M_f wants to move to the right of $\#$ going to
353 some state q and producing γ , \mathcal{T}' moves also to the right of $\#$ producing
354 γ and then all the way right producing ε . After moving to the right of \dashv ,
355 it accepts if q is a final state of M_f and rejects otherwise.

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

358 $\llbracket \mathcal{A}' \rrbracket = \llbracket f \sqcap g \rrbracket$. Using the equivalence of 2NUFT and 2DFT, we can convert \mathcal{A}'
 359 into an equivalent 2DFT \mathcal{A} . In a similar way, to obtain $\llbracket f \overline{\sqcap} g \rrbracket$, the 2NUFT \mathcal{B}'
 360 is obtained as a composition of \mathcal{D} and \mathcal{T}' and is then converted to an equivalent
 361 2DFT \mathcal{B} .

362 (4) **Kleene-plus.** The proof is similar to case (3). First, we show that $\text{dom}(f^{\boxplus})$
 363 is regular. Notice that if $\varepsilon \in \text{dom}(f)$ then $\text{dom}(f^{\boxplus}) = \emptyset$, hence we assume below
 364 that $\varepsilon \notin \text{dom}(f)$. As in case (3), the language K of words w having at least
 365 two factorizations $w = u_1v_1 = u_2v_2$ with $u_1, u_2 \in \text{dom}(f)$, $v_1, v_2 \in \text{dom}(f)^*$ and
 366 $u_1 \neq u_2$ is regular. Hence, $K' = \text{dom}(f)^* \cdot K$ is regular and contains all words
 367 in $\text{dom}(f)^+$ having several factorizations as products of words in $\text{dom}(f)$. We
 368 deduce that $\text{dom}(f^{\boxplus}) = \text{dom}(f)^+ \setminus K'$ is regular and we can construct a complete
 369 DFA \mathcal{C} recognizing this domain.

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

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

382 (5) **2-chained Kleene-plus.** As in case (4), we construct the 1NUFT \mathcal{D} which
 383 takes as input w and outputs $u_1\#u_2\#\dots\#u_n$ iff there is an unambiguous
 384 decomposition of w as $u_1u_2\cdots u_n$, with each $u_i \in K$. We then construct a 2DFT
 385 \mathcal{D}' that takes as input words of the form $u_1\#u_2\#\dots\#u_n$ with each $u_i \in \Sigma^*$ and
 386 produces $u_1u_2\#u_2u_3\#\dots\#u_{n-1}u_n$. The 2NUFT \mathcal{A}' is then obtained as the
 387 composition of \mathcal{D}' with the 2DFT \mathcal{T} constructed for case (4). Finally, a 2DFT \mathcal{A}
 388 equivalent to the 2NUFT \mathcal{A}' is constructed. The output produced by \mathcal{A} is thus
 389 $\llbracket M_f \rrbracket(u_1u_2) \cdot \llbracket M_f \rrbracket(u_2u_3) \cdots \llbracket M_f \rrbracket(u_{n-1}u_n)$. We proceed similarly for \mathcal{B} . \square

390 2.4. Unambiguous forest factorization

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

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

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

396 where $a \in \Sigma$. For reasons that will be clear below, we prefer to use the Kleene-
 397 plus instead of the Kleene-star, hence we also add ε explicitly in the syntax. An
 398 expression is said to be ε -free if it does not use ε .

399 Let $(S, \cdot, \mathbf{1}_S)$ be a *finite* monoid and $\varphi: \Sigma^* \rightarrow S$ be a morphism. We say
 400 that a rational expression F is φ -good (or simply *good* when φ is clear from the
 401 context) when

- 402 1. the rational expression F is unambiguous,
 403 2. for each subexpression E of F we have $\varphi(\mathcal{L}(E)) = \{s_E\}$ is a singleton set.
 404 Notice that \emptyset cannot be used in a good expression since it does not satisfy the
 405 second condition. Also, the second condition implies that for each subexpression
 406 E^+ of F we have $s_E \cdot s_E = s_E$ is an idempotent.

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

411 Theorem 12 can be seen as an unambiguous version of Imre Simon’s forest
 412 factorization theorem [23]. Its proof, which can be found in [21], follows the
 413 same lines of the recent proofs of Simon’s theorem, see e.g. [12, 13]. For the
 414 sake of completeness, we summarize the proof idea and contributions in [21]
 415 here. Given a semigroup morphism $\varphi: \Sigma^+ \rightarrow S$, [21] constructs a universal,
 416 unambiguous automaton \mathcal{A} , which is “good” wrt φ in the following sense: (1)
 417 \mathcal{A} is unambiguous and accepts all words in $\Sigma^* \cup \Sigma^\omega$, (2) \mathcal{A} has a unique initial
 418 state i which has no incoming transitions to it, as well as a unique final state f
 419 with no outgoing transitions from it, (3) the states of \mathcal{A} are totally ordered as
 420 $Q \setminus \{i, f\} < f < i$, where Q is the set of states of \mathcal{A} , (4) for each state q , the set
 421 of words that have a run originating at q and ending at q , visiting only states
 422 lower than q in the ordering are mapped to a unique idempotent $e_q \in S$. These
 423 properties of \mathcal{A} ensure that, for any word $w \in \Sigma^* \cup \Sigma^\omega$, the unique accepting run
 424 of w produces a Ramsey split in the sense of [13], with the height of the split
 425 being bounded by the number of states of \mathcal{A} . The construction of \mathcal{A} proceeds
 426 according to the local divisor technique, which uses a lexicographic induction on
 427 $(|S|, |\varphi(\Sigma)|)$. While the base cases (i) when S is a group, and (ii) $|\varphi(\Sigma)| = 1$ are
 428 easy, the inductive cases are non trivial. The inductive cases follow by identifying
 429 an element $c \in S$ for which $Sc \subsetneq S$ or $cS \subsetneq S$, and the details are in [21].

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

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

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

447 2.5. Transition monoid of 2NFAs

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

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

454 In a one way automaton, the canonical morphism $\text{Tr}: \Sigma^* \rightarrow \text{TrM}$ is such that
 455 $\text{Tr}(w)$ consists of the set of pairs (p, q) such that there is a run from state p
 456 to state q reading w . In the case of two-way automaton, we also consider the
 457 starting side (left/right) and ending side (left/right) of the reading head while
 458 going from state p to q . This is represented with a *direction* d amongst “left-left”
 459 (\rhd) , “left-right” (\rightarrow) , “right-left” (\leftarrow) and “right-right” (ζ) . Hence, an element of
 460 TrM is a set X of tuples (p, d, q) with $p, q \in Q$ states of \mathcal{A} and $d \in \{\rightarrow, \rhd, \zeta, \leftarrow\}$.
 461 The canonical morphism $\text{Tr}: \Sigma^* \rightarrow \text{TrM}$ is such that $\text{Tr}(w)$ is the set of triples
 462 (p, d, q) which are compatible with w . For instance, $(p, \rightarrow, q) \in \text{Tr}(w)$ iff \mathcal{A} has a
 463 run starting in state p on the left most symbol of w and which exits w on its
 464 right and in state q . Likewise, $(p, \zeta, q) \in \text{Tr}(w)$ iff \mathcal{A} has a run starting in state
 465 p on the right most symbol of w and which exits w on its right and in state q .
 466 The explanation is similar for other directions.

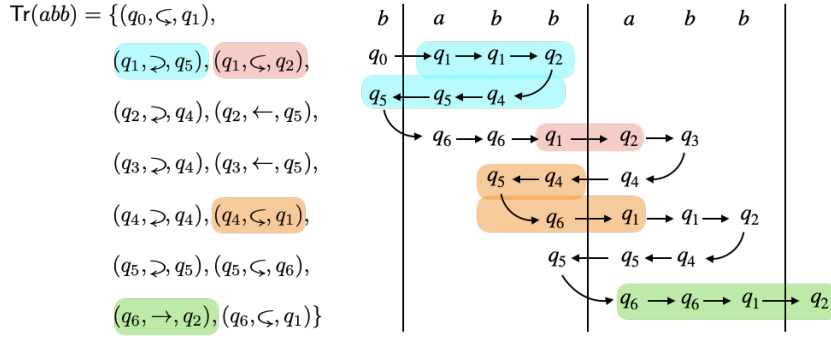


Figure 3: Illustrations of subset of $\text{Tr}(abb)$

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

$$\text{Tr}(abb) = \{(q_0, \zeta, q_1), (q_1, \rhd, q_5), (q_1, \zeta, q_2), (q_2, \rhd, q_4), (q_2, \leftarrow, q_5),$$

$$(q_3, \rhd, q_4), (q_3, \leftarrow, q_5), (q_4, \rhd, q_4), (q_4, \zeta, q_1),$$

$$(q_5, \rhd, q_5), (q_5, \zeta, q_6), (q_6, \rightarrow, q_2), (q_6, \zeta, q_1)\}.$$

467 Some of these triples are highlighted in Figure 3.

468 It is well-known that TrM is a monoid and that Tr is a morphism, see
 469 for instance [7]. The left-right and right-right relations were already used by
 470 Shepherdson to prove the equivalence between two-way and one-way automata
 471 [22]. These relations define a right-congruence. We obtain a congruence by
 472 considering also the right-left and left-left relations. The quotient of the free
 473 monoid by this congruence is the transition monoid of the 2NFA.

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

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

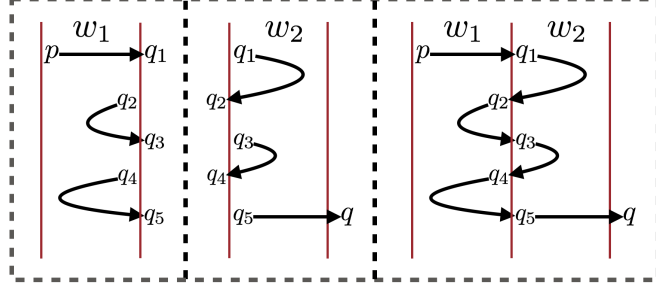
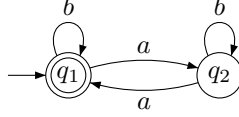


Figure 4: The first and second pictures are illustrations of subsets of $\text{Tr}(w_1)$ and $\text{Tr}(w_2)$ respectively. $(p, \rightarrow, q_1), (q_2, \curvearrowright, q_3), (q_4, \curvearrowright, q_5) \in \text{Tr}(w_1)$ while $(q_1, \curvearrowright, q_2), (q_3, \curvearrowright, q_4), (q_5, \rightarrow,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

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



481
 482 Let TrM be the transition monoid of \mathcal{A} and let $\text{Tr}: \Sigma^* \rightarrow \text{TrM}$ be the canonical
 483 morphism. The expression $F = a^+(ba)^+$ is not Tr -good: one of the reasons
 484 why F is not Tr -good is that the subexpression a^+ is such that $\text{Tr}(a)$ is not an
 485 idempotent since $\text{Tr}(a) = \{(q_1, q_2), (q_2, q_1)\}$ and $\text{Tr}(a^2) = \{(q_1, q_1), (q_2, q_2)\}$, thus
 486 $\text{Tr}(a^2) \neq \text{Tr}(a)$. We have omitted the direction in the tuples as the underlying
 487 automaton is one way. Similarly, the subexpression $(ba)^+$ is also not Tr -good.
 488 The expression $F' = aba \cup aaba \cup a(aa)^+ba \cup a(baba)^+ \cup a(aa)^+(baba)^+$ is not
 489 Tr -good, even though each of the expressions $aba, aaba, a(aa)^+ba, a(baba)^+$ and
 490 $a(aa)^+(baba)^+$ are Tr -good. F' is not Tr -good since $\text{Tr}(\mathcal{L}(F'))$ is not a singleton.
 491 The expression $F'' = aba \cup (aa)^+ \cup a(aa)^+ba$ is Tr -good.

492 2.6. 2DFT to RTE

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

495 Consider a deterministic and complete two-way transducer \mathcal{A} . Let TrM
 496 be the transition monoid of the underlying input automaton. We can apply
 497 the unambiguous factorization theorem to the morphism $\text{Tr}: \Sigma^* \rightarrow \text{TrM}$ in
 498 order to obtain, for each $s \in \text{TrM}$, an ε -free good rational expression F_s for
 499 $\text{Tr}^{-1}(s) \setminus \{\varepsilon\}$. We use the unambiguous expression $G = \varepsilon \cup \bigcup_{s \in \text{TrM}} F_s$ as a *guide*
 500 when constructing RTEs corresponding to the 2DFT \mathcal{A} .

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

- 505 (I₁) $\text{dom}(C_F(x)) = \mathcal{L}(F)$,
- 506 (I₂) for each $u \in \mathcal{L}(F)$, $\llbracket C_F(x) \rrbracket(u)$ is the output produced by \mathcal{A} when running
 507 step x on u (i.e., running \mathcal{A} on u from p to q following direction d).

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

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

- 514 • either $\delta(p, a) = (q, \gamma, 1)$ and we let $C_a((p, \rightarrow, q)) = C_a((p, \zeta, q)) =$
515 $a ? \gamma : \perp$,
- 516 • or $\delta(p, a) = (q, \gamma, -1)$ and we let $C_a((p, \rhd, q)) = C_a((p, \leftarrow, q)) = a ? \gamma :$
517 \perp .

518 Clearly, invariants (I₁) and (I₂) hold for all $x \in \text{Tr}(a) = s_E$.

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 (I₁) for E_1
and E_2 , we deduce that

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

519 Therefore, (I₁) holds for E . Now, for each $u \in \mathcal{L}(E)$, either $u \in \mathcal{L}(E_1)$ and
520 $\llbracket C_E(x) \rrbracket(u) = \llbracket C_{E_1}(x) \rrbracket(u)$ or $u \in \mathcal{L}(E_2)$ and $\llbracket C_E(x) \rrbracket(u) = \llbracket C_{E_2}(x) \rrbracket(u)$.
521 In both cases, applying (I₂) for E_1 or E_2 , we deduce that $\llbracket C_E(x) \rrbracket(u)$ is
522 the output produced by \mathcal{A} when running step x on u .

523 **concatenation** Assume that $E = E_1 \cdot E_2$ is a concatenation. Since the expres-
524 sion is good, we deduce that $s_E = s_{E_1} \cdot s_{E_2}$. Let $x \in s_E$.

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

$$\begin{aligned} C_E(x) &= (C_{E_1}(x_1) \boxtimes C_{E_2}(x_2)) \odot (C_{E_1}(x_3) \boxtimes C_{E_2}(x_4)) \odot \dots \odot \\ &\quad (C_{E_1}(x_i) \boxtimes C_{E_2}(x_{i+1})). \end{aligned}$$

525 Notice that when $i = 1$ we simply have $C_E(x) = C_{E_1}(x_1) \boxtimes C_{E_2}(x_2)$
526 with $x_2 = (q_1, \rightarrow, q)$.

527 The concatenation $\mathcal{L}(E) = \mathcal{L}(E_1) \cdot \mathcal{L}(E_2)$ is unambiguous. Therefore,
528 for all $y \in s_{E_1}$ and $z \in s_{E_2}$, using (I₁) for E_1 and E_2 , we obtain
529 $\text{dom}(C_{E_1}(y) \boxtimes C_{E_2}(z)) = \mathcal{L}(E)$. We deduce that $\text{dom}(C_E(x)) = \mathcal{L}(E)$
530 and (I₁) holds for E .

531 Now, let $u \in \mathcal{L}(E)$ and let $u = u_1 u_2$ be its unique factorization
532 with $u_1 \in \mathcal{L}(E_1)$ and $u_2 \in \mathcal{L}(E_2)$. The step $x = (p, \rightarrow, q)$ per-
533 formed by \mathcal{A} on u is actually the concatenation of steps x_1 on
534 u_1 , followed by x_2 on u_2 , followed by x_3 on u_1 , followed by x_4
535 on u_2 , \dots , until x_{i+1} on u_2 . Using (I₂) for E_1 and E_2 , we de-
536 deduce that the output produced by \mathcal{A} while running step x on u is
537 $\llbracket C_{E_1}(x_1) \rrbracket(u_1) \cdot \llbracket C_{E_2}(x_2) \rrbracket(u_2) \cdot \dots \cdot \llbracket C_{E_1}(x_i) \rrbracket(u_1) \cdot \llbracket C_{E_2}(x_{i+1}) \rrbracket(u_2) =$
538 $\llbracket C_E(x) \rrbracket(u)$.

539
540
541
542
543
544
545
546
547

- If $x = (p, \succ, q)$ 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 $C_E(x) = C_{E_1}(x) \boxtimes (E_2 ? \varepsilon : \perp)$. Since $\text{dom}(E_2 ? \varepsilon : \perp) = \mathcal{L}(E_2)$, we deduce as above that $\text{dom}(C_E(x)) = \mathcal{L}(E)$. 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 $x = (p, \succ, q)$ 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 $\llbracket C_{E_1}(x) \rrbracket(u_1) = \llbracket C_E(x) \rrbracket(u)$.

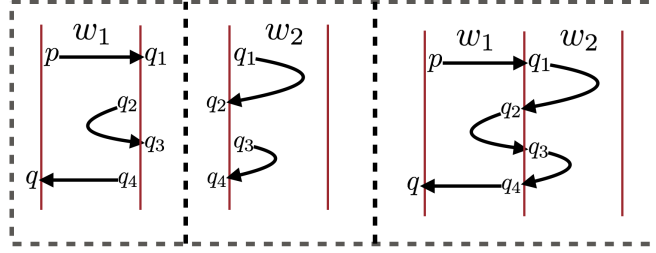


Figure 5: Let $w = w_1 \cdot w_2 \in \mathcal{L}(E)$ with $w_1 \in \mathcal{L}(E_1)$, $w_2 \in \mathcal{L}(E_2)$. We have $(p, \rightarrow, q_1), (q_2, \varsigma, q_3), (q_4, \leftarrow, q) \in \text{Tr}(w_1)$ and $(q_1, \succ, q_2), (q_3, \succ, q_4) \in \text{Tr}(w_2)$. Then (p, \succ, q) is composed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

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

$$C_E(x) = (C_{E_1}(x_1) \boxtimes C_{E_2}(x_2)) \odot (C_{E_1}(x_3) \boxtimes C_{E_2}(x_4)) \odot \dots \odot (C_{E_1}(x_i) \boxtimes (E_2 ? \varepsilon : \perp)).$$

548
549

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 $x = (p, \varsigma, q)$ are handled symmetrically. For instance, when $x = (p, \leftarrow, q)$, the unique sequence of steps is $x_1 = (p, \leftarrow, q_1)$, $x_2 = (q_1, \varsigma, q_2)$, $x_3 = (q_2, \succ, q_3)$, $x_4 = (q_3, \varsigma, q_4)$, \dots , $x_i = (q_{i-1}, \succ, q_i)$, $x_{i+1} = (q_i, \leftarrow, q)$ with $i \geq 1$, $x_1, x_3, \dots, x_i \in s_{E_2}$ and $x_2, x_4, \dots, x_{i+1} \in s_{E_1}$ (see Figure 6). We define

$$C_E(x) = ((E_1 ? \varepsilon : \perp) \boxtimes C_{E_2}(x_1)) \odot (C_{E_1}(x_2) \boxtimes C_{E_2}(x_3)) \odot \dots \odot (C_{E_1}(x_{i-1}) \boxtimes C_{E_2}(x_i)) \odot (C_{E_1}(x_{i+1}) \boxtimes (E_2 ? \varepsilon : \perp)).$$

550
551
552

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 $x \in s$.

553
554
555
556
557
558

- If $x = (p, \succ, q)$. Since F^+ is unambiguous, a word $u \in \mathcal{L}(F^+)$ admits a unique factorization $u = u_1 u_2 \dots u_n$ with $n \geq 1$ and $u_i \in \mathcal{L}(F)$. Now, $\text{Tr}(u_1) = s_E$ and since $x = (p, \succ, q) \in s_E$ 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 $u = u_1 u_2 \dots u_n$ only visits u_1 and is actually ρ itself. Therefore, we

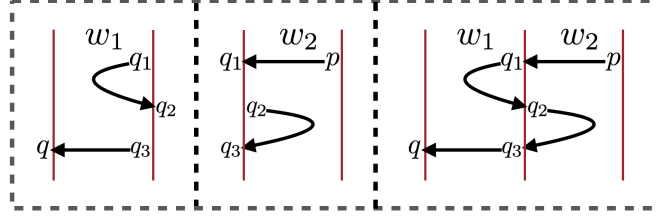


Figure 6: Let $w = w_1 \cdot w_2 \in \mathcal{L}(E)$ with $w_1 \in \mathcal{L}(E_1)$, $w_2 \in \mathcal{L}(E_2)$. We have $(p, \leftarrow, q_1), (q_2, \succ, q_3) \in \text{Tr}(w_2)$ and $(q_1, \swarrow, q_2), (q_3, \leftarrow, q) \in \text{Tr}(w_1)$. Then $(p, \leftarrow, q) \in \text{Tr}(w)$ is composed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

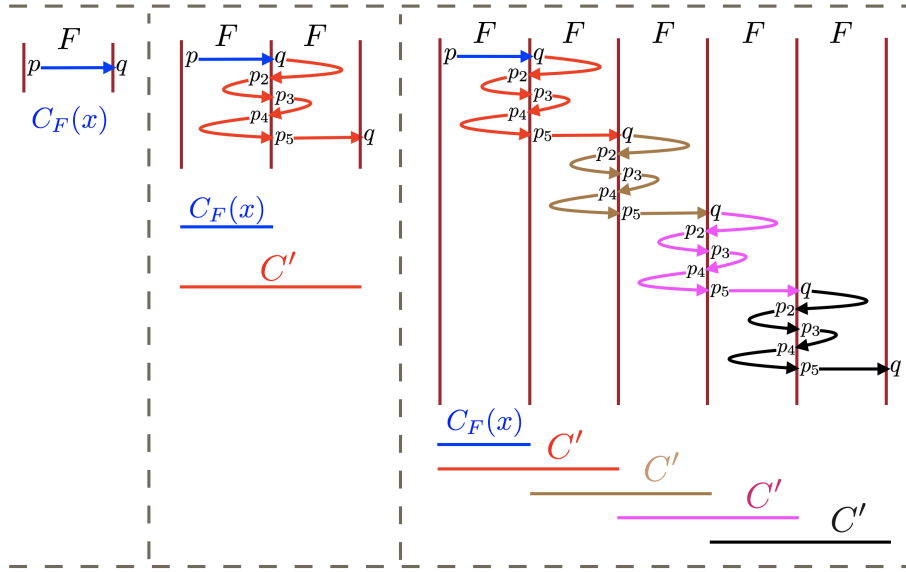


Figure 7: In the Kleene-plus $E = F^+$, a step $x = (p, \rightarrow, q) \in s_E$ on some $u = u_1 u_2 \cdots u_n$ with $u_\ell \in \mathcal{L}(F)$ is obtained by composing the following steps in s_F : $x_1 = x$, $x_2 = (q, \succ, p_2)$, $x_3 = (p_2, \swarrow, p_3)$, $x_4 = (p_3, \succ, p_4)$, $x_5 =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

559
560
561
562
563

set $C_E(x) = C_F(x) \boxtimes (F^* ? \varepsilon : \perp)$ and we can easily check that (I_1-I_2) are satisfied.

- Similarly for $x = (p, \swarrow, q)$ we set $C_E(x) = (F^* ? \varepsilon : \perp) \boxtimes C_F(x)$.
- 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

$$C_E(x) = F ? C_F(x) : (C_F(x) \boxtimes (C_F(y))^{\boxplus}).$$

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

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

$$\begin{aligned} C_E(x) &= (C_F(x) \boxtimes (F^* ? \varepsilon : \perp)) \odot [F, C']^{2\boxplus} \\ C' &= ((F ? \varepsilon : \perp) \boxtimes C_F(x_2)) \odot (C_F(x_3) \boxtimes C_F(x_4)) \odot \cdots \odot \\ &\quad (C_F(x_i) \boxtimes C_F(x_{i+1})) \end{aligned}$$

564 Since the expression is good, the Kleene-plus $E = F^+$ is unambigu-
 565 ous. We have $\text{dom}(C_F(x_j)) = \mathcal{L}(F)$ for $1 \leq j \leq i+1$ by (I_1) . Also
 566 $\text{dom}(F^* ? \varepsilon : \perp) = \mathcal{L}(F^*)$. Since F^+ is unambiguous, the concatena-
 567 tion $F \cdot F^*$ is also unambiguous and we get $\text{dom}(C_F(x) \boxtimes (F^* ? \varepsilon : \perp)) = \mathcal{L}(F) \cdot \mathcal{L}(F^*) = \mathcal{L}(E)$. Also, the product $F \cdot F$ is unam-
 568 biguous and we deduce that $\text{dom}(C_F(x_j) \boxtimes C_F(x_{j+1})) = \mathcal{L}(F)^2$ for
 569 $1 \leq j \leq i$ and $\text{dom}((F ? \varepsilon : \perp) \boxtimes C_F(x_2)) = \mathcal{L}(F)^2$. Therefore,
 570 $\text{dom}(C') = \mathcal{L}(F)^2$ and using once again that F^+ is unambiguous,
 571 we deduce that $\text{dom}([F, C']^{2\boxplus}) = \mathcal{L}(F^+) = \mathcal{L}(E)$. We deduce that
 572 $\text{dom}(C_E(x)) = \mathcal{L}(E)$ and (I_1) holds for E .
 573

574 Let now $u \in \mathcal{L}(F^+) = \text{dom}(C_E(x))$. We have to show that the output
 575 $\gamma \in \mathbb{D}$ produced by \mathcal{A} when running step x on u is $\llbracket C_E(x) \rrbracket(u)$. There
 576 is a unique factorization $u = u_1 u_2 \cdots u_n$ with $n \geq 1$ and $u_\ell \in \mathcal{L}(F)$
 577 for $1 \leq \ell \leq n$.

578 Assume first that $n = 1$ (see Figure 7 left). By definition, we have
 579 $\llbracket [F, C']^{2\boxplus} \rrbracket(u) = \varepsilon$ and $\llbracket C_F(x) \boxtimes (F^* ? \varepsilon : \perp) \rrbracket(u) = \llbracket C_F(x) \rrbracket(u)$ which,
 580 by induction, is the output γ produced by \mathcal{A} running step x on u .
 581 Therefore, $\llbracket C_E(x) \rrbracket(u) = \gamma \cdot \varepsilon = \gamma$.

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 $\gamma_j^\ell = \llbracket C_F(x_j) \rrbracket(u_\ell)$
 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

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

582 We have $\llbracket C' \rrbracket(u_\ell u_{\ell+1}) = \gamma_2^{\ell+1} \gamma_3^\ell \gamma_4^{\ell+1} \cdots \gamma_i^\ell \gamma_{i+1}^{\ell+1}$ for $1 \leq \ell < n$. There-
 583 fore, we obtain $\gamma = \gamma_1^1 \llbracket C' \rrbracket(u_1 u_2) \llbracket C' \rrbracket(u_2 u_3) \cdots \llbracket C' \rrbracket(u_{n-1} u_n)$. Since
 584 $\llbracket C_F(x) \boxtimes (F^* ? \varepsilon : \perp) \rrbracket(u) = \gamma_1^1$ we deduce that $\gamma = \llbracket C_E(x) \rrbracket(u)$.

585 • The case of $x = (p, \leftarrow, q)$ can be handled similarly. \square

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

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

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

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

592 right in some state q . Or the step (p, ζ, q) starts on the right in some state p
 593 and exits on the right in some state q . See Figure 8.

Let s_{\vdash} be the set of steps $(p, \rightarrow, q), (p, \zeta, q)$ 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 $x_1 = (q_0, \rightarrow, q_1), x_2 = (q_1, \supset, q_2), x_3 = (q_2, \zeta, q_3), x_4 = (q_3, \supset, q_4), \dots, x_i = (q_{i-1}, \zeta, q_i), x_{i+1} = (q_i, \rightarrow, q)$ with $i \geq 1, x_1, x_3, \dots, x_i \in s_{\vdash}$ and $x_2, x_4, \dots, x_{i+1} \in s$ (see Figure 8 left). We define

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

594 Notice that when $i = 1$ we simply have $C_{\vdash F_s}((q_0, \rightarrow, q)) = \gamma_{q_0} \odot C_{F_s}((q_1, \rightarrow, q))$.
 595 Since $\text{dom}(C_{F_s}(x_i)) = \mathcal{L}(F_s) = \text{Tr}^{-1}(s) \setminus \{\varepsilon\}$ for $i = 2, 4, \dots, i+1$, we deduce that
 596 $\text{dom}(C_{\vdash F_s}((q_0, \rightarrow, q))) = \text{Tr}^{-1}(s) \setminus \{\varepsilon\}$. Moreover, for each $u \in \text{Tr}^{-1}(s) \setminus \{\varepsilon\}$, the
 597 output produced by \mathcal{A} performing step (q_0, \rightarrow, q) on $\vdash u$ is $\llbracket C_{\vdash F_s}((q_0, \rightarrow, q)) \rrbracket(u)$.

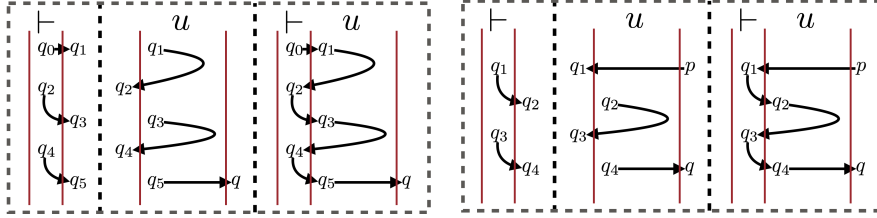


Figure 8: (Left) Given steps of s , a step (q_0, \rightarrow, q) of $\vdash u$ for some $u \in F_s$, is obtained by composing the following steps alternatively from s_{\vdash} and s : $x_1 = (q_0, \rightarrow, q_1), x_2 = (q_1, \supset, q_2), x_3 = (q_2, \zeta, q_3), x_4 = (q_3, \supset, q_4), x_5 =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

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

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

598 As above, we have $\text{dom}(C_{\vdash F_s}((p, \zeta, q))) = \text{Tr}^{-1}(s) \setminus \{\varepsilon\}$. Moreover, for each
 599 $u \in \text{Tr}^{-1}(s) \setminus \{\varepsilon\}$, the output produced by \mathcal{A} performing step (p, ζ, q) on $\vdash u$ is
 600 $\llbracket C_{\vdash F_s}((p, \zeta, q)) \rrbracket(u)$.

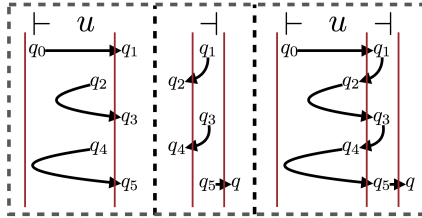


Figure 9: On input $\vdash u$, a step $x = (q_0, \rightarrow, q)$ is obtained by composing the following steps alternatively from steps of $\vdash u$ and s_{\vdash} : $x_1 = (q_0, \rightarrow, q_1), x_2 = (q_1, \supset, q_2), x_3 = (q_2, \zeta, q_3), x_4 = (q_3, \supset, q_4), x_5 = (q_4, \zeta, q_5)$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

601 Similarly, let s_{\dashv} be the set of steps (p, \succ, q) such that there is a transition
602 $\delta(p, \dashv) = (q, \gamma_p, -1)$ in \mathcal{A} or steps (p, \rightarrow, q) such that there is a transition
603 $\delta(p, \dashv) = (q, \gamma_p, +1)$ in \mathcal{A} . From the initial state q_0 of \mathcal{A} , there is a unique
604 sequence of steps $x_1 = (q_0, \rightarrow, q_1)$, $x_2 = (q_1, \succ, q_2)$, $x_3 = (q_2, \swarrow, q_3)$, $x_4 = (q_3, \succ$
605 $, q_4)$, \dots , $x_i = (q_{i-1}, \swarrow, q_i)$, $x_{i+1} = (q_i, \rightarrow, q)$ with $i \geq 1$, and x_1, x_3, \dots, x_i are
606 steps where $C_{\vdash F_s}$ is defined and $x_2, x_4, \dots, x_{i+1} \in s_{\dashv}$ (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 $\text{dom}(\mathcal{A}) \cap (\text{Tr}^{-1}(s) \setminus \{\varepsilon\}) = \emptyset$ so we set $C_s = \perp$. Or, $q \in F$ and $\text{Tr}^{-1}(s) \setminus \{\varepsilon\} \subseteq \text{dom}(\mathcal{A})$ so we define

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

607 We have $\text{dom}(C_s) = \text{Tr}^{-1}(s) \setminus \{\varepsilon\}$ and for all $u \in \text{dom}(C_s)$ we have $\llbracket C_s \rrbracket(u) =$
608 $\llbracket \mathcal{A} \rrbracket(u)$. \square

609 3. Infinite Words

610 In this section, we start looking at regular functions on infinite words. As
611 in Section 2, we restrict our attention to two-way transducers as the model for
612 computing regular functions. Given a finite alphabet Σ , let Σ^ω denote the set of
613 infinite words over Σ , and let $\Sigma^\infty = \Sigma^* \cup \Sigma^\omega$ be the set of all finite or infinite
614 words over Σ .

615 3.1. Two-way transducers over ω -words (ω -2DMT_{la})

616 Let Σ be a finite input alphabet and let Γ be a finite output alphabet. Let \vdash
617 be a left end marker symbol not in Σ and let $\Sigma_{\vdash} = \Sigma \cup \{\vdash\}$. The input word is
618 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_{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 $\delta: Q \times \Sigma_{\vdash} \times \mathcal{R} \mapsto Q \times \Gamma^* \times \{-1, +1\}$ is the partial transition function. We request that for every pair $(q, a) \in Q \times \Sigma_{\vdash}$, the subset $\mathcal{R}(q, a)$ of languages $R \in \mathcal{R}$ such that $\delta(q, a, R)$ is defined forms a partition of Σ^ω . This ensures that \mathcal{A} is complete and behaves deterministically. The set $\mathcal{F} \subseteq 2^Q$ specifies the Muller acceptance condition. As in the finite case, the reading head cannot move left while on \vdash . A configuration is represented by $w'qaw''$ where $w'a \in \vdash\Sigma^*$, $w'' \in \Sigma^\omega$ and q is the current state, scanning letter a . From configuration $w'qaw''$, let R be the unique ω -regular language in $\mathcal{R}(q, a)$ such that $w'' \in R$, the automaton outputs γ and moves to

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

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

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

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

623 The notation $\omega\text{-2DMT}_{\text{la}}$ signifies the use of the look-ahead (la) using the ω -
624 regular languages in \mathcal{R} . It must be noted that without look-ahead, the expressive
625 power of two-way transducers over infinite words is lesser than regular transfor-
626 mations over infinite words [4]. A classical example of this is given in Example
627 15, where the look-ahead is necessary to obtain the required transformation.

628 **Example 15.** Figure 1 shows an $\omega\text{-2DMT}_{\text{la}}$ \mathcal{A}' over $\Sigma = \{a, b, \#\}$ that defines
629 the transformation $\llbracket \mathcal{A}' \rrbracket(u_1 \# u_2 \# \cdots \# u_n \# v) = u_1^R u_1 \# u_2^R u_2 \# \cdots \# u_n^R u_n \# v$
630 where $u_1, \dots, u_n \in (a+b)^*$, $v \in (a+b)^\omega$ and u^R denotes the reverse of u . The
631 Muller acceptance set is $\{\{q_5\}\}$. From state q_1 reading \vdash , or state q_4 reading $\#$,
632 \mathcal{A}' uses the look ahead partition $\mathcal{R}(q_1, \vdash) = \mathcal{R}(q_4, \#) = \{\Sigma^* \# \Sigma^\omega, (\Sigma \setminus \{\#\})^\omega\}$,
633 which indicates the presence or absence of a $\#$ in the remaining suffix of the
634 word being read. For all other transitions, the look-ahead language is Σ^ω , hence
635 it is omitted. Also, to keep the picture light, the automaton is not complete, i.e.,
636 we have omitted the transitions going to a sink state. It can be seen that any
637 maximal string u between two consecutive occurrences of $\#$ is replaced with $u^R u$;
638 the infinite suffix over $\{a, b\}^\omega$ is then reproduced as it is.

639 **Remark 16.** The model used here is a two-way, deterministic Muller automaton,
640 which has for each pair (q, a) consisting of a state and symbol, a tuple of look-
641 ahead ω -regular languages which are mutually exclusive. The model (denoted
642 2WST_{la}) used in [4] however is a two-way deterministic Muller automaton which
643 is equipped with a look-behind automaton (a NFA) and a look-ahead automaton
644 (a possibly non-deterministic Muller automaton). It is easy to see that the two
645 models are equivalent, see [16] for details.

646 3.2. ω -Regular Transducer Expressions (ω -RTE)

647 As in the case of finite words, we define regular transducer expressions for
648 infinite words. Let Σ and Γ be finite input and output alphabets and let \perp stand
649 for undefined. We define the output domain as $\mathbb{D} = \Gamma^\infty \cup \{\perp\}$, with the usual
650 concatenation of a finite word on the left with a finite or infinite word on the
651 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:

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

652 where $K \subseteq \Sigma^+$ ranges over regular languages of finite non-empty words, $L \subseteq \Sigma^\omega$
653 ranges over ω -regular languages of infinite words and E is an RTE over finite
654 words as defined in Section 2.2. The semantics $\llbracket E \rrbracket: \Sigma^* \rightarrow \Gamma^* \cup \{\perp\}$ of the
655 finitary combinator expressions $E \in \text{RTE}$ is unchanged (see Section 2.2). The
656 semantics of an ω -RTE C is a function $\llbracket C \rrbracket: \Sigma^\omega \rightarrow \mathbb{D}$. Given a regular language
657 $K \subseteq \Sigma^+$, an ω -regular language $L \subseteq \Sigma^\omega$, and functions $f: \Sigma^* \rightarrow \Gamma^* \cup \{\perp\}$,
658 $g, h: \Sigma^\omega \rightarrow \mathbb{D}$, we define

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

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

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

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

664 **Unambiguous Cauchy product.** If $w \in \Sigma^\omega$ admits a unique factorization
665 $w = u \cdot v$ with $u \in \text{dom}(f)$ and $v \in \text{dom}(g)$ then we set $(f \boxtimes g)(w) =$
666 $f(u) \cdot g(v)$. Otherwise, we set $(f \boxtimes g)(w) = \perp$.

667 **Unambiguous ω -iteration.** If $w \in \Sigma^\omega$ admits a unique infinite factorization
668 $w = u_1 u_2 u_3 \cdots$ with $u_i \in \text{dom}(f)$ for all $i \geq 1$ then we set $f^\omega(w) =$
669 $f(u_1) f(u_2) f(u_3) \cdots \in \Gamma^\infty$. Otherwise, we set $f^\omega(w) = \perp$.

670 **Unambiguous 2-chained ω -iteration.** If $w \in \Sigma^\omega$ admits a unique factoriza-
671 tion $w = u_1 u_2 u_3 \cdots$ with $u_i \in K$ for all $i \geq 1$ and if moreover $u_i u_{i+1} \in$
672 $\text{dom}(f)$ for all $i \geq 1$ then we set $[K, f]^{2\omega}(w) = f(u_1 u_2) f(u_2 u_3) f(u_3 u_4) \cdots$.
673 Otherwise, we set $[K, f]^{2\omega}(w) = \perp$.

674 **Remark 17.** Let $C_\varepsilon = (\Sigma ? \varepsilon : \perp)^\omega$. We have $\text{dom}(C_\varepsilon) = \Sigma^\omega$ and $\llbracket C_\varepsilon \rrbracket(w) = \varepsilon$
675 for all $w \in \Sigma^\omega$. Now, for $\gamma \in \Gamma^+$, let $C_\gamma = (\Sigma ? \gamma : \perp) \boxtimes C_\varepsilon$. We have
676 $\text{dom}(C_\gamma) = \Sigma^\omega$ and $\llbracket C_\gamma \rrbracket(w) = \gamma$ for all $w \in \Sigma^\omega$. Therefore, we can freely use
677 constants $\gamma \in \Gamma^*$ when defining ω -RTEs.

678 **Remark 18.** We can express the ω -iteration with the 2-chained ω -iteration as
679 follows: $f^\omega = [\text{dom}(f), f \boxtimes (\text{dom}(f) ? \varepsilon : \perp)]^{2\omega}$.

680 **Remark 19.** In a similar manner to $[K, f]^{k\boxplus}$, we can extend 2-chained ω -
681 iteration as well to k -chained ω -iteration for any $k \geq 3$. It is defined as fol-
682 lows: If w admits a unique factorization $w = u_1 u_2 \dots$, with $u_i \in K$ for all
683 $i \geq 1$, then $[K, f]^{k\omega}(w) = f(u_1 u_2 \dots u_k) f(u_2 u_3 \dots u_{k+1}) \dots$. Otherwise, we set
684 $[K, f]^{k\omega}(w) = \perp$. In [16], we have shown that adding k -chained ω -iteration does
685 not increase the expressive power of ω -RTEs.

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

$$E_1 = a ? a : (b ? b : (\# ? \# : \perp))$$

$$E_2 = a ? a : (b ? b : \perp)$$

$$E_3 = a ? a : (b ? b : (\# ? \varepsilon : \perp)).$$

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

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

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

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

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

$$\llbracket C_1 \rrbracket(u_1 \# u_2 \# \cdots u_n \# v) = u_1^R u_1 \# u_2^R u_2 \# \cdots \# u_n^R u_n \# v$$

when $u_i \in (a + b)^*$ and $v \in (a + b)^\omega$. Finally, let

$$C = (a + b)^\omega ? E_2^\omega : C_1.$$

686 We have $\text{dom}(C) = [(a + b)^* \#]^*(a + b)^\omega$ and $\llbracket C \rrbracket = \llbracket \mathcal{A}' \rrbracket$ where \mathcal{A}' is the transducer
687 of Figure 1.

688 The main theorem connecting ω -2DMT_{la} and ω -RTE is as follows.

689 **Theorem 21.** ω -2DMT_{la} and ω -RTEs define the same class of functions. More
690 precisely,

- 691 1. given an ω -RTE C , we can construct an ω -2DMT_{la} \mathcal{A} such that $\llbracket \mathcal{A} \rrbracket = \llbracket C \rrbracket$.
- 692 2. given an ω -2DMT_{la} \mathcal{A} , we can construct an ω -RTE C such that $\llbracket \mathcal{A} \rrbracket = \llbracket C \rrbracket$,

693 The proof of (1) is given in the next section, while the proof of (2) will be
694 given in Section 3.7 after some preparatory work on backward deterministic Büchi
695 automata (Section 3.4) which are used to remove the look-ahead of ω -2DMT_{la}
696 (Section 3.5), and the notion of transition monoid for ω -2DMT_{la} (Section 3.6)
697 used in the unambiguous forest factorization theorem extended to infinite words
698 (Theorem 28).

699 3.3. ω -RTE to ω -2DMT_{la}

700 In this section, we prove one direction of Theorem 21: given an ω -RTE C , we
701 can construct an ω -2DMT_{la} \mathcal{A} such that $\llbracket \mathcal{A} \rrbracket = \llbracket C \rrbracket$. The proof is by structural
702 induction and follows immediately from

703 **Lemma 22.** Let $K \subseteq \Sigma^*$ be regular and $L \subseteq \Sigma^\omega$ be ω -regular. Let f be an RTE
704 with $\llbracket f \rrbracket = \llbracket M_f \rrbracket$ for some 2DFT M_f . Let g, h be ω -RTEs with $\llbracket g \rrbracket = \llbracket M_g \rrbracket$ and
705 $\llbracket h \rrbracket = \llbracket M_h \rrbracket$ for ω -2DMT_{la} M_g and M_h respectively. Then, one can construct

- 706 1. an ω -2DMT_{la} \mathcal{A} such that $\llbracket L?g : h \rrbracket = \llbracket \mathcal{A} \rrbracket$,
- 707 2. an ω -2DMT_{la} \mathcal{A} such that $\llbracket \mathcal{A} \rrbracket = \llbracket g \odot h \rrbracket$,
- 708 3. an ω -2DMT_{la} \mathcal{A} such that $\llbracket \mathcal{A} \rrbracket = \llbracket f \boxplus g \rrbracket$,
- 709 4. an ω -2DMT_{la} \mathcal{A} such that $\llbracket \mathcal{A} \rrbracket = \llbracket f^\omega \rrbracket$,
- 710 5. an ω -2DMT_{la} \mathcal{A} such that $\llbracket \mathcal{A} \rrbracket = \llbracket [K, f]^{2\omega} \rrbracket$.

711 *Proof.* Throughout the proof, we let $M_g = (Q_g, \Sigma, \Gamma, s_g, \delta_g \mathcal{F}_g, \mathcal{R}_g)$ and $M_h =$
712 $(Q_h, \Sigma, \Gamma, s_h, \delta_h \mathcal{F}_h, \mathcal{R}_h)$ be the ω -2DMT_{la} such that $\llbracket M_g \rrbracket = \llbracket g \rrbracket$ and $\llbracket M_h \rrbracket = \llbracket h \rrbracket$.

713 (1) **If then else.** The set of states of \mathcal{A} is $Q_{\mathcal{A}} = \{q_0\} \cup Q_g \cup Q_h$ with $q_0 \notin$
714 $Q_g \cup Q_h$. In state q_0 , we have the transitions $\delta_{\mathcal{A}}(q_0, (\vdash, R \cap L)) = (q, \gamma, +1)$ if
715 $\delta_g(s_g, (\vdash, R)) = (q, \gamma, +1)$ and $\delta_{\mathcal{A}}(q_0, (\vdash, R' \setminus L)) = (q', \gamma', +1)$ if $\delta_h(s_h, (\vdash, R')) =$
716 $(q', \gamma', +1)$. This invokes M_g (M_h) iff the input w is in L (not in L). The Muller
717 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
718 is clear that $\llbracket \mathcal{A} \rrbracket$ coincides with $\llbracket M_g \rrbracket$ iff the input string is in L , and otherwise,
719 $\llbracket \mathcal{A} \rrbracket$ coincides with $\llbracket M_h \rrbracket$.

720 (2) **Hadamard product.** Recall that for a word w to be in $\text{dom}(g \odot h)$ we
721 should have $w \in \text{dom}(g) \cap \text{dom}(h)$ and also $\llbracket g \rrbracket(w) \in \Gamma^*$. Hence, M_g will produce
722 $\llbracket g \rrbracket(w)$ after reading a finite prefix of w . We create a look ahead which indicates
723 the position where the transducer M_g can stop reading the input word w so that
724 we can reset the head to the left most position and start M_h . The look ahead
725 should satisfy two conditions for this purpose:

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

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

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

735 We explain the construction of the ω -2DMT_{la} \mathcal{A} such that $\llbracket g \odot h \rrbracket = \llbracket \mathcal{A} \rrbracket$. The
736 set of states of \mathcal{A} are $Q_{\mathcal{A}} = Q_g \cup Q_h \cup \{\text{reset}\}$. Backward transitions in \mathcal{A} and
737 M_g are the same: $\delta_{\mathcal{A}}(q, a, R) = (q', \gamma, -1)$ iff $\delta_g(q, a, R) = (q', \gamma, -1)$. Forward
738 transitions of M_g are divided into two depending on the look ahead. If we have
739 $\delta_g(q, a, R) = (q', \gamma, +1)$ in M_g for an $a \in \Sigma_{\vdash}$, then

$$740 \quad \delta_{\mathcal{A}}(q, a, R \setminus L_{q'}) = (q', \gamma, +1) \quad \text{and} \quad \delta_{\mathcal{A}}(q, a, R \cap L_{q'}) = (\text{reset}, \gamma, +1).$$

741 From the `reset` state, we go to the left until \vdash is reached and then start running M_h .
742 So, $\delta_{\mathcal{A}}(\text{reset}, a, \Sigma^\omega) = (\text{reset}, \varepsilon, -1)$ for all $a \in \Sigma$ and $\delta_{\mathcal{A}}(\text{reset}, \vdash, R) = (q'', \gamma, +1)$
743 if $\delta_h(s_h, \vdash, R) = (q'', \gamma, +1)$. The accepting set is the same as the Muller accepting
744 set \mathcal{F}_h of M_h .

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

748 Now, the set L of words w having at least two factorizations $w = u_1 v_1 = u_2 v_2$
749 with $u_1, u_2 \in \text{dom}(f)$, $v_1, v_2 \in \text{dom}(g)$ and $u_1 \neq u_2$ is ω -regular. This is easy
750 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

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

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

760 First we construct an ω -1DMT_{la} \mathcal{D} such that $\text{dom}(\mathcal{D}) = \text{dom}(f \boxtimes g)$ and
761 on an input word $w = uv$ with $u \in \text{dom}(f)$ and $v \in \text{dom}(g)$, it produces the
762 output $u\#v$ where $\# \notin \Sigma$ is a new symbol. From its initial state while reading
763 \vdash , \mathcal{D} uses the look-ahead to check whether the input word w is in $\text{dom}(f \boxtimes g)$
764 or not. If yes, it moves right and enters the initial state of \mathcal{D}_f . If not, it goes
765 to a sink state and rejects. While running \mathcal{D}_f , \mathcal{D} copies each input letter to
766 output. Upon reaching a final state of \mathcal{D}_f , we use the look-ahead $\text{dom}(g)$ to see
767 whether we should continue running \mathcal{D}_f or we should switch to \mathcal{D}_g . Formally, if
768 $\delta_f(q, a) = q' \in F_f$ the corresponding transitions of \mathcal{D} are

$$769 \quad \delta_{\mathcal{D}}(q, a, \text{dom}(g)) = (s_g, a\#, +1) \quad \text{and} \quad \delta_{\mathcal{D}}(q, a, \Sigma^\omega \setminus \text{dom}(g)) = (q', a, +1).$$

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

773 Next we construct an ω -2DMT_{la} \mathcal{T} which takes input words of the form $u\#v$
774 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
775 in either direction depending on M_f and the symbol $\#$ is interpreted as right
776 end marker \dashv for M_f . While simulating a transition of M_f moving right of \dashv ,
777 producing the output γ and reaching state q , there are two possibilities. If q is
778 not a final state of M_f then \mathcal{T} moves to the right of $\#$, goes to some sink state
779 and rejects. If q is a final state of M_f , then \mathcal{T} stays on $\#$ producing the output
780 γ and goes to the initial state of M_g . Then, \mathcal{T} runs M_g on v interpreting $\#$ as

781 \vdash . The Muller accepting set of \mathcal{T} is the same as M_g .

782 We construct an ω -2DMT $_{\text{la}}$ \mathcal{A} as the composition of \mathcal{D} and \mathcal{T} . Regular
783 transformations are definable by ω -2DMT $_{\text{la}}$ [4] and are closed under composition
784 [14]. Thus the composition of an ω -1DMT $_{\text{la}}$ and an ω -2DMT $_{\text{la}}$ is an ω -2DMT $_{\text{la}}$.
785 We deduce that \mathcal{A} is an ω -2DMT $_{\text{la}}$. Moreover $\llbracket \mathcal{A} \rrbracket = \llbracket f \boxplus g \rrbracket$.

786 (4) **ω -iteration.** By Remark 18, this is a derived operator and hence the result
787 follows from the next case.

788 (5) **2-chained ω -iteration.** First we show that the set of words w in Σ^ω having
789 an unambiguous decomposition $w = u_1 u_2 \cdots$ with $u_i \in K$ for each i is ω -regular.
790 As in case (3) above, the language L of words w having at least two factorizations
791 $w = u_1 v_1 = u_2 v_2$ with $u_1, u_2 \in K$, $v_1, v_2 \in K^\omega$ and $u_1 \neq u_2$ is ω -regular. Hence,
792 $L' = K^* \cdot L$ is ω -regular and contains all words in Σ^ω having several factorizations
793 as products of words in K . We deduce that $\Sigma^\omega \setminus L'$ is ω -regular.

794 As in case (3) above, we construct an ω -1DMT $_{\text{la}}$ \mathcal{D} which takes as input w and
795 outputs $u_1 \# u_2 \# \cdots$ iff there is an unambiguous decomposition of w as $u_1 u_2 \cdots$
796 with each $u_i \in K$. We then construct an ω -2DMT \mathcal{D}' that takes as input words
797 of the form $u_1 \# u_2 \# \cdots$ with each $u_i \in \Sigma^*$ and produces $u_1 u_2 \# u_2 u_3 \# \cdots$.

798 Next we construct an ω -2DMT \mathcal{T} that takes as input words of the form
799 $w_1 \# w_2 \# \cdots$ with each $w_i \in \Sigma^*$ and runs M_f on each w_i from left to right. The
800 transducer \mathcal{T} interprets $\#$ as \vdash (resp. \dashv) when it is reached from the right (resp.
801 from left). While simulating a transition of M_f moving right of \dashv , we proceed
802 as in case (3) above, except that \mathcal{T} goes to the initial state of M_f instead.

803 The ω -2DMT $_{\text{la}}$ \mathcal{A} is then obtained as the composition of \mathcal{D} , \mathcal{D}' and \mathcal{T} . The
804 output produced by \mathcal{A} is thus $\llbracket M_f \rrbracket(u_1 u_2) \llbracket M_f \rrbracket(u_2 u_3) \cdots$. \square

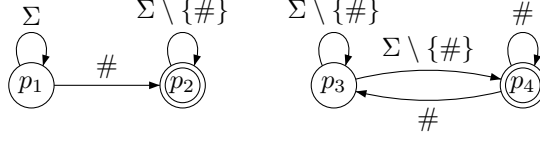
805 3.4. Backward deterministic Büchi automata (BDBA)

806 A Büchi automaton over the input alphabet Σ is a tuple $\mathcal{B} = (P, \Sigma, \Delta, \text{Fin})$
807 where P is a finite set of states, $\text{Fin} \subseteq P$ is the set of final (accepting) states,
808 and $\Delta \subseteq P \times \Sigma \times P$ is the transition relation. A run of \mathcal{B} over an infinite word
809 $w = a_1 a_2 a_3 \cdots$ is a sequence $\rho = p_0, a_1, p_1, a_2, p_2, \dots$ such that $(p_{i-1}, a_i, p_i) \in \Delta$
810 for all $i \geq 1$. The run is final (accepting) if $\text{inf}(\rho) \cap \text{Fin} \neq \emptyset$ where $\text{inf}(\rho)$ is the
811 set of states visited infinitely often by ρ . This is a Büchi acceptance condition.

812 The Büchi automaton \mathcal{B} is *backward deterministic* (BDBA) or *complete*
813 *unambiguous* (CUBA) if for all infinite words $w \in \Sigma^\omega$, there is *exactly one*
814 run ρ of \mathcal{B} over w which is final, this run is denoted $\mathcal{B}(w)$. The fact that we
815 request at least/most one final run on w explains why the automaton is called
816 complete/unambiguous. Wlog, we may assume that all states of \mathcal{B} are *useful*, i.e.,
817 for all $p \in P$ there exists some $w \in \Sigma^\omega$ such that $\mathcal{B}(w)$ starts from state p . In
818 that case, it is easy to check that the transition relation is *backward deterministic*
819 *and complete*: for all $(p, a) \in P \times \Sigma$ there is exactly one state p' such that
820 $(p', a, p) \in \Delta$. We write $p' \xleftarrow{a} p$ and state p' is denoted $\Delta^{-1}(p, a)$. In other words,
821 the inverse of the transition relation $\Delta^{-1}: P \times \Sigma \rightarrow P$ is a total function.

822 For each state $p \in P$, we let $\mathcal{L}(\mathcal{B}, p)$ be the set of infinite words $w \in \Sigma^\omega$ such
823 that $\mathcal{B}(w)$ starts from p . Notice that, $\Sigma^\omega = \bigsqcup_{p \in P} \mathcal{L}(\mathcal{B}, p)$, i.e., words in Σ^ω are
824 partitioned according to the starting state of their unique final run. For every
825 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.

826 **Example 23.** For instance, the automaton \mathcal{B} below is a BDBA. Moreover,
 827 we have $\mathcal{L}(\mathcal{B}, p_2) = (\Sigma \setminus \{\#\})^\omega$, $\mathcal{L}(\mathcal{B}, p_4) = (\#\Sigma^*)^\omega$, and $\mathcal{L}(\mathcal{B}, \{p_1, p_3, p_4\}) =$
 828 $\Sigma^* \#\Sigma^\omega$.



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

832 **Theorem 24** (Carton & Michel [10]). A language $L \subseteq \Sigma^\omega$ is ω -regular iff
 833 $L = \mathcal{L}(\mathcal{B}, I)$ for some BDBA \mathcal{B} and initial set I .

834 The proof in [10] is constructive, starting with an NBA with m states, they
 835 construct an equivalent BDBA with $(3m)^m$ states.

836 A crucial fact on BDBA is that they are easily closed under Boolean operations.
 837 In particular, the complement, which is quite difficult for NBAs, becomes trivial
 838 with BDBAs: $\mathcal{L}(\mathcal{B}, P \setminus I) = \Sigma^\omega \setminus \mathcal{L}(\mathcal{B}, I)$. For intersection and union, we simply
 839 use the classical cartesian product of two automata \mathcal{B}_1 and \mathcal{B}_2 . This clearly
 840 preserves the backward determinism. For intersection, we use a generalized
 841 Büchi acceptance condition, i.e., a conjunction of Büchi acceptance conditions.
 842 For BDBAs, generalized and classical Büchi acceptance conditions are equivalent
 843 [10]. We obtain immediately

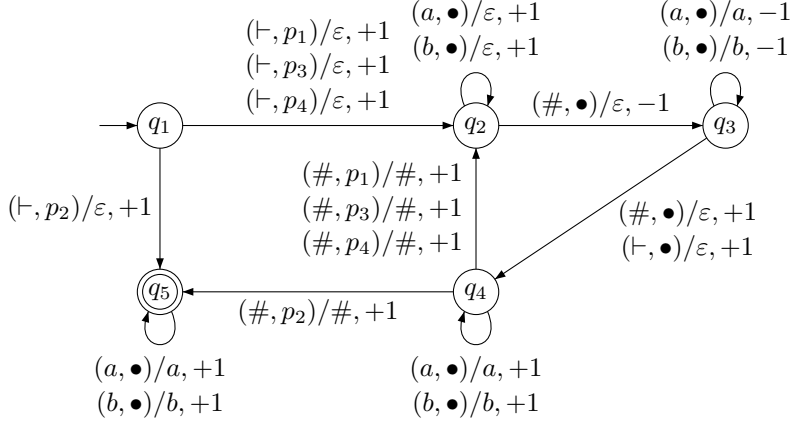
844 **Corollary 25.** Let \mathcal{R} be a finite family of ω -regular languages. There is a BDBA
 845 \mathcal{B} and a tuple of initial sets $(I_R)_{R \in \mathcal{R}}$ such that $R = \mathcal{L}(\mathcal{B}, I_R)$ for all $R \in \mathcal{R}$.

846 3.5. Replacing the look-ahead of an ω -2DMT $_{\text{la}}$ with a BDBA

847 Let $\mathcal{A} = (Q, \Sigma, \Gamma, q_0, \delta, \mathcal{F}, \mathcal{R})$ be an ω -2DMT $_{\text{la}}$. By Corollary 25 there is a
 848 BDBA $\mathcal{B} = (P, \Sigma, \Delta, \text{Fin})$ and a tuple $(I_R)_{R \in \mathcal{R}}$ of initial sets for the finite family
 849 \mathcal{R} of ω -regular languages used as look-ahead by the transducer \mathcal{A} . Recall that
 850 for every pair $(q, a) \in Q \times \Sigma_+$, the subset $\mathcal{R}(q, a)$ of languages $R \in \mathcal{R}$ such that
 851 $\delta(q, a, R)$ is defined forms a partition of Σ^ω . We deduce that $(I_R)_{R \in \mathcal{R}(q, a)}$ is a
 852 partition of P .

853 We construct an ω -2DMT $\tilde{\mathcal{A}} = (Q, \tilde{\Sigma}, \Gamma, q_0, \tilde{\delta}, \mathcal{F})$ without look-ahead over the
 854 extended alphabet $\tilde{\Sigma} = \Sigma \times P$ which is equivalent to \mathcal{A} in some sense made
 855 precise below. Intuitively, in a pair $(a, p) \in \tilde{\Sigma}_+$, the state p of \mathcal{B} gives the
 856 look-ahead information required by \mathcal{A} . Formally, the deterministic transition
 857 function $\tilde{\delta}: Q \times \tilde{\Sigma}_+ \rightarrow Q \times \Gamma^* \times \{-1, +1\}$ is defined as follows: for $q \in Q$ and
 858 $(a, p) \in \tilde{\Sigma}_+$ we let $\tilde{\delta}(q, (a, p)) = \delta(q, a, R)$ for the unique $R \in \mathcal{R}(q, a)$ such that
 859 $p \in I_R$.

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



Let $w = a_1 a_2 a_3 \dots \in \Sigma^\omega$ and let $\mathcal{B}(w) = p_0, a_1, p_1, a_2, p_2, \dots$ be the unique final run of \mathcal{B} on w . We define $\widetilde{\vdash} w = (\vdash, p_0)(a_1, p_1)(a_2, p_2) \dots \in \widetilde{\Sigma}_+^\omega$. We can easily check by induction that the unique run of \mathcal{A} on w

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

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

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

863 where for all $i > 0$ we have $\widetilde{\vdash} w = \widetilde{w}'_i \widetilde{w}''_i$ and $|w'_i| = |\widetilde{w}'_i|$. Indeed, assume
864 that in a configuration $w' q a w''$ with $\vdash w = w' a w''$ the transducer \mathcal{A} takes the
865 transition $q \xrightarrow{(a, R)} (q', \gamma, +1)$ and reaches configuration $w' a q' w''$. Then, $w'' \in R$
866 and the corresponding configuration $\widetilde{w}' q(a, p) \widetilde{w}''$ with $\widetilde{\vdash} w = \widetilde{w}'(a, p) \widetilde{w}''$ and
867 $|w'| = |\widetilde{w}'|$ is such that $p \in I_R$. Therefore, the transducer $\widetilde{\mathcal{A}}$ takes the transition
868 $q \xrightarrow{(a, p)} (q', \gamma, +1)$ and reaches configuration $\widetilde{w}'(a, p) q' \widetilde{w}''$. The proof is similar
869 for backward transitions. We have shown that \mathcal{A} and $\widetilde{\mathcal{A}}$ are equivalent in the
870 following sense:

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

874 3.6. Transition monoid of an ω -2DMT $_{I_a}$

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

881 The ω -2DMT $\widetilde{\mathcal{A}}$ does not use look-ahead, hence, we may use for $\widetilde{\mathcal{A}}$ the classical
882 notion of transition monoid. Actually, in order to handle the Muller acceptance
883 condition of $\widetilde{\mathcal{A}}$, we need a slight extension of the transition monoid defined in
884 Section 2.5. More precisely, the abstraction of a finite word $\tilde{u} \in \widetilde{\Sigma}^+$ will be the
885 set $\widetilde{\text{Tr}}(\tilde{u})$ of tuples (q, d, X, q') with $q, q' \in Q$, $X \subseteq Q$ and $d \in \{\rightarrow, \supseteq, \subseteq, \leftarrow\}$ such

886 that the unique run of $\tilde{\mathcal{A}}$ on \tilde{u} starting in state q on the left of \tilde{u} if $d \in \{\rightarrow, \triangleright\}$
 887 (resp. on the right if $d \in \{\zeta, \leftarrow\}$) exits in state q' on the left of \tilde{u} if $d \in \{\triangleright, \leftarrow\}$
 888 (resp. on the right if $d \in \{\rightarrow, \zeta\}$) and visits the set of states X while in \tilde{u} (i.e.,
 889 including q but not q' unless q' is also visited before the run exits \tilde{u}).

890 For instance, with the automaton $\tilde{\mathcal{A}}$ of Example 26, we have $(q_4, \rightarrow, \{q_2, q_3, q_4\},$
 891 $q_5) \in \tilde{\text{Tr}}(\tilde{u})$ when $\tilde{u} \in ((a, p_1) + (b, p_1))^*(\#, p_1)((a, p_1) + (b, p_1))^*(\#, p_2)$.

892 We denote by $\widetilde{\text{TrM}} = \{\tilde{\text{Tr}}(\tilde{u}) \mid \tilde{u} \in \tilde{\Sigma}^+\} \cup \{\mathbf{1}_{\widetilde{\text{TrM}}}\}$ the transition monoid of
 893 $\tilde{\mathcal{A}}$ with unit $\mathbf{1}_{\widetilde{\text{TrM}}}$. The classical product of the transition monoid of a two-
 894 way automaton [7] is extended by taking the union of the sets X occurring
 895 in a sequence of steps. For instance, if we have steps $(q_0, \rightarrow, X_1, q_1), (q_2, \zeta$
 896 $, X_3, q_3), \dots, (q_{i-1}, \zeta, X_i, q_i)$ in $\tilde{\text{Tr}}(\tilde{u})$ and $(q_1, \triangleright, X_2, q_2), (q_3, \triangleright, X_4, q_4), \dots,$
 897 $(q_i, \rightarrow, X_{i+1}, q_{i+1})$ in $\tilde{\text{Tr}}(\tilde{v})$ then there is a step $(q_0, \rightarrow, X_1 \cup \dots \cup X_{i+1}, q_{i+1})$ in
 898 $\tilde{\text{Tr}}(\tilde{u} \cdot \tilde{v}) = \tilde{\text{Tr}}(\tilde{u}) \cdot \tilde{\text{Tr}}(\tilde{v})$. We denote by $\tilde{\text{Tr}}: \tilde{\Sigma}^* \rightarrow \widetilde{\text{TrM}}$ the canonical morphism.

899 Let $u = a_1 \dots a_n \in \Sigma^+$ be a finite word of length $n > 0$ and let $p \in P$.
 900 We define the sequence of states p_0, p_1, \dots, p_n by $p_n = p$ and for all $0 \leq$
 901 $i < n$ we have $p_i \xleftarrow{a_{i+1}} p_{i+1}$ in \mathcal{B} . Notice that for all infinite words $w \in$
 902 $\mathcal{L}(\mathcal{B}, p)$, the unique run $\mathcal{B}(uw)$ starts with $p_0, a_1, p_1, \dots, a_n, p_n$. We define $\tilde{u}^p =$
 903 $(a_1, p_1)(a_2, p_2) \dots (a_n, p_n) \in \tilde{\Sigma}^+$.

904 We are now ready to define the finite abstraction $\text{Tr}(u)$ of a finite word
 905 $u \in \Sigma^+$ with respect to the pair $(\mathcal{A}, \mathcal{B})$: we let $\text{Tr}(u) = (r^p, b^p, s^p)_{p \in P}$ where
 906 for each $p \in P$, $s^p = \tilde{\text{Tr}}(\tilde{u}^p) \in \widetilde{\text{TrM}}$ is the abstraction of \tilde{u}^p with respect to $\tilde{\mathcal{A}}$,
 907 $r^p \in P$ is the unique state of \mathcal{B} such that $r^p \xleftarrow{u} p$, $b^p = 1$ if the word \tilde{u}^p contains
 908 a final state of \mathcal{B} and $b^p = 0$ otherwise.

909 We define the transition monoid of $(\mathcal{A}, \mathcal{B})$ as the set $\text{TrM} = \{\text{Tr}(u) \mid u \in$
 910 $\Sigma^+\} \cup \{\mathbf{1}_{\text{TrM}}\}$ where $\mathbf{1}_{\text{TrM}}$ is the unit. The product of $\sigma_1 = (r_1^p, b_1^p, s_1^p)_{p \in P}$ and
 911 $\sigma = (r^p, b^p, s^p)_{p \in P}$ is defined to be $\sigma_1 \cdot \sigma = (r_1^p, b_1^p \vee b^p, s_1^p \cdot s^p)_{p \in P}$. We can check
 912 that this product is associative, so that $(\text{TrM}, \cdot, \mathbf{1}_{\text{TrM}})$ is a monoid. Moreover,
 913 let $u, v \in \Sigma^+$ be such that $\text{Tr}(u) = \sigma_1$ and $\text{Tr}(v) = \sigma$. For each $p \in P$, we can
 914 check that $\tilde{u}v^p = \tilde{u}^{r^p} \cdot \tilde{v}^p$. We deduce easily that $\text{Tr}(uv) = \sigma_1 \cdot \sigma = \text{Tr}(u) \cdot \text{Tr}(v)$.
 915 Therefore, $\text{Tr}: \Sigma^* \rightarrow \text{TrM}$ is a morphism.

916 3.7. ω -2DMT_{la} to ω -RTE

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

921 **Theorem 28** (Unambiguous Forest Factorization [21]). *Let $\varphi: \Sigma^* \rightarrow S$ be*
 922 *a morphism to a finite monoid $(S, \cdot, \mathbf{1}_S)$. There is an unambiguous rational*
 923 *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*
 924 *φ -good rational expressions for all $1 \leq k \leq m$.*

925 We will apply this theorem to the morphism $\text{Tr}: \Sigma^* \rightarrow \text{TrM}$ defined in
 926 Section 3.6. We use the unambiguous expression $G = \bigcup_{k=1}^m F_k \cdot G_k^\omega$ as a *guide*
 927 when constructing ω -RTEs corresponding to the ω -2DMT_{la} \mathcal{A} .

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

930 **Lemma 29.** *Let G be an ε -free Tr -good rational expression and let $\text{Tr}(G) =$
 931 $\sigma_G = (r_G^p, b_G^p, s_G^p)_{p \in P}$ be the corresponding element of the transition monoid TrM*

932 of $(\mathcal{A}, \mathcal{B})$. For each state $p \in P$, we can construct a map $C_G^p: s_G^p \rightarrow \text{RTE}$ such
 933 that for each step $x = (q, d, X, q') \in s_G^p$ the following invariants hold:

934 (J₁) $\text{dom}(C_G^p(x)) = \mathcal{L}(G)$,

935 (J₂) for each $u \in \mathcal{L}(G)$, $\llbracket C_G^p(x) \rrbracket(u)$ is the output produced by $\tilde{\mathcal{A}}$ when running
 936 step x on \tilde{u}^p (i.e., running $\tilde{\mathcal{A}}$ on \tilde{u}^p from q to q' following direction d).

937 *Proof.* The proof is by structural induction on the rational expression. For each
 938 subexpression E of G we let $\text{Tr}(E) = \sigma_E = (r_E^p, b_E^p, s_E^p)_{p \in P}$ be the corresponding
 939 element of the transition monoid TrM of $(\mathcal{A}, \mathcal{B})$. We start with atomic regular
 940 expressions. Since G is ε -free and \emptyset -free, we do not need to consider $E = \varepsilon$
 941 or $E = \emptyset$. The construction is similar to the one given in Section 2.6. The
 942 interesting cases are concatenation and Kleene-plus.

943 **atomic** Assume that $E = a \in \Sigma$ is an atomic subexpression. Notice that
 944 $\tilde{a}^p = (a, p)$ for all $p \in P$. Since the ω -2DMT $\tilde{\mathcal{A}}$ is deterministic and
 945 complete, for each state $q \in Q$ we have

- 946 • either $\tilde{\delta}(q, (a, p)) = (q', \gamma, 1)$ and we let $C_a^p((q, \rightarrow, \{q\}, q')) = C_a^p((q, \hookrightarrow$
 947 $, \{q\}, q')) = a ? \gamma : \perp$,
- 948 • or $\tilde{\delta}(q, (a, p)) = (q', \gamma, -1)$ and we let $C_a^p((q, \rhd, \{q\}, q')) = C_a^p((q, \leftarrow$
 949 $, \{q\}, q')) = a ? \gamma : \perp$.

950 Clearly, invariants (J₁) and (J₂) hold for all $x \in s_E^p$.

951 **union** Assume that $E = E_1 \cup E_2$. Since E is good, we deduce that $\sigma_E = \sigma_{E_1} =$
 952 σ_{E_2} . For each $p \in P$ and $x \in s_E^p$ we define $C_E^p(x) = E_1 ? C_{E_1}^p(x) : C_{E_2}^p(x)$.
 953 Since E is unambiguous we have $\mathcal{L}(E_1) \cap \mathcal{L}(E_2) = \emptyset$. As in Section 2.6 we
 954 can prove easily that invariants (J₁) and (J₂) hold for all $x \in s_E^p$.

955 **concatenation** Assume that $E = E_1 \cdot E_2$ is a concatenation. Since E is good,
 956 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
 957 $s_E^p = s_{E_1}^{p_1} \cdot s_{E_2}^p$. Let $x \in s_E^p$.

If $x = (q, \rightarrow, X, q')$ then, by definition of the product in the transition
 monoid TrM , there is a unique sequence of steps $x_1 = (q, \rightarrow, X_1, q_1)$,
 $x_2 = (q_1, \rhd, X_2, q_2)$, $x_3 = (q_2, \hookrightarrow, X_3, q_3)$, $x_4 = (q_3, \rhd, X_4, q_4)$, \dots , $x_i =$
 $(q_{i-1}, \hookrightarrow, X_i, q_i)$, $x_{i+1} = (q_i, \rightarrow, X_{i+1}, q')$ with $i \geq 1$, $x_1, x_3, \dots, x_i \in s_{E_1}^{p_1}$
 and $x_2, x_4, \dots, x_{i+1} \in s_{E_2}^p$ and $X = X_1 \cup \dots \cup X_{i+1}$ (see Figure 10). We
 define

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

958 Notice that when $i = 1$ we have $C_E^p(x) = C_{E_1}^{p_1}(x_1) \boxtimes C_{E_2}^p(x_2)$ with $x_2 =$
 959 $(q_1, \rightarrow, X_2, q')$.

960 The concatenation $\mathcal{L}(E) = \mathcal{L}(E_1) \cdot \mathcal{L}(E_2)$ is unambiguous. Therefore, for
 961 all $y \in s_{E_1}^{p_1}$ and $z \in s_{E_2}^p$, using (J₁) for E_1 and E_2 , we obtain $\text{dom}(C_{E_1}^{p_1}(y) \boxtimes$
 962 $C_{E_2}^p(z)) = \mathcal{L}(E)$. We deduce that $\text{dom}(C_E(x)) = \mathcal{L}(E)$ and (J₁) holds for
 963 E and $x = (q, \rightarrow, X, q')$.

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

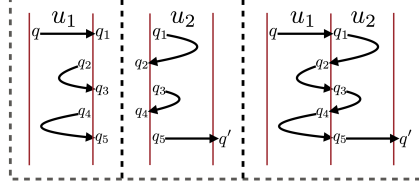


Figure 10: In the concatenation $E = E_1 \cdot E_2$, a step $x = (q, \rightarrow, X, q') \in s_E^p$ on some $u_1 u_2$ with $u_1 \in E_1$ and $u_2 \in E_2$, is obtained by composing the following steps alternatively from $s_{E_1}^{p_1}$ and $s_{E_2}^p$ for a unique state p_1 : $x_1 = (q, \rightarrow,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

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 , \dots , until x_{i+1} on \widetilde{u}_2^p . Using (J₂) 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

$$\begin{aligned} \llbracket C_{E_1}^{p_1}(x_1) \rrbracket(u_1) \cdot \llbracket C_{E_2}^p(x_2) \rrbracket(u_2) \cdots \llbracket C_{E_1}^{p_1}(x_i) \rrbracket(u_1) \cdot \llbracket C_{E_2}^p(x_{i+1}) \rrbracket(u_2) \\ = \llbracket C_E^p(x) \rrbracket(u) \end{aligned}$$

964 Therefore, (J₂) holds for E and step $x = (q, \rightarrow, X, q')$. The proof is obtained
 965 mutatis mutandis for the other cases $x = (q, \succ, X, q')$ or $x = (q, \zeta, X, q')$
 966 or $x = (q, \leftarrow, X, q')$.

967 **Kleene-plus** Assume that $E = F^+$. Since E is good, we deduce that $\sigma_E =$
 968 $\sigma_F = \sigma = (r^p, b^p, s^p)_{p \in P}$ is an idempotent of the transition monoid TrM .
 969 Notice that for all $p \in P$, since σ is an idempotent, we have $r^{r^p} = r^p$.

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

- 971 • If $x = (q, \succ, X, q')$. Since F^+ is unambiguous, a word $u \in \mathcal{L}(F^+)$
 972 admits a unique factorization $u = u_1 u_2 \cdots u_n$ with $n \geq 1$ and $u_i \in$
 973 $\mathcal{L}(F)$. Now, $\text{Tr}(u_i) = \sigma$ for all $1 \leq i \leq n$ and since $p = r^p$ we deduce
 974 that $\widetilde{u}^p = \widetilde{u}_1^p \widetilde{u}_2^p \cdots \widetilde{u}_n^p$. Since $x = (q, \succ, X, q') \in s^p$, the unique
 975 run ρ of $\widetilde{\mathcal{A}}$ starting in state q on the left of \widetilde{u}_1^p exits on the left in
 976 state q' . Therefore, the unique run of $\widetilde{\mathcal{A}}$ starting in state q on the
 977 left of \widetilde{u}^p only visits \widetilde{u}_1^p and is actually ρ itself. Therefore, we set
 978 $C_E^p(x) = C_F^p(x) \boxtimes (F^* ? \varepsilon : \perp)$ and we can easily check that (J₁–J₂)
 979 are satisfied.
 - 980 • Similarly for $x = (q, \zeta, X, q')$ we set $C_E^p(x) = (F^* ? \varepsilon : \perp) \boxtimes C_F^p(x)$.
 - 981 • If $x = (q, \rightarrow, X, q')$. Since σ is an idempotent, we have $x \in s^p \cdot s^p$. We
 982 distinguish two cases depending on whether the step $y \in s^p$ starting
 983 in state q' from the left goes to the right or goes back to the left.
 984 First, if $y = (q', \rightarrow, X_2, q_2) \in s^p$ goes to the right. Since s^p is an
 985 idempotent, following x in $s^p \cdot s^p$ is the same as following x in (the
 986 first) s^p and then y in (the second) s^p . Therefore, we must have
 987 $q_2 = q'$ and $X_2 \subseteq X$. In this case, we set $C_E^p(x) = F ? C_F^p(x) :$
 988 $(C_F^p(x) \boxtimes (C_F^p(y))^{\boxplus})$.
- Second, if $y = (q', \succ, X_2, q_2) \in s^p$ goes to the left. Since s^p is an
 idempotent, there exists a unique sequence of steps in s^p : $x_1 = x$,
 $x_2 = y$, $x_3 = (q_2, \zeta, X_3, q_3)$, $x_4 = (q_3, \succ, X_4, q_4)$, \dots , $x_i = (q_{i-1}, \zeta$

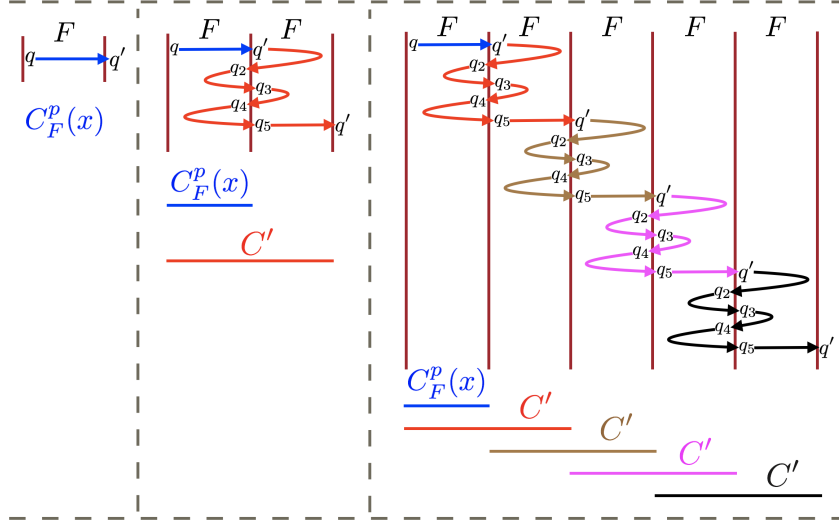


Figure 11: In the Kleene-plus $E = F^+$, a step $x = (q, \rightarrow, X, q') \in s_E^p$ on some $u = u_1 u_2 \cdots u_n$ with $u_\ell \in \mathcal{L}(F)$ is obtained by composing the following steps in s_F^p : $x_1 = x$, $x_2 = (q', \supset, X_2, q_2)$, $x_3 = (q_2, \subset, X_3, q_3)$, $x_4 = (q_3,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

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

$$C_E^p(x) = (C_F^p(x) \boxtimes (F^* ? \varepsilon : \perp)) \odot [F, C']^{2\boxplus}$$

$$C' = ((F ? \varepsilon : \perp) \boxtimes C_F^p(x_2)) \odot (C_F^p(x_3) \boxtimes C_F^p(x_4)) \odot \cdots \odot (C_F^p(x_i) \boxtimes C_F^p(x_{i+1}))$$

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

- 991 • If $x = (q, \leftarrow, X, q')$, the proof is obtained mutatis mutandis, using the
992 backward unambiguous (2-chained) Kleene-plus C^{\boxminus} and $[K, C]^{2\boxminus}$.

993 Now, we consider $p \in P$ with $r^p \neq p$. We let $p' = r^p$. We have already
994 noticed that since σ is idempotent we have $r^{p'} = p'$. Consider a word
995 $u \in \mathcal{L}(F^+)$. Since F^+ is unambiguous, u admits a unique factorization
996 $u = u_1 \cdots u_{n-1} u_n$ with $n \geq 1$ and $u_i \in \mathcal{L}(F)$. Now, $\text{Tr}(u_i) = \sigma$ for all $1 \leq$
997 $i \leq n$. Using $r^p = p'$ and $r^{p'} = p'$ we deduce that $\widetilde{u}^p = \widetilde{u}_1^{p'} \cdots \widetilde{u}_{n-1}^{p'} \widetilde{u}_n^p$.
998 So when $n > 1$, the expression C_E^p that we need to compute is like the
999 concatenation of $C_E^{p'}$ on the first $n - 1$ factors with C_F^p on the last factor.
1000 Since $r^{p'} = p'$ we have already seen how to compute $C_E^{p'}$. We also know
1001 how to handle concatenation. So it should be no surprise that we can
1002 compute C_E^p when $p \neq r^p$. We define now formally $C_E^p(x)$ for $x \in s^p$.

- 1003 • If $x = (q, \supset, X, q') \in s^p$. There are two cases depending on whether
1004 the step $y \in s^{p'}$ starting in state q from the left goes back to the left
1005 or goes to the right.
If it goes back to the left, then $y = (q, \supset, X, q') = x$ since $s^p = s^{p'} \cdot s^p$

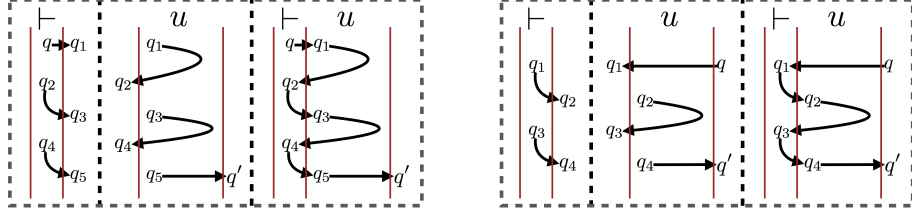


Figure 12: (Left) Given a look-ahead $p \in P$, a step (q, \rightarrow, q') of $\vdash u$ for some u with $\text{Tr}(u) = (r^p, b^p, s^p)_{p \in P}$, is obtained by composing the following steps alternatively from s^p and s^p : $x_1 = (q, \rightarrow, q_1)$, $x_2 = (q_1, \rightarrow, X_2, q_2)$, $x_3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

(recall that σ is idempotent) and we define

$$C_E^p(x) = F ? C_F^p(x) : (C_E^{p'}(x) \boxtimes (F^+ ? \varepsilon : \perp)).$$

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

$$C' = (C_E^{p'}(x_1) \boxtimes C_F^p(x_2)) \odot \dots \odot (C_E^{p'}(x_{i-2}) \boxtimes C_F^p(x_{i-1})) \odot (C_E^{p'}(x_i) \boxtimes (F ? \varepsilon : \perp)).$$

1006

We can check that (J_1-J_2) are satisfied for (E, p, x) .

1007

- 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.

1008

1009

If it goes to the left, then $y = (q', \leftarrow, X', q')$ with $X' \subseteq X$ and we define

$$C_E^p(x) = F ? C_F^p(x) : (C_E^{p'}(y) \overleftarrow{\boxtimes} C_F^p(x)).$$

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

$$C' = (C_E^{p'}(x_2) \overleftarrow{\boxtimes} C_F^p(x_1)) \odot \dots \odot (C_E^{p'}(x_{i-1}) \overleftarrow{\boxtimes} C_F^p(x_{i-2})) \odot (C_E^{p'}(x_{i+1}) \overleftarrow{\boxtimes} C_F^p(x_i)).$$

1010

We can check that (J_1-J_2) are satisfied for (E, p, x) .

1011

- The cases $x = (q, \rightarrow, X, q') \in s^p$ and $x = (q, \leftarrow, X, q') \in s^p$ can be handled similarly. \square

1012

1013

We now define RTEs corresponding to the left part of the computation of the ω -2DMT_{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} .

1014

1015

1016

1017 **Lemma 30.** *Let F be an ε -free Tr -good rational expression. For each state*
1018 *$p \in P$ and $q \in Q$, there is a unique state $q' \in Q$ and RTEs $C_{\vdash F}^p((q, \rightarrow, q'))$ (resp.*
1019 *$C_{\vdash F}^p((q, \zeta, q'))$) such that the following invariants hold:*
1020 *(K₁) $\text{dom}(C_{\vdash F}^p((q, \rightarrow, q'))) = \mathcal{L}(F)$ (resp. $\text{dom}(C_{\vdash F}^p((q, \zeta, q'))) = \mathcal{L}(F)$),*
1021 *(K₂) for each $u \in \mathcal{L}(F)$, $\llbracket C_{\vdash F}^p((q, \rightarrow, q')) \rrbracket(u)$ (resp. $\llbracket C_{\vdash F}^p((q, \zeta, q')) \rrbracket(u)$) is the*
1022 *output produced by $\tilde{\mathcal{A}}$ on $\widetilde{\vdash}^p u$ when starting on the left (resp. right) in state*
1023 *q until it exists on the right in state q' .*

1024 *Proof.* Let $\sigma = (r^p, b^p, s^p)_{p \in P} = \text{Tr}(F)$. We fix some state $p \in P$. For all words
1025 $u \in \mathcal{L}(F)$, we have $\widetilde{\vdash}^p u = (\vdash, r^p) \tilde{u}^p$. Let s_{\vdash}^p be the set of steps (q, \rightarrow, q') , (q, ζ, q')
1026 such that $\tilde{\delta}(q, (\vdash, r^p)) = (q', \gamma_q^p, +1)$ in $\tilde{\mathcal{A}}$.

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

$$C_{\vdash F}^p((q, \rightarrow, q')) = \gamma_q^p \odot C_F^p(x_2) \odot \gamma_{q_2}^p \odot C_F^p(x_4) \odot \dots \odot \gamma_{q_{i-1}}^p \odot C_F^p(x_{i+1}).$$

1027 Using Lemma 29, we can show that $\mathcal{L}(F) = \text{dom}(C_{\vdash F}^p((q, \rightarrow, q')))$ and also that
1028 for each $u \in \mathcal{L}(F)$, $\llbracket C_{\vdash F}^p((q, \rightarrow, q')) \rrbracket(u)$ is the output produced by $\tilde{\mathcal{A}}$ on $\widetilde{\vdash}^p u$
1029 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 $x_1 = (q, \leftarrow, X_1, q_1)$,
 $x_2 = (q_1, \zeta, q_2)$, $x_3 = (q_2, \supset, X_3, q_3)$, $x_4 = (q_3, \zeta, q_4)$, \dots , $x_i = (q_{i-1}, \zeta, q_i)$,
 $x_{i+1} = (q_i, \rightarrow, X_{i+1}, q')$ with $i \geq 2$, $x_2, x_4, \dots, x_i \in s_{\vdash}^p$ and $x_1, x_3, \dots, x_{i+1} \in s^p$
(see Figure 12 right). We define

$$C_{\vdash F}^p((q, \zeta, q')) = C_F^p(x_1) \odot \gamma_{q_1}^p \odot C_F^p(x_3) \odot \gamma_{q_3}^p \odot \dots \odot \gamma_{q_{i-1}}^p \odot C_F^p(x_{i+1}).$$

1030 Using Lemma 29, we can show that $\mathcal{L}(F) = \text{dom}(C_{\vdash F}^p((q, \zeta, q')))$ and also that
1031 for each $u \in \mathcal{L}(F)$, $\llbracket C_{\vdash F}^p((q, \zeta, q')) \rrbracket(u)$ is the output produced by $\tilde{\mathcal{A}}$ on $\widetilde{\vdash}^p u$
1032 when starting on the right in state q until it exists on the right in state q' . \square

1033 **Lemma 31.** *Let $F \cdot G^\omega$ be an unambiguous rational expression such that F and*
1034 *G are ε -free Tr -good rational expressions and $\text{Tr}(G) = \sigma = (r^p, b^p, s^p)_{p \in P}$ is an*
1035 *idempotent in the transition monoid TrM of $(\mathcal{A}, \mathcal{B})$. We can construct an ω -RTE*
1036 *C_{FG^ω} such that $\text{dom}(C_{FG^\omega}) = \mathcal{L}(FG^\omega) \cap \text{dom}(\mathcal{A})$ and for each $w \in \text{dom}(C_{FG^\omega})$,*
1037 *$\llbracket C_{FG^\omega} \rrbracket(w) = \llbracket \mathcal{A} \rrbracket(w)$.*

1038 *Proof.* We first show that there exists one and only one state $p \in P$ such that
1039 $r^p = p$ and $b^p = 1$. For the existence, consider a word $w = u_1 u_2 u_3 \dots \in \mathcal{L}(FG^\omega)$
1040 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
1041 unique final run of \mathcal{B} over w : $p_0, u_1, p_1, u_2, p_2, \dots$. Let us show first that $p_n = p_1$
1042 for all $n \geq 1$. Since σ is idempotent, we have $\text{Tr}(u_2 \dots u_{n+1}) = \text{Tr}(u_{n+1})$. Since
1043 $p_1 \xleftarrow{u_2 \dots u_{n+1}} p_{n+1}$ and $p_n \xleftarrow{u_{n+1}} p_{n+1}$, we deduce that $p_1 = r^{p_{n+1}} = p_n$. This
1044 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
1045 w is $p_0, u_1, p, u_2, p, \dots$. Now, for all $n \geq 2$ we have $\text{Tr}(u_n) = \sigma$ and we deduce
1046 that $p \xleftarrow{u_n} p$ visits a final state from Fin iff $b^p = 1$. Since the run is accepting,
1047 we deduce that indeed $b^p = 1$. To prove the unicity, let $p \in P$ with $p = r^p$ and
1048 $b^p = 1$. Let $v \in \mathcal{L}(G)$. We have $p \xleftarrow{v} p$ and this subrun visits a final state from
1049 Fin . Therefore, $p, v, p, v, p, v, p, \dots$ is a final run of \mathcal{B} on v^ω . Since \mathcal{B} is BDBA,
1050 there is a unique final run of \mathcal{B} on v^ω , which proves the unicity of p .

1051 We apply Lemma 30. We denote by $s'_{\leftarrow F}$ the set of triples $(q, d, q') \in Q \times \{\rightarrow$
1052 $, \leftarrow\} \times Q$ such that the RTE $C'_{\leftarrow F}(q, d, q')$ is defined.

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

$$C_1 = (C'_{\leftarrow F}(x'_1) \boxtimes C_G^p(x'_2)) \odot (C'_{\leftarrow F}(x'_3) \boxtimes C_G^p(x'_4)) \odot \dots \odot (C'_{\leftarrow F}(x'_i) \boxtimes C_G^p(x'_{i+1}))$$

$$C_2 = C_1 \boxtimes (G^\omega ? \varepsilon : \perp).$$

1053 We have $\text{dom}(C_1) = FG$ and $\widetilde{\vdash u_1 u_2}^p = \widetilde{\vdash u_1}^p \widetilde{u_2}^p$ for all $u_1 \in F$ and $u_2 \in G$.

1054 Moreover, $\llbracket C_1 \rrbracket(u_1 u_2)$ is the output produced by $\tilde{\mathcal{A}}$ on $\widetilde{\vdash u_1 u_2}^p$ when starting on
1055 the left in the initial state q_0 until it exists on the right in state q . Now, C_2 is
1056 an ω -RTE with $\text{dom}(C_2) = FG^\omega$ and for all $w = u_1 u_2 u_3 \dots \in FG^\omega$ with $u_1 \in F$
1057 and $u_n \in G$ for all $n > 1$, we have $\llbracket C_2 \rrbracket(w) = \llbracket C_1 \rrbracket(u_1 u_2) \in \Gamma^*$.

Now, we distinguish two cases. First, assume that there is a step $x =$
 $(q, \rightarrow, X, q') \in s^p$. Since σ is idempotent, so is s^p , and since $x'_{i+1} = (q'_i, \rightarrow$
 $, X'_{i+1}, q) \in s^p$ we deduce that $q' = q$. Therefore, the unique run of $\tilde{\mathcal{A}}$ on
 $\widetilde{\vdash w} = \widetilde{\vdash u_1}^p \widetilde{u_2}^p \widetilde{u_3}^p \dots$ follows the steps $x'_1 x'_2 \dots x'_i x'_{i+1} x x x \dots$. 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 $FG^\omega \cap \text{dom}(\mathcal{A}) = \emptyset$ and we
set $C_{FG^\omega} = \perp$. Now, if $X \in \mathcal{F}$ we have $FG^\omega \subseteq \text{dom}(\mathcal{A})$ and we set

$$C_{FG^\omega} = C_2 \odot ((FG ? \varepsilon : \perp) \boxtimes C_G^p(x)^\omega).$$

We have $\text{dom}(C_{FG^\omega}) = FG^\omega$ and for all $w = u_1 u_2 u_3 \dots \in FG^\omega$ with $u_1 \in F$
and $u_n \in G$ for all $n > 1$, we have

$$\llbracket C_{FG^\omega} \rrbracket(w) = \llbracket C_1 \rrbracket(u_1 u_2) \llbracket C_G^p(x) \rrbracket(u_3) \llbracket C_G^p(x) \rrbracket(u_4) \dots.$$

1058 By (J₂), we know that for all $n \geq 3$, $\llbracket C_G^p(x) \rrbracket(u_n)$ is the output produced by
1059 $\tilde{\mathcal{A}}$ when running step $x = (q, \rightarrow, X, q)$ on $\widetilde{u_n}^p$. We deduce that $\llbracket C_{FG^\omega} \rrbracket(w) =$
1060 $\llbracket \tilde{\mathcal{A}} \rrbracket(\widetilde{\vdash w}) = \llbracket \mathcal{A} \rrbracket(w)$ as desired.

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

$$C_3 = ((G ? \varepsilon : \perp) \boxtimes C_G^p(x_1)) \odot (C_G^p(x_2) \boxtimes C_G^p(x_3)) \odot \dots \odot$$

$$(C_G^p(x_j) \boxtimes C_G^p(x_{j+1}))$$

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

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

$$\llbracket C_{FG^\omega} \rrbracket(w) = \llbracket C_1 \rrbracket(u_1u_2) \llbracket C_3 \rrbracket(u_2u_3) \llbracket C_3 \rrbracket(u_3u_4) \cdots .$$

1061 Using (J₂), we can check that this is the output produced by $\tilde{\mathcal{A}}$ when running
1062 on $\tilde{\vdash}w$. We deduce that $\llbracket C_{FG^\omega} \rrbracket(w) = \llbracket \tilde{\mathcal{A}} \rrbracket(\tilde{\vdash}w) = \llbracket \mathcal{A} \rrbracket(w)$ as desired. \square

1063 We are now ready to prove that ω -2DMT_{la} are no more expressive than
1064 ω -RTEs.

Proof of Theorem 21 (2). We use the notations of the previous sections, in particular for the ω -2DMT_{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 $\text{Tr}(G_k) = \{\sigma_{G_k}\}$. For each $1 \leq k \leq m$, let $C_k = C_{F_k G_k^\omega}$ be the ω -RTE given by Lemma 31. We define the final ω -RTE as

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

1065 From Lemma 31, we can easily check that $\text{dom}(C) = \text{dom}(\mathcal{A})$ and $\llbracket C \rrbracket(w) =$
1066 $\llbracket \mathcal{A} \rrbracket(w)$ for all $w \in \text{dom}(C)$. \square

1067 4. Conclusion

1068 The main contribution of the paper is to give a characterisation of regular
1069 string transductions using some combinators, giving rise to regular transducer
1070 expressions (RTE). Our proof uniformly works well for finite and infinite string
1071 transformations. RTE are a succinct specification mechanism for regular transfor-
1072 mations just like regular expressions are for regular languages. It is worthwhile
1073 to consider extensions of our technique to regular tree transformations, or in
1074 other settings where more involved primitives such as sorting or counting are
1075 needed. The minimality of our combinators in achieving expressive completeness,
1076 as well as computing complexity measures for the conversion between RTEs and
1077 two-way transducers are open.

1078 [1] Rajeev Alur and Pavol Černý. Expressiveness of streaming string transduc-
1079 ers. In Kamal Lodaya and Meena Mahajan, editors, *30th IARCS Annual*
1080 *Conference on Foundations of Software Technology and Theoretical Com-*
1081 *puter Science (FSTTCS 2010)*, volume 8 of *Leibniz International Proceedings*
1082 *in Informatics (LIPIcs)*, pages 1–12. Schloss Dagstuhl–Leibniz-Zentrum fuer
1083 Informatik, 2010.

1084 [2] Rajeev Alur and Loris D’Antoni. Streaming tree transducers. *J. ACM*,
1085 64(5):31:1–31:55, 2017.

1086 [3] Rajeev Alur, Loris D’Antoni, and Mukund Raghothaman. DReX: A declara-
1087 tive language for efficiently evaluating regular string transformations. In Sri-
1088 ram K. Rajamani and David Walker, editors, *42nd Annual ACM SIGPLAN-*
1089 *SIGACT Symposium on Principles of Programming Languages - POPL’15*,
1090 pages 125–137. ACM Press, 2015.

- 1091 [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.
- 1092
1093
1094
- 1095 [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.
- 1096
1097
1098
1099
1100
- 1101 [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.
- 1102
1103
1104
1105
- 1106 [7] Jean-Camille Birget. Concatenation of inputs in a two-way automaton. *Theoretical Computer Science*, 63(2):141–156, 1989.
- 1107
- 1108 [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.
- 1109
1110
1111
1112
- 1113 [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.
- 1114
1115
1116
- 1117 [10] Olivier Carton and Max Michel. Unambiguous Büchi automata. *Theoretical Computer Science*, 297(1-3):37–81, Mar 2003.
- 1118
- 1119 [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.
- 1120
1121
1122
- 1123 [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.
- 1124
1125
- 1126 [13] Thomas Colcombet. The factorisation forest theorem. To appear in Handbook “Automata: from Mathematics to Applications”, 2013.
- 1127
- 1128 [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.
- 1129
1130
1131

- 1132 [15] Vrunda Dave, Paul Gastin, and ShankaraNarayanan Krishna. Regular
1133 Transducer Expressions for Regular Transformations. In *Proceedings of*
1134 *the 33rd Annual ACM/IEEE Symposium on Logic In Computer Science*
1135 *(LICS'18)*, pages 315–324, Oxford, UK, July 2018. ACM Press.
- 1136 [16] Vrunda Dave, Paul Gastin, and ShankaraNarayanan Krishna. Regular
1137 transducer expressions for regular transformations. *CoRR*, abs/1802.02094,
1138 2018.
- 1139 [17] Vrunda Dave, Shankara Narayanan Krishna, and Ashutosh Trivedi. FO-
1140 definable transformations of infinite strings. In Akash Lal, S. Akshay, Saket
1141 Saurabh, and Sandeep Sen, editors, *36th IARCS Annual Conference on*
1142 *Foundations of Software Technology and Theoretical Computer Science*
1143 *(FSTTCS 2016)*, volume 65 of *Leibniz International Proceedings in Informatics*
1144 *(LIPIcs)*, pages 12:1–12:14. Schloss Dagstuhl–Leibniz-Zentrum fuer
1145 Informatik, 2016.
- 1146 [18] Manfred Droste, Werner Kuich, and Heiko Vogler. *Handbook of Weighted*
1147 *Automata*. Springer Publishing Company, 1st edition, 2009.
- 1148 [19] Joost Engelfriet and Hendrik Jan Hoogeboom. MSO definable string trans-
1149 ductions and two-way finite-state transducers. *ACM Transactions on Com-*
1150 *putational Logic*, 2(2):216–254, Apr 2001.
- 1151 [20] Emmanuel Filiot, Shankara Narayanan Krishna, and Ashutosh Trivedi.
1152 First-order Definable String Transformations. In Venkatesh Raman and S. P.
1153 Suresh, editors, *34th International Conference on Foundation of Software*
1154 *Technology and Theoretical Computer Science (FSTTCS 2014)*, volume 29
1155 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 147–159.
1156 Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2014.
- 1157 [21] Paul Gastin and ShankaraNarayanan Krishna. Unambiguous forest factor-
1158 ization. *CoRR*, abs/1810.07285, 2018.
- 1159 [22] J. C. Shepherdson. The reduction of two-way automata to one-way automata.
1160 *IBM Journal of Research and Development*, 3(2):198–200, 1959.
- 1161 [23] Imre Simon. Factorization forests of finite height. *Theoretical Computer*
1162 *Science*, 72(1):65–94, Apr 1990.
- 1163 [24] Thomas Wilke. Backward deterministic Büchi automata on infinite words.
1164 In *37th IARCS Annual Conference on Foundations of Software Technology*
1165 *and Theoretical Computer Science (FSTTCS'17)*, volume 93 of *LIPIcs*,
1166 pages 6:1–6:9. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.

1167 Appendix A. 2DFT to RTE: A practical example

1168 We show in this section how one computes an RTE equivalent to the 2DFT
1169 \mathcal{A} of Figure 2.

- 1170 1. We work with the morphism $\text{Tr}: \Sigma^* \rightarrow \text{TrM}$ which maps words $w \in \Sigma^*$ to
1171 the transition monoid TrM of \mathcal{A} . An element $X \in \text{TrM}$ is a set consisting of
1172 triples (p, d, q) , where d is a direction $\{\succ, \zeta, \rightarrow, \leftarrow\}$. Given a word $w \in \Sigma^*$,
1173 a triple $(p, \succ, q) \in \text{Tr}(w)$ iff when starting in state p on the left most symbol
1174 of w , the run of \mathcal{A} leaves w on the left in state q . The other directions ζ
1175 (start at the rightmost symbol of w in state p and leave w on the right
1176 in state q), \leftarrow and \rightarrow are similar. In general, we have $w \in \text{dom}(\mathcal{A})$ iff on
1177 input $\vdash w \dashv$, starting on \vdash in the initial state of \mathcal{A} , the run exits on the
1178 right of \dashv in some final state of \mathcal{A} . With the automaton \mathcal{A} of Figure 2 we
1179 have $w \in \text{dom}(\mathcal{A})$ iff $(q_0, \rightarrow, q_2) \in \text{Tr}(w)$.
- 1180 2. For each $X \in \text{TrM}$ such that $(q_0, \rightarrow, q_2) \in X$, we find an RTE C_X whose
1181 domain is $\text{Tr}^{-1}(X)$ and such that $\llbracket \mathcal{A} \rrbracket(w) = \llbracket C_X \rrbracket(w)$ for all $w \in \text{Tr}^{-1}(X)$.
1182 The RTE corresponding to $\llbracket \mathcal{A} \rrbracket$ is the disjoint union of all these RTEs and is
1183 written using the if-then-else construct iterating over for all such elements X .
1184 For instance, if the monoid elements containing (q_0, \rightarrow, q_2) are X_1, X_2, X_3
1185 then we set $C = \text{Tr}^{-1}(X_1) ? C_{X_1} : (\text{Tr}^{-1}(X_2) ? C_{X_2} : (\text{Tr}^{-1}(X_3) ? C_{X_3} : \perp))$
1186 where \perp stands for a nowhere defined function, i.e., $\text{dom}(\perp) = \emptyset$.
- 1187 3. Consider the language $L = (ba^+)^+b \subseteq \text{dom}(\mathcal{A})$. Notice that the regular
1188 expression $(ba^+)^+b$ is not “good”. For instance, condition (ii) is violated
1189 since $\text{Tr}(bab) \neq \text{Tr}(babab)$. Indeed, we can see in Figure A.13 that if we
1190 start on the right of bab in state q_3 then we exit on the left in state q_5 :
1191 $(q_3, \leftarrow, q_5) \in \text{Tr}(bab)$. On the other hand, if we start on the right of $babab$
1192 in state q_3 then we exit on the right in state q_2 : $(q_3, \zeta, q_2) \in \text{Tr}(babab)$.
1193 Also, $(q_5, \rightarrow, q_1) \in \text{Tr}(bab)$ while $(q_5, \rightarrow, q_2) \in \text{Tr}(babab)$. It can be seen
1194 that $\text{Tr}(a)^2$ is an idempotent, hence $\text{Tr}(a^+) = \text{Tr}(a)$. We deduce also
1195 $\text{Tr}(ba^+b) = \text{Tr}(bab)^3$. Finally, we have $\text{Tr}((ba^+)^n b) = \text{Tr}(babab)^4$ for all
1196 $n \geq 2$. Therefore, to obtain the RTE corresponding to L , we compute
1197 RTEs corresponding to ba^+b and $(ba^+)^+ba^+b$ satisfying conditions (i) and
1198 (ii) of “good” rational expressions.
- 1199 4. While ba^+b is good since $\text{Tr}(a)$ is an idempotent, $(ba^+)^+ba^+b$ is not good,
1200 the reason being that $\text{Tr}(ba^+)$ is not an idempotent. We can check that
1201 $\text{Tr}(ba^+ba^+)^5$ is still not idempotent, while $\text{Tr}((ba^+)^i) = \text{Tr}((ba^+)^3)$ for
1202 all $i \geq 3$, (see Figure A.13: we only need to argue for $(q_0, \rightarrow, q_3), (q_5, \rightarrow$
1203 $, q_3)$ and (q_6, \rightarrow, q_3) in $\text{Tr}((ba^+)^i)$, $i \geq 3$, all other entries trivially carry
1204 over). In particular, $\text{Tr}((ba^+)^3)$ is an idempotent⁶. Thus, to compute
1205 the RTE for $L = (ba^+)^+b$, we consider the RTEs corresponding to the

² $\text{Tr}(a) = \{(q_1, \rightarrow, q_1), (q_1, \zeta, q_1), (q_2, \rightarrow, q_3), (q_2, \zeta, q_3), (q_3, \rightarrow, q_3), (q_3, \zeta, q_3),$
 $(q_4, \leftarrow, q_4), (q_4, \succ, q_4), (q_5, \leftarrow, q_5), (q_5, \succ, q_5), (q_6, \rightarrow, q_6), (q_6, \zeta, q_6)\}$

³ $\text{Tr}(ba^+b) = \{(q_0, \rightarrow, q_2), (q_0, \zeta, q_1), (q_1, \succ, q_5), (q_1, \zeta, q_2), (q_2, \succ, q_4), (q_2, \leftarrow, q_5),$
 $(q_3, \succ, q_4), (q_3, \leftarrow, q_5), (q_4, \succ, q_5), (q_4, \zeta, q_1), (q_5, \rightarrow, q_1), (q_5, \zeta, q_6), (q_6, \rightarrow, q_2), (q_6, \zeta, q_1)\}$

⁴ $\text{Tr}(ba^+ba^+b) = \{(q_0, \rightarrow, q_2), (q_0, \zeta, q_1), (q_1, \succ, q_5), (q_1, \zeta, q_2), (q_2, \succ, q_4), (q_2, \zeta, q_2),$
 $(q_3, \succ, q_4), (q_3, \zeta, q_2), (q_4, \succ, q_5), (q_4, \zeta, q_1), (q_5, \rightarrow, q_2), (q_5, \zeta, q_6), (q_6, \rightarrow, q_2), (q_6, \zeta, q_1)\}$

⁵ $\text{Tr}(ba^+ba^+)^3 = \{(q_0, \rightarrow, q_3), (q_1, \succ, q_5), (q_1, \zeta, q_1), (q_2, \succ, q_4), (q_2, \zeta, q_3), (q_3, \succ, q_4),$
 $(q_3, \zeta, q_3), (q_4, \succ, q_5), (q_4, \zeta, q_1), (q_5, \rightarrow, q_1), (q_5, \zeta, q_6), (q_6, \rightarrow, q_3), (q_6, \zeta, q_6)\}$

⁶ $\text{Tr}((ba^+)^3) = \{(q_0, \rightarrow, q_3), (q_1, \succ, q_5), (q_1, \zeta, q_1), (q_2, \succ, q_4), (q_2, \zeta, q_3), (q_3, \succ, q_4),$
 $(q_3, \zeta, q_3), (q_4, \succ, q_5), (q_4, \zeta, q_1), (q_5, \rightarrow, q_3), (q_5, \zeta, q_6), (q_6, \rightarrow, q_3), (q_6, \zeta, q_6)\}$

1206
1207

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

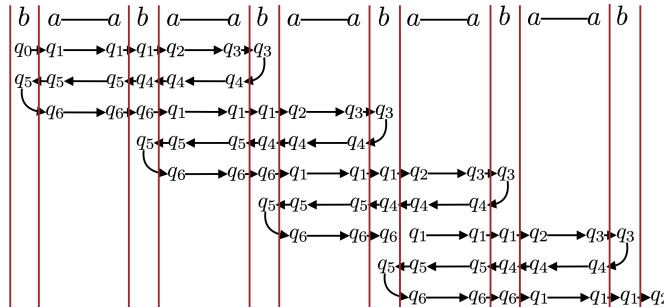


Figure A.13: Run of \mathcal{A} on an input word in $(ba^+)^+b$.

1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225

- We define by induction, for each “good” expression E and “step” $x = (p, d, q)$ in the monoid element $X = \text{Tr}(E)$ associated with E , an RTE $C_E(x)$ whose domain is E and, given a word $w \in E$, it computes $\llbracket C_E(x) \rrbracket(w)$ 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 $C_a(q_5, \leftarrow, q_5) = (a ? b : \perp)$. The if-then-else ensures that the domain is a . Similarly, we get the RTE associated with all atomic expressions and steps. For instance, $C_b(q_1, \rightarrow, q_2) = (b ? \varepsilon : \perp) = C_b(q_3, \triangleright, q_4)$. For $u, v \in \Sigma^*$, we introduce the macro $u/v = u ? v : \perp$. We have $\text{dom}(u/v) = \{u\}$ and $\llbracket u/v \rrbracket(u) = v$.

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 $C_{a^+}(q_5, \leftarrow, q_5) = (C_a(q_5, \leftarrow, q_5))^{\boxplus} = (a/b)^{\boxplus}$. Similarly, $C_{a^+}(q_4, \leftarrow, q_4) = (a/a)^{\boxplus}$, $C_{a^+}(q_1, \rightarrow, q_1) = (a/\varepsilon)^{\boxplus} = C_{a^+}(q_6, \rightarrow, q_6)$. 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 $C_{a^+}(q_2, \rightarrow, q_3) = a ? C_a(q_2, \rightarrow, q_3) : (C_a(q_2, \rightarrow, q_3) \boxplus (C_a(q_3, \rightarrow, q_3))^{\boxplus}) = a ? (a/\varepsilon) : ((a/\varepsilon) \boxplus (a/\varepsilon)^{\boxplus})$, 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)

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

More interesting is the step $y = (q_4, \leftarrow, q_1)$ 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

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

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 $E_2 = ba^+ba^+b \subseteq \text{dom}(\mathcal{A})$:

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

Therefore, $\llbracket C_2 \rrbracket (ba^{m_1}ba^{m_2}b) = a^{m_2}b^{m_1} = \llbracket \mathcal{A} \rrbracket (ba^{m_1}ba^{m_2}b)$.

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

1226
1227
1228

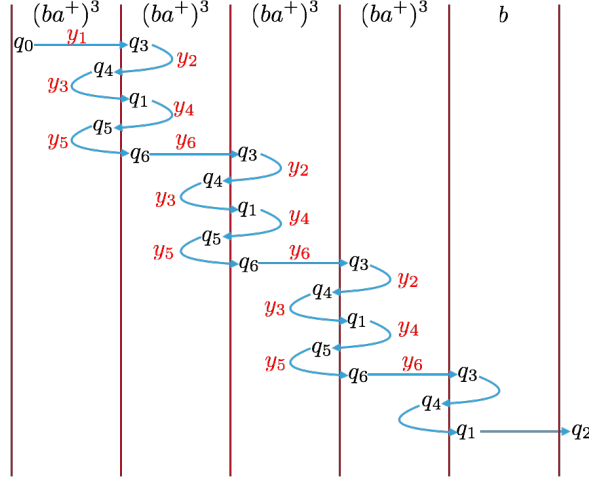


Figure A.14: run of a word in $E_3 = [(ba^+)^3]^+b$

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),

$$\begin{aligned}
C_{E_3}(q_0, \rightarrow, q_2) &= (C_{F^+}(q_0, \rightarrow, q_3) \square C_b(q_3, \rightarrow, q_4)) \odot \\
&\quad (C_{F^+}(q_4, \leftarrow, q_1) \square C_b(q_1, \rightarrow, q_2)).
\end{aligned}$$

1229
1230

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

1231
1232
1233
1234
1235
1236
1237

with atomic expressions and steps which are going to be used in constructing $C_{E_3}(q_0, \rightarrow, q_2)$. They are $C_b(q_3, \rhd, q_4) = C_b(q_6, \rightarrow, q_1) = C_b(q_1, \rightarrow, q_2) = C_b(q_5, \rightarrow, q_6) = (b/\varepsilon)$ and $C_{a^+}(q_2, \rightarrow, q_3) = C_{a^+}(q_1, \rightarrow, q_1) = (a/\varepsilon)^{\boxplus}$, $C_{a^+}(q_4, \leftarrow, q_4) = (a/a)^{\boxplus}$, $C_{a^+}(q_5, \leftarrow, q_5) = (a/b)^{\boxplus}$. We compute RTE $C_F(x)$ for the relevant steps x in the monoid element $X = \text{Tr}(F)$. 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 $y_1 = (q_0, \rightarrow, q_3)$

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

1238
1239

where $C_{E_2}(q_0, \rightarrow, q_2)$ has been computed in Section 1.

For example, $\llbracket C_F(y_1) \rrbracket (ba^{m_1}ba^{m_2}ba^{m_3}) = a^{m_2}b^{m_1}$.

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

$$C_F(y_2) = C_b(q_3, \rhd, q_4) \boxtimes (a^+ba^+ba^+? \varepsilon : \perp) \approx ((ba^+)^3? \varepsilon : \perp)$$

1240

$$\llbracket C_F(y_2) \rrbracket (ba^{m_1}ba^{m_2}ba^{m_3}) = \varepsilon.$$

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

$$\begin{aligned} C_F(y_3) &= (ba^+? \varepsilon : \perp) \xleftarrow{\boxplus} C_{ba^+ba^+}(y_3) \\ &= (ba^+? \varepsilon : \perp) \xleftarrow{\boxplus} ((b/\varepsilon) \xleftarrow{\boxplus} (a/b)^{\boxplus} \xleftarrow{\boxplus} (b/\varepsilon) \xleftarrow{\boxplus} (a/a)^{\boxplus}) \\ &\approx (ba^+b? \varepsilon : \perp) \boxtimes ((a/b)^{\boxplus} \xleftarrow{\boxplus} (b/\varepsilon) \xleftarrow{\boxplus} (a/a)^{\boxplus}) \end{aligned}$$

1241
1242

where $C_{ba^+ba^+}(y_3)$ is already computed in Section 1.

As an example, $\llbracket C_F(y_3) \rrbracket (ba^{m_1}ba^{m_2}ba^{m_3}) = a^{m_3}b^{m_2}$.

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

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

1243

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

(e) For $y_5 = (q_5, \swarrow, q_6)$, 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.

$$\begin{aligned} C_{ba^+}(y_5) &= ((b/\varepsilon) \overleftarrow{\square} C_{a^+}(q_5, \leftarrow, q_5)) \odot (C_b(q_5, \rightarrow, q_6) \square C_{a^+}(q_6, \rightarrow, q_6)) \\ &= ((b/\varepsilon) \overleftarrow{\square} (a/b) \overleftarrow{\square}) \odot ((b/\varepsilon) \square (a/\varepsilon) \overleftarrow{\square}) \approx (b/\varepsilon) \overleftarrow{\square} (a/b) \overleftarrow{\square} \end{aligned}$$

We can compute $C_F(y_5)$ as

$$C_F(y_5) = (ba^+ba^+? \varepsilon : \perp) \square C_{ba^+}(y_5) \approx (ba^+ba^+b? \varepsilon : \perp) \overleftarrow{\square} (a/b) \overleftarrow{\square}$$

1244

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

- (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.

$$\begin{aligned} C_{ba^+ba^+}(q_6, \rightarrow, q_3) &= C_b(q_6, \rightarrow, q_1) \square C_{a^+}(q_1, \rightarrow, q_1) \square \\ &\quad C_b(q_1, \rightarrow, q_2) \square C_{a^+}(q_2, \rightarrow, q_3) \\ &= (b/\varepsilon) \square (a/\varepsilon) \overleftarrow{\square} \square (b/\varepsilon) \square (a/\varepsilon) \overleftarrow{\square} \\ &\approx (ba^+ba^+? \varepsilon : \perp) \\ C_{ba^+ba^+b}(q_6, \rightarrow, q_2) &= (C_{ba^+ba^+}(q_6, \rightarrow, q_3) \square C_b(q_3, \rightarrow, q_4)) \odot \\ &\quad (C_{ba^+ba^+}(q_4, \leftarrow, q_1) \square C_b(q_1, \rightarrow, q_2)) \\ &\approx (ba^+ba^+b? \varepsilon : \perp) \odot \\ &\quad (((b/\varepsilon) \overleftarrow{\square} (a/b) \overleftarrow{\square}) \overleftarrow{\square} (b/\varepsilon) \overleftarrow{\square} (a/a) \overleftarrow{\square}) \square (b/\varepsilon) \\ &\approx ((b/\varepsilon) \overleftarrow{\square} (a/b) \overleftarrow{\square}) \overleftarrow{\square} (b/\varepsilon) \overleftarrow{\square} (a/a) \overleftarrow{\square} \square (b/\varepsilon) \end{aligned}$$

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

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

1245

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

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.

$$\begin{aligned} C_{F^+}(y_1) &= (C_F(y_1) \square (F^*? \varepsilon : \perp)) \odot [F, C']^{2\boxplus} \\ C' &= ((F? \varepsilon : \perp) \square C_F(y_2)) \odot (C_F(y_3) \square C_F(y_4)) \odot (C_F(y_5) \square C_F(y_6)) \end{aligned}$$

1246

As an example,

1247

$\llbracket C_{F^+}(y_1) \rrbracket (ba^{m_1}ba^{m_2}ba^{m_3}ba^{m_4}ba^{m_5}ba^{m_6}) = a^{m_2}b^{m_1}a^{m_3}b^{m_2}a^{m_4}b^{m_3}a^{m_5}b^{m_4}$. 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.

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

Notice that $C_{F^+}(q_4, \zeta, q_1) = (F^* ? \varepsilon : \perp) \boxtimes C_F(y_3)$.

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,

$$\begin{aligned} \llbracket C_{E_3}(y) \rrbracket (ba^{m_1}ba^{m_2}ba^{m_3}ba^{m_4}ba^{m_5}ba^{m_6}b) \\ = a^{m_2}b^{m_1}a^{m_3}b^{m_2}a^{m_4}b^{m_3}a^{m_5}b^{m_4}a^{m_6}b^{m_5} \end{aligned}$$