UNIVERSITY OF OSLO
Department of Informatics

Unification and Matching in Complex Theories

Research Report No.

Sergiu Bursuc
Cristian Prisacariu

Isbn 82-7368-330-3
Issn 0806-3036

March 10, 2010

March 10, 2010

Unification and Matching in Complex Theories

Sergiu Bursuc[*]   Cristian Prisacariu[†]

March 10, 2010

Abstract

The present work studies problems of unification and matching in equational theories based on idempotent semirings. These theories include Kleene algebras and extensions of this to model forms of concurrency, contraint semirings, and synchronous actions algebra. Generaly the unification problems are undecidable (but different undecidability proofs are required) whereas the matching problems are decidable.

[*]School of Computer Science – Univ. of Birmingham, Birmingham B15 2TT, UK. E-mail: s.bursuc@cs.bham.ac.uk

[†]Dept. of Informatics – Univ. of Oslo, P.O. Box 1080 Blindern, N-0316 Oslo, Norway. E-mail: cristi@ifi.uio.no

Contents

1 Introduction

Semirings, but mostly idempotent semirings (*IS*) are the basis of several recent equational theories of higher complexity (i.e., with more defining equations or extra operators). The theories that we are particulary interested in are the *Kleene algebras* (*KA*) [Con71, Koz79] with their several extensions with tests [Koz97], types [Koz98], non-local flow of control [Koz08], or local variables [AHK08]; the concurrency extensions of *synchronous Kleene algebras* (*SKA*) [Pri10] and *concurrent Kleene algebras* (*CKA*) [HMSW09b]; *synchronous actions algebras* (*SAA*) [Pri10]; *constraints semirings* (*CS*) and combinations [BMR97, NFM$^+$05].

Kleene algebras appear in various formalisms in computer science: relation algebras, logics of programs like propositional dynamic logic [FL77, Pra90], or regular expressions and formal language theory [KS86]. *Kleene algebra with tests* (*KAT*) can express while programs [Koz00] and can encode propositional Hoare logic using a Horn-style inference system. Concurrent Kleene algebra was recently proposed in [HMSW09b] as a general formalism for reasoning about concurrent programs. In the same lines, synchronous Kleene algebra captures the notion of synchrony and synchonously executing entities. The simplification that looses the regularity of the Kleene star is the synchronous actions algebra (also investigated in [Pri10]); these actions are the basis of the contract logic from [PS09]. Contraint semirings have been proposed in [BMR97] to model various notions of contraints and have been used in [NFM$^+$05] to define a process algebra with contraints. Semirings alone have been used in [Moh02] for a general algebraic framework for shortest-distance problems.

For all these formalisms unification and matching problems appear naturally. For example, matching in Kleene algebras is used when on the one side of the equality we have the specification of the required regular behaviour of the system and on the other side we have the behaviour of part of our system (the part that we know) and the unknown behaviour is filled in by variables. Resolving the matching problem means to discover the unknown parts of our composed system. In the concurrent extensions of Kleene algebra (the *SKA* or the *CKA*) matching is more desirable as the variables may represent entire unknown components that run concurrently with the known components.

For the synchronous actions algebra *SAA* the matching problem was left open in [Pri10] where a particular matching problem is used in the definition of the semantics of the contract logic $\mathcal{CL}$ in [PS09]. The decidability of the $\mathcal{CL}$ logic was proven relative with the decidability of this particular matching problem. The matching algorithm that we give here solves in afirmative the decidability of $\mathcal{CL}$.

(1) $\alpha + (\beta + \gamma) = (\alpha + \beta) + \gamma$
(2) $\alpha + \beta = \beta + \alpha$
(3) $\alpha + \mathbf{0} = \mathbf{0} + \alpha = \alpha$
(4) $\alpha + \alpha = \alpha$
(5) $\alpha \cdot (\beta \cdot \gamma) = (\alpha \cdot \beta) \cdot \gamma$
(6) $\alpha \cdot \mathbf{1} = \mathbf{1} \cdot \alpha = \alpha$
(7) $\alpha \cdot \mathbf{0} = \mathbf{0} \cdot \alpha = \mathbf{0}$
(8) $\alpha \cdot (\beta + \gamma) = \alpha \cdot \beta + \alpha \cdot \gamma$
(9) $(\alpha + \beta) \cdot \gamma = \alpha \cdot \gamma + \beta \cdot \gamma$

Table 1: Axioms of idempotent semirings (*IS*)

2 Equational Theories Under Consideration

2.1 Idempoten semirings

Definition 2.1 (idempotent semiring) *An* idempotent semiring *is an algebraic structure* $(\mathcal{A}, +, \cdot, \mathbf{0}, \mathbf{1})$ *that respects axioms (1)-(9) of Table 1. We understand the operations as representing respectively* nondeterministic choice *and* sequence*. Henceforth we denote elements of* $\mathcal{A}$ *by* α, β, γ*, and call them* (compound) actions*. The constants* $\mathbf{0}$ *and* $\mathbf{1}$ *are sometimes called the* fail *action respectively the* skip *action. For an idempotent semiring the* natural order $\leq$ *is defined as:*

$$\alpha \leq \beta \overset{\Delta}{=} \alpha + \beta = \beta;$$

and in this paper we usually say that "β is preferable *to α".*

An intuitive understanding of the natural order of a semiring is that $\leq$ states that the left operand has less behavior than the right operand, or in other words, the right operand specifies behavior which includes all the behavior specified by the left operand (and possibly more).

Remarks: It is easy to check that $\leq$ is a partial order and that it forms a semilattice with least element $\mathbf{0}$ and with $\alpha + \beta$ the least upper bound of α and β. Moreover, the three operators are monotone w.r.t. $\leq$.

The axioms (1)-(4) define the choice operator $+$ to be associative, commutative, with neutral element $\mathbf{0}$, and idempotent. Axioms (5)-(7) define the sequence operator $\cdot$ to be associative, with neutral element $\mathbf{1}$, and with annihilator $\mathbf{0}$ both on the left and right side. Axioms (8) and (9) give the distributivity of $\cdot$ over $+$.

(10) $\mathbf{1} + \alpha \cdot \alpha^* \leq \alpha^*$
(11) $\mathbf{1} + \alpha^* \cdot \alpha \leq \alpha^*$
(12) $\beta + \alpha \cdot \gamma \leq \gamma \ \rightarrow\ \alpha^* \cdot \beta \leq \gamma$
(13) $\beta + \gamma \cdot \alpha \leq \gamma \ \rightarrow\ \beta \cdot \alpha^* \leq \gamma$

Table 2: Axioms of Kleene algebra (*KA*)

All axioms of Kleene algebra from Table 2
(14) $\alpha\times(\beta\times\gamma) = (\alpha\times\beta)\times\gamma$
(15) $\alpha\times\beta = \beta\times\alpha$
(16) $\alpha\times\mathbf{1} = \mathbf{1}\times\alpha = \alpha$
(17) $\alpha\times\mathbf{0} = \mathbf{0}\times\alpha = \mathbf{0}$
(18) $a\times a = a \quad \forall a \in \mathcal{A}_B$
(19) $\alpha\times(\beta + \gamma) = \alpha\times\beta + \alpha\times\gamma$
(20) $(\alpha + \beta)\times\gamma = \alpha\times\gamma + \beta\times\gamma$
(21) $(\alpha_\times \cdot \alpha)\times(\beta_\times \cdot \beta) = (\alpha_\times\times\beta_\times)\cdot(\alpha\times\beta),\ \forall \alpha_\times, \beta_\times \in \mathcal{A}_B^\times$

Table 3: Axioms of synchronous Kleene algebra (*SKA*)

2.2 Kleene algebras

Definition 2.2 (Kleene algebras) *A* Kleene algebra $(\mathcal{A}, +, \cdot, ^*, \mathbf{0}, \mathbf{1})$ *is an idempotent semiring with one extra unary (postfix) operation* **, which respects the extra axioms (10)-(13) of Table 2. We understand the Kleene* * *operation as representing* iteration.

The equations (10) and (11) and equational implications (12) and (13) are the standard axiomatization of * [Sal66, Koz94] which say that $\alpha^* \cdot \beta$ is the least solution w.r.t. the preference relation $\leq$ for the equation $\beta + \alpha \cdot X \leq X$ (and dually $\beta \cdot \alpha^*$ is the least solution to the equation $\beta + X \cdot \alpha \leq X$).

2.3 Synchronous Kleene algebras

Definition 2.3 (synchronous Kleene algebra) *A* synchronous Kleene algebra *(SKA) is a structure* $(\mathcal{A}, +, \cdot, \times, ^*, \mathbf{0}, \mathbf{1}, \mathcal{A}_B)$ *obtained from a Kleene algebra by adding a "$\times$" operation for* synchronous composition *of two actions. The new operation* $\times$ *respects the axioms (14)-(21) of Table 3.*

Notation: Consider the set $\mathcal{A}_B^\times \subset \mathcal{A}$ to be the set $\mathcal{A}_B$ closed under application of $\times$. We call the elements of $\mathcal{A}_B^\times$ $\times$-*actions* and denote them generically by $\alpha_\times$ (e.g., a, $a\times b \in \mathcal{A}_B^\times$ but $a + b, a\times b + c, a \cdot b \notin \mathcal{A}_B^\times$ and $\mathbf{0}, \mathbf{1} \notin \mathcal{A}_B^\times$). Note that $\mathcal{A}_B^\times$ is finite because there is a finite number of basic actions in $\mathcal{A}_B$

which may be combined with the synchrony operator $\times$ in a finite number of ways (due to the weak idempotence of $\times$ over basic actions; see axiom (18) of Table 3). Note the inclusion of sorts $\mathcal{A}_B \subseteq \mathcal{A}_B^{\times} \subset \mathcal{A}$. For brevity we often drop the sequence operator and instead of $\alpha \cdot \beta$ we write $\alpha\beta$. To avoid unnecessary parentheses we use the following precedence over the constructors: $+ < \cdot < \times < {}^*$.

Axioms (14)-(17) give the properties of $\times$ to be associative, commutative, with identity element $\mathbf{1}$, and annihilator element $\mathbf{0}$ (i.e., $(\mathcal{A}, \times, \mathbf{1}, \mathbf{0})$ is a commutative monoid with an annihilator element). Axioms (14) and (15) basically say that the syntactic ordering of actions in a $\times$-action does not matter. Axiom (18) defines $\times$ to be weakly idempotent over the basic actions $a \in \mathcal{A}_B$. Note that this does *not* imply that we have an idempotent monoid. Axioms (19) and (20) define the distributivity of $\times$ over $+$. From axioms (14)-(20) together with (1)-(4) we conclude that $(\mathcal{A}, +, \times, \mathbf{0}, \mathbf{1})$ is a commutative and idempotent semiring (NB: idempotence comes from axiom (4), and the axiom (18) is just an extra property of the semiring).

At this point we give an informal intuition for the actions (elements) of $\mathcal{A}$: we consider that the actions are "done" by somebody (be that a person, a program, or an agent). One should not think exclusively of processes "executing" instructions as this is only one way of viewing the actions. Moreover, we do not discuss in this paper operational semantics nor bisimulation equivalences (like is done in SCCS [Mil83]).

With this non-algebraic intuition of actions we can elaborate on the purpose of $\times$, which models the fact that two actions are *done at the same time*. Doing actions at the same time should not depend on the syntactic ordering of the concurrent actions; thus the associativity and commutativity axioms (14) and (15) of $\times$. Intuitively, if a component does a skip action $\mathbf{1}$ then this should not be visible in the synchronous action of the whole system (thus the axiom (16)); whereas, if a component fails (i.e., does action $\mathbf{0}$) then the whole system fails (thus the axiom (17)).

Particular to $\times$ is the axiom (18) which defines a weak form of idempotence for the synchrony operator. The idempotence is natural for basic actions but it is not desirable for complex actions. Take as example a choice action performed synchronously with itself, $(a + b) \times (a + b)$. The first entity may choose a and the second entity may choose b thus performing the synchronous action $a \times b$. Therefore, the complex action is the same as $a + a \times b + b$ (by the distributivity axiom (19), the commutativity of $\times$ and $+$, idempotence of $\times$ over basic actions (18), and idempotence of $+$).

Particular to our concurrency model is axiom (21) which synchronizes sequences of actions by working in steps given by the $\cdot$ constructor. This encodes the synchrony model.

$\alpha + (\beta + \gamma) = (\alpha + \beta) + \gamma$	$\alpha \times (\beta \times \gamma) = (\alpha \times \beta) \times \gamma$
$\alpha + \beta = \beta + \alpha$	$\alpha \times \beta = \beta \times \alpha$
$\alpha + \mathbf{0} = \mathbf{0} + \alpha = \alpha$	$\alpha \times \mathbf{1} = \mathbf{1} \times \alpha = \alpha$
$\alpha + \alpha = \alpha$	$\alpha \times \mathbf{0} = \mathbf{0} \times \alpha = \mathbf{0}$
$\alpha \cdot (\beta \cdot \gamma) = (\alpha \cdot \beta) \cdot \gamma$	$a \times a = a \quad \forall a \in \mathcal{A}_B$
$\alpha \cdot \mathbf{1} = \mathbf{1} \cdot \alpha = \alpha$	$\alpha \times (\beta + \gamma) = \alpha \times \beta + \alpha \times \gamma$
$\alpha \cdot \mathbf{0} = \mathbf{0} \cdot \alpha = \mathbf{0}$	$(\alpha + \beta) \times \gamma = \alpha \times \gamma + \beta \times \gamma$
$\alpha \cdot (\beta + \gamma) = \alpha \cdot \beta + \alpha \cdot \gamma$	$(\alpha_\times \cdot \alpha) \times (\beta_\times \cdot \beta) = (\alpha_\times \times \beta_\times) \cdot (\alpha \times \beta),\ \forall \alpha_\times, \beta_\times \in \mathcal{A}_B^\times$
$(\alpha + \beta) \cdot \gamma = \alpha \cdot \gamma + \beta \cdot \gamma$	

Table 4: Axioms of SAA

Definition 2.4 *Consider $SKA \vdash \alpha = \beta$ to mean that the SKA equation can be deduced from the axioms of SKA using the standard rules of equational reasoning (reflexivity, symmetry, transitivity, and substitution), instantiation, and introduction and elimination of implication. Consider henceforth the relation $\equiv \subseteq T_{SKA} \times T_{SKA}$ defined as: $\alpha \equiv \beta \Leftrightarrow SKA \vdash \alpha = \beta$.*

Remark: The proof that $\equiv$ is a congruence is straightforward, based on the deduction rules, and we leave it to the reader.

2.4 Synchronous actions algebras

Definition 2.5 (synchronous actions algebra) *The* synchronous actions algebra *is the synchronous Kleen algebra from Definition 2.3 where we remove the Kleene * operator. The complete axiomatization of SAA is given by axioms (1)-(9) of Table 2 together with axioms (14)-(21) of Table 3. We collect these axioms in Table 4.*

SAA is finitely generated by a fixed set $\mathcal{A}_B \cup \{\mathbf{0}, \mathbf{1}\}$. The synchronous actions of SAA are sometimes called *-free actions and are terms constructed with the grammar below:

$$\alpha \quad ::= \quad a \mid \mathbf{0} \mid \mathbf{1} \mid \alpha + \alpha \mid \alpha \cdot \alpha \mid \alpha \times \alpha$$

where $a \in \mathcal{A}_B$ is a basic action. $SAA = (\mathcal{A}, +, \cdot, \times, \mathbf{0}, \mathbf{1})$ is formed as two idempotent semirings $(\mathcal{A}, +, \cdot, \mathbf{0}, \mathbf{1})$ and $(\mathcal{A}, +, \times, \mathbf{0}, \mathbf{1})$ with three extra axioms: commutativity of $\times$ (15), weak idempotence for $\times$ (18) over basic actions $\mathcal{A}_B$, and the synchrony axiom (21).

The canonical form (defined below) gives us a more structured way of viewing the *-free actions and makes easier the formulation and proofs of the unification and matching results for SAA.

Definition 2.6 (canonical form for SAA**)** *We say that a* **-free action* α *is in* canonical form*, denoted by* $\underline{\alpha}$*, iff it has the following form:*

$$\underline{\alpha} = \mathop{+}_{i \in I} \alpha^i_\times \cdot \underline{\alpha^i}$$

where $\alpha^i_\times \in \mathcal{A}^\times_B$ *are pairwise distinct and* $\underline{\alpha^i} \in SAA$ *is in canonical form. The indexing set* I *is finite as the compound actions* α *are finite; i.e., there is a finite number of applications of the* $+$ *operator. Actions* $\mathbf{0}$ *and* $\mathbf{1}$ *are considered in canonical form.*

Theorem 2.7 ([Pri10]) *For every* **-free action* $\alpha \in SAA$ *there is corresponding* $\underline{\alpha}$ *in canonical form and equivalent to* α *(i.e.,* $SKA \vdash \alpha = \underline{\alpha}$*).*

Corollary 2.8 ([Pri10]) *For any* **-free action* α *there exists an equivalent action* $\beta \in SAA$ *(i.e.,* $\beta \equiv \alpha$*) which is of the following form:*

$$\beta = \mathop{+}_{i \in I} \mathop{\cdot}_{j \in J} \alpha^{ij}_\times .$$

Lemma 2.9 ([Pri10]) *For any action* α *that has the form of a sequence of synchronous actions (i.e.,* $\alpha = \alpha^1_\times \cdot \cdots \cdot \alpha^n_\times$*) then* $\alpha \times \alpha = \alpha$*. In other words,* $\times$ *is idempotent for actions of this form.*

2.5 Concurrent Kleene algebras

The definitions that we give here are taken from [HMSW09b], and more recent theoretical results can be found in [HMSW09a].

Definition 2.10 (quantales) *A* quantale $(\mathcal{A}, +, \cdot, \mathbf{0}, \mathbf{1})$ *is an idempotent semiring where, in addition,* $(\mathcal{A}, \leq)$ *forms a complete lattice under the natural order of the semiring (*$+$ *coincides with the supremum operation of the lattice) and sequential composition distributes over arbitrary suprema.*

A quantale has $\mathbf{0}$ as its least element and being a complete lattice it has also a greatest element.

Definition 2.11 (concurrent Kleene algebra) *A* concurrent Kleene algebra $(\mathcal{A}, +, \cdot, \times, \mathbf{0}, \mathbf{1})$ *is two quantales,* $(\mathcal{A}, +, \cdot, \mathbf{0}, \mathbf{1})$ *and* $(\mathcal{A}, +, \times, \mathbf{0}, \mathbf{1})$*, linked by the* exchange axiom*:*

$$(\alpha \times \beta) \cdot (\alpha' \times \beta') \leq (\beta \cdot \alpha') \times (\alpha \cdot \beta') \tag{1}$$

$\alpha + (\beta + \gamma) = (\alpha + \beta) + \gamma$
$\alpha + \beta = \beta + \alpha$
$\alpha + \mathbf{0} = \mathbf{0} + \alpha = \alpha$
$\alpha + \alpha = \alpha$
$\alpha \cdot (\beta \cdot \gamma) = (\alpha \cdot \beta) \cdot \gamma$
$\alpha \cdot \mathbf{1} = \mathbf{1} \cdot \alpha = \alpha$
$\alpha \cdot \mathbf{0} = \mathbf{0} \cdot \alpha = \mathbf{0}$
$\alpha \cdot (\beta + \gamma) = \alpha \cdot \beta + \alpha \cdot \gamma$
$(\alpha + \beta) \cdot \gamma = \alpha \cdot \gamma + \beta \cdot \gamma$
(22) $\alpha + \mathbf{1} = \mathbf{1}$
(23) $\alpha \cdot \beta = \beta \cdot \alpha$

Table 5: Axioms of constraint semirings (CS)

Proposition 2.12 ([HMSW09b]) *The following equalities hold for a CKA.*

$$\begin{aligned}
\alpha \times \beta &= \beta \times \alpha & (2)\\
(\alpha \times \beta) \cdot (\alpha' \times \beta') &\leq (\alpha \cdot \alpha') \times (\beta \cdot \beta') & (3)\\
\alpha \cdot \beta &\leq \alpha \times \beta & (4)\\
(\alpha \times \beta) \cdot \gamma &\leq \alpha \times (\beta \cdot \gamma) & (5)\\
\alpha \cdot (\beta \times \gamma) &\leq (\alpha \cdot \beta) \times \gamma & (6)
\end{aligned}$$

2.6 Constraint semirings

The definitions and examples that we consider are taken from [NFM+05] where is presented an application of contraint semirings in the definition of a process algebra for specification of systems where the quality of service is an important factor. Technical foundations of constraint semirings have been given in [BMR97].

Definition 2.13 (contraint semirings) *A* constraint semiring $(\mathcal{A}, +, \cdot, \mathbf{0}, \mathbf{1})$ *is an idempotent semiring where* $\mathbf{1}$ *is absorbing element from* $+$*, i.e., axiom (22), and* $\cdot$ *is commutative, i.e., axiom (23).*

Examples of useful constraint semirings for specifying quality of service parameters are:

- Network availability (boolean): $(\{true, false\}, \vee, \wedge, false, true)$.
- Consts and delays (opmization): $(\mathbb{R}^+, min, +, \infty, 0)$.
- Bandwidth: $(\mathbb{R}^+, max, min, 0, \infty)$.

- Performance and failure rates (probabilistic): $([0,1], max, \cdot, 0, 1)$.
- Access rights (powerset): $(2^S, \cup, \cap, \emptyset, S)$.

Combinations of constraint semirings offers the posibility to reason about different types of QoS requirements.

3 Results

In this section we show that unification in idempotent semirings is undecidable. On the other hand, we give a simple algorithm for the matching problem in idempotent semirings which runs in NP-time. We add a third operator which makes a second idempotent and commutative semiring. This theory is the basis for many theories used in practice, like: Q-algebras, c-semirings, concurrent Kleene algebras, or synchronous Kleene algebra (*SKA*). For the later one we show that general unification remains undecidable by using a similar argument as for one idempotent semiring. This argument is possible because of the semantics of *SKA*. The matching problem for *SKA* is decidable.

To prove undecidability we use the method (addapted to our theories) of [ANR04] which uses reduction from the modified Post correspondence problem (MPCP). We recall this problem here. Consider a list of pairs (w_i, w'_i), for $0 \leq i \leq n$, of nonempty finite strings over some alphabet. Determine whether there exist $i_1 \ldots i_k$ with $0 \leq i_j \leq n$ for $j \in \overline{1,k}$ s.t.

$$w_0 w_{i_1} w_{i_2} \ldots w_{i_k} = w'_0 w'_{i_1} w'_{i_2} \ldots w'_{i_k}.$$

This problem is undecidable. Intuitively the problem says that from the n pairs of strings one can pick some of these pairs (maybe several times the same pair and not necessarily all the n pairs, i.e., there is no condition on k and i_j may denote the same index) and concatenate the left strings together and the paired right strings together to obtain the same string. The modified PCP says that these two strings must start with some choosen pair (w_0, w'_0).

Proposition 3.1 *For a MPCP one can build another MPCP' which has the w_0 and w'_0 at the end of the strings s.t. any solution for MPCP' is a solution for MPCP and vice versa.*

Proof: The proof is simple by just mirroring all the strings in all the pairs in the original MPCP. For these new pairs the MPCP' problem is

$$\overleftarrow{w}_{i_1} \overleftarrow{w}_{i_2} \ldots \overleftarrow{w}_{i_k} \overleftarrow{w}_0 = \overleftarrow{w'}_{i_1} \overleftarrow{w'}_{i_2} \ldots \overleftarrow{w'}_{i_k} \overleftarrow{w'}_0,$$

where $\overline{w}$ represent the mirror words comming from w. The solution for MPCP' is also a solution for MPCP. □

Remark: For any term $\alpha \in IS$ (i.e., built with $+$ and $\cdot$ in the theory of IS) one can find a normal form $\alpha\!\downarrow$ that looks like a sum of products:

$$\alpha \equiv \alpha\!\downarrow = \underset{i\in I}{+} \underset{j\in J}{\cdot}\, t_{ij}$$

where $t_{ij} \in \mathcal{A}_B$ if α is ground, otherwise t_{ij} is a variable. Moreover, all the products $\cdot_{j\in J}\, t_{ij}$ are different because of the idempotence (4) of $+$.

Definition 3.2 (types of variables) *We define a* type $\mathcal{A}^L$, *where L is a list of operators, to be the set of all terms built only with the operators from L. For a variable X we say that X is of some type $\mathcal{A}^L$, and denote it $X : \mathcal{A}^L$, iff X can take values only from $\mathcal{A}^L$.*

Theorem 3.3 (undecidability of *IS*) *Unification with constants, modulo the theory IS of idempotent semirings is undecidable.*

Proof: The proof that we give here is basically the same as in [ANR04] only that we need to treat also the axiom (6) for the unity element $\mathbf{1}$.

Consider a MPCP $(w_0, w'_0)\dots(w_n, w'_n)$ in the mirrored style of Proposition 3.1 with the assumption, wlog., that $w_0 \neq w'_0$. Construct the following *IS*-unification problem:

$$\begin{aligned} w_1\cdot X_1\cdot \overline{1} + \dots + w_n\cdot X_n\cdot \overline{n} + w_0\cdot \# &= X_1 + \dots + X_n + V,\\ w'_1\cdot Y_1\cdot \overline{1} + \dots + w'_n\cdot Y_n\cdot \overline{n} + w'_0\cdot \# &= Y_1 + \dots + Y_n + V,\end{aligned}$$

where $\overline{1}, \dots \overline{n}, \#$ are new constants not appearing in w_i or w'_i and the capital letters are variables of type $\mathcal{A}^{\{+,\cdot\}}$ (this means, cf. remark above, that any substitution replaces these variables with tearms $\alpha\!\downarrow$ in normal form drawn from $\mathcal{A}^{\{+,\cdot\}}$; i.e., are sets of strings over ground terms and variables of $\mathcal{A}^{\{+,\cdot\}}$).

The key observation for the theory of *IS* is that none of the variables can contain $\mathbf{1}$ (one can think of $\mathbf{1}$ as the empty string) among their summands. This is because otherwise we would have on the rhs a $\mathbf{1}$ which has to match on the lhs. This is not possible because all the summands on the lhs start with some w_i or w'_i and all these are just strings over a set of constants and $\mathbf{1}$ is not part of them because of the axiom (6) which removes any occurence of $\mathbf{1}$ in a sequence.

The rest of the proof follows [ANR04] faithfuly. We restate it here to be complete. Another important observation is that not all the X_i can be $\mathbf{0}$ in the solution of the unification problem because then V would be $w_0\cdot\#$. This means that also all Y_i have to be $\mathbf{0}$ because V does not contain any string ending in one of the special constants $\{\overline{1},\ldots,\overline{n}\}$. But this would imply that $V = w'_0 \cdot \#$ which is a contradiction as $w_0 \neq w'_0$.

Therefore, V contains at least one word ending in one of the constants $\{\overline{1},\ldots,\overline{n}\}$. Take from the rhs of the first equation the word v of maximal length; this must come from V and not be in any of the X_i. Otherwize, if v is in X_i then in the lhs there must be the word $w_i \cdot v \cdot \overline{i}$ and hence also in the rhs. But his word has greater length than v and ends in the same constant $\overline{i}$, hence the contradiction.

Take the word $v \in V$ of maximal length which is of the form $w_i \cdot w \cdot \overline{i}$, for some $i \in \overline{1,n}$; otherwize, for $v = w_0 \cdot \#$ our recursive resoning stops. Therefore, $v = w_i \cdot w \cdot \overline{i}$ with $w \in X_i$ appears in both parts and can be removed. We remain with w in the rhs and all the words in (the solution for) V have now length smaller than v. We continue a similar reasoning for w which must also be a the form $w_j \cdot w \cdot \overline{j}$ with $j \neq i$. In the end we reach $w_0 \cdot \#$ and stop with a word in V of the form $w_{i_1} \cdot \cdots \cdot w_{i_n} \cdot w_0 \cdot \# s$, where s is a word over the special constants $\{\overline{1},\ldots,\overline{n}\}$ (corresponding to the indexes $\{i_1,\ldots,i_n\}$). With the same reasoning we find for the second equation that the word $w'_{i_1} \cdot \cdots \cdot w'_{i_n} \cdot w_0 \cdot \# s$ is in V with the same s. This would give the solution to the MPCP. Therefore, if we can decide the unification problems as the above we ca decide the MPCP. $\Box$

We go on to prove the decidability of the matching problem for *IS*. The matching problem is to decide if there exists a substitution solution (and give an algorithm to find it if the answer is afirmative) for an equation $u =_{IS} t$ where t is a ground term and u may contain variables. One can normalize the two terms in *IS*, cf. the remark above, and the problem becomes

$$\underset{k}{+}\underset{l}{\cdot}\, u_{kl} =_{IS} \underset{i}{+}\underset{j}{\cdot}\, t_{ij} \qquad (M_{IS})$$

for some $i,j,k,l \in \mathbb{N}$ and $i \in \overline{1,n}$. Moreover, we can search for only ground substitutions (i.e., substitutions that for any variable introduce a ground term) that introduce only terms in normal form.

3.1 Stratified theories

Definition 3.4 (factors,subterms) *Let $\mathcal{F}$ be a signature and $f \in \mathcal{F}$. For a term u, we define the multiset of its f-factors by:*

- $\mathsf{Fact}_f(u) = \{u\}$, *if $top(u) \neq f$*
- $\mathsf{Fact}_f(u) = \mathsf{Fact}_f(u_1) \uplus \ldots \uplus \mathsf{Fact}_f(u_n)$, *if $u = f(u_1,\ldots,u_n)$*

The multiset of f-subterms of u is given by:

- $\mathsf{St}_f(u) = \mathsf{St}_f(u_1) \uplus \ldots \uplus \mathsf{St}_f(u_n)$, *if* $u = g(u_1, \ldots, u_n)$ *and* $g \neq f$
- $\mathsf{St}_f(u) = \mathsf{Fact}_f(u) \uplus \mathsf{St}_f(\mathsf{Fact}_f(u))$, *otherwise.*

We have used the notation $\mathsf{St}_f(M) = \uplus_{t \in M} \mathsf{St}_f(t)$, *for a multiset of terms* M.

The goal of f-subterms is to collect all the occurences of terms that are not headed with f and are under an f in a given term.

Example 3.1

Definition 3.5 (Stratified theories) *Let f be a function symbol from a signature $\mathcal{F}$. We say that a theory $\mathcal{E}$, based on $\mathcal{F}$, is* f-stratified *if for each term t there exists a term u s.t.*

- $t =_E u$ *and*
- $u = C[u_1, \ldots, u_n]$, *where* $C \in \mathcal{T}(\{f\}, \mathcal{X})$ *and* $u_1, \ldots, u_n \in \mathcal{T}(\mathcal{F} \setminus \{f\}, \mathcal{X})$.
- $|\mathsf{St}_f(t)| \leq n$

We will say that u is a canonical form of t, denoted by $\overline{t}$.

The first two conditions express the fact that any term can be transformed into an equivalent one where the symbol f occurs only in the upper part. The third condition ensures that, in this transformation, the number of f-subterms does not decrease. This intuitively means that the rewriting did not destroy the existing f-subterms, but only possibly recombined them into new ones.

Example 3.2 (or lemma) *IS, SKA and SAA are $+$-stratified.*

In the following, we consider a $+$-stratified theory, for a given AC-symbol $+$, with a neutral element 0. The $+$-factors will be called simply factors and denoted by Fact instead of Fact_+. Our first goal is to show that the matching problem in such a theory is NP-reducible to the matching problem in the "bottom" layer:

Definition 3.6 (bottom layer) *Given a $+$-stratified equational theory $\mathcal{E}$, its* bottom layer *is defined by the set of equations $\mathcal{E}_\perp \subseteq \mathcal{E}$ such that $+$ does not appear in $\mathcal{E}_\perp$.*

Example 3.3 *$IS_\perp$, $SKA_\perp$ and $SAA_\perp$ are ...*

To achieve full-separation of the bottom layer from $+$ we need the following independence conditions:

Definition 3.7 (+-separability) *Let $\mathcal{E}$ be an $+$-stratified theory and $\mathcal{E}_\perp$ be its bottom layer. We say that $\mathcal{E}_\perp$ is* $+$-separable *if:*

- *for all terms $u, t \in \mathcal{T}(\mathcal{F} \setminus \{+\})$, we have*

  $$u =_\mathcal{E} t \Leftrightarrow u =_{\mathcal{E}_\perp} t$$

- *for all terms $u_1, \dots, u_k, t_1, \dots, t_n \in \mathcal{T}(\mathcal{F} \setminus \{+\})$ s.t. $t_1 \neq_\mathcal{E} 0, \dots, t_n \neq_\mathcal{E} 0$, we have*

  $$u_1 + \dots + u_k =_\mathcal{E} t_1 + \dots + t_n \Leftrightarrow \{u_1, \dots, u_k\} =_\mathcal{E} \{t_1, \dots, t_n\} \uplus \{u_i \,|u_i =_\mathcal{E} 0\}$$

  where the second $=_\mathcal{E}$ is the equality of multisets modulo $\mathcal{E}$ and $\uplus$ is the multiset union.

- *for all term t, we have*

  $$top(t) \neq + \ \& \ 0 \notin \mathsf{St}_+(t) \ \& \ 0 \in \mathsf{Fact}_+(\bar{t}) \implies t =_{\mathcal{E}_\perp} 0$$

The first separability condition is natural. The second condition ensures that, once $+$ is on top, it does not interfere with other operators. The last condition is more technical: it will be used to show that we can concentrate on solutions with "persistent" variables.

Example 3.4 (or lemma?) *$IS_\perp, SKA_\perp$ and $SAA_\perp$ are $+$-separable.*

Definition 3.8 (persistent variables, persistent solutions) *Let $\mathcal{E}$ be an $+$-stratified theory. Let u be a term and σ be a ground substitution. We call a variable X of u* persistent *for σ iff X appears in a factor v of u such that $v\sigma \neq_\mathcal{E} 0$.*

Given a matching problem $u = t$ and one of its solutions σ, we say that σ is a persistent solution *of $u = t$ if all the variables of u are persistent for σ.*

We first show that, when reducing matching modulo $\mathcal{E}$ to matching modulo $\mathcal{E}_\perp$, it is sufficient to consider persistent solutions:

Lemma 3.9 *Let $\mathcal{E}$ be a $+$-stratified theory such that $\mathcal{E}_\perp$ is separable. A matching problem $u = t$ is $\mathcal{E}$-solvable iff there are terms $u_1, \dots, u_n \in \mathsf{Fact}(u)$ and a substitution ρ such that:*

- *ρ is a solution of $u_1 = 0, \ldots, u_n = 0$ modulo $\mathcal{E}_\perp$*
- *$u\rho\!\downarrow = t$ has a persistent solution modulo $\mathcal{E}$.*

Proof: The "if" direction is obvious.

Now let σ be a (normalized) solution of $u = t$. We choose $u_1, \ldots, u_n$ to be the set $\{v \mid v \in \mathsf{Fact}(u), v\sigma =_{\mathcal{E}} 0\}$. We define the substitution ρ as follows .

Let us show first that ρ is a solution of $u_1 = 0, \ldots, u_n = 0$ modulo $\mathcal{E}$. By the choice of $u_1, \ldots, u_n$, we have $u_1\sigma\!\downarrow = 0, \ldots, u_n\sigma\!\downarrow = 0$. □

Therefore, in the following, we only consider persistent solutions.

Because of the axiom (7) the other variables dissapear whenever they appear in a product with some variable substituted with $\mathbf{0}$.

Lemma 3.10 (width lemma) *For any solution σ for the matching problem M_{IS} it holds that for any X a persistent variable for σ in the lhs of M_{IS}, the ground term $\sigma(X)$ is a $v_1 + \cdots + v_{n(X)}$, with $v_i : \mathcal{A}^{\{\cdot\}}$, s.t. $n(X) \leq n$.*

Proof: The lemma puts a bound on the width (i.e., the number of summands in a normal form) of the variables in the lhs of M_{IS}.

Recall that all t_{ij} are different (and the same for u_{kl}). This means that in the rhs of M_{IS} there is a sum of n products. We prove the lemma by *reductio ad absurdum* and assume that for some X, $\sigma(X) = v_1 + \cdots + v_{n+1}$. This X appears in the lhs in one of the products, say in $u_{1l} = u' \cdot X \cdot u''$. After the substitution, by way of normalization, u_{1l} becomes $u' \cdot v_1 \cdot u'' + \cdots + u' \cdot v_{n+1} \cdot u''$. Because v_i are different pairwise and are not $\mathbf{0}$, it implies that the products $u' \cdot v_i \cdot u''$ are different. Hence we have $n + 1$ different products on the lhs and only n products on the rhs of M_{IS}. □

Theorem 3.11 (matching in *IS*) *The matching problem modulo IS is decidable in NP-time.*

Proof : We give a nondeterministic algorithm for finding (if one exists) a substitution solution for the matching problem M_{IS}. We think of the matching problem from before as

$$\underset{k}{+}\, u_k =_{IS} \underset{i}{+}\, t_i \qquad (M_{IS})$$

where u_k is a product containing variables and t_i are ground products.

1. Guess some of the variables X_i in M_{IS} and replace them in lhs of M_{IS} by $\mathbf{0}$. Remove all products u_k in lhs of M_{IS} that contain $\mathbf{0}$. We are left with a problem M'_{IS} where variables are of type $\mathcal{A}^{\{+,\cdot\}} \setminus \{\mathbf{0}\}$.

2. For each variable X in the lhs of M'_{IS},

   (a) guess a number $n(X) \leq n$, cf. Lemma 3.10;

   (b) introduce $n(X)$ fresh variables $X_1, \ldots, X_{n(X)}$ of type $\mathcal{A}^{\{\cdot\}} \setminus \{\mathbf{0}\}$ (i.e., the new variables can be substituted only with terms that are products; no $+$ involved and no $\mathbf{0}$);

   (c) replace X with $X_1 + \cdots + X_{n(X)}$ in M'_{IS};

   (d) normalize the lhs of M'_{IS};

   We have now a new matching problem:

   $$\underset{k'}{+} u'_{k'} =_{IS} \underset{i}{+} t_i \qquad (M''_{IS})$$

   where all variables are of type $\mathcal{A}^{\{\cdot\}} \setminus \{\mathbf{0}\}$.

3. Guess an n-partition of the set of products $u'_{k'}$.

   Test using an NP-algorithm for matching modulo A($\cdot$)+U(1) for the $\cdot$ to see if each t_i can be matched with all the u_k from the corresponding partition (see Narendran [KN87]). Return the answer.

□

The synchronous actions algebra *SAA* of Table 4 increases the complexity of the *IS* theory by adding an operator that models synchronous execution of actions. This algebra was investigated in [Pri10] and is essential in giving the semantics of the action-based contracts language $\mathcal{CL}$ of [PS09].

Theorem 3.12 (undecidability of *SAA***)** *Unification with constants modulo the theory of SAA is undecidable.*

Proof : We use the same reduction from the MPCP as in the proof of Theorem 3.3. In the case of *SAA* the alphabet over which the words w_i and w'_i are built is $\mathcal{P}(\mathcal{A}_B)$; or equivalently, elements of $\mathcal{A}_B^{\times}$. Consider the following unification problem in *SAA*:

$$\begin{gathered} w_1 \cdot X_1 \cdot \overline{1} + \cdots + w_n \cdot X_n \cdot \overline{n} + w_0 \cdot \# = X_1 + \cdots + X_n + V, \\ w'_1 \cdot Y_1 \cdot \overline{1} + \cdots + w'_n \cdot Y_n \cdot \overline{n} + w'_0 \cdot \# = Y_1 + \cdots + Y_n + V, \end{gathered}$$

The same arguments as before can be made to say that no X of Y can be $\mathbf{1}$ and there must exists at least one X and one Y not equat to $\mathbf{0}$.

The proof is based on Corollary 2.8; i.e., ground terms of SAA can be rewritten as a sum of products of synchronous actions:

$$\underset{i\in I}{+}\ \underset{j\in J}{\cdot}\ \alpha_{\times}^{ij}$$

with $\alpha_{\times}^{ij} \in \mathcal{A}^{\{\times\}}$. We consider as before only ground substitutions.

Thus, the solution for V must contain products ending with symbols from $\{\overline{1}, \ldots, \overline{n}\}$. It is clear that products of maximal length from the rhs of the equality must come from V, for otherwise if they come from some X then there would be some longer word in lhs which would have to match some word in the rhs; impossible.

Take a word of maximal length from the rhs, thus from V; this must be of the form $w_i \times w \times \overline{i}$ with $w = \cdot_j\, \alpha_{\times}^{j}$ for some j; i.e., w is a product of synchronous actions. In consequence, w is part of the solution for X_i. The argument continues as in [ANR04]. □

Definition 3.13 (length of terms) *The* length of a ground term α *is defined (inductively) as a function* $len : SAA \to \mathbb{N}$*:*

- $len(\mathbf{1}) = len(\mathbf{0}) = 0$,
- $len(a) = 1$, *for any constant* a *of* $\mathcal{A}_B$,
- $len(\alpha \times \beta) = len(\alpha + \beta) = max(len(\alpha), len(\beta))$,
- $len(\alpha \cdot \beta) = len(\alpha) + len(\beta)$.

The length of a ground term is calculated in time linear in the number of constants and function symbols that constitute the term. Intuitively, for a term in normal form as in Corollary 2.8 (i.e., a sum of products of synchronous actions) the length function counts the dimension of the maximal product among all the summands.

Lemma 3.14 (length lemma) *For a matching problem*

$$\underset{k}{+}\, u_k =_{SAA} \underset{i}{+}\, t_i \qquad (M_{SAA})$$

with $len(+_i\, t_i) = m$ *and all variables of type* $\mathcal{A}^{\{\cdot,\times\}}$*, if* σ *is some solution for* M_{SAA} *then for all variables* X *the value* $\sigma(X) = w_1 \cdot \cdots \cdot w_{len(\sigma(X))}$*, with* $w_i \in \mathcal{A}^{\{\times\}}$*, has length* $len(\sigma(X)) \leq m$*.*

Proof : The proof is simple using *reductio ad absurdum.* The main idea is that if any variable is assigned a term with length greater than m then it is not possible to find a matching product in the rhs as all have length less or equal to m. $\square$

Theorem 3.15 (matching in *SAA***)** *The matching problem modulo SAA is decidable in NP-time.*

Proof: We follow similar arguments as those from the proof of Theorem 3.11 ony that for SAA we use the normal form of Corollary 2.8. The matching problem for SAA is translated as:

$$\underset{k}{+} u_k =_{SAA} \underset{i}{+} t_i \qquad (M_{SAA})$$

for some $i, k \in \mathbb{N}$ and $u_k \in \mathcal{A}^{\{\cdot,\times\}}$ with variables of type $X : \mathcal{A}^{\{+,\cdot,\times\}}$. Consider the bound on i to be n, where n can be calculated in time linear in the size of the rhs of M_{SAA}.

1. Guess some of the variables X_i in M_{SAA} and replace them in lhs of M_{SAA} by $\mathbf{0}$. Remove all products u_k in lhs of M_{SAA} that contain $\mathbf{0}$. We are left with a problem M'_{SAA} where variables are of type $\mathcal{A}^{\{+,\cdot,\times\}} \setminus \{\mathbf{0}\}$.

2. For each variable X in the lhs of M'_{SAA},

   (a) guess a number $n(X) \leq n$, cf. width Lemma 3.10;

   (b) introduce $n(X)$ fresh variables $X_1, \ldots, X_{n(X)}$ of type $\mathcal{A}^{\{\cdot,\times\}} \setminus \{\mathbf{0}\}$ (i.e., the new variables can be substituted only with terms that do no contain $+$ or $\mathbf{0}$, but only $\cdot$ and/or $\times$);

   (c) replace X with $X_1 + \cdots + X_{n(X)}$ in M'_{SAA};

   (d) normalize the lhs of M'_{SAA}.

   We have now a new matching problem:

   $$\underset{k'}{+} u'_{k'} =_{SAA} \underset{i}{+} t_i \qquad (M''_{SAA})$$

   where all variables are of type $\mathcal{A}^{\{\cdot,\times\}} \setminus \{\mathbf{0}\}$. We can use the length Lemma 3.14 and simplify the matching problem even more.

3. For each variable X' in the lhs of M''_{SAA},

   (a) guess a number $len(X') \leq len(rhs)$, cf. length Lemma 3.14;

(b) introduce $len(X')$ fresh variables $X'_1, \ldots, X'_{len(X')}$ of type $\mathcal{A}^{\{\times\}} \setminus \{\mathbf{0}\}$ (i.e., all the variables in the lhs can be replaced only with terms build from basic actions of $\mathcal{A}_B$ using $\times$);

(c) replace X' with $X'_1 \cdot \cdots \cdot X'_{len(X')}$ in M''_{SAA};

(d) normalize the lhs of M''_{SAA}.

The more simple matching problem is now

$$\underset{k''}{+} u''_{k''} =_{SAA} \underset{i}{+} t_i \qquad (M'''_{SAA})$$

where all variables in the lhs are of type $\mathcal{A}^{\{\times\}}$.

4. Guess an n-partition of the set of products $u''_{k''}$.

   We are left with a matching problem over the two operators $\cdot$ and $\times$ which do not interact. This means that we have disjoint theories for the two: for $\cdot$ we have AU($\mathbf{1}$) and for $\times$ we have the theory ACU($\mathbf{1}$)I($\mathcal{A}_B$). We can apply a technique for combining the NP algorithms for the two theories into one as described in [BS96]. In this way we check to see if each t_i can be matched with all the $u''_{k''}$ from the corresponding partition. Return the answer.

□

Theorem 3.16 (general matching algorithm)

4 Conclusion

References

[AHK08] Kamal Aboul-Hosn and Dexter Kozen. Local variable scoping and kleene algebra with tests. *J. Log. Algebr. Program.*, 76(1):3–17, 2008.

[ANR04] Siva Anantharaman, Paliath Narendran, and Michaël Rusinowitch. Unification Modulo ACUI Plus Distributivity Axioms. *J. Autom. Reasoning*, 33(1):1–28, 2004.

[BMR97] Stefano Bistarelli, Ugo Montanari, and Francesca Rossi. Semiring-based constraint satisfaction and optimization. *Journal of ACM*, 44(2):201–236, 1997.

[BS96] Franz Baader and Klaus U. Schulz. Unification in the Union of Disjoint Equational Theories: Combining Decision Procedures. *J. Symbolic Computation*, 21(2):211–243, 1996.

[Con71] John Horton Conway. *Regular Algebra and Finite Machines*. Chapman and Hall, 1971.

[FL77] Michael J. Fischer and Richard E. Ladner. Propositional modal logic of programs. In *9th ACM Symposium on Theory of Computing (STOC'77)*, pages 286–294. ACM, 1977.

[HMSW09a] C. A. R. Hoare, Bernhard Möller, Georg Struth, and Ian Wehrman. Foundations of Concurrent Kleene Algebra. In Rudolf Berghammer, Ali Jaoua, and Bernhard Möller, editors, *International Conference on Relations and Kleene Algebra in Computer Science (RelMiCS'09)*, volume 5827 of *Lecture Notes in Computer Science*, pages 166–186. Springer, 2009.

[HMSW09b] Tony Hoare, Bernhard Möller, Georg Struth, and Ian Wehrman. Concurrent Kleene Algebra. In Mario Bravetti and Gianluigi Zavattaro, editors, *20th International Conference on Concurrency Theory (CONCUR'09)*, volume 5710 of *LNCS*, pages 399–414. Springer, 2009.

[KN87] Deepak Kapur and Paliath Narendran. Matching, Unification and Complexity. *ACM SIGSAM Bulletin*, 21(4):6–9, 1987.

[Koz79] Dexter Kozen. On the duality of dynamic algebras and kripke models. In *Logic of Programs, Workshop*, volume 125 of *LNCS*, pages 1–11. Springer-Verlag, 1979.

[Koz94] Dexter Kozen. A completeness theorem for kleene algebras and the algebra of regular events. *Information and Computation*, 110(2):366–390, 1994.

[Koz97] Dexter Kozen. Kleene algebra with tests. *ACM Transactions on Programming Languages and Systems (TOPLAS'97)*, 19(3):427–443, 1997.

[Koz98] Dexter Kozen. Typed kleene algebra. Technical Report 1669, Computer Science Department, Cornell University, March 1998.

[Koz00] Dexter Kozen. On hoare logic and kleene algebra with tests. *Transactions on Computational Logic*, 1(1):60–76, July 2000.

[Koz08] Dexter Kozen. Nonlocal flow of control and kleene algebra with tests. In *23rd IEEE Symposium on Logic in Computer Science (LICS'08)*, pages 105–117. IEEE Computer Society, 2008.

[KS86] Werner Kuich and Arto Salomaa. *Semirings, Automata, Languages*. Springer-Verlag, Berlin, 1986.

[Mil83] Robin Milner. Calculi for synchrony and asynchrony. *Theorethical Computer Science*, 25:267–310, 1983.

[Moh02] Mehryar Mohri. Semiring frameworks and algorithms for shortest-distance problems. *J. Automata, Languages and Combinatorics*, 7(3):321–350, 2002.

[NFM$^+$05] Rocco De Nicola, Gian Luigi Ferrari, Ugo Montanari, Rosario Pugliese, and Emilio Tuosto. A Process Calculus for QoS-Aware Applications. In Jean-Marie Jacquet and Gian Pietro Picco, editors, *7th International Conference Coordination Models and Languages (COORDINATION'05)*, volume 3454 of *Lecture Notes in Computer Science*, pages 33–48. Springer, 2005.

[Pra90] Vaughan R. Pratt. Dynamic algebras as a well-behaved fragment of relation algebras. In Clifford H. Bergman, Roger D. Maddux, and Don L. Pigozzi, editors, *Algebraic Logic and Universal Algebra in Computer Science*, volume 425 of *LNCS*, pages 77–110. Springer-Verlag, 1990.

[Pri10] Cristian Prisacariu. Synchronous Kleene Algebra. *The Journal of Logic and Algebraic Programming*, 2010. (to appear).

[PS09] Cristian Prisacariu and Gerardo Schneider. CL: An Action-based Logic for Reasoning about Contracts. In *Workshop on Logic, Language, Informations and Computation (WOLLIC'09)*, volume 5514 of *LNCS*, pages 335–349. Springer, 2009.

[Sal66] Arto Salomaa. Two complete axiom systems for the algebra of regular events. *Journal of ACM*, 13(1):158–169, 1966.