Derived-Term Automata of Multitape Rational Expressions (Long version*)

Akim Demaille akim@lrde.epita.fr

EPITA Research and Development Laboratory (LRDE)
14-16, rue Voltaire, 94276 Le Kremlin-Bicêtre, France

Abstract. We consider (weighted) rational expressions to denote series over Cartesian products of monoids. We define an operator $\mid$ to build multitape expressions such as $(a^+ \mid x + b^+ \mid y)^*$. We introduce *expansions*, which generalize the concept of derivative of a rational expression, but relieved from the need of a free monoid. We propose an algorithm based on expansions to build multitape automata from multitape expressions.

1 Introduction

Automata and rational (or regular) expressions share the same expressive power, with algorithms going from one to the other. This fact made rational expressions an extremely handy practical tool to specify some rational languages in a concise way, from which acceptors (automata) are built. There are many largely used implementations, probably starting with Ken Thompson [15], the creator of Unix, grep, etc.

There are numerous algorithms to build an automaton from an expression. We are particularly interested in the derivative-based family of algorithms [3–5, 7, 10], because they offer a very natural interpretation to states (they are labeled by an expression that denotes the future of the states, i.e., the language/series accepted from this state). This allowed to support several extensions: extended operators (intersection, complement) [4, 5], weights [10], additional products (shuffle, infiltration), etc.

Multitape automata, including transducers, share many properties with "single-tape" automata, in particular the Fundamental Theorem [14, Theorem 2.1, p. 409]: under appropriate conditions, multitape automata and rational (multitape) series share the same expressive power. However, as far as the author knows, there is no definition of multitape rational expressions that allows expressions such as $\mathsf{E}_2 \coloneqq (a^+ \mid x + b^+ \mid y)^*$ (Example 5). To denote such a binary relation between words, one had to build a (usual) rational expression in "normal form", without tupling of expressions but only tuples of letters such as a set of generators.

* This report is an extended version of the paper published in CIAA 2016 under the same name.

So for instance instead of E_2, one must use $\mathsf{E}_2' \coloneqq \big((a \mid \varepsilon)^+(\varepsilon \mid x) + (b \mid \varepsilon)^+(\varepsilon \mid y)\big)^*$, which is larger, as is its derived-term automaton.

The contributions of this paper are twofold: we define (weighted) multitape rational expressions featuring a $\mid$ operator, and we provide an algorithm to build an equivalent automaton. This algorithm is a generalization of the derived-term based algorithms, freed from the requirement that the monoid is free.

We first settle the notations in Sect. 2, provide an algorithm to compute the expansion of an expression in Sect. 3, which is used in Sect. 4 to propose an alternative construction of the derived-term automaton.

The constructs exposed in this paper are implemented in Vcsn[1]. Vcsn is a free-software platform dedicated to weighted automata and rational expressions [8]; its lowest layer is a C++ library, on top of which Python/IPython bindings provide an interactive graphical environment.

2 Notations

Our purpose is to define (weighted) multitape rational expressions, such as $\mathsf{E}_1 \coloneqq \langle 5\rangle 1|1 + \langle 4\rangle a\,d\,e^*|x + \langle 3\rangle b\,d\,e^*|x + \langle 2\rangle a\,c\,e^*|x\,y + \langle 6\rangle b\,c\,e^*|x\,y$ (weights are written in angle brackets). It relates *ade* with x, with weight 4. We introduce an algorithm to build a multitape automaton (aka *transducer*) from such an expression, e.g., Fig. 1. This algorithm relies on *rational expansions*. They are to the derivatives of rational expressions what differential forms are to the derivatives of functions. Defining expansions requires several concepts, defined bottom-up in this section. The following figure presents these different entities, how they relate to each other, and where we are heading to: given a weighted multitape rational expression such as E_1, compute *its* expansion:

$$\underbrace{\underbrace{\overbrace{\langle 5\rangle}^{\text{Weight}}}_{\text{Constant term}} \oplus \underbrace{\underbrace{\overbrace{a|x}^{\text{Label}}}_{\text{First}} \odot \left[\langle 2\rangle \odot \underbrace{\overbrace{ce^*|y}^{\substack{\text{Expression}\\ \text{(Sect. 2.2)}}}}_{\text{Derived term}} \oplus \overbrace{\langle 4\rangle \odot de^*|1}^{\text{Monomial}}\right] \oplus b|x \odot \left[\overbrace{\langle 6\rangle \odot ce^*|y \oplus \langle 3\rangle \odot de^*|1}^{\text{Polynomial (Sect. 2.3)}}\right]}_{\text{Proper part of the expansion}}}_{\text{Expansion (Sect. 2.4)}}$$

from which we build its derived-term automaton (Fig. 1).

It is helpful to think of expansions as a normal form for expressions.

[1] See the interactive environment, http://vcsn-sandbox.lrde.epita.fr, or its documentation, http://vcsn.lrde.epita.fr/dload/2.3/notebooks/expression.derived_term.html, or this paper's companion notebook, http://vcsn.lrde.epita.fr/dload/2.3/notebooks/CIAA-2016.html.

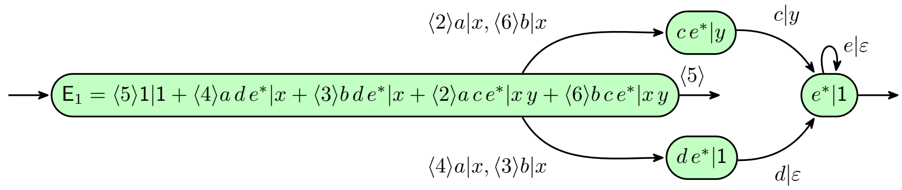


Fig. 1. The derived-term automaton of E_1 (see Examples 1 to 3) with $\mathsf{E}_1 \coloneqq \langle 5\rangle 1|1 + \langle 4\rangle a\,d\,e^*|x + \langle 3\rangle b\,d\,e^*|x + \langle 2\rangle a\,c\,e^*|x\,y + \langle 6\rangle b\,c\,e^*|x\,y$.

2.1 Rational Series

Series will be used to define the semantics of the forthcoming structures: they are to weighted automata what languages are to Boolean automata. Not all languages are rational (denoted by an expression), and similarly, not all series are rational (denoted by a weighted expression). We follow Sakarovitch [14, Chap. III].

In order to cope with (possibly) several tapes, we cannot rely on the traditional definitions based on the free monoid A^* for some alphabet A.

Labels Let M be a monoid (e.g., A^* or $A^* \times B^*$), whose neutral element is denoted ε_M, or ε when clear from the context. For consistency with the way transducers are usually represented, we use $m \mid n$ rather than (m, n) to denote the pair of m and n. For instance $\varepsilon_{A^* \times B^*} = \varepsilon_{A^*} \mid \varepsilon_{B^*}$, and $\varepsilon_M \mid a \in M \times \{a\}^*$. A *set of generators* G of M is a subset of M such that $G^* = M$. A monoid M is of *finite type* (or *finitely generated*) if it admits a finite set of generators. A monoid M is *graded* if it admits a *gradation* function $|\cdot| \in M \to \mathbb{N}$ such that $\forall m, n \in M$, $|m| = 0$ iff $m = \varepsilon$, and $|mn| = |m| + |n|$. Cartesian products of graded monoids are graded, and Cartesian products of finitely generated monoids are finitely generated. Free monoids and Cartesian products of free monoids are graded and finitely generated.

Weights Let $\langle \mathbb{K}, +, \cdot, 0_{\mathbb{K}}, 1_{\mathbb{K}} \rangle$ (or $\mathbb{K}$ for short) be a semiring whose (possibly non commutative) multiplication will be denoted by juxtaposition. $\mathbb{K}$ is *commutative* if its multiplication is. $\mathbb{K}$ is a *topological semiring* if it is equipped with a topology, and both addition and multiplication are continuous. It is *strong* if the product of two summable families is summable.

Series A (formal power) *series* over M with *weights* (or *multiplicities*) in $\mathbb{K}$ is a map from M to $\mathbb{K}$. The weight of $m \in M$ in a series s is denoted $s(m)$. The *null* series, $m \mapsto 0_{\mathbb{K}}$, is denoted 0; for any $m \in M$ (including ε_M), m denotes the series $u \mapsto 1_{\mathbb{K}}$ if $u = m, 0_{\mathbb{K}}$ otherwise. If M is of finite type, then we can define the Cauchy product of series. $s \cdot t \coloneqq m \mapsto \sum_{u,v \in M | uv = m} s(u) \cdot t(v)$. Equipped with the pointwise addition ($s + t \coloneqq m \mapsto s(m) + t(m)$) and $\cdot$ as multiplication, the set of these series forms a semiring denoted $\left\langle \mathbb{K}\langle\langle M \rangle\rangle, +, \cdot, 0, \varepsilon \right\rangle$.

The *constant term* of a series s, denoted s_ε, is $s(\varepsilon)$, the weight of the empty word. A series s is *proper* if $s_\varepsilon = 0_{\mathbb{K}}$. The *proper part* of s is the proper series s_p such that $s = s_\varepsilon + s_p$.

Star The *star* of a series is an infinite sum: $s^* := \sum_{n\in\mathbb{N}} s^n$. To ensure semantic soundness, we need M to be graded monoid and $\mathbb{K}$ to be a strong topological semiring.

Proposition 1. *Let M be a graded monoid and $\mathbb{K}$ a strong topological semiring. Let $s \in \mathbb{K}\langle\langle M\rangle\rangle$, s^* is defined iff s_ε^* is defined and then $s^* = s_\varepsilon^* + s_\varepsilon^* s_p s^*$.*

Proof. By [14, Prop. 2.6, p. 396] s^* is defined iff s_ε^* is defined and then $s^* = (s_\varepsilon^* s_p)^* s_\varepsilon^* = s_\varepsilon^*(s_p s_\varepsilon^*)^*$. The result then follows directly from $s^* = \varepsilon + ss^*$: $s^* = s_\varepsilon^*(s_p s_\varepsilon^*)^* = s_\varepsilon^*(\varepsilon + (s_p s_\varepsilon^*)(s_p s_\varepsilon^*)^*) = s_\varepsilon^* + s_\varepsilon^* s_p(s_\varepsilon^*(s_p s_\varepsilon^*)^*) = s_\varepsilon^* + s_\varepsilon^* s_p s^*$. ☐

Tuple We suppose $\mathbb{K}$ is commutative. The *tupling* of two series $s \in \mathbb{K}\langle\langle M\rangle\rangle, t \in \mathbb{K}\langle\langle N\rangle\rangle$, is the series $s \mid t := m \mid n \in M \times N \mapsto s(m)t(n)$. It is a member of $\mathbb{K}\langle\langle M \times N\rangle\rangle$.

Proposition 2. *For all series $s, s' \in \mathbb{K}\langle\langle M\rangle\rangle$ and $t, t' \in \mathbb{K}\langle\langle N\rangle\rangle$, $(s + s') \mid t = s \mid t + s' \mid t$ and $s \mid (t + t') = s \mid t + s \mid t'$.*

Proof. Let $m|n \in M\times N$. $((s+s')|t)(m|n) = (s+s')(m)\cdot t(n) = (s(m)+s'(m))\cdot t(n) = s(m)\cdot t(n)+s'(m)\cdot t(n) = (s \mid t)(m \mid n)\cdot(s' \mid t)(m \mid n) = (s \mid t+s' \mid t)(m \mid n)$. Likewise for right distributivity. ☐

From now on, M is a graded monoid of finite type, and $\mathbb{K}$ a commutative strong topological semiring.

2.2 Weighted Rational Expressions

Contrary to the usual definition, we do not require a finite alphabet: any set of generators $G \subseteq M$ will do. For expressions with more than one tape, we required $\mathbb{K}$ to be commutative; however, for single tape expressions, our results apply to non-commutative semirings, hence there are two exterior products.

Definition 1 (Expression). *A* rational expression E *over G is a term built from the following grammar, where $a \in G$ denotes any non empty label, and $k \in \mathbb{K}$ any weight:* $\mathsf{E} ::= \mathsf{0} \mid \mathsf{1} \mid a \mid \mathsf{E}+\mathsf{E} \mid \langle k\rangle\mathsf{E} \mid \mathsf{E}\langle k\rangle \mid \mathsf{E}\cdot\mathsf{E} \mid \mathsf{E}^* \mid \mathsf{E}|\mathsf{E}$.

Expressions are syntactic; they are finite notations for (some) series.

Definition 2 (Series Denoted by an Expression). *Let E be an expression. The series denoted by E, noted $[\![\mathsf{E}]\!]$, is defined by induction on E:*

$$[\![\mathsf{0}]\!] := 0 \qquad [\![\mathsf{1}]\!] := \varepsilon \qquad [\![a]\!] := a \qquad [\![\mathsf{E}+\mathsf{F}]\!] := [\![\mathsf{E}]\!] + [\![\mathsf{F}]\!] \qquad [\![\langle k\rangle\mathsf{E}]\!] := k[\![\mathsf{E}]\!]$$
$$[\![\mathsf{E}\langle k\rangle]\!] := [\![\mathsf{E}]\!]k \quad [\![\mathsf{E}\cdot\mathsf{F}]\!] := [\![\mathsf{E}]\!]\cdot[\![\mathsf{F}]\!] \quad [\![\mathsf{E}^*]\!] := [\![\mathsf{E}]\!]^* \quad [\![\mathsf{E} \mid \mathsf{F}]\!] := [\![\mathsf{E}]\!] \mid [\![\mathsf{F}]\!]$$

An expression is *valid* if it denotes a series. More specifically, there are two requirements. First, the expression must be well-formed, i.e., concatenation and disjunction must be applied to expressions of appropriate number of tapes. For instance, $a + b|c$ and $a(b|c)$ are ill-formed, $(a \mid b)^* \mid c + a \mid (b \mid c)^*$ is well-formed. Second, to ensure that $[\![\mathsf{F}]\!]^*$ is well defined for each subexpression of the form F^*, the constant term of $[\![\mathsf{F}]\!]$ must be *starrable* in $\mathbb{K}$ (Proposition 1). This definition, which involves series (semantics) to define a property of expressions (syntax), will be made effective (syntactic) with the appropriate definition of the constant term $d_\varepsilon(\mathsf{E})$ *of an expression* E (Definition 6).

Let $[n]$ denote $\{1, \dots, n\}$). The *size* (aka *length*) of a (valid) expression E, $|\mathsf{E}|$, is its total number of symbols, not counting parenthesis; for a given tape number $i \in [k]$ the *width on tape* i, $\|\mathsf{E}\|_i$, is the number of occurrences of labels on the tape i, the *width* of E (aka *literal length*), $\|\mathsf{E}\| := \sum_{i\in[k]}\|\mathsf{E}\|_i$ is the total number of occurrences of labels.

Two expressions E and F are *equivalent* iff $[\![\mathsf{E}]\!] = [\![\mathsf{F}]\!]$. Some expressions are "trivially equivalent"; any candidate expression will be rewritten via the following *trivial identities*. Any subexpression of a form listed to the left of a '$\Rightarrow$' is rewritten as indicated on the right.

$$\begin{gathered}
\mathsf{E}+\mathsf{0} \Rightarrow \mathsf{E} \qquad \mathsf{0}+\mathsf{E} \Rightarrow \mathsf{E} \\
\langle 0_\mathbb{K}\rangle\mathsf{E} \Rightarrow \mathsf{0} \quad \langle 1_\mathbb{K}\rangle\mathsf{E} \Rightarrow \mathsf{E} \quad \langle k\rangle\mathsf{0} \Rightarrow \mathsf{0} \quad \langle k\rangle\langle h\rangle\mathsf{E} \Rightarrow \langle kh\rangle\mathsf{E} \\
\mathsf{E}\langle 0_\mathbb{K}\rangle \Rightarrow \mathsf{0} \quad \mathsf{E}\langle 1_\mathbb{K}\rangle \Rightarrow \mathsf{E} \quad \mathsf{0}\langle k\rangle \Rightarrow \mathsf{0} \quad \mathsf{E}\langle k\rangle\langle h\rangle \Rightarrow \mathsf{E}\langle kh\rangle \\
(\langle k\rangle\mathsf{E})\langle h\rangle \Rightarrow \langle k\rangle(\mathsf{E}\langle h\rangle) \qquad \ell\langle k\rangle \Rightarrow \langle k\rangle\ell \\
\mathsf{E}\cdot\mathsf{0} \Rightarrow \mathsf{0} \qquad \mathsf{0}\cdot\mathsf{E} \Rightarrow \mathsf{0} \\
(\langle k\rangle^?\mathsf{1})\cdot\mathsf{E} \Rightarrow \langle k\rangle^?\mathsf{E} \qquad \mathsf{E}\cdot(\langle k\rangle^?\mathsf{1}) \Rightarrow \mathsf{E}\langle k\rangle^? \\
\mathsf{0}^\star \Rightarrow \mathsf{1} \\
(\langle k\rangle^?\mathsf{E}) \mid (\langle h\rangle^?\mathsf{F}) \Rightarrow \langle kh\rangle^?\mathsf{E} \mid \mathsf{F}
\end{gathered}$$

where E is a rational expression, $\ell \in G \cup \{\mathsf{1}\}$ a label, $k, h \in \mathbb{K}$ weights, and $\langle k\rangle^?\ell$ denotes either $\langle k\rangle\ell$, or ℓ in which case $k = 1_\mathbb{K}$ in the right-hand side of $\Rightarrow$. The choice of these identities is beyond the scope of this paper (see [14]), however note that they are limited to trivial properties; in particular *linearity* ("weighted ACI": associativity, commutativity and $\langle k\rangle\mathsf{E} + \langle h\rangle\mathsf{E} \Rightarrow \langle k+h\rangle\mathsf{E}$) is not enforced. In practice, additional identities help reducing the automaton size [12].

2.3 Rational Polynomials

At the core of the idea of "partial derivatives" introduced by Antimirov [3], is that of *sets* of rational expressions, later generalized in *weighted sets* by Lombardy and Sakarovitch [10], i.e., functions (partial, with finite domain) from the set of rational expressions into $\mathbb{K} \setminus \{0_\mathbb{K}\}$. It proves useful to view such structures as "polynomials of expressions". In essence, they capture the linearity of addition.

Definition 3 (Rational Polynomial). *A* polynomial *(of rational expressions) is a finite (left) linear combination of expressions. Syntactically it is a term built from the grammar* $\mathsf{P} ::= \mathsf{0} \mid \langle k_1\rangle \odot \mathsf{E}_1 \oplus \cdots \oplus \langle k_n\rangle \odot \mathsf{E}_n$ *where* $k_i \in \mathbb{K} \setminus \{0_\mathbb{K}\}$ *denote* non-null *weights, and* E_i *denote* non-null *expressions. Expressions may not appear more than once in a polynomial. A* monomial *is a pair* $\langle k_i\rangle \odot \mathsf{E}_i$.

We use specific symbols ($\odot$ and $\oplus$) to clearly separate the outer polynomial layer from the inner expression layer. Let $\mathsf{P} = \bigoplus_{i\in[n]} \langle k_i \rangle \odot \mathsf{E}_i$ be a polynomial of expressions. The "*projection*" of P is the expression $\mathsf{expr}(\mathsf{P}) \coloneqq \langle k_1 \rangle \mathsf{E}_1 + \cdots + \langle k_n \rangle \mathsf{E}_n$ (or $\mathsf{0}$ if P is null); this operation is performed on a canonical form of the polynomial (expressions are sorted in a well defined order). Polynomials denote series: $[\![\mathsf{P}]\!] \coloneqq [\![\mathsf{expr}(\mathsf{P})]\!]$. The *terms* of P is the set $\mathsf{exprs}(\mathsf{P}) \coloneqq \{\mathsf{E}_1, \ldots, \mathsf{E}_n\}$.

Example 1. Let $\mathsf{E}_1 \coloneqq \langle 5 \rangle 1|1 + \langle 4 \rangle a\,d\,e^*|x + \langle 3 \rangle b\,d\,e^*|x + \langle 2 \rangle a\,c\,e^*|x\,y + \langle 6 \rangle b\,c\,e^*|x\,y$. Polynomial '$\mathsf{P}_{1,a|x} \coloneqq \langle 2 \rangle \odot ce^* \mid y \oplus \langle 4 \rangle \odot de^* \mid 1$' has two monomials: '$\langle 2 \rangle \odot ce^* \mid y$' and '$\langle 4 \rangle \odot de^* \mid 1$'. It denotes the (left) quotient of $[\![\mathsf{E}_1]\!]$ by $a \mid x$, and '$\mathsf{P}_{1,b|x} \coloneqq \langle 6 \rangle \odot ce^* \mid y \oplus \langle 3 \rangle \odot de^* \mid 1$' the quotient by $b \mid x$.

Let $\mathsf{P} = \bigoplus_{i\in[n]} \langle k_i \rangle \odot \mathsf{E}_i, \mathsf{Q} = \bigoplus_{j\in[m]} \langle h_i \rangle \odot \mathsf{F}_i$ be polynomials, k a weight and F an expression, all possibly null, we introduce the following operations:

$$\mathsf{P} \cdot \mathsf{F} \coloneqq \bigoplus_{i\in[n]} \langle k_i \rangle \odot (\mathsf{E}_i \cdot \mathsf{F}) \quad \langle k \rangle \mathsf{P} \coloneqq \bigoplus_{i\in[n]} \langle k k_i \rangle \odot \mathsf{E}_i \quad \mathsf{P}\langle k \rangle \coloneqq \bigoplus_{i\in[n]} \langle k_i \rangle \odot (\mathsf{E}_i \langle k \rangle)$$

$$\mathsf{P} \mid 1 \coloneqq \bigoplus_{i\in[n]} \langle k_i \rangle \odot \mathsf{E}_i \mid 1 \qquad 1 \mid \mathsf{P} \coloneqq \bigoplus_{i\in[n]} \langle k_i \rangle \odot 1 \mid \mathsf{E}_i$$

$$\mathsf{P} \mid \mathsf{Q} \coloneqq \bigoplus_{(i,j)\in[n]\times[m]} \langle k_i \cdot h_j \rangle \odot \mathsf{E}_i \mid \mathsf{F}_j$$

Trivial identities might simplify the result. Note the asymmetry between left and right exterior products. The addition of polynomials is commutative, multiplication by zero (be it an expression or a weight) evaluates to the null polynomial, and the left-multiplication by a weight is distributive.

Lemma 1. $[\![\mathsf{P} \cdot \mathsf{F}]\!] = [\![\mathsf{P}]\!] \cdot [\![\mathsf{F}]\!] \qquad [\![\langle k \rangle \mathsf{P}]\!] = \langle k \rangle [\![\mathsf{P}]\!] \qquad [\![\mathsf{P}\langle k \rangle]\!] = [\![\mathsf{P}]\!]\langle k \rangle$
$[\![\mathsf{P} \mid \mathsf{Q}]\!] = [\![\mathsf{P}]\!] \mid [\![\mathsf{Q}]\!]$.

Proof. See Appendix A.1.

2.4 Rational Expansions

Definition 4 (Rational Expansion). *A* rational expansion X *is a term* $\mathsf{X} ::= \langle \mathsf{X}_\varepsilon \rangle \oplus a_1 \odot [\mathsf{X}_{a_1}] \oplus \cdots \oplus a_n \odot [\mathsf{X}_{a_n}]$ *where* $\mathsf{X}_\varepsilon \in \mathbb{K}$ *is a weight (possibly null),* $a_i \in G \setminus \{\varepsilon\}$ *non-empty labels (occurring at most once), and* X_{a_i} *non-null polynomials. The* constant term *is* X_ε*, the* proper part *is* $\mathsf{X}_p \coloneqq a_1 \odot [\mathsf{X}_{a_1}] \oplus \cdots \oplus a_n \odot [\mathsf{X}_{a_n}]$*, the* firsts *is* $f(\mathsf{X}) \coloneqq \{a_1, \ldots, a_n\}$ *(possibly empty) and the* terms $\mathsf{exprs}(\mathsf{X}) \coloneqq \bigcup_{i\in[n]} \mathsf{exprs}(\mathsf{X}_{a_i})$.

To ease reading, polynomials are written in square brackets. Contrary to expressions and polynomials, there is no specific term for the null expansion: it is represented by $\langle 0_{\mathbb{K}} \rangle$, the null weight. Except for this case, null constant terms are left implicit. Expansions will be written: $\mathsf{X} = \langle \mathsf{X}_\varepsilon \rangle \oplus \bigoplus_{a\in f(\mathsf{X})} a \odot [\mathsf{X}_a]$. When more convenient, we write $\mathsf{X}(\ell)$ instead of X_ℓ for $\ell \in f(\mathsf{X}) \cup \{\varepsilon\}$.

An expansion X can be "projected" as a rational expression $\mathsf{expr}(\mathsf{X})$ by mapping weights, labels and polynomials to their corresponding rational expressions, and $\oplus/\odot$ to the sum/concatenation of expressions. Again, this is performed on a canonical form of the expansion: labels are sorted. Expansions also denote series: $[\![\mathsf{X}]\!] \coloneqq [\![\mathsf{expr}(\mathsf{X})]\!]$. An expansion X is *equivalent* to an expression E iff $[\![\mathsf{X}]\!] = [\![\mathsf{E}]\!]$.

Example 2 (Example 1 continued). Expansion $\mathsf{X}_1 \coloneqq \langle 5\rangle \oplus a|x \odot [\mathsf{P}_{1,a|x}] \oplus b|x \odot [\mathsf{P}_{1,b|x}]$ has $\mathsf{X}_1(\varepsilon) = \langle 5\rangle$ as constant term, and maps the generator $a|x$ (resp. $b|x$) to the polynomial $\mathsf{X}_1(a|x) = \mathsf{P}_{1,a|x}$ (resp. $\mathsf{X}_1(b|x) = \mathsf{P}_{1,b|x}$). X_1 can be proved to be equivalent to E_1.

Let X, Y be expansions, k a weight, and E an expression (all possibly null):

$$\mathsf{X} \oplus \mathsf{Y} \coloneqq \langle \mathsf{X}_\varepsilon + \mathsf{Y}_\varepsilon\rangle \oplus \bigoplus_{a \in f(\mathsf{X}) \cup f(\mathsf{Y})} a \odot [\mathsf{X}_a \oplus \mathsf{Y}_a] \tag{1}$$

$$\langle k\rangle\mathsf{X} \coloneqq \langle k\mathsf{X}_\varepsilon\rangle \oplus \bigoplus_{a \in f(\mathsf{X})} a \odot [\langle k\rangle \mathsf{X}_a] \qquad \mathsf{X}\langle k\rangle \coloneqq \langle \mathsf{X}_\varepsilon k\rangle \oplus \bigoplus_{a \in f(\mathsf{X})} a \odot [\mathsf{X}_a \langle k\rangle] \tag{2}$$

$$\mathsf{X} \cdot \mathsf{E} \coloneqq \bigoplus_{a \in f(\mathsf{X})} a \odot [\mathsf{X}_a \cdot \mathsf{E}] \qquad \text{with } \mathsf{X} \text{ proper: } \mathsf{X}_\varepsilon = 0_{\mathbb{K}} \tag{3}$$

$$\begin{aligned} \mathsf{X} \mid \mathsf{Y} \coloneqq{}& \langle \mathsf{X}_\varepsilon \mathsf{Y}_\varepsilon\rangle \oplus \langle \mathsf{X}_\varepsilon\rangle \bigoplus_{b \in f(\mathsf{Y})} (\varepsilon|b) \odot (1 \mid \mathsf{Y}_b) \oplus \langle \mathsf{Y}_\varepsilon\rangle \bigoplus_{a \in f(\mathsf{X})} (a|\varepsilon) \odot (\mathsf{X}_a \mid 1) \\ &\oplus \bigoplus_{a|b \in f(\mathsf{X}) \times f(\mathsf{Y})} (a|b) \odot (\mathsf{X}_a \mid \mathsf{Y}_b) \end{aligned} \tag{4}$$

Since by definition expansions never map to null polynomials, some firsts might be smaller that suggested by these equations. For instance in $\mathbb{Z}$ the sum of $\langle 1\rangle \oplus a \odot [\langle 1\rangle \odot b]$ and $\langle 1\rangle \oplus a \odot [\langle -1\rangle \odot b]$ is $\langle 2\rangle$.

The following lemma is simple to establish: lift semantic equivalences, such as Proposition 2, to syntax, using Lemma 1.

Lemma 2. $[\![\mathsf{X} \oplus \mathsf{Y}]\!] = [\![\mathsf{X}]\!] + [\![\mathsf{Y}]\!]$ $\qquad [\![\langle k\rangle\mathsf{X}]\!] = \langle k\rangle[\![\mathsf{X}]\!]$ $\qquad [\![\mathsf{X}\langle k\rangle]\!] = [\![\mathsf{X}]\!]\langle k\rangle$
$[\![\mathsf{X} \cdot \mathsf{E}]\!] = [\![\mathsf{X}]\!] \cdot [\![\mathsf{E}]\!]$ $\qquad [\![\mathsf{X} \mid \mathsf{Y}]\!] = [\![\mathsf{X}]\!] \mid [\![\mathsf{Y}]\!]$

2.5 Finite Weighted Automata

Definition 5 (Weighted Automaton). *A* weighted automaton $\mathcal{A}$ *is a tuple* $\langle M, G, \mathbb{K}, Q, E, I, T\rangle$ *where:*

- *M is a monoid,*
- *G (the labels) is a set of generators of M,*
- *$\mathbb{K}$ (the set of weights) is a semiring,*
- *Q is a finite set of states,*
- *I and T are the* initial *and* final *functions from Q into $\mathbb{K}$,*
- *E is a (partial) function from $Q \times G \times Q$ into $\mathbb{K} \setminus \{0_{\mathbb{K}}\}$;*
  its domain represents the transitions: $(source, label, destination)$.

An automaton is proper *if no label is ε_M.*

A *computation* $p = (q_0, a_0, q_1)(q_1, a_1, q_2)\cdots(q_n, a_n, q_{n+1})$ in an automaton is a sequence of transitions where the source of each is the destination of the previous one; its *label* is $a_0 a_1 \cdots a_n \in M$, its *weight* is $I(q_0) \otimes E(q_0, a_0, q_1) \otimes \cdots \otimes E(q_n, a_n, q_{n+1}) \otimes T(q_{n+1}) \in \mathbb{K}$. The *evaluation* of word u by $\mathcal{A}$, $\mathcal{A}(u)$, is the sum of the weights of all the computations labeled by u, or $0_\mathbb{K}$ if there are none. The *behavior* of an automaton $\mathcal{A}$ is the series $[\![\mathcal{A}]\!] \coloneqq m \mapsto \mathcal{A}(m)$. A state q is *initial* if $I(q) \neq 0_\mathbb{K}$. A state q is *accessible* if there is a computation from an initial state to q. The *accessible* part of an automaton $\mathcal{A}$ is the subautomaton whose states are the accessible states of $\mathcal{A}$. The size of an automaton, $|\mathcal{A}|$, is its number of states.

We are interested, given an expression E, in an algorithm to compute an automaton $\mathcal{A}_\mathsf{E}$ such that $[\![\mathcal{A}_\mathsf{E}]\!] = [\![\mathsf{E}]\!]$ (Definition 7). To this end, we first introduce a simple recursive procedure to compute *the* expansion of an expression.

3 Expansion of a Rational Expression

Definition 6 (Expansion of a Rational Expression). *The* expansion of a rational expression E*, written* $d(\mathsf{E})$*, is defined inductively as follows:*

$$d(\mathsf{0}) \coloneqq \langle 0_\mathbb{K} \rangle \qquad d(\mathsf{1}) \coloneqq \langle 1_\mathbb{K} \rangle \qquad d(a) \coloneqq a \odot [\langle 1_\mathbb{K} \rangle \odot \mathsf{1}] \tag{5}$$

$$d(\mathsf{E} + \mathsf{F}) \coloneqq d(\mathsf{E}) \oplus d(\mathsf{F}) \tag{6}$$

$$d(\langle k \rangle \mathsf{E}) \coloneqq \langle k \rangle d(\mathsf{E}) \qquad d(\mathsf{E}\langle k \rangle) \coloneqq d(\mathsf{E})\langle k \rangle \tag{7}$$

$$d(\mathsf{E} \cdot \mathsf{F}) \coloneqq d_p(\mathsf{E}) \cdot \mathsf{F} \oplus \big\langle d_\varepsilon(\mathsf{E}) \big\rangle d(\mathsf{F}) \tag{8}$$

$$d(\mathsf{E}^*) \coloneqq \big\langle d_\varepsilon(\mathsf{E})^* \big\rangle \oplus \big\langle d_\varepsilon(\mathsf{E})^* \big\rangle d_p(\mathsf{E}) \cdot \mathsf{E}^* \tag{9}$$

$$d(\mathsf{E} \mid \mathsf{F}) \coloneqq d(\mathsf{E}) \mid d(\mathsf{F}) \tag{10}$$

where $d_\varepsilon(\mathsf{E}) \coloneqq d(\mathsf{E})_\varepsilon, d_p(\mathsf{E}) \coloneqq d(\mathsf{E})_p$ *are the constant term/proper part of* $d(\mathsf{E})$*.*

The right-hand sides are indeed expansions. The computation trivially terminates: induction is performed on strictly smaller subexpressions. These formulas are enough to compute the expansion of an expression; there is no secondary process to compute the firsts — indeed $d(a) \coloneqq a \odot [\langle 1_\mathbb{K} \rangle \odot \mathsf{1}]$ suffices and every other case simply propagates or assembles the firsts — or the constant terms.

Of course, in an implementation, a single recursive call to $d(\mathsf{E})$ is performed for (8) and (9), from which $d_\varepsilon(\mathsf{E})$ and $d_p(\mathsf{E})$ are obtained, and additional expansions are computed only when needed. So they should rather be written:

$$d(\mathsf{E} \cdot \mathsf{F}) \coloneqq \texttt{let } \mathsf{X} = d(\mathsf{E}) \texttt{ in if } \langle \mathsf{X}_\varepsilon \rangle \neq 0_\mathbb{K} \texttt{ then } \mathsf{X}_p \cdot \mathsf{F} \oplus \langle \mathsf{X}_\varepsilon \rangle d(\mathsf{F}) \texttt{ else } \mathsf{X}_p \cdot \mathsf{F}$$

$$d(\mathsf{E}^*) \coloneqq \texttt{let } \mathsf{X} = d(\mathsf{E}) \texttt{ in } \langle \mathsf{X}_\varepsilon^* \rangle \oplus \langle \mathsf{X}_\varepsilon^* \rangle \mathsf{X}_p \cdot \mathsf{E}^*$$

Besides, existing expressions should be referenced to, not duplicated. In the previous piece of code, E^* is not built again, the input argument is reused.

Note that the firsts are a subset of the labels of the expression, hence of $G \setminus \{\varepsilon\}$. In particular, no first includes ε.

Proposition 3. *The expansion of a rational expression is equivalent to the expression.*

Proof. We prove that $\llbracket d(\mathsf{E}) \rrbracket = \llbracket \mathsf{E} \rrbracket$ by induction on the expression. The equivalence is straightforward for (5) to (7) and (10), viz., $\llbracket d(\mathsf{E} \mid \mathsf{F}) \rrbracket = \llbracket d(\mathsf{E}) \mid d(\mathsf{F}) \rrbracket$ (by (10)) $= \llbracket d(\mathsf{E}) \rrbracket \mid \llbracket d(\mathsf{F}) \rrbracket$ (by Lemma 2) $= \llbracket \mathsf{E} \rrbracket \mid \llbracket \mathsf{F} \rrbracket$ (by induction hypothesis) $= \llbracket \mathsf{E} \mid \mathsf{F} \rrbracket$ (by Lemma 2) . The case of multiplication, (8), follows from:

$$\begin{aligned}
\llbracket d(\mathsf{E} \cdot \mathsf{F}) \rrbracket &= \left\llbracket d_p(\mathsf{E}) \cdot \mathsf{F} \oplus \big\langle d_\varepsilon(\mathsf{E}) \big\rangle \cdot d(\mathsf{F}) \right\rrbracket & &= \llbracket d_p(\mathsf{E}) \rrbracket \cdot \llbracket \mathsf{F} \rrbracket + \big\langle d_\varepsilon(\mathsf{E}) \big\rangle \cdot \llbracket d(\mathsf{F}) \rrbracket \\
&= \llbracket d_p(\mathsf{E}) \rrbracket \cdot \llbracket \mathsf{F} \rrbracket + \big\langle d_\varepsilon(\mathsf{E}) \big\rangle \cdot \llbracket \mathsf{F} \rrbracket & &= \Big(\llbracket \langle d_\varepsilon(\mathsf{E}) \rangle \rrbracket + \llbracket d_p(\mathsf{E}) \rrbracket \Big) \cdot \llbracket \mathsf{F} \rrbracket \\
&= \left\llbracket \big\langle d_\varepsilon(\mathsf{E}) \big\rangle + d_p(\mathsf{E}) \right\rrbracket \cdot \llbracket \mathsf{F} \rrbracket & &= \llbracket d(\mathsf{E}) \rrbracket \cdot \llbracket \mathsf{F} \rrbracket \\
&= \llbracket \mathsf{E} \rrbracket \cdot \llbracket \mathsf{F} \rrbracket & &= \llbracket \mathsf{E} \cdot \mathsf{F} \rrbracket
\end{aligned}$$

It might seem more natural to exchange the two terms (i.e., $\big\langle d_\varepsilon(\mathsf{E}) \big\rangle \cdot d(\mathsf{F}) \oplus d_p(\mathsf{E}) \cdot \mathsf{F}$), but an implementation first computes $d(\mathsf{E})$ and then computes $d(\mathsf{F})$ *only if* $d_\varepsilon(\mathsf{E}) \neq 0_{\mathbb{K}}$. The case of Kleene star, (9), follows from Proposition 1. □

4 Expansion-Based Derived-Term Automaton

Definition 7 (Expansion-Based Derived-Term Automaton). *The* derived-term automaton *of an expression* E *over* G *is the* accessible part *of the automaton* $\mathcal{A}_{\mathsf{E}} \coloneqq \langle M, G, \mathbb{K}, Q, E, I, T \rangle$ *defined as follows:*

- Q *is the set of rational expressions on alphabet* A *with weights in* $\mathbb{K}$*,*
- $I = \mathsf{E} \mapsto 1_{\mathbb{K}}$*,*
- $E(\mathsf{F}, a, \mathsf{F}') = k$ *iff* $a \in f(d(\mathsf{F}))$ and $\langle k \rangle \mathsf{F}' \in d(\mathsf{F})(a)$*,*
- $T(\mathsf{F}) = k$ *iff* $\langle k \rangle = d(\mathsf{F})(\varepsilon)$*.*

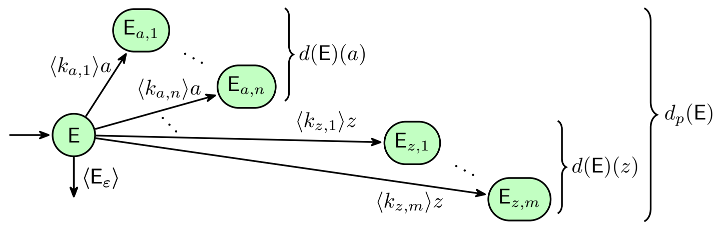
Since the firsts exclude ε, this automaton is proper. It is straightforward to extract an algorithm from Definition 7, using a work-list of states whose outgoing transitions to compute (see Appendix A.2). The Fig. 2 illustrates the process. This approach admits a natural lazy implementation: the whole automaton is not computed at once, but rather, states and transitions are computed on-the-fly, on demand, for instance when evaluating a word [7]. However, we must justify Definition 7 by proving that this automaton is finite (Theorem 1).

Example 3 (Examples 1 and 2 continued). With $\mathsf{E}_1 \coloneqq \langle 5 \rangle 1|1 + \langle 4 \rangle a\,d\,e^*|x + \langle 3 \rangle b\,d\,e^*|x + \langle 2 \rangle a\,c\,e^*|x\,y + \langle 6 \rangle b\,c\,e^*|x\,y$, one has:

$$\begin{aligned}
d(\mathsf{E}_1) &= \langle 5 \rangle \oplus a|x \odot [\langle 2 \rangle \odot ce^* \mid y \oplus \langle 4 \rangle \odot de^* \mid \varepsilon] \oplus b|x \odot [\langle 6 \rangle \odot ce^* \mid y \oplus \langle 3 \rangle \odot de^* \mid \varepsilon] \\
&= \mathsf{X}_1 \qquad \text{(from Example 2)}
\end{aligned}$$

Fig. 1 shows the resulting derived-term automaton.

Theorem 1. *For any* k*-tape expression* E*,* $|\mathcal{A}_{\mathsf{E}}| \leq \prod_{i \in [k]} (\|\mathsf{E}\|_i + 1) + 1$*.*



$$d(\mathsf{E}) = \underbrace{\langle \mathsf{E}_\varepsilon\rangle}_{d_\varepsilon(\mathsf{E})} \oplus \underbrace{a \odot [\overbrace{\langle k_{a,1}\rangle \odot \mathsf{E}_{a,1} \oplus \cdots \oplus \langle k_{a,n}\rangle \odot \mathsf{E}_{a,n}}^{d(\mathsf{E})(a)}] \oplus \cdots \oplus z \odot [\langle k_{z,1}\rangle \odot \mathsf{E}_{z,1} \oplus \cdots \oplus \langle k_{z,m}\rangle \odot \mathsf{E}_{z,m}]}_{d_p(\mathsf{E})}$$

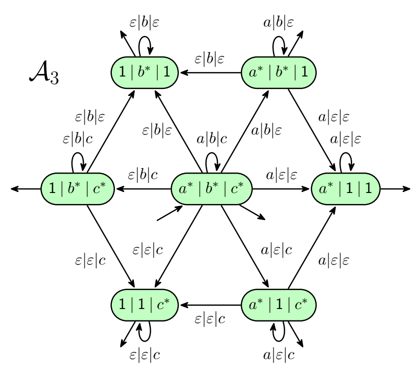
Fig. 2. Initial part of $\mathcal{A}_{\mathsf{E}}$, the derived-term automaton of E. This figure is somewhat misleading in that some $\mathsf{E}_{a,i}$ might be equal to an $\mathsf{E}_{z,j}$, or E (but never another $\mathsf{E}_{a,j}$).

Proof. The detailed proof is available in Appendix A.3. The proof goes in several steps. First introduce the *true derived terms* of E, a set of expressions noted $\mathrm{TD}(\mathsf{E})$, and the *derived terms* of E, $\mathrm{D}(\mathsf{E}) \coloneqq \mathrm{TD}(\mathsf{E}) \cup \{\mathsf{E}\}$. $\mathrm{TD}(\mathsf{E})$ admits a simple inductive definition similar to [2, Def. 3], to which we add $\mathrm{TD}(\mathsf{E} \mid \mathsf{F}) \coloneqq (\mathrm{TD}(\mathsf{E}) \mid \mathrm{TD}(\mathsf{F})) \cup (\{\mathsf{1}\} \mid \mathrm{TD}(\mathsf{F})) \cup (\mathrm{TD}(\mathsf{E}) \mid \{\mathsf{1}\})$, where for two sets of expressions E, F we introduce $E \mid F \coloneqq \{\mathsf{E} \mid \mathsf{F}\}_{(\mathsf{E},\mathsf{F}) \in E \times F}$. Second, verify that $|\mathrm{TD}(\mathsf{E})| \leq \prod_{i \in [k]}(\|\mathsf{E}\|_i + 1)$ (hence finite). Third, prove that $\mathrm{D}(\mathsf{E})$ is "stable by expansion", i.e., $\forall \mathsf{F} \in \mathrm{D}(\mathsf{E}), \mathsf{exprs}\big(d(\mathsf{F})\big) \subseteq \mathrm{D}(\mathsf{E})$. Finally, observe that the states of $\mathcal{A}_{\mathsf{E}}$ are therefore members of $\mathrm{D}(\mathsf{E})$, whose size is less than or equal to $1 + |\mathrm{TD}(\mathsf{E})|$. □

Theorem 2. *Any expression* E *and its expansion-based derived-term automaton* $\mathcal{A}_{\mathsf{E}}$ *denote the same series, i.e.,* $[\![\mathcal{A}_{\mathsf{E}}]\!] = [\![\mathsf{E}]\!]$.

Example 4. Let $\mathcal{A}_k$ be the derived-term automaton of the k-tape expression $a_1^* \mid \cdots \mid a_k^*$. The states of $\mathcal{A}_k$ are all the possible expressions where the tape i features $\mathsf{1}$ or a_i^*, except $\mathsf{1} \mid \cdots \mid \mathsf{1}$. Therefore $|\mathcal{A}_k| = 2^k - 1$, and $\prod_{i \in [k]}(\|\mathsf{E}\|_i + 1) = 2^k$.

$\mathcal{A}_3$, the derived-term automaton of $a^* \mid b^* \mid c^*$, is depicted on the right.



Proof (Theorem 2). We will prove $[\![\mathcal{A}_{\mathsf{E}}]\!](m) = [\![\mathsf{E}]\!](m)$ by induction on $m \in M$. If $m = \varepsilon$, then $[\![\mathcal{A}_{\mathsf{E}}]\!](m) = \mathsf{E}_\varepsilon = d(\mathsf{E})(\varepsilon) = \big[\!\big[d(\mathsf{E})\big]\!\big](\varepsilon) = [\![\mathsf{E}]\!](\varepsilon)$.

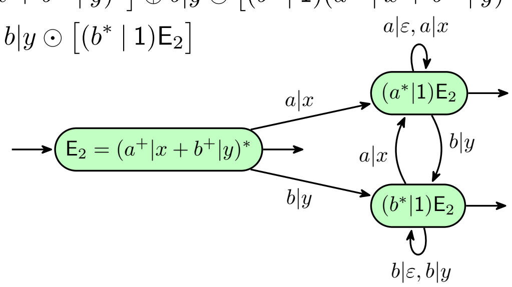
If m is not ε, then it can be generated in a (finite) number of ways: let $F(\mathsf{E}, m) \coloneqq \{(a, m_a) \in f(d(\mathsf{E})) \times M \mid m = a m_a\}$. $F(\mathsf{E}, m)$ is a function: for a given a, there is at most one m_a such that $(a, m_a) \in F(\mathsf{E}, m)$. Fig. 2 is helpful.

$$
\begin{aligned}
[\![\mathcal{A}_{\mathsf{E}}]\!](m) &= \sum_{(a,m_a)\in F(\mathsf{E},m)} \sum_{i\in[n_a]} \langle k_{a,i}\rangle [\![\mathcal{A}_{\mathsf{E}_{a,i}}]\!](m_a) && \text{by definition of } \mathcal{A}_{\mathsf{E}} \\
&= \sum_{(a,m_a)\in F(\mathsf{E},m)} \sum_{i\in[n_a]} \langle k_{a,i}\rangle [\![\mathsf{E}_{a,i}]\!](m_a) && \text{by induction hypothesis} \\
&= \sum_{(a,m_a)\in F(\mathsf{E},m)} \Big[\!\!\Big[\sum_{i\in[n_a]} \langle k_{a,i}\rangle \mathsf{E}_{a,i} \Big]\!\!\Big](m_a) && \text{by Lemma 1} \\
&= \sum_{(a,m_a)\in F(\mathsf{E},m)} \big[\!\big[d(\mathsf{E})(a) \big]\!\big](m_a) = \sum_{(a,m_a)\in F(\mathsf{E},m)} \big[\!\big[a \odot d(\mathsf{E})(a) \big]\!\big](a m_a) \\
&= \sum_{a\in f(d(\mathsf{E}))} \big[\!\big[a \odot d(\mathsf{E})(a) \big]\!\big](m) && F(\mathsf{E}, m) \text{ is a function} \\
&= \Big[\!\!\Big[\sum_{a\in f(d(\mathsf{E}))} a \odot d(\mathsf{E})(a) \Big]\!\!\Big](m) && \text{by Lemma 2} \\
&= \big[\!\big[d_\varepsilon(\mathsf{E}) \big]\!\big](m) && \text{by definition} \\
&= \big[\!\big[d(\mathsf{E}) \big]\!\big](m) && \text{since } m \neq \varepsilon \\
&= [\![\mathsf{E}]\!](m) && \text{by Proposition 3} \qquad \square
\end{aligned}
$$

Example 5. Let $\mathsf{E}_2 \coloneqq (a^+ \mid x + b^+ \mid y)^*$, where $\mathsf{E}^+ \coloneqq \mathsf{E}\mathsf{E}^*$. Its expansion is

$$
\begin{aligned}
d(\mathsf{E}_2) &= \langle 1\rangle \oplus a|x \odot \big[(a^* \mid 1)(a^+ \mid x + b^+ \mid y)^*\big] \oplus b|y \odot \big[(b^* \mid 1)(a^+ \mid x + b^+ \mid y)^*\big] \\
&= \langle 1\rangle \oplus a|x \odot \big[(a^* \mid 1)\mathsf{E}_2\big] \oplus b|y \odot \big[(b^* \mid 1)\mathsf{E}_2\big]
\end{aligned}
$$

Its derived-term automaton is:



5 Related Work

Multitape rational expressions have been considered early [11], but "an n-way regular expression is simply a regular expression whose terms are n-tuples of alphabetic symbols or ε" [9]. However, Kaplan and Kay [9] do consider the full generality of the semantics of operations on rational languages and rational relations, including $\times$, the Cartesian product of languages, and even use rational expressions more general than their definition. They do not, however, provide an explicit automaton construction algorithm, apparently relying on the simple inductive construction (using the Cartesian product between automata). Our $|$ operator on series was defined as the *tensor product*, denoted $\otimes$, by Sakarovitch [14, Sec. III.3.2.5], but without equivalent for expressions.

Brzozowski [4] introduced the idea of derivatives of expressions as a means to construct an equivalent automaton. The method applies to extended (unweighted) rational expressions, and constructs a deterministic automaton. Antimirov [3] modified the computation to rely on parts of the derivatives ("partial derivatives"), which results in nondeterministic automata.

Lombardy and Sakarovitch [10] extended this approach to support weighted expressions; independently, and with completely different foundations, Rutten [13] proposed a similar construction. Caron et al. [5] introduced support for (unweighted) extended expressions. Demaille [7] provides support for weighted extended expressions; expansions, originally mentioned by Brzozowski [4], are placed at the center of the construct, replacing derivatives, to gain independence with respect to the size of the alphabet, and efficiency. However, the proofs still relied on derivatives, contrary to the present work.

Based on (10) one could attempt to define a derivative-based version, with $\partial_{a|b}(\mathsf{E} \mid \mathsf{F}) \coloneqq \partial_a \mathsf{E} \mid \partial_b \mathsf{F}$, however this is troublesome on several regards. First, it would also require $\partial_{a|\varepsilon}$ and $\partial_{\varepsilon|b}$, whose semantics is dubious. Second, from an implementation point of view, that would lead to repeated computations of $\partial_a \mathsf{E}$ and of $\partial_b \mathsf{F}$, unless one would cache them, but that's exactly what expansions do. And finally observe that in the derived-term automaton in Example 5, the state $(a^* \mid \mathsf{1})(a^+ \mid x + b^+ \mid y)^*$ accepts words starting with a on the first tape, and y on the second, yet an outgoing transition on $a|y$ would result in a more complex automaton.

Alternative definitions of derivatives may exist[2], but anyway they would no longer be equivalent to taking the left-quotient of the corresponding language: $a|y$ *is* a viable prefix from this state.

Different constructions of the derived-term automaton have been discovered [1, 6]. They do not rely on derivatives at all. It is an open question whether these approaches can be adapted to support a tuple operator.

6 Conclusion

Our work is in the continuation of derivative-based computations of the derived-term automaton [3–5, 10]. However, we replaced the derivatives by expansions, which lifted the requirement for the monoid of labels to be free.

In order to support k-tape (weighted) rational expressions, we introduced a tupling operator, which is more compact and readable than simple expressions on k-tape letters. We demonstrated how to build the derived-term automaton for any such expressions.

Vcsn[1] implements the techniques exposed in this paper. Our future work aims at other operators, and studying more closely the complexity of the algorithm. The usual state-elimination method to compute an expression from an automaton works perfectly, however we are looking for means to reduce the expression size.

[2] Makarevskii and Stotskaya [11] define derivatives, but (i) in the case of expressions over tuples of letters, and (ii) only when in so-called "standard form", for which he notes "no method of constructing [an] n-expression in standard form for a regular n-expression is known."

Acknowledgments The author thanks the anonymous reviewers for their constructive comments, and A. Duret-Lutz, S. Lombardy, L. Saiu and J. Sakarovitch for their feedback during this work.

References

1. C. Allauzen and M. Mohri. A unified construction of the Glushkov, follow, and Antimirov automata. In *MFCS*, vol. 4162 of *LNCS*, pp. 110–121. Springer, 2006.
2. P.-Y. Angrand, S. Lombardy, and J. Sakarovitch. On the number of broken derived terms of a rational expression. *Journal of Automata, Languages and Combinatorics*, 15(1/2):27–51, 2010.
3. V. Antimirov. Partial derivatives of regular expressions and finite automaton constructions. *TCS*, 155(2):291–319, 1996.
4. J. A. Brzozowski. Derivatives of regular expressions. *J. ACM*, 11(4):481–494, 1964.
5. P. Caron, J.-M. Champarnaud, and L. Mignot. Partial derivatives of an extended regular expression. In *LATA*, vol. 6638 of *LNCS*, pp. 179–191. Springer, 2011.
6. J.-M. Champarnaud, F. Ouardi, and D. Ziadi. An efficient computation of the equation $\mathbb{K}$-automaton of a regular $\mathbb{K}$-expression. In *DLT*, vol. 4588 of *LNCS*. Springer, 2007.
7. A. Demaille. Derived-term automata for extended weighted rational expressions. Technical Report 1605.01530, arXiv, May 2016. URL `http://arxiv.org/abs/1605.01530`.
8. A. Demaille, A. Duret-Lutz, S. Lombardy, and J. Sakarovitch. Implementation concepts in Vaucanson 2. In *CIAA'13*, vol. 7982 of *LNCS*, pp. 122–133, July 2013.
9. R. M. Kaplan and M. Kay. Regular models of phonological rule systems. *Comput. Linguist.*, 20(3):331–378, Sept. 1994.
10. S. Lombardy and J. Sakarovitch. Derivatives of rational expressions with multiplicity. *TCS*, 332(1-3):141–177, 2005.
11. A. Y. Makarevskii and E. D. Stotskaya. Representability in deterministic multi-tape automata. *Cybernetics and System Analysis*, 5(4):390–399, 1969.
12. S. Owens, J. Reppy, and A. Turon. Regular-expression derivatives re-examined. *J. Funct. Program.*, 19(2):173–190, Mar. 2009.
13. J. J. M. M. Rutten. Behavioural differential equations: a coinductive calculus of streams, automata, and power series. *TCS*, 308(1-3):1–53, 2003.
14. J. Sakarovitch. *Elements of Automata Theory*. Cambridge University Press, 2009. Corrected English translation of *Éléments de théorie des automates*, Vuibert, 2003.
15. K. Thompson. Programming techniques: Regular expression search algorithm. *Commun. ACM*, 11(6):419–422, 1968.

A Appendix

A.1 Proof of Lemma 1

Proof (Lemma 1). The first three equations are straightforward to prove.

$$[\![\mathsf{P} \mid \mathsf{Q}]\!] = \left[\!\!\left[\bigoplus_{(i,j)\in[n]\times[m]} \langle k_i \cdot h_j\rangle \odot \mathsf{E}_i \mid \mathsf{F}_j \right]\!\!\right]$$

$$
\begin{aligned}
&= \sum_{(i,j)\in[n]\times[m]} \langle k_i \cdot h_j \rangle [\![\mathsf{E}_i \mid \mathsf{F}_j]\!] \\
&= \sum_{(i,j)\in[n]\times[m]} \langle k_i \cdot h_j \rangle [\![\mathsf{E}_i]\!] \mid [\![\mathsf{F}_j]\!] \\
&= \Big(\sum_{i\in[n]} \langle k_i \rangle [\![\mathsf{E}_i]\!]\Big) \mid \Big(\sum_{j\in[m]} \langle h_j \rangle [\![\mathsf{F}_j]\!]\Big) \\
&= \Big[\!\!\Big[\bigoplus_{i\in[n]} \langle k_i \rangle \odot \mathsf{E}_i\Big]\!\!\Big] \mid \Big[\!\!\Big[\bigoplus_{j\in[m]} \langle h_j \rangle \odot \mathsf{F}_j\Big]\!\!\Big] \\
&= [\![\mathsf{P}]\!] \mid [\![\mathsf{Q}]\!] \qquad \square
\end{aligned}
$$

A.2 Derived-Term Algorithm

Input : E, a rational expression
Output : $\langle E, I, T \rangle$ an automaton (simplified notation)

```
I(E) := 1_𝕂 ;                                   // Unique initial state
Q := Queue(E) ;                  // A work list (queue) loaded with E
while Q is not empty do
    E := pop(Q) ;                // A new state/expression to complete
    X := d(E) ;                                    // The expansion of E
    T(E) := X(ε) ;                // Final weight: the constant term
    foreach a ⊙ [P_a] ∈ X do       // For each first/polynomial in X
        foreach ⟨k⟩ ⊙ F ∈ P_a do   // For each monomial of P_a = X(a)
            E(E, a, F) := k ;                           // New transition
            if F ∉ Q then
                push(Q, F) ;  // F is a new state, to complete later
            end
        end
    end
end
```

A.3 Derived Terms

We will prove that the states of $\mathcal{A}_{\mathsf{E}}$ are actually members of $\mathrm{TD}(\mathsf{E})$ (and E itself), a *finite* set of expressions, called the *derived terms* of E. $\mathrm{TD}(\mathsf{E})$ admits a simple inductive definition.

Definition 8 (Derived Terms). *The* true derived terms *of an expression* E *is* $\mathrm{TD}(\mathsf{E})$*, the set of expressions defined inductively below:*

$$
\begin{aligned}
\mathrm{TD}(0) &:= \emptyset \\
\mathrm{TD}(1) &:= \emptyset \\
\mathrm{TD}(a) &:= \{1\} \qquad \forall a \in A \\
\mathrm{TD}(\mathsf{E}+\mathsf{F}) &:= \mathrm{TD}(\mathsf{E}) \cup \mathrm{TD}(\mathsf{F})
\end{aligned}
$$

$$\begin{aligned}
\mathrm{TD}(\langle k\rangle \mathsf{E}) &\coloneqq \mathrm{TD}(\mathsf{E}) \qquad \forall k \in \mathbb{K} \\
\mathrm{TD}(\mathsf{E}\langle k\rangle) &\coloneqq \{\mathsf{E}_i\langle k\rangle \mid \mathsf{E}_i \in \mathrm{TD}(\mathsf{E})\} \qquad \forall k \in \mathbb{K} \\
\mathrm{TD}(\mathsf{E}\cdot\mathsf{F}) &\coloneqq \{\mathsf{E}_i\cdot\mathsf{F} \mid \mathsf{E}_i \in \mathrm{TD}(\mathsf{E})\} \cup \mathrm{TD}(\mathsf{F}) \\
\mathrm{TD}(\mathsf{E}^*) &\coloneqq \{\mathsf{E}_i\cdot\mathsf{E}^* \mid \mathsf{E}_i \in \mathrm{TD}(\mathsf{E})\} \\
\mathrm{TD}(\mathsf{E}\mid\mathsf{F}) &\coloneqq (\mathrm{TD}(\mathsf{E}) \mid \mathrm{TD}(\mathsf{F})) \cup (\{1\} \mid \mathrm{TD}(\mathsf{F})) \cup (\mathrm{TD}(\mathsf{E}) \mid \{1\})
\end{aligned}$$

The derived terms *of an expression* E *is* $\mathrm{D}(\mathsf{E}) \coloneqq \mathrm{TD}(\mathsf{E}) \cup \{\mathsf{E}\}$.

Lemma 3 (Number of Derived Terms). *For any k-tape expression* E,

$$|\mathrm{TD}(\mathsf{E})| \leq \prod_{i\in[k]} (\|\mathsf{E}\|_i + 1) \quad .$$

Proof. It is simple to check by induction on E that for all cases, except tuple, $\mathrm{TD}(\mathsf{E}) \leq \|\mathsf{E}\|$ (which is the classical result for single-tape expressions). In the case of $|$, it is clear that $|\mathrm{TD}(\mathsf{E}\mid\mathsf{F})| \leq (|\mathrm{TD}(\mathsf{E})|+1)\cdot(|\mathrm{TD}(\mathsf{F})|+1)$, hence the result.

Lemma 4 (True Derived Terms and Single Expansion). *For any expression* E, $\mathsf{exprs}\big(d(\mathsf{E})\big) \subseteq \mathrm{TD}(\mathsf{E})$.

Proof. Established by a simple verification of Definition 6. □

The derived terms of derived terms of E are derived terms of E. In other words, repeated expansions never "escape" the set of derived terms.

Lemma 5 (True Derived Terms and Repeated Expansions). *Let* E *be an expression. For all* $\mathsf{F} \in \mathrm{TD}(\mathsf{E})$, $\mathsf{exprs}\big(d(\mathsf{F})\big) \subseteq \mathrm{TD}(\mathsf{E})$.

Proof. This will be proved by induction over E.

Case $\mathsf{E} = 0$ or $\mathsf{E} = 1$. Impossible, as then $\mathrm{TD}(\mathsf{E}) = \emptyset$.

Case $\mathsf{E} = a$. Then $\mathrm{TD}(\mathsf{E}) = \{1\}$, hence $\mathsf{F} = 1$ and therefore $d(\mathsf{F}) = d(1) = \langle 0_{\mathbb{K}}\rangle$, so $\mathsf{exprs}\big(d(\mathsf{F})\big) = \emptyset \subseteq \mathrm{TD}(\mathsf{E})$.

Case $\mathsf{E} = \mathsf{G} + \mathsf{H}$. Then $\mathrm{TD}(\mathsf{E}) = \mathrm{TD}(\mathsf{G}) \cup \mathrm{TD}(\mathsf{H})$. Suppose, without loss of generality, that $\mathsf{F} \in \mathrm{TD}(\mathsf{G})$. Then, by induction hypothesis, $\mathsf{exprs}\big(d(\mathsf{F})\big) \subseteq \mathrm{TD}(\mathsf{G}) \subseteq \mathrm{TD}(\mathsf{E})$.

Case $\mathsf{E} = \langle k\rangle\mathsf{G}$. Then if $\mathsf{F} \in \mathrm{TD}(\langle k\rangle\mathsf{G}) = \mathrm{TD}(\mathsf{G})$, so by induction hypothesis $\mathsf{exprs}\big(d(\mathsf{F})\big) \subseteq \mathrm{TD}(\mathsf{G}) = \mathrm{TD}(\langle k\rangle\mathsf{G}) = \mathrm{TD}(\mathsf{E})$.

Case $\mathsf{E} = \mathsf{G}\langle k\rangle$. Then $\forall \mathsf{F} \in \mathrm{TD}(\mathsf{G}\langle k\rangle) = \{\mathsf{G}_i\langle k\rangle \mid \mathsf{G}_i \in \mathrm{TD}(\mathsf{G})\}$, there exists an i such that $\mathsf{F} = \mathsf{G}_i\langle k\rangle$. Then $d(\mathsf{F}) = d(\mathsf{G}_i\langle k\rangle) = d(\mathsf{G}_i)\langle k\rangle$ hence $\mathsf{exprs}\big(d(\mathsf{F})\big) = \mathsf{exprs}\big(d(\mathsf{G}_i)\langle k\rangle\big)$.

Since $\mathsf{G}_i \in \mathrm{TD}(\mathsf{G})$, by induction hypothesis $\mathsf{exprs}\big(d(\mathsf{G}_i)\big) \subseteq \mathrm{TD}(\mathsf{G})$, so by definition of the right exterior product of expansions (and polynomials), $\mathsf{exprs}\big(d(\mathsf{G}_i)\langle k\rangle\big) \subseteq \mathrm{TD}(\mathsf{G}\langle k\rangle) = \mathrm{TD}(\mathsf{E})$.

Hence $\mathsf{exprs}\big(d(\mathsf{F})\big) \subseteq \mathrm{TD}(\mathsf{E})$.

Case $\mathsf{E} = \mathsf{G}\cdot\mathsf{H}$. Then $\mathrm{TD}(\mathsf{E}) = \{\mathsf{G}_i\cdot\mathsf{H} \mid \mathsf{G}_i \in \mathrm{TD}(\mathsf{G})\} \cup \mathrm{TD}(\mathsf{H})$.

- If $\mathsf{F} = \mathsf{G}_i \cdot \mathsf{H}$ with $\mathsf{G}_i \in \mathrm{TD}(\mathsf{G})$, then $d(\mathsf{F}) = d(\mathsf{G}_i \cdot \mathsf{H}) = d_p(\mathsf{G}_i) \cdot \mathsf{H} \oplus \big\langle d_\varepsilon(\mathsf{G}_i)\big\rangle d(\mathsf{H})$.
  Since $\mathsf{G}_i \in \mathrm{TD}(\mathsf{G})$ by induction hypothesis $\mathsf{exprs}\big(d_p(\mathsf{G}_i)\big) = \mathsf{exprs}\big(d(\mathsf{G}_i)\big) \subseteq \mathrm{TD}(\mathsf{G})$. By definition of the product of an expansion by an expression, $\mathsf{exprs}\big(d_p(\mathsf{G}_i) \cdot \mathsf{H}\big) \subseteq \{\mathsf{G}_j \cdot \mathsf{H} \mid \mathsf{G}_j \in \mathrm{TD}(\mathsf{G})\} \subseteq \mathrm{TD}(\mathsf{G} \cdot \mathsf{H}) = \mathrm{TD}(\mathsf{E})$.
- If $\mathsf{F} \in \mathrm{TD}(\mathsf{H})$, then by induction hypothesis $\mathsf{exprs}\big(d(\mathsf{F})\big) \subseteq \mathrm{TD}(\mathsf{H}) \subseteq \mathrm{TD}(\mathsf{E})$.

Case $\mathsf{E} = \mathsf{G}^*$. If $\mathsf{F} \in \mathrm{TD}(\mathsf{E}) = \{\mathsf{G}_i \cdot \mathsf{G}^* \mid \mathsf{G}_i \in \mathrm{TD}(\mathsf{G})\}$, i.e., if $\mathsf{F} = \mathsf{G}_i \cdot \mathsf{G}^*$ with $\mathsf{G}_i \in \mathrm{TD}(\mathsf{G})$, then $d(\mathsf{F}) = d(\mathsf{G}_i \cdot \mathsf{G}^*) = d_p(\mathsf{G}_i) \cdot \mathsf{G}^* \oplus \big\langle d_\varepsilon(\mathsf{G}_i)\big\rangle d(\mathsf{G}^*)$, so $\mathsf{exprs}\big(d(\mathsf{F})\big) \subseteq \mathsf{exprs}\big(d_p(\mathsf{G}_i) \cdot \mathsf{G}^*\big) \cup \mathsf{exprs}\big(d(\mathsf{G}^*)\big)$.[3] We will show that both are subsets of $\mathrm{TD}(\mathsf{E})$, which will prove the result.
Since $\mathsf{G}_i \in \mathrm{TD}(\mathsf{G})$, by induction hypothesis, $\mathsf{exprs}\big(d_p(\mathsf{G}_i)\big) = \mathsf{exprs}\big(d(\mathsf{G}_i)\big) \subseteq \mathrm{TD}(\mathsf{G})$, so by definition of a product of an expansion by an expression, $\mathsf{exprs}\big(d_p(\mathsf{G}_i) \cdot \mathsf{G}^*\big) \subseteq \{\mathsf{G}_j \cdot \mathsf{G}_j^* \mid \mathsf{G}_j \in \mathrm{TD}(\mathsf{G})\} = \mathrm{TD}(\mathsf{E})$.
By Lemma 4 $\mathsf{exprs}\big(d(\mathsf{G}^*)\big) \subseteq \mathrm{TD}(\mathsf{G}^*) = \mathrm{TD}(\mathsf{E})$.

Case $\mathsf{E} = \mathsf{G} \mid \mathsf{H}$. Let $\mathsf{F} \in \mathrm{TD}(\mathsf{E}) = \mathrm{TD}(\mathsf{G}) \mid \mathrm{TD}(\mathsf{H})$, i.e., let $\mathsf{F} = \mathsf{G}_i \mid \mathsf{H}_j$ with $\mathsf{G}_i \in \mathrm{TD}(\mathsf{G}), \mathsf{H}_j \in \mathrm{TD}(\mathsf{H})$, then by induction hypothesis $\mathsf{exprs}\big(d(\mathsf{G}_i)\big) \subseteq \mathrm{TD}(\mathsf{G})$ and $\mathsf{exprs}\big(d(\mathsf{H}_j)\big) \subseteq \mathrm{TD}(\mathsf{H})$. So, by definition of the tupling of expansions $\mathsf{exprs}\big(d(\mathsf{G}_i) \mid d(\mathsf{H}_j)\big) \subseteq \mathrm{TD}(\mathsf{G}) \mid \mathrm{TD}(\mathsf{H}) = \mathrm{TD}(\mathsf{E})$.
We have $d(\mathsf{F}) = d(\mathsf{G}_i|\mathsf{H}_j) = d(\mathsf{G}_i)|d(\mathsf{H}_j)$, so $\mathsf{exprs}\big(d(\mathsf{F})\big) = \mathsf{exprs}\big(d(\mathsf{G}_i) \mid d(\mathsf{H}_j)\big) \subseteq \mathrm{TD}(\mathsf{E})$. ☐

Lemma 6 (Derived Terms and Repeated Expansions). *Let* E *be an expression. For all* $\mathsf{F} \in \mathrm{D}(\mathsf{E})$, $\mathsf{exprs}\big(d(\mathsf{F})\big) \subseteq \mathrm{TD}(\mathsf{E})$.

Proof. Since $\mathrm{D}(\mathsf{E}) = \mathrm{TD}(\mathsf{E}) \cup \{\mathsf{E}\}$, this is an immediate consequence of Lemmas 4 and 5.

[3] Given two expansions $\mathsf{X}_1, \mathsf{X}_2$, $\mathsf{exprs}(\mathsf{X}_1 \oplus \mathsf{X}_2) \subseteq \mathsf{exprs}(\mathsf{X}_1) \cup \mathsf{exprs}(\mathsf{X}_2)$, but they may be different; consider for instance $\mathsf{X}_1 = a \odot [\langle 1\rangle \odot 1]$ and $\mathsf{X}_2 = a \odot [\langle -1\rangle \odot 1]$ with $\mathbb{K} = \mathbb{Z}$.