Rewriting Extended Regular Expressions

Valentin M. Antimirov[*] Peter D. Mosses[†]

November 1993

Abstract

We concider an extened algebra of regular events (languages) with intersection besides the usual operations. This algebra has the structure of a distributive lattice with monotonic operations; the latter property is crucial for some applications. We give a new complete Horn equational axiomatiztion of the algebra and develop some term-rewriting techniques for constructing logical inferences of valid equations.

A shorter version of this paper is to appear in the proceedings of *Developments in Language Theory, Univ. of Turku, July 1993*, published by World Scientific. The present version has been submitted for publication elsewhere.

1 Introduction

In this paper we consider an extended algebra of regular events (languages) on a given alphabet with *intersection* besides the usual operations (union, concatenation, Kleene star, empty, and the regular unit). This algebra has the structure of a *distributive lattice* (join is union, meet is intersection) with only *monotonic* operations. The latter property is crucial for some applications, for instance in the algebraic specification of abstract data types

[*] V.M.Glushkov Institute of Cybernetics, 252650, GSP, Kiev-207, Ukraine (SNG)

[†] Computer Science Department, Aarhus University, Ny Munkegade Bldg. 540, DK-8000 Aarhus C, Denmark

in the framework of so-called *unified algebras* [19], where sorts of values are themselves treated algebraically as values. Such specifications are used in *action semantic descriptions* of programming languages [20]; our extended algebra of regular events has been found to be particularly appropriate in connection with the description of various operations on semantic entities, as well as with that of abstract syntax.

In Sect. 3 we give a new Horn-equational axiomatization of the extended algebra of regnlar events on a possibly infinite alphabet $\mathcal{A}$, and prove its completeness for the ground equational theory of the algebra. The axiomatization is finite when $\mathcal{A}$ is finite. The axioms concerning the usual algebra of regular events are based on Salomaa's system [24], but the inference rule depending on the negation of the empty word property is replaced by an equational implication involving the meet operation. A new collection of equations then characterizes the meet operation. Our axiomatization exploits order-sorted equational logic [10] by introducing $\mathcal{A}$ as a subsort of the sort of all regular events.

In Sect. 4 we develop some term-rewriting techniques which lead to a simple and practical algebraic calculus for proving/disproving equations between extended regular expressions - avoiding construction of finite automata. The calculus is based on several rewrite systems; we have used the algebraic programming language OBJ [11] to implement and to experiment with these. We provide some examples of inferences obtained with the help of this calculus.

Finally, in Sect. 5 we review a large amount of related work, and consider possible improvements and extensions of our results. We also discuss a possible complexity advantage of our approach for deciding equations between particular forms of regular expressions, compared to any approach based on explicit construction of finite automata.

First we need some notational preliminaries.

2 Preliminaries

We are going to present the algebras of ordinary and extended regular expressions within the framework of order-sorted equational logic [10].

The signature *REG* of (ordinary) regular expressions includes: a sort *Reg* for regular expressions; a sort *Alph* for an alphabet which is a subsort of *Reg*; and constants and operation symbols:

$$\begin{array}{lcll}
\emptyset & : & \to \mathit{Reg} & \text{- the empty event (zero);} \\
\lambda & : & \to \mathit{Reg} & \text{- the regular unit;} \\
\cdot & : & \mathit{Reg} \times \mathit{Reg} \to \mathit{Reg} & \text{- concatenation;} \\
+ & : & \mathit{Reg} \times \mathit{Reg} \to \mathit{Reg} & \text{- union (join);} \\
{}^{*} & : & \mathit{Reg} \to \mathit{Reg} & \text{- iteration (Kleene star).}
\end{array}$$

To obtain the syntax of regular expressions over some alphabet $\mathcal{A}$, the signature *REG* is extended with an enumerated set of constants α_i of sort *Alph* in one-to-one correspondence with $\mathcal{A}$ (so we do not make distinction between the constants and the letters from $\mathcal{A}$). We do not assume in general that the alphabet is finite; we shall say explicitly when this assumption is made.

Given a set of variables Var (including those for both sorts *Reg* and *Alph*), let $\mathcal{T}(X)$ denote the set of all *REG*-terms (of sort *Reg*) with variables from $X \subset Var$, and $\mathcal{T}$ denote the set $\mathcal{T}(\emptyset)$ of *ground* *REG*-terms (without variables).

The signature $\mathit{REG}^{\&}$ is the enrichment of *REG* by the operation symbol:

$$\cap \; : \; \mathit{Reg} \times \mathit{Reg} \to \mathit{Reg} \quad \text{- intersection (meet).}$$

The set of all $\mathit{REG}^{\&}$-terms with variables from $X \subset Var$ is denoted as $\mathcal{T}^{\&}(X)$, similarly $\mathcal{T}^{\&}$ is the set of ground $\mathit{REG}^{\&}$-terms.

Let $\mathbf{Reg}[\mathcal{A}]$ be the set of regular events (sets of words, languages) over $\mathcal{A}$ with the standard interpretation of the sorts, constants and operation symbols of the signature *REG*. To make it an order-sorted *REG*-algebra, we identify each element $\alpha_i \in \mathcal{A}$ with the corresponding one-element event $\{\alpha_i\} \in \mathbf{Reg}[\mathcal{A}]$.

The algebra $\mathbf{Reg}[\mathcal{A}]$ is known to be closed under intersections, so it can be enriched to the $\mathit{REG}^{\&}$-algebra $\mathbf{Reg}^{\&}[\mathcal{A}]$ with the operation symbol $\cap$ interpreted as intersection. (Note that this holds true even for an *infinite* alphabet $\mathcal{A}$ - in contrast to the case when one is going to enrich $\mathbf{Reg}[\mathcal{A}]$ with the complement operation.)

The restriction of (the interpretation of) $\cap$ to the (carrier of the) sort *Alph* is to satisfy the following condition:

$$\alpha \cap \beta = \emptyset \iff \alpha \neq \beta \qquad (1)$$

for all constants α, β from $\mathcal{A}$. Similarly, the intersection of non-equal words from $\mathcal{A}^*$ is $\emptyset$. Thus the algebra $\mathbf{Reg}^{\&}[\mathcal{A}]$ has the structure of an *atomic distributive lattice* with the join $+$, the meet $\cap$, the bottom $\emptyset$, and atoms from $\mathcal{A}^*$.

This provides a standard interpretation of ground $REG^{\&}$-terms for any given alphabet $\mathcal{A}$; let

$$int : \mathcal{T}^{\&} \rightarrow \mathbf{Reg}^{\&}[\mathcal{A}]$$

denote the corresponding interpretation function (in fact, the uniquely defined homomorphism from the absolutely free term algebra $\mathcal{T}^{\&}$). The interpretation of (non-ground) $REG^{\&}$-terms $\mathcal{T}^{\&}(X)$ is defined by the unique homomorphic extension $\theta^{\#} : \mathcal{T}^{\&}(X) \rightarrow \mathbf{Reg}^{\&}[\mathcal{A}]$ of a given variable assignment $\theta : X \rightarrow \mathbf{Reg}^{\&}[\mathcal{A}]$.

Equations and other first-order formulas are interpreted in $\mathbf{Reg}^{\&}[\mathcal{A}]$ as usual, e.g.:

$$\mathrm{Reg}^{\&}[\mathcal{A}] \models \mathbf{t_1} = \mathbf{t_2} \iff \theta^{\#}(t_1) = \theta^{\#}(t_2) \qquad (2)$$

for all assignments $\theta : X \rightarrow \mathbf{Reg}^{\&}[\mathcal{A}]$, etc.

An equation is called ground if it consists of ground terms. The set of (ground) equations valid in an algebra, or in a class of those, $\mathcal{B}$ is denoted by $Eg(\mathcal{B})$ (correspondingly, by $GEq(\mathcal{B})$); this set is also called the (ground) equational theory of $\mathcal{B}$.

Given a set of universal Horn-equations (i.e., equations and equational implications) E, we write $E \vdash t_1 = t_2$ to assert that the equation $t_1 = t_2$ can be inferred from the set E using order-sorted equational calculus [10]. Given a set of equations Γ, the notation $E \vdash \Gamma$ means, as usual, that $E \vdash e$ for each $e \in \Gamma$.

The set of order-sorted Horn-equations E is said to be *complete* for (the ground equational theory of) the algebra $\mathcal{B}$ if $E \vdash GEq(\mathcal{B})$ holds. (Regarding E valid in $\mathcal{B}$, this completeness holds exactly when $\mathcal{B}$ is the *initial algebra* of the quasivariety defined by E.)

In the second half of this paper some standard notions from term-rewriting theory will be used. We shall keep our notations and terminology compatible with those of Dershowitz and Jouannaud [7].

In what follows we sometimes omit the alphabet and denote the algebras introduced above just **Reg** and $\mathbf{Reg}^{\&}$.

Table1: The axiom system AX

For all $a, b, c : Reg$, for alI $x, y : Alph$,

$$\begin{aligned}
a + (b + c) &= (a + b) + c && (A1)\\
(a \cdot b) \cdot c &= a \cdot (b \cdot c) && (A2)\\
a + b &= b + a && (A3)\\[1em]
a \cdot (b + c) &= a \cdot b + a \cdot c && (A4)\\
(a + b) \cdot c &= a \cdot c + b \cdot c && (A5)\\[1em]
a + a &= a && (A6)\\
a \cdot \lambda &= a && (A7)\\
a \cdot \emptyset &= \emptyset && (A8)\\
a + \emptyset &= a && (A9)\\[1em]
\lambda + a \cdot a^* &= a^* && (A10)\\
(\lambda + a)^* &= a^* && (A11)\\[1em]
(\lambda \cap b = \emptyset) \wedge (a = b \cdot a + c) \Longrightarrow a &= b^* \cdot c && (A12)\\[1em]
\lambda \cap (a \cdot b) &= (\lambda \cap a) \cap b && (A13)\\
\lambda \cap a^* &= \lambda && (A14)\\
\lambda \cap x &= \emptyset && (A15)\\[1em]
\emptyset \cap a &= \emptyset && (A16)\\
a \cap a &= a && (A17)\\
a \cap b &= b \cap a && (A18)\\[1em]
a \cap (b \cap c) &= (a \cap b) \cap c && (A19)\\
a \cap (b + c) &= (a \cap b) + (a \cap c) && (A20)\\
a + (a \cap b) &= a && (A21)\\[1em]
(x \cdot a) \cap (y \cdot b) &= (x \cap y) \cdot (a \cap b) && (A22)\\
(a \cdot x) \cap (b \cdot y) &= (a \cap b) \cdot (x \cap y) && (A23)\\
\alpha_i \cap \alpha_j &= \emptyset \text{ for all } \alpha_i \neq \alpha_j && (A24)
\end{aligned}$$

3 Axiomatization of the Algebra of Extended Regular Expressions $\mathbf{Reg}^{\&}$

The problem we consider in this section can be formulsted precisely as follows: given the alphabet $\mathcal{A}$, to find a finite set AX of Horn-equations over the signature $\mathrm{REG}^{\&}$ valid in the algebra $\mathbf{Reg}^{\&}[\mathcal{A}]$ and complete for its ground equational theory $GEq(\mathbf{Reg}^{\&}[\mathcal{A}])$. Moreover, we would like to avoid axioms involving the use of all letters from $\mathcal{A}$ (such as those given by Salomaa [25]) so as to be able to capture the case of an infinite (countable) alphabet $\mathcal{A}$ as well. In other words, we are looking for a "generic algebraic specification" of $\mathbf{Reg}^{\&}[\mathcal{A}]$ for all possible $\mathcal{A}$.

We approach the problem in two steps. First we consider a finite set of Horn-equations which is intended to axiomatize $\mathbf{Reg}[\mathcal{A}]$. Then we extend this by new axioms for the meet in order to get a complete axiomatization of the ground equational theory of $\mathbf{Reg}^{\&}[\mathcal{A}]$.

3.1 Axioms for Reg

Several different axiomatizations of the algebra of regular events have been suggested [24, 6, 14, 12, 3, 17, 15]. Some of these are finitary, some involve infinite sets of identities presented by finite numbers of schemes. (A deep analysis of this latter kind of systems of identities has been presented by Conway [6] and Krob[17].)

Since our main concern is axiomatization of the meet, it does not make much difference for us which particular system of axioms, complete for $\mathbf{Reg}[\mathcal{A}]$, to choose: the only requirement is that it should be presented by a finite set of Horn-equations. We are going to obtain such a system by an easy modification of the system F_1 suggested by Salomaa [24]. The corresponding set of equational axioms is given by $(A1) - (A11)$ in Table 1.

Salomaa used $\emptyset^*$ instead of λ, but in fact the equation $\emptyset^* = \lambda$ will be derivable from the full set of our axioms. For technical reasons we take the equations $(A7)$, $(A8)$, and $(A10)$ to be dual to the corresponding ones in F_1. This duality is determined by the automorphism *rev* on $\mathbf{Reg}$ that maps each word to the reverse one, i.e.,

$$rev(\lambda) \;=\; \lambda \qquad (3)$$

$$
\begin{aligned}
rev(x) &= x && (4)\\
rev(a \cdot b) &= rev(b) \cdot rev(a) && (5)\\
rev(a + b) &= rev(a) + rev(b) && (6)\\
rev(a^*) &= rev(a)^* && (7)
\end{aligned}
$$

for all $x \in \mathcal{A}, a, b \in \mathbf{Reg}$. Therefore, each inference in F_1 can be translated to a "dual" one in our system.

The system F_1 included two inference rules: the substitution rule and the "solution of equations" rule. We do not need the first one as it is a part of the order-sorted equational logic. As for the second one, it was based on the *empty word property* (e.w.p.). This property of a regular expression r can be expressed equationally as follows:

$$r \text{ possesses e.w.p.} \iff r + \lambda = r. \qquad (8)$$

The problem, however, is that the inference rule involves the *negation* of e.w.p.:

$$\frac{b \text{ does not possess e.w.p.}, \; a \;=\; a \;\cdot\; b \;+\; c}{a \;=\; c \;\cdot\; b^*} \qquad (9)$$

The use of such a non-logical premise in this rule - "does not possess e.w.p." has given rise to certain objections [6, 15]. Really, the negation of e.w.p. cannot be expressed by a universal equation within *REG*, so this rule can't be considered as a Horn-equation either.

However, this problem disappears in the enrichment of *REG* by the meet, for it becomes possible to express the negation of the e.w.p. equationally (this was already noticed by Salomaa and Tixier [26]):

$$a\text{does not possess e.w.p.} \iff a \cap \lambda = \emptyset \qquad (10)$$

This allows us to introduce the equational implication $(A12)$ in Table 1, which plays the same role in our system as the rule (9) plays in F_1. (For the reasons discussed above, the equations in the premise and the conclusion of the implication are slight modifications of those in the inference rule.)

Note that the set of axioms $(A1) - (A12)$ is *not* complete even for the ground equational theory of $\mathbf{Reg}[\mathcal{A}]$: the last axiom involves the meet, hence one obviously needs further axioms. However, our goal is to axiomatize $GEq(\mathbf{Reg}^{\&}[\mathcal{A}])$ (which includes $GEq(\mathbf{Reg}[\mathcal{A}])$), so we need some axioms for the meet anyway. Let's turn to this problem now.

3.2 Axioms for Meet and Completeness.

To axiomatize $\mathbf{Reg}^{\&}$, we take the axioms $(A1)-(A12)$ and add the remaining equations given in Table 1, $(A13)-(A24)$, reflecting properties of the meet. Note that $(A24)$ is a scheme describing a family of equations. Note aIso that the axioms $(A22)-(A23)$ reflect the "restricted" distributivity of meet w.r.t. concatenation and this cannot be extended to the full distributive law

$$(\forall a, b, c, d : \mathit{Reg}) \quad (a \cdot c) \cap (b \cdot d) \;=\; (a \cap b) \cdot (c \cap d) \tag{11}$$

which is is not valid in $\mathbf{Reg}^{\&}[\mathcal{A}]$. (This was one of the main motivations for us to exploit the order-sorted language, which allows to express the restricted distributivity in a natural and still strictly formal way.)

Now let AX denote the extended set of axioms $(A1)-(A24)$. The main result of this part of the paper is that AX is complete (and, of course, sound) for the ground equational theory of $\mathbf{Reg}^{\&}[\mathcal{A}]$. Moreover, this is a finite axiomatization whenever the alphabet $\mathcal{A}$ is finite.

A direct proof of this statement would be rather long and tedious, so we prefer to make use of the completeness of the system F_1. Yet we should be careful at this point, since we have reformulated the rule for solving equations into the implication $(A12)$, which uses the meet in the antecedent. Still the following fact holds:

Proposition 1. *The set of axioms* $(A1)-(A20)$ *is complete for* $\mathit{GEq}(\mathbf{Reg}[\mathcal{A}])$.
Proof. One can observe that for any ground REG-term t the expression $\lambda \cap t$ can be reduced to either $\emptyset$ or λ by succinct applications of $(A13)-(A15)$ together with the following three logical consequences of $(A13)-(A20)$:

$$\lambda \cap \emptyset, \quad \lambda \cap \lambda, \quad \lambda \cap (a+b) = (\lambda \cap a) + (\lambda \cap b) \tag{12}$$

(proof by induction on the structure of $t \in \mathcal{T}$). This allows to eliminate the premise $\lambda \cap a = \emptyset$ from the antecedent of $(A12)$ for any ground REG-term a satisfying e.w.p., and then one can apply literally all the constructions of Salomaa [24] used in the completeness proof of F_1. □

The following theorem states a "sufficient completeness" result: the set of new axioms we have introduced suffices to eliminate meets from ground $\mathit{REG}^{\&}$-terms. (This is, of course, a *proof-theoretic* statement about the specific set of axioms, rather than a reformulation of the corresponding model-theoretic property that **Reg** is closed under the meet; the latter can be easily proved by well-known finite automata techniques.)

Theorem 2. *For any ground REG$^{\&}$-term t there exists a ground REG-term t' such that* $AX \vdash t = t'$.
Proof. In the Appendix. (We put the proof in the Appendix because it uses some notions which will be introduced in the next section.) □

Corollary 3. *For all ground terms* $t_1, t_2 \in \mathcal{T}^{\&}$ *we have*

$$AX \vdash t_1 = t_2 \iff \mathbf{Reg}^{\&}[\mathcal{A}] \models t_1 = t_2. \tag{13}$$

In other words, AX is a sound and complete axiomatization of GEq($\mathbf{Reg}^{\&}[\mathcal{A}]$).
Proof. The non-trivial direction of the equivalence (completeness) can be proved as follows.

Let $\mathbf{Reg}^{\&} \models t_1 = t_2$ for some $t_1, t_2 \in \mathcal{T}^{\&}$. By Thm. 2 there exist some terms $t'_1, t'_2 \in \mathcal{T}$ such that $AX \vdash t_1 = t'_1$ and $AX \vdash t_2 = t'_2$. By Prop. 1 the equation $t'_1 = t'_2$ can be deduced from AX. Combining these two facts, we obtain $AX \vdash t_1 = t_2$. □

Thus we reach our goal to finitely axiomatize $\mathbf{Reg}^{\&}[\mathcal{A}]$ in the case of a finite alphabet $\mathcal{A}$.

Let's consider now what happens when $\mathcal{A}$ is infinite (countable). In this case our axiomatization becomes infinite, since the scheme ($A24$) then describes an infinite set of ground identities.

One obvious way to amend this is to replace the scheme ($A24$) by the equivalence in (1), or, at least, by the implication

$$(\forall\ x, y : \mathit{Alph})\ x \neq y \implies x \cap y = \emptyset \tag{14}$$

which is closely related to the very last inference rule (*Rule 3*) of Salomaa and Tixier [26].

However, this implication is not a Horn-equation: actually, it is equivalent to a universally quantified disjunction $(\forall x, y : \mathit{Alph}) x = y \vee x \cap y = \emptyset$. Thus one would need a richer logic than the order-sorted equational one to deal with such an axiom, e.g., the full first-order logic with equality or some universal fragment including disjunction. It seems not quite appropriate to involve such a general logical system in order to axiomatize just an *equational* theory.

But we can approach the problem from another side. It seems to be more reasonable to consider the infinite alphabet $\mathcal{A}$ not just as a set of constants, but as a set of terms over some *finite* signature $\sum_{\mathcal{A}}$, or even more generally - a finitely-generated $\sum_{\mathcal{A}}$-algebra axiomatized by a set of (Horn-) equations $E_{\mathcal{A}}$. Now the algebras $\mathbf{Reg}[\mathcal{A}]$ and $\mathbf{Reg}^{\&}[\mathcal{A}]$ are supposed to be enrichments of the "alphabet algebra" $\mathcal{A}$, and one could hope to axiomatize the property (1) by a finite set of (Horn-) equations (over the signature $\sum_{\mathcal{A}} \cup REG^{\&}$) which then could replace (14) and give a truly Horn-equational finite axiomatization of $\mathbf{Reg}^{\&}[\mathcal{A}]$.

A detailed implementation of this programme goes beyond the scope of this paper; let us just point out that the unified algebra of tuples of natural numbers [20] gives a good illustration of this construction.

Now we turn to the question of proving equations in $\mathbf{Reg}^{\&}$ using our axiomatization. This is the subject of the next section.

4 Inferring Equations in $\mathbf{Reg}^{\&}$ by Rewriting

The word problem in **Reg** is decidable and it is well known how to (dis)prove a REG-equation $t_1 = t_2$: to construct minimal deterministic finite automata (DFA) for both regular expressions t_1, t_2 and to check whether they are isomorphic. The same holds true for ground $REG^{\&}$-equations: there are known procedures for constructing an "intersection" of a given pair of DFA.

However, we are going to address a somewhat different problem: how to prove equations in the algebra $\mathbf{Reg}^{\&}$ by *logical* methods.

Once we have a complete set of axioms, we can, in principle, infer from it any valid ground equation in $\mathbf{Reg}^{\&}$. The only problem is how to find such an inference. Actually, the completeness proof of Salomaa [24] is constructive and offers an algorithm for producing inferences of valid equations in **Reg**. The same is true for our Prop. 1 and Thm. 2, so combining these we do have an algorithm for constructing logical inferences from AX. But this would give rather long and complicated inferences. and it would be too tedious to use it in practice, even for solving small exercises.

In this section we suggest a much more practical method for proving and disproving ground equations in $\mathbf{Reg}^{\&}$. We shall present it in the form of yet another inference system still closely related to the above axiomatization. To describe it, we first need to introduce some term-rewriting techniques dealing with ground $REG^{\&}$-terms.

4.1 Linear Forms for Extended Regular Expressions

We define a set of *linear* terms $Lin \subset \mathcal{T}^{\&}$ by the following grammar:

$$Item \ ::= \ Alph \cdot Reg \tag{15}$$

$$Sum \ ::= \ ltem \mid Item + Sum \tag{16}$$

$$Lin \ ::= \ \emptyset \mid Sum \tag{17}$$

(Within) the order-sorted or unified algebras framework this can be naturally formulated as enriching the signature $REG^{\&}$ by a chain of new subsorts of $Reg : Item < Sum < Lin < Reg$.)

This implies that any term $l \in Lin$ is either $\emptyset$ or has the form of a sum of items $x_1 \cdot r_1 + \ldots + x_n \cdot r_n$ for some constants $x_i \in \mathcal{A}$ and terms $r_i \in \mathcal{T}^{\&}$, $i = 1, \ldots, n$. We say that an item $x \cdot r$ has the *head* x and the *tail* r. We also say that a $REG^{\&}$-term t is *in linear form* if $t \in Lin$.

Deflnition 4. *Given a linear term $t \in Lin$, let Hd(t) denote the set of all heads of items of t, and Tl(t) denote the set of all tails of items of t.*

Definition 5. *The linear term t is said to be* deterministic *(or* in deterministic linear form*) iff either it is $\emptyset$ or all the heads of its items are distinct and $Tl(t) \subset \mathcal{T}^{\&} \setminus \{\emptyset\}$.*

We shall use the notation $\sum_{x \in Hd(l)} x \cdot r_x$ to denote a deterministic linear term l as a (possibly empty) sum of its items $x \cdot r_x$. This sum denotes the term $\emptyset$ if the set of heads $Hd(l)$ is empty.

The following facts can be proved by straightforward induction on the structure of ground terms.

Proposition 6. *For any $t \in \mathcal{T}^{\&}$ there is some $l \in Lin$ such that $AX \vdash t = \lambda \cap t + l$. For any $l \in Lin$ there is some deterministic $l' \in Lin$ such that $AX \vdash l = l'$.* □

It follows that any ground $REG^{\&}$-term t can be represented in the following form:

$$t = o(t) + \sum_{x \in hd(l)} x \cdot r_x \tag{18}$$

where $o(t)$ denotes the "constant part" of t, which is equal to either $\emptyset$ or λ. (This recalls a classical result [24, 26, 25] that every (extended) regular expression can be *equationally characterized*; note however that we do not use the sum over the whole alphabet $\mathcal{A}$, which may be infinite!) Yet the representation is not unique (even modulo all the equations in AX): there are different but equal in $\mathbf{REG}^{\&}$ linear terms. This requires us to be more specific in the definition of the representation in (18) and to provide a constructive procedure for calculating linear forms.

One possible way to do this is to introduce special operations on the term algebra $\mathcal{T}^{\&}$ such as *deriuatives* [4, 6] (also called *left quotients* [5] and *left residuals* [21]), but this would involve new equations to define these operations and would make the inference system more complicated.

We prefer to resolve the problem by providing a parlicular *strategy* for applying equations from AX to reduce a term to the required form; on the way we shall also obtain the derivatives, without introducing special equations for them.

The strategy is presented through the rewrite system LF given in Table 2, modulo associativity of the concatenation '$\cdot$' and associativity and commutativity of the join $+$ and the meet $\cap$. (In fact, this is an algebraic program - we have implemented it in OBJ3 [11].) The system includes an auxiliary unary function

$$f : Reg \rightarrow Lin$$

whose rôle is to calculate the non-constant part of the representation in (18). Note how this function is used in the rewrite rules ($L23$) and ($L28$) to control applications of the axiom ($A10$) in Table 1, in order to provide limited "unfolding" of starred expressions.

The system LF is terminating and provides the (unique) normal form $LF(t)$ of any term over the signature $REC^{\&} \cup \{f\}$. The following proposition (which is just a constructive variant of Prop. 6) ensures that the system LF

does its job properly.

Proposition 7. *Given $t \in \mathcal{T}^{\&}$, the following facts hold:*

1. $LF(\lambda \cap t) \in \{\emptyset, \lambda\}$.
2. $LF(f(t))$ *is in deterministic linear form.*
3. $AX \vdash t = LF(\lambda \cap t) + LF(f(t))$. □

Table 2: The rewrite system LF

For $x, y, z : \mathit{Alph}$ and $a, b, c : \mathit{Reg}$,

$$\emptyset^* \rightarrow \lambda \tag{$L1$}$$
$$\lambda^* \rightarrow \lambda \tag{$L2$}$$
$$\emptyset + a \rightarrow a \tag{$L3$}$$
$$a + a \rightarrow a \tag{$L4$}$$
$$\emptyset \cdot a \rightarrow \emptyset \tag{$L5$}$$
$$a \cdot \emptyset \rightarrow \emptyset \tag{$L6$}$$
$$\lambda \cdot a \rightarrow a \tag{$L7$}$$

$$\emptyset \cap a \rightarrow \emptyset \tag{$L8$}$$
$$a \cap a \rightarrow a \tag{$L9$}$$
$$\lambda \cap x \rightarrow \emptyset \tag{$L10$}$$
$$\lambda \cap (a \cdot b) \rightarrow (\lambda \cap a) \cap b \tag{$L11$}$$
$$\lambda \cap a^* \rightarrow \lambda \tag{$L12$}$$

$$x \cap y \rightarrow \textbf{if } x = y \textbf{ then } x \textbf{ else } \emptyset \tag{$L13$}$$

$$(x \cdot a) \cap (y \cdot b) \rightarrow (x \cap y) \cdot (a \cap b) \tag{$L14$}$$
$$(x \cdot a) \cap y \rightarrow (x \cap y) \cdot (a \cap \lambda) \tag{$L15$}$$
$$(a + b) \cap c \rightarrow (a \cap c) + (b \cap c) \tag{$L16$}$$

$$x \cdot b + x \rightarrow x \cdot (b + \lambda) \tag{$L17$}$$
$$x \cdot b + x \cdot c \rightarrow x \cdot (b + c) \tag{$L18$}$$

$$f(\emptyset) \rightarrow \emptyset \tag{$L19$}$$
$$f(\lambda) \rightarrow \emptyset \tag{$L20$}$$
$$f(x) \rightarrow x \cdot \lambda \tag{$L21$}$$
$$f(x \cdot a) \rightarrow x \cdot a \tag{$L22$}$$
$$f(a^* \cdot b) \rightarrow f(a) \cdot a^* \cdot b + f(b) \tag{$L23$}$$

$$f((a + b) \cdot c) \rightarrow f(a \cdot c) + f(b \cdot c) \tag{$L24$}$$
$$f((a \cap b) \cdot c) \rightarrow (f(a) \cap f(b)) \cdot f(c) + (\lambda \cap a \cap b) \cdot f(c) \tag{$L25$}$$
$$f(a + b) \rightarrow f(a) + f(b) \tag{$L26$}$$
$$f(a \cap b) \rightarrow f(a) \cap f(b) \tag{$L27$}$$
$$f(a^*) \rightarrow f(a) \cdot a^* \tag{$L28$}$$

This provides a particular representation as in (18) for each ground *REG*-term t and we can now define unambiguously the functions

$$o : \mathcal{T}^{\&} \to \{\emptyset, \lambda\},\ \ell : \mathcal{T}^{\&} \to Lin,\quad \partial : \mathcal{A} \times \mathcal{T}^{\&} \to \mathcal{T}^{\&}$$

as follows:

$$o(t) \;=\; LF(\lambda \cap t); \tag{19}$$

$$\ell(t) \;=\; LF(f(t)); \tag{20}$$

$$\partial_x(t) \;=\; \begin{cases} r & \text{if } x \cdot r \text{ occurs in } \ell(t) \\ \emptyset & \text{otherwise} \end{cases} \tag{21}$$

The latter function calculates derivatives (left residuals) of its second argument, because

$$init(\partial_x(t)) \;=\; \{w \in \mathcal{A}^* \mid x \cdot w \in init(t)\} \tag{22}$$

holds for all $x \in \mathcal{A}, t \in \mathcal{T}^{\&}$.

The following inductive definition extends the function ∂ on its first argument to the whole set $\mathcal{A}^*$:

$$\partial_\lambda(t) \;=\; t, \tag{23}$$

$$\partial_{x \cdot w}(t) \;=\; \partial_w(\partial_x(t)) \tag{24}$$

for any $x \in \mathcal{A}$, $w \in \mathcal{A}^*$. Here we get the *word derivatives* of t.

The fundamental fact about the word derivatives is that only a finite number of those of a given term $t \in \mathcal{T}^{\&}$ are *dissimilar*, i.e., distinct w.r.t. a restricted subset of equations $E \subset AX$ [14, 24]. (Recall that it suffices to include into E only the basic monoid and lattice axioms, even without distributivity and absorption - i.e., $(A1) - (A3)$, $(A6) - (A9)$ and $(A13) - (A19)$. But the set can (and should!) be extended for practical purposes, cf. discussion of this point in the next subsection.) Given such a set E, let $\mathcal{D}_E(t)$ denote this finite set of dissimilar w.r.t. E word derivatives of the ground $REG^{\&}$-term t.

The use of linear forms leads us to a respectably simple method for eliminating meets from extended regular expressions presented in the proof of Thm. 2 (cf. the Appendix), as well as to a new inference system for proving equations in $\mathbf{Reg}^{\&}$.

4.2 Inferring Equations in $\mathbf{Reg}^{\&}$

Here we present a new inference system which includes two components: a set of rewrite rules SIM for "simplifying" regular expressions and a set of transformation rules TR implementing a complete strategy for proving/refuting ground $REG^{\&}$-equations. (The system TR also involves the rewrite system LF for computing linear forms.)

The rewrite system SIM may be chosen more or less arbitrarily; the only requirements are that 1) it should be terminating and 2) the congruence $\equiv_{SIM}$ on $\mathcal{T}^{\&}$, generated by SIM, must be sufficiently strong to make the set of derivatives $\mathcal{D}_{SIM}(t)$ finite for any ground $REG^{\&}$-term t.

For instance, the rewrite system consisting of the (oriented from left to right) equations $(A6) - (A9)$ and $(A13) - (A17)$ modulo (non-oriented) equations $(A1) - (A3)$ and $(A18) - (A19)$ would satisfy both requirements. However, in order to make the inferences shorter, it is useful to include in SIM also such (oriented) equations as $(A10) - (A12)$, $(A21)$, as well as

$$\begin{aligned} (a^*)^* &\rightarrow a^*, & (25)\\ (a^* + b)^* &\rightarrow (a+b)^*, & (26)\\ (a^* \cap b^*)^* &\rightarrow a^* \cap b^*, & (27) \end{aligned}$$

and possibly some further rewrite rules. (The more rules are used here, the more equations $a = b$ valid in $\mathbf{Reg}^{\&}$ can be proved by just reducing a and b to the same normal form by SIM, but the more expensive the calculations become - due to the use of associative-commutative pattern matching etc.)

The idea of the second - actually, the main - component TR of our inference system comes from the following observation. Suppose we are going to check whether the ground equation $a = b$ is valid in $\mathbf{Reg}^{\&}$. We have $\mathbf{Reg}^{\&} \models a = b$ if and only if $o(a) = o(b)$ and each item of $\ell(a)$ is equal (in $\mathbf{Reg}^{\&}$) to some item of $\ell(b)$ and vice versa. Then, an item $x \cdot \partial_x(a)$ is equal to an item $x \cdot \partial_x(b)$ if and only if the equation $\partial_x(a) = \partial_x(b)$ is valid in $\mathbf{Reg}^{\&}$. Proceeding in this way, we can "unfold" the initial equation into an equivalent conjunction of equations of corresponding derivatives of a and b. The crucial point here is that when proving the latter, the initial equation $a = b$ can be used as a kind of "inductive hypothesis": if $a = b$ reappears as a member of the conjunction, it can be removed from the set of equations to be proved. This can be formulated more precisely as the following inference rule:

$$\frac{\lambda \cap c = \emptyset,\ a = c \cdot a + a',\ b = c \cdot b + b',\ a' = b'}{a = b} \tag{28}$$

for all $a, a', b, b', c : Reg$.

Proposition 8. *(28) i.s a derived inference rule for the theory AX, i.e., $AX \vdash a = b$ whenever the premises are derivable from AX.*

Proof. By obvious application of $(A12)$ to the second and third premises. □

This derived inference rule, combined with the use of linear forms, leads to a pretty simple strategy for inferring ground equations in $\mathbf{Reg}^{\&}$. To describe it, we need a couple of auxiliary constructions.

Let $Eq = \mathcal{T}^{\&} \times \mathcal{T}^{\&}$ be the set of ground equations represented as pairs of terms, we denote a pair $e \in Eq$ as $t_1 \simeq t_2$. Let $Set[Eq]$ be a data structure representing conjunctions (sets) of equations $e \in Eq$ (so that *true* corresponds to the empty set and $\cup$ corresponds to conjunction).

Table 3: The transformation system TR

$$\begin{array}{lcll}
\langle (\emptyset = \lambda) \wedge S, H \rangle & \Rightarrow & \langle false,\ H \rangle; & (DIS) \\
\langle (a = b) \wedge S,\ H \rangle & \Rightarrow & \langle S,\ H \rangle \ \mathbf{if}\ a \equiv_{SIM} b; & (SIM) \\
\langle (a = b) \wedge S,\ H \rangle & \Rightarrow & \langle S,\ H \rangle \ \mathbf{if}\ a = b)\ in\ H; & (IND) \\
\langle (a = b) \wedge S,\ H \rangle & \Rightarrow & \langle split(a = b) \wedge S,\ H \wedge (a = b) \rangle & \\
 & & \mathbf{if} \neg ((a = b)\ in\ H) & (SPL)
\end{array}$$

We also need a special membership predicate

$$in : Eq \times Set[Eq] \rightarrow Bool$$

defined as follows: $t_1 \simeq t_2 inH$ iff there is a pair $t_1' \simeq t_2' \in H$ such that

$$(t_1 \equiv_{SIM} t_1' \wedge t_2 \equiv_{SIM} t_2') \quad \vee \quad (t_1 \equiv_{SIM} t_2 \wedge t_2 \equiv_{SIM} t_1'). \tag{29}$$

The following equations define a function $conj$: Lin × Lin → $Set[Eq]$

$$\begin{aligned} conj(\emptyset, \emptyset) &= true; & (30)\\ conj(\emptyset, x \cdot a + l) &= \{\emptyset \simeq a\} \cup conj(\emptyset, l) & (31)\\ conj(x \cdot a + l_1, l_2) &= \{a \simeq \partial_x(l_2)\} \cup conj(l_1, l_2) & (32) \end{aligned}$$

Finally, we define an operation for splitting an equation $a \simeq b$ (where $a, b \in \mathcal{T}^{\&}$) into a conjunction of equations:

$$split(a \simeq b) = \ \textbf{if} \ \ o(a) \equiv o(b) \ \ \textbf{then} \ conj(\ell(a), \ell(b)) \ \ \textbf{else} \ false; \quad (33)$$

Proposition 9. *Given $t_1, t_2 \in \mathcal{T}^{\&}$ with $o(t_1) = o(t_2)$, then $AX \vdash t_1 = t_2$ iff $AX \vdash e$ for each equation $e \in split(t_1 \simeq t_2)$.* □

Now we are in a position to formulate our transformation system TR. It consists of the (conditional) rewrite rules given in Table 3 - "disprove", "simplify", "induction", "splitting" - which transform pairs $\langle S, H\rangle$ of sets (conjunctions) of equations $S, H \in Set[Eq]$. The set S includes equations to be proved, while the set H accumulates "inductive hypotheses". To simplify notations, from here on we denote S and H just as conjunctions of equations, rather than sets of pairs.

Note that the second rule in Table 3 involves the rewrite system *SIM* discussed above. The fourth rule involves calculations of linear forms through the function *split*.

Let ⇒ denote the rewrite relation defined by TR, then $\Rightarrow^*$ denotes its reflexive transitive closure. A derivation in TR is a chain of applications of the rules to a given pair.

Theorem 10. *The following facts hold:*

1. *The rewrite system TR is terminating.*
2. *Given $a, b \in \mathcal{T}^{\&}$, let $\langle S,\ H\rangle$ be the result (a normal form) of the following derivation in TR:*

$$\langle a = b,\ true\rangle \Rightarrow^* \langle S, H\rangle.$$

Then $\mathbf{Reg}^{\&} \models a = b$ *iff S is the empty set (i.e., true).*

Proof. 1) Consider the following (partial) ordering $\succ$ on pairs $\langle S, H\rangle$:

$$\langle S_1,\ H_1\rangle \succ \langle S_1,\ H_2\rangle \quad \textbf{iff} \quad |H_1| < |H_2| \vee (|H_1| = |H_2| \wedge |S_1| > |S_2|) \; (34)$$

where $|X|$ stands for the cardinality of a set X. For any given $S_0, H_0 \in Set[Eq]$ this ordering is noetherian (well-founded) on the set of pairs

$$\{\ \langle S,\ H\rangle \mid \langle S_0,\ H_0\rangle \Rightarrow \langle S,\ H\rangle\ \}$$

due to the fact that H is a subset of the finite set

$$\bigcup_{a=b\in S_0} \mathcal{D}_{SIM}(a) \times \mathcal{D}_{SIM}(b)$$

of pairs of all possible delivatives of terms in initial equations.

Each rule in *TR* either increases $|H|$ or, otherwise, reduces $|S|$, therefore the system is terminating.

2) The rules *(DIS)*, *(SIM)* obviously keep validity in $\mathbf{Reg}^{\&}$ (and deducibility in AX) of all equations in S. The same is true for the rule *(SPL)*, due to Prop. 9, and the rule *(IND)*, due to Prop. 8. □

Thus the use of TR to prove a $REG^{\&}$-equation $a = b$ is supposed to be as follows: take the pair $\langle a = b, true\rangle$ and apply the rules in some order until the first component of the pair becomes equal either to true or to false. Apparently it is reasonable to use first *(DIS)*, if possible, then *(SIM)* and *(IND)*, and *(SPL)* in the last turn. Still the procedure remains nondeterministic: the rules can be applied to different equations in the set S.

We next consider some examples to illustrate the use of *TR*.

4.3 Examples

The first example below is a rather simple introductory one. Examples 2 and 3 demonstrate the treatment of meet. Examples 4 and 5 show how equations involving the same extended regular expression are respectively confirmed and refuted. Finally, Examples 6 and 7 consider a couple of “classical” equations known from the literature.

We shall use regular expressions on the alplabet $\mathcal{A} = \{a, b, c, \ldots\}$. To simplify notation, we omit the concatenation sign from the expressions and in some cases introduce auxiliary meta-variables $X, Y, \ldots$ denoting (parts of) the regular expressions under consideration. Given a positive natural k and a regular expression r, let r^k stand for the k-times concatenation of r.

Derivations $\langle S_1,\ H_1\rangle \Rightarrow \ldots \Rightarrow \langle S_n, H_n\rangle$ are presented below in tabular form: row i of the table shows S_i, H_i and the rule (R_i) to be applied. When S_i has more than one conjunct, the index j of the conjunct $(S_i)_j$ to which the rule is applied is indicated thus: $(R_i)_j$. The result of a full derivation is either *true* or *false*, and H_n is irrelevant so we omit it from the table.

Example 1. To prove $b(ab)^* = (ba)^*b$, one can obtain the following inference in TR:

i	S_i	H_i	(R_i)
1.	$b(ab)^* = (ba)^*b$	*true*	(SPL)
2.	$(ab)^* = a(ba)^*b + \lambda$	S_1	(SPL)
3.	$b(ab)^* = (ba)^*b$	$H_2 \wedge S_2$	(IND)
4.	*true*		

Note that the equation at step 2 is a classical axiom, used by Conway [6] and Krob [17].

Example 2. To prove $(aaa)^* \cap (aa)^* = (aaaaaa)^*$, one can obtain the following inference in TR:

i	S_i	H_i	(R_i)
1.	$(a^3)^* \cap (a^2)^* = (a^6)^*$	*true*	(SPL)
2.	$a(a(a^3)^* \cap (a^2)^*) = a^5(a^6)^*$	S_1	(SPL)
3.	$a(a^3)^* \cap (a^2)^* = a^4(a^6)^*$	$H_2 \wedge S_2$	(SPL)
4.	$(a^3)^* \cap a(a^2)^* = a^3(a^6)^*$	$H_3 \wedge S_3$	(SPL)
5.	$a^2(a^3)^* \cap (a^2)^* = a^2(a^6)^*$	$H_4 \wedge S_4$	(SPL)
6.	$a((a^3)^* \cap (a^2)^*) = a(a^6)^*$	$H_5 \wedge S_5$	(SPL)
7.	$(a^3)^* \cap (a^2)^* = (a^6)^*$	$H_6 \wedge S_6$	(IND)
8.	*true*		

Example 3. To prove $X = (a + bb)^* \cap (aa + b)^* = (aa + bb)^* = Y$, one can obtain the following inference in TR:

i	S_i	H_i	(R_i)
1.	$X = Y$	$true$	(SPL)
2.	$X_1 = Y_1 \wedge X_2 = Y_2$	S_1	$(SPL)_1$
3.	$X = Y \wedge X_2 = Y_2$	$H_2 \wedge (S_2)_1$	$(IND)_1$
4.	$X_2 = Y_2$	H_3	(SPL)
5.	$X = Y$	$H_4 \wedge S_4$	(IND)
6.	$true$		

where

$$X_1 = (a+bb)^* \cap (a(aa+b)^*) \tag{35}$$

$$Y_1 = a(aa+bb)^* \tag{36}$$

$$X_2 = (b(a+bb)^*) \cap (aa+b)^* \tag{37}$$

$$Y_2 = b(aa+bb)^* \tag{38}$$

Example 4. Let $X = (a^*b)^*,\ Y = (ab^*)^*$. To prove $X \cap Y = a(a+b)^*b + \lambda$ one can obtain the following inference in TR:

i	S_i	H_i	(R_i)
1.	$X \cap Y = a(a+b)^*b + \lambda$	$true$	(SPL)
2.	$a^*bX \cap b^*Y = (a+b)^*b$	S_1	(SPL)
3.	$a^*bX \cap b^*Y = (a+b)^*b \wedge X \cap b^*Y = (a+b)^*b + \lambda$	$H_2 \wedge S_2$	$(IND)_1$
4.	$X \cap b^*Y = (a+b)^*b + \lambda$	$H_3 \wedge (S_3)_1$	(SPL)
5.	$a^*bX \cap b^*Y = (a+b)^*b \wedge X \cap b^*Y = (a+b)^*b + \lambda$	$H_4 \wedge S_4$	$(IND)_1$
6.	$X \cap b^*Y = (a+b)^*b + \lambda$	H_5	$(IND)_1$
7.	$true$		

Example 5. Let $X = (a^*b)^*,\ Y = (ab^*)^*$. To disprove $X \cap Y = (ab)^*$, one can obtain the following inference in TR:

i	S_i	H_i	(R_i)
1.	$X \cap Y = (ab)^*$	$true$	(SPL)
2.	$a^*bX \cap b^*Y = b(ab)^*$	S_1	(SPL)
3.	$a^*bX \cap b^*Y = \emptyset \wedge X \cap b^*Y = (ab)^*$	$H_2 \wedge S_2$	$(SPL)_1$
4.	$a^*bX \cap b^*Y = \emptyset \wedge X \cap b^*Y = \emptyset \wedge X \cap b^*Y = (ab)^*$	$H_3 \wedge (S_3)_1$	$(SPL)_2$
5.	$false$		

Example 6. The following family of *cyclic identities*

$$C_k : \quad a^* = \quad (a^k)^*(\lambda + a + a^2 + \ldots a^{k-1}) \tag{39}$$

for all $k > 0$, forms a set of equations in **Reg** which is not derivable from any finite et of equational axioms [23, 6]. Consider the inference of C_3 produced by *TR*:

i	S_i	H_i	(R_i)
1.	$a^* = (a^3)^*(\lambda + a + a^2)$	*true*	(SPL)
2.	$a^* = a(a(a^3)^*(\lambda + a + a^2) + \lambda) + \lambda$	S_1	(SPL)
3.	$a^* = a(a^3)^*(\lambda + a + a^2) + \lambda$	$H_2 \wedge S_2$	(SPL)
4.	$a^* = (a^3)^*(\lambda + a + a^2$	$H_3 \wedge S_3$	(IND)
5.	*true*		

Obviously, any of the C_k can be derived in *TR* in $k+1$ steps in the same manner.

Example 7. Conway [6] suggested a family of identities R'_n to provide a complete infinite equational basis for **Reg**. He pointed out that for each $n = 1, \ldots, 4$, R'_n is deducible from other classical equational axioms, but for $n = 4$ he doubted that “a completely written out proof could be fitted into 10 pages” (Conway [6], page 119). R'_4 is the following equation:

$$\begin{aligned}(a+b+c)^* \quad &= \quad (a(b+c)^*a + b(a+c)^*b + c(a+b)^*c)^*. \\ &\qquad (\lambda + a(b+c)^* + b(a+c)^* + c(a+b)^*\end{aligned} \tag{40}$$

Let’s consider its proof produced by *TR*. We use the following abbreviations here: X is the left-hand side of (40), Y is its right-hand side and

$$Y_1 = (b+c)^*aY + (b+c)^* \tag{41}$$

$$Y_2 = (a+c)^*bY + (a+c)^* \tag{42}$$

$$Y_3 = (a+b)^*cY + (a+b)^* \tag{43}$$

The inference in *TR* is:

i	S_i	H_i	(R_i)
1.	$X = Y$	$true$	(SPL)
2.	$X = Y_1 \wedge X = Y_2 \wedge X = Y_3$	S_1	$(SPL)_1$
3.	$X = Y \wedge X = Y_1 \wedge X = Y_2 \wedge X = Y_3$	$H_2 \wedge (S_2)_1$	$(IND)_1$
4.	$X = Y_1 \wedge X = Y_3 \wedge X = Y_3$	H_3	$(IND)_1$
5.	$X = Y_2 \wedge X = Y_3$	H_4	$(SPL)_1$
6.	$X = Y_2 \wedge X = Y \wedge X = Y_3$	$H_5 \wedge (S_5)_1$	$(IND)_1$
7.	$X = Y \wedge X = Y_3$	H_6	$(IND)_1$
8.	$X = Y_3$	H_7	(SPL)
9.	$X = Y_3 \wedge X = Y$	$H_8 \wedge S_8$	$(IND)_1$
10.	$X = Y$	H_9	(IND)
11.	$true$		

This inference is the longest one amongst our examples, still it is respectably short and presents a completely written out formal proof of R'_4. Of course, the main point is that the proof is not purely equational.

It is also worth noting that each of the identities R'_n (for all $n > O$) can be derived in TR in the same manner.

5 Conclusion

Let us first summarize what we have achieved in this paper:

- We have given a new system of Horn-equational axioms AX for the extended algebra of regular events $\mathbf{Reg}^{\&}[\mathcal{A}]$, and proved that it is complete for the ground equational theory of this algebra; the axiomatization is finite when $\mathcal{A}$ is finite.
- We have described a transformation system TR for (dis)proving ground equations in $\mathbf{Reg}^{\&}[\mathcal{A}]$, and proved its correctness and completeness, i.e., that it is terminating and that the result corresponds to whether a ground equation is satisfied or not. Our method is based on term-rewriting techniques and avoids explicit construction and minimalization of deterministic finite automata (DFA) or non-deterministic ones (NFA).

The primary application envisaged for this work is in the implementation of term rewriting in frameworks that allow algebras of sorts - in particular, for unified algebras [19]. Extended regular expressions denoting sorts are much exploited in action semantics [20], whose foundations are also specified using unified algebras.

Our work may also be seen as a contribution to the theory of regular expressions. Let us briefly review previous related work. At the end, we shall consider possible improvements and extensions of our approach.

5.1 Related Work

There has been much research on the axiomatization of **Reg**[$\mathcal{A}$], whose (ground) equational theory is not finitely based for alphabets with more than one letter, as proved by Redko [23] and Conway [6] (cf. also Salomaa [25]). Infinite equational axiomatizations were first provided by Conway [6] and shown to be complete by Krob [17]. To obtain a finite axiomatization, several approaches have been explored:

- Using special (non-logical) inference rules: Salomaa [24] gave two complete axiomatic calculi. One refers to the negation of the empty word property, the other one uses the number of letters occurring in regular expressions. See also Salomaa [25], Salomaa and Tixier [26].

- Using equational implications: Conway [6] gave a finite Horn-equational axiomatization of **Reg**; he conjectured, but did not prove, completeness. Gorshkov and Archangelsky [12] gave a different one using 10 equations and two equational implications, proving its completeness. Boffa [3] linked the completeness of Conway's and Salomaa's systems and suggested an "intermediate" inference rule (which can be taken as an equational implication). Krob [17] proved completeness of several axiomatic systems for **Reg** (including those by Conway and Boffa). Kozen [15] gave yet another finite axiomatization of **Reg** by 13 equations and two equational implications.

- Extending **Reg** with further operations: Salomaa and Tixier [26] gave two complete axiomatizations of the extension of **Reg** with intersection and complement, one depending on a particular alphabet and referring to the negation of e.w.p. (see also Salomaa [25]), the other one (developed by Tixier in his thesis) getting rid of e.w.p. through intersection.

Pratt [22] considered *action algebras* equipped with *residuations* (left and right) and gave a finite *equational* axiomatization (in the enriched signature) of the equational theory of the variety of action algebras (which conservatively extends the ground equational theory of $\mathbf{Reg}[\mathcal{A}]$ on the countable alphabet $\mathcal{A}$) Kozen [16] extended the above with intersection to obtain *action lattices*, but his axioms do not axiomatize $GEq(\mathbf{Reg}^{\&}[\mathcal{A}])$.

- Using order-sorted algebras: In the present paper we exploit the rather natural idea that the alphabet should be a *subsort* of the sort of regular events over that alphabet. This can also be done in the framework of *unified algebras* [19] which treats sorts as values and uses a binary predicate symbol for sort inclusion. In both cases, we exploit Horn-equations freely in order to get a complete axiomatization.

Concerning calculation and proof techniques in **Reg** and its extensions - avoiding explicit construction and minimalization of deterministic finite automata - we find the following work:

- Using derivatives: Brzozowski [4] and Conway [6] showed how to use derivatives to carry on some calculations in **Reg**, also when extended with meet and complement. See also Ginzburg [9] for "mechanical methods" for proving equivalence in **Reg**.
- Calculating normal forms: Johansen [13] provided "algebraic normal forms" for **Reg** (actually, not unique in general). Other papers have developed normal forms for some *proper subclasses* of **Reg** (cf. further references given by Johansen [13]).
- Solving systems of equations: Brzozowski and Leiss [5] showed how to do this for linear equations in **Reg** extended with intersection and complement. Leiss [18] has subsequently demonstrated some advantages that arise in the *absence* of complement.

The method for inferring equations given in the present paper involves several rewrite systems - *LF* for calculating linear forms, *SIM* for simplifying regular expressions, and *TR* for reducing sets of equations. All these are modulo associativity/commutativity and thus based on the corresponding matching algorithm, which is known to be NP-complete [1]. Therefore, the most adequate complexity measure seems to be the *length* of inferences providing by the system *TR*. An analysis of the proof of Thm. 10 shows that in the worst case the length of the inference of the equation $a = b$ can be *exponential* in the size of the expressions involved (more precisely, it can be equal to the product of the numbers of dissimilar derivatives corresponding to a and b).

This is not surprising since the problem of non-equivalence of two ground *REG*-expressions is known to be PSPACE-complete [8]. On many particular examples, however, the system *TR* produces respectably short inferences. The use of the rewriting system *SIM* for simplifying regular expressions may be crucial here. Consider for example the equation

$$(a^* + b)^* a(a + b)^k \quad = \quad (a + b^*)^* a(b + a)^k \qquad (44)$$

for some positive natural k. Note that the minimal DFA for either of the sides of (44) has 2^{k+1} states and corresponding exponential time is needed to construct one. However, both sides can be reduced to normal forms, equivalent modulo associativity/commutativity of the join, by *one application* of the rule (26). So the inference of (44) in *TR* may consist of just one step - the application of rule *(SIM)*. Perhaps the term rewriting approach that we have suggested leads to a better average-case complexity algorithm than known ones constructing automata to solve the word problems in **Reg** and **Reg**$^{\&}$?

This possibility is supported by a result of Birget [2], proving that the size of a minimal DFA may increase *exponentially* for both sums $A_1 + A_2 + \ldots + A_k$ and intersections $A_1 \cap A_2 \cap \ldots \cap A_k$ of minimal DFA A_i of the same size n (i.e., the resulting DFA may have n^k states). The same holds true even for the size of NFA for intersections. Now imagine that one is going to (dis)prove a regular equation

$$a_1 \cap a_2 \cap \ldots \cap a_k \quad = \quad r \qquad (45)$$

using automata methods. Then one is supposed first to unfold each side into a DFA or an NFA, and this may take exponential time and space. In contrast to this, our *TR* “unfolds” both sides together in a “lazy” manner using on the way simplification. This can help to obtain a rather short inference (not in the worst case, of course).

5.2 Open problems and possible extensions

Finally, let us mention a couple of aspects of this work that have been left open here:

- It should be investigated whether the term rewriting approach we have suggested does in fact lead to a better average-case complexity algorithm than those based on the minimal DFA construction.

- Our axiomatization of $\mathbf{Reg}^{\&}[\mathcal{A}]$ is complete for inferring ground equations (using letters from $\mathcal{A}$ as constants). It would be useful to extend it to one that is complete also for the whole equational theory $Eq(\mathbf{Reg}^{\&}[\mathcal{A}])$. For instance, the universal equation

  $$(\forall a, b : Reg)(a^* \cap b^*)^* \quad = \quad a^* \cap b^* \tag{46}$$

  is valid in $\mathbf{Reg}^{\&}[\mathcal{A}]$ and should be derivable from such an extension.

Acknowledgments: This work was partially supported by a Research Fellowship (J.nr. 11 − 9479) and by the DART project (5.21.08.03), both funded by the Danish Science Research Council. Nils Andersen, Torben Mogensen, sincerely thankful to DAIMI (Computer Science Department, Aarhus University), where he started the work reported in this paper, and to DIKU (Computer Science Department, Copenhagen University), which gave him the opportunity to complete the work and to present it at the conference on *Developments in Language Theory*, Univ. of Turku, 12-15 July 1993.

Appendix

The following proof of Thm. 2 exploits linear forms and derivatives of $REG^{\&}$-terms – cf. Sect. 4 for definitions.
Proof. Given an extended regular expression $t \in \mathcal{T}$, we show how to find $t' \in \mathcal{T}$ such that $AX \vdash t = t'$.

First, consider the linear form $\ell(t)$. It may happen that it doesn't contain the meet – then $t = o(t) + \ell(t)$ and we are done. Otherwise, consider the following finite non-empty set of equations $LS(t)$:

$$\{r = o(r) + \ell(r) \quad | \quad r \in \mathcal{D}_E(t) \setminus \mathcal{T}\} \tag{47}$$

It follows from Prop. 7 that $AX \vdash LS(t)$. Note that the meet appears in these equations only inside the expressions r (which can also occur in right-hand sides as tails of some items).

Now replace all the occurrences of each expression r in $LS(t)$ (in both left- and right-hand sides) by corresponding fresh variables x_r and consider the result as a system of linear equations for these x_r. This system doesn't contain meet, so it follows from Prop. 1 that it can be solved in **Reg** by the classical method (cf., e.g., Salomaa [24, 25] using axioms $(A1) - (A20)$. The solution t' for the component x_t is just the required REG-term, since the equation $t = t'$ is derived from $LS(t)$ using $(A1) - (A20)$. □

References

[1] D. Benanav, D. Kapur, and P. Narendran, Complexity of matching problems, *J. Symbolic Computation* **3** (1987) 203-216.

[2] J.-C. Birget, Intersection and union of regular languages and state complexity, *Inf. Process. Lett.* **43** (1992) 185-190.

[3] M. Boffa, Une remarque sur les systèmes complets d'identités rationnelles, *Theor. Inf. Applic.* **24** (1990) 419-423.

[4] J. A. Brzozowski, Derivatives of regular expressions, *J. ACM* **11** (1964) 481-494.

[5] J. A. Brzozowski and E. L. Leiss, On equations for regular languages, finite automata, and sequential networks, *Theoret. Comput. Sci.* **10** (1980) 19-35.

[6] J. H. Conway, *Regular Algebra and Finite Machines* (Chapman and Hall, 1971).

[7] N. Dershowitz and J.-P. Jouannaud, Rewrite systems, in: J. van Leeuwen, ed., *Handbook of Theoretical Computer Science, Vol. B* (Elsevier, Amsterdam, 1990) Ch. 6.

[8] M. R. Garey and D. S. Johnson, *Computers and Intractability - A Guide to the Theory of NP-Completeness* (W. H. Freeman & Co., 1979).

[9] A. Ginzburg, A procedure for checking equality of regular expressions, *J. ACM* **14** (1967) 355-362.

[10] J. A. Goguen and J. Meseguer, Order-sorted algebra I: Equational deduction for multiple inheritance, overloading, exceptions and partial operations, *Theoret. Comput. Sci.* **105** (1992) 217-273.

[11] J. A. Goguen and T. Winkler, Introducing OBJ3, Tech. Report SRI-CSL-88-9, Computer Science Lab., SRI International, 1988.

[12] P. V. Gorshkov and K. V. Archangelsky, Conditional equations in an algebra of regular languages, *Reports of Ukrainian Academy of Sci., series A* **10** (1987) 67-69.

[13] P. Johansen, *An Algebraic Normal Form for Regular Events* (Polyteknisk Forlag, Lyngby, Denmark, 1972).

[14] D. Kozen, On induction vs. *-continuity, in: D. Kozen, ed., *Proc. Workshop on Logic of Programs,* Lecture Notes in Computer Science, Vol. 131 (Springer, Berlin, 1981) 167-176.

[15] D. Kozen, A completeness theorem for Kleene algebras and the algebra of regular events, in: *Proc. LICS'91* (IREE, 1991) 214-225.

[16] D. Kozen, On action algebras, Tech. Report DAIMI PB-381, Computer Science Dept., Aarhus University, 1992.

[17] D. Krob, Complete systems of B-rational identities, *Theoret. Comput. Sci.* **89** (1991) 207-343.

[18] E. L. Leiss, Generalized language equations with multiple solutions, *Theoret. Comput. Sci.* **44** (1986) 155-174.

[19] P. D. Mosses, Unified algebras and institutions. in: *Proc. LICS'89* (IEEE. 1989) 304-312.

[20] P. D. Mosses, *Action Semantics*, Cambridge Tracts in Theoretical Computer Science, Vol. 26 (Cambridge University Press, 1992).

[21] D. Perrin, Finite automata, in: J. van Leeuwen, ed., *Handbook of Theoretical Computer Science, Vol. B* (Elsevier, Amsterdam, 1990) Ch. 1.

[22] V. Pratt, Action logic and pure induction, in: *Proc. JELIA'90*, Lecture Notes in Computer Science, Vol 478 (Springer, Berlin, l990) 97-120.

[23] V. N. Redko, On defining relations for the algebra of regular events, *Ukrainian Mat. Z.* **16** (1964) 120-126.

[24] A. Salomaa, Two complete axiom systems for the algebra of regular events, *J. ACM* **13** (1966) 158-169.

[25] A. Salomaa, *Theory of Automata* (Pergamon, 1969).

[26] A. Salomaa and V. Tixier, Two complete axiom systems for the algebra of regular events, *IEEE Trans. Comp.* **C-17** (1968) 700-701.