Unification and Matching in Separable Theories*

Sergiu Bursuc[1]** and Cristian Prisacariu[2]***

[1] School of Computer Science, Univ. of Birmingham, Birmingham B15 2TT, UK.
s.bursuc@cs.bham.ac.uk
[2] Dept. of Informatics, Univ. of Oslo, P.O. Box 1080 Blindern, 0316 Oslo, Norway.
cristi@ifi.uio.no

Abstract. We study unification and matching in equational theories based on semirings, which include Kleene algebra and extensions with different forms of concurrency, constraint semirings, and synchronous actions algebra. Generally the unification problems are undecidable in this setting, but different undecidability proofs are required. On the other hand, the matching problems are decidable and a general pattern can be drawn. This pattern is developed into a matching algorithm, relying on a new way of combining non-disjoint theories, which we call stratification, and on a relaxation of the finite variant property, which we call separability. Consequently, we believe that our algorithm and the notions that we introduce have an importance i) beyond theories based on semirings; ii) for other problems related to unification and matching.

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 consider in this paper are *synchronous actions algebras* (*SAA*) [16] and *constraints semirings* (*CS*) and combinations [3,15]. Not considered here, but in the scope of our work are *Kleene algebra* [7] and concurrent extensions of Kleene algebra [16,9].

The synchronous actions of *SAA* are the basis of the contract logic from [17]. Constraint semirings have been proposed in [3] to model various notions of constraints and have been used in [15] to define a process algebra with constraints. Semirings alone have been used in [14] for a general algebraic framework for shortest-distance problems.

For all these formalisms unification and matching problems occur naturally. Even if matching is a restricted version of unification, sometimes it is more relevant in practice. For example, matching in Kleene algebras or *SAA* can be used in formal synthesis of systems: on the one side of the equality we have the

* An extended version appeared as the technical report [4].

** This author is supported by the EP/G02684X/1 project on Trustworthy Voting Systems

*** This author was partially supported by the Nordunet3 project "COSoDIS – Contract-Oriented Software Development for Internet Services".

(1)	$x + (y + z) = (x + y) + z$	(10)	$x \times (y \times z) = (x \times y) \times z$
(2)	$x + y = y + x$	(11)	$x \times y = y \times x$
(3)	$x + \mathbf{0} = \mathbf{0} + x = x$	(12)	$x \times \mathbf{1} = \mathbf{1} \times x = x$
(4)	$x + x = x$	(13)	$x \times \mathbf{0} = \mathbf{0} \times x = \mathbf{0}$
(5)	$x \cdot (y \cdot z) = (x \cdot y) \cdot z$	(14)	$a \times a = a \quad \forall a \in \mathcal{A}_B$
(6)	$x \cdot \mathbf{1} = \mathbf{1} \cdot x = x$	(15)	$x \times (y + z) = x \times y + x \times z$
(7)	$x \cdot \mathbf{0} = \mathbf{0} \cdot x = \mathbf{0}$	(16)	$(x + y) \times z = x \times z + y \times z$
(8)	$x \cdot (y + z) = x \cdot y + x \cdot z$	(17)	$(t_\times \cdot x) \times (t'_\times \cdot y) = (t_\times \times t'_\times) \cdot (x \times y)$,
(9)	$(x + y) \cdot z = x \cdot z + y \cdot z$		$\forall t_\times, t'_\times \in \mathcal{A}_B^\times$

(18) $x + \mathbf{1} = \mathbf{1}$
(19) $x \cdot y = y \cdot x$

Table 1. Axioms of idempotent semirings *IS* (1)-(9), synchronous actions algebras *SAA* (1)-(17), and constraint semirings *CS* (1)-(9),(18),(19).

specification of the required behaviour of the system and on the other side we have the behaviour of (a part) of our system, where the unknown behaviour is represented by variables. Discovering the unkown parts of the system amounts then to solving the corresponding matching problem. The matching problem for *SAA* was left open in [16] and a particular matching problem is used in giving semantics to the contract logic $\mathcal{CL}$ in [17]. Hence, the decidability of the $\mathcal{CL}$ logic was proven relative to the decidability of this particular matching problem. The matching algorithm that we propose here for *SAA* solves the decidability of $\mathcal{CL}$.

2 Decomposition algorithm for separable theories

In this section we show that there are algorithms for matching in semiring based theories (even if unification is undecidable for such theories) and that these algorithms can be cast in a general setting. More precisely, we propose a non-disjoint combination scheme: these theories are stratified in such a way that their layers, even though non-disjoint, can be separated when one is interested in a solution for a matching problem.

Definition 1 (examples of theories considered). *An* idempotent semiring *is the algebraic structure* $IS = (\mathcal{A}, +, \cdot, \mathbf{0}, \mathbf{1})$ *that respects axioms (1)-(9) of Table 1. A* synchronous actions algebra *[16]* $SAA = (\mathcal{A}, +, \cdot, \times, \mathbf{0}, \mathbf{1})$ *is generated from a set of constants* $\mathcal{A}_B$ *and is axiomatized by (1)-(17) of Table 1.* $\mathcal{A}_B^\times$ *is the set of ground terms constructed from* $\mathcal{A}_B$ *closed under* $\times$*. A* constraint semiring *[3]* $CS = (\mathcal{A}, +, \cdot, \mathbf{0}, \mathbf{1})$ *is an idempotent semiring where* $\mathbf{1}$ *is absorbing element for* $+$ *and* $\cdot$ *is commutative (i.e., axioms (1)-(9),(18),(19) of Table 1).*

Unification in idempotent semirings and synchronous actions algebras is undecidable. To prove undecidability we addapt to our theories the method of [1] which uses reduction from the modified Post correspondence problem. (We cojecture that unification in constraint semirings is also undecidable and the proof can make use of Minski machines in the style of the undecidability result for ACUIH^C from [1].) Nevertheless, matching is decidable for these theories and the main ideas behind our combination scheme and general decomposition algorithm for matching are the following.

Definition 2 (stratification). *We say that an equational theory $\mathcal{E}$ is* stratified w.r.t. *f, where $f \in \mathcal{F}$ is a functional symbol from $\mathcal{E}$ (or simply call $\mathcal{E}$ f-*stratified*), iff $\mathcal{E}$ is the union of*

- *an* upper layer*: $\mathcal{E}_\top \subseteq \mathcal{E}$ – which contains all and only the equations of $\mathcal{E}$ that contain only f, variables, and constants;*
- *a* bottom layer*: $\mathcal{E}_\bot \subseteq \mathcal{E}$ – which contains all and only the equations of $\mathcal{E}$ in which f does not occur;*
- *an* interface layer*: $\mathcal{E}_\vdash \subseteq \mathcal{E}$ – which consists of all the other equations; i.e., $\mathcal{E}_\vdash = \mathcal{E} \setminus (\mathcal{E}_\top \cup \mathcal{E}_\bot)$ – and where the interface layer has to respect the following* stratification restriction*:*
  $\mathcal{E}_\vdash$ can be directed into a convergent rewrite system $\mathcal{R}_\vdash$ s.t. for any term u its interface canonical form *$\overline{u}$ (defined as $\overline{u} = u{\downarrow_{\mathcal{R}_\vdash}}$) is of the form $C[u_1, \ldots, u_k]$, where $C \in \mathcal{T}(\{f\}, \mathcal{X})$ and $u_1, \ldots, u_k \in \mathcal{T}(\mathcal{F} \setminus \{f\}, \mathcal{X})$.*

Example 1. Suppose that $\mathcal{E}_\vdash$ contains only equations where f distributes over all other function symbols g, e.g. of the form $f(g(x_1, x_2)) = g(f(x_1), f(x_2))$. Consider the system $\mathcal{R}_\vdash$ obtained by orienting all the equations from $\mathcal{E}_\vdash$ as $l \to r$ with $top(r) = f$, i.e. $\mathcal{R}_\vdash$ moves the symbols f to the top. It is easy to see that $\mathcal{R}_\vdash$ is terminating and the canonical form gathers all the f symbols at the top. Hence, the *stratification restriction* is respected.

Example 2. The theories *IS* and *SAA* are particular cases of the previous example. They are stratified with respect to $+$, where $IS_\vdash$ is made of equations (8), (9) and $SAA_\vdash$ is made of (8), (9), (15), (16). Moreover, $SAA_\bot$ is stratified as well, with respect to $\cdot$, where $SAA_{\bot\vdash}$ is made of the equation (17).

We will prove that matching modulo a stratified theory $\mathcal{E}$ can be reduced to matching modulo its upper layer $\mathcal{E}_\top$ and matching modulo its bottom layer $\mathcal{E}_\bot$. To be able to separate $\mathcal{E}_\top$ from $\mathcal{E}_\bot$, we need their interaction to be handled only by the interface layer $\mathcal{E}_\vdash$, and in a particular way described in the following.

Definition 3 (separability). *We say that an f-stratified theory $\mathcal{E}$ is* f-separable *iff for any term u and ground term t, there is a finite set of substitutions $\theta_1, \ldots, \theta_n$ such that:*

$$\exists \sigma : \ u\sigma =_{\mathcal{E}} t \Leftrightarrow \exists i, \sigma' : (\overline{u\theta_i})\sigma' =_{\mathcal{E}_\top \cup \mathcal{E}_\bot} t$$

Intuitively, separability states that we can guess in advance all the reductions in the interface layer: they are determined by the set of substitutions $\theta_1, \ldots, \theta_n$.

Example 3. *IS* and *SAA* are separable. The set of substitutions $\theta_1, \ldots, \theta_n$ is given by the possible ways to assign $\mathbf{0}$ to variables and then by bounds given depending on a measure of t w.r.t. the $+$ operator, which we call *width*. For example, for the matching problem $x + y = a + b + c \cdot d \times d'$, two of the substitutions would be $\theta_1 = \{x \mapsto 0, y \mapsto y_1 + y_2 + y_3\}$ and $\theta_2 = \{y \mapsto y_1 + y_2\}$.

Furthermore, $SAA_\bot$ is also separable. For example, one of the $SAA_\bot$-matching problems obtained above, $y_3 = c \cdot d \times d'$, would further be reduced by the substitution $\theta_3 = \{y_3 \mapsto z_1 \cdot z_2\}$.

Decomposition algorithm for matching. Let $u =_{\mathcal{E}} t$ be a matching problem modulo an f-stratified and f-separable theory $\mathcal{E}$. We may assume, without loss of generality, that u and t are in interface canonical form.

Step 1. By separability, we know that $u =_{\mathcal{E}} t$ is solvable iff there is a computable term v in interface canonical form s.t. $v =_{\mathcal{E}_\top \cup \mathcal{E}_\bot} t$ is solvable. Separability ensures that there are finitely many v's (i.e., $v = \overline{u\theta_i}$ for some i), hence we can just choose such a v.

Step 2. Because v, t are in interface canonical form, they have the form $v = C[v_1, \ldots, v_n]$ and $t = C'[t_1, \ldots, t_m]$, with $C, C' \in \mathcal{T}(\{f\}, \mathcal{X})$ and $v_1, \ldots, v_n$, $t_1, \ldots, t_m \in \mathcal{T}(\mathcal{F} \setminus \{f\}, \mathcal{X})$. Since $\mathcal{E}_\bot$ and $\mathcal{E}_\top$ share only constants, we can prove that $v =_{\mathcal{E}_\top \cup \mathcal{E}_\bot} t$ is solvable iff there is a function F that assigns to $v_1, \ldots, v_n$ either a term in $\{t_1, \ldots, t_m\}$ or a constant such that
- $C[F(v_1), \ldots, F(v_n)] =_{\mathcal{E}_\top} C'[t_1, \ldots, t_m]$; (*)
- $v_1 =_{\mathcal{E}_\bot} F(v_1) \wedge \ldots \wedge v_n =_{\mathcal{E}_\bot} F(v_n)$; (**)

Step 3. Guess a function F (there are only finitely many trials to make) and check that it satisfies the ground $\mathcal{E}_\top$-word problem (*).

Step 4. If (*) is satisfied, solve the $\mathcal{E}_\bot$-matching problem (**).

3 Related work and conclusion

Stratification and the corresponding decompositon algorithm relate to the works on the combination of disjoint theories, e.g. [2]. Indeed, after a first step that eliminates the interface layer, our algorithm separates the (almost) disjoint combination of the bottom layer and the upper layer. In the context of formal verification of security protocols, [5] proposes a way to hierarchize a theory. However, their notion of hierarchy assumes that the two theories cannot be swapped in the same equation. This is not the case in semiring-based theories, because of distribitivity axioms.

Separability is related to the finite variant property of [6]. It has already been shown in [8] how the finite variant property is useful for solving unification problems. However, as we have seen, unification is undecidable for our theories and also the finite variant property does not hold: [6] shows that this is the case for any theory that has homomorphism-like properties. Separability can therefore be seen as a first relaxation of the finite variant property, suitable in the context of matching and possibly other applications.

An approach to unification in the light of notions simmilar to stratification and separability is followed by [12]. The structure that is ensured by stratification here is ensured by a sort theory there. However, even if a sort theory could be assigned to semiring-based theories, what we call here the bottom layer would not be separable in the sense of [12]. Indeed, they require for none of the separated symbols to appear in the rest of the equations. This is not the case for $\cdot$ from the bottom layer, which also appears in the interface layer.

In a broader setting, stratification and separability relate to hierarchical and local reasoning in logical theories [13,10,11]. For instance, separability can be

seen as a locality notion with respect to the ground right hand side, where the *local* set is determined by the bounded set of substitutions.

In the long version of this work, we will show that our general approach can be applied to any semiring based theory. In future, we will try to tighten the links with the above-mentioned related works. We will also investigate how stratification and separability relate to other theories and to restricted unification problems, possibly decidable and relevant in practice, e.g. where we bound the number of variables, the number of constants, or both.

References

1. S. Anantharaman, P. Narendran, and M. Rusinowitch. Unification Modulo ACUI Plus Distributivity Axioms. *J. Autom. Reasoning*, 33(1):1–28, 2004.
2. F. Baader and K. U. Schulz. Unification in the Union of Disjoint Equational Theories: Combining Decision Procedures. *J. Symbolic Computation*, 21(2):211–243, 1996.
3. S. Bistarelli, U. Montanari, and F. Rossi. Semiring-based constraint satisfaction and optimization. *Journal of ACM*, 44(2):201–236, 1997.
4. S. Bursuc and C. Prisacariu. Unification and Matching in Separable Theories – technicalities. Technical Report 398, Dept. Info., Univ. Oslo, May 2010.
5. Y. Chevalier and M. Rusinowitch. Hierarchical combination of intruder theories. *Inf. Comput.*, 206(2-4):352–377, 2008.
6. H. Comon-Lundh and S. Delaune. The Finite Variant Property: How to Get Rid of Some Algebraic Properties. In *RTA'05*, volume 3467 of *LNCS*, pages 294–307. Springer, 2005.
7. J. H. Conway. *Regular Algebra and Finite Machines*. Chapman and Hall, 1971.
8. S. Escobar, J. Meseguer, and R. Sasse. Variant narrowing and equational unification. In *WRLA'08*, volume 238 of *ENTCS*, pages 103–119. Elsevier, 2009.
9. C. Hoare, B. Möller, G. Struth, and I. Wehrman. Concurrent Kleene Algebra. In *CONCUR'09*, volume 5710 of *LNCS*, pages 399–414. Springer, 2009.
10. C. Ihlemann, S. Jacobs, and V. Sofronie-Stokkermans. On local reasoning in verification. In *TACAS*, pages 265–281, 2008.
11. C. Ihlemann and V. Sofronie-Stokkermans. On hierarchical reasoning in combinations of theories. In *IJCAR*, 2010, to appear.
12. C. Lynch and B. Morawska. Faster Basic Syntactic Mutation with Sorts for Some Separable Equational Theories. In J. Giesl, editor, *RTA*, volume 3467 of *LNCS*, pages 90–104. Springer, 2005.
13. D. A. McAllester. Automatic recognition of tractability in inference relations. *J. ACM*, 40(2):284–303, 1993.
14. M. Mohri. Semiring frameworks and algorithms for shortest-distance problems. *J. Automata, Languages and Combinatorics*, 7(3):321–350, 2002.
15. R. D. Nicola, G. L. Ferrari, U. Montanari, R. Pugliese, and E. Tuosto. A Process Calculus for QoS-Aware Applications. In *COORDINATION'05*, volume 3454 of *LNCS*, pages 33–48. Springer, 2005.
16. C. Prisacariu. Synchronous Kleene Algebra. *The Journal of Logic and Algebraic Programming*, 2010. (to appear).
17. C. Prisacariu and G. Schneider. CL: An Action-based Logic for Reasoning about Contracts. In *WOLLIC'09*, volume 5514 of *LNCS*, pages 335–349. Springer, 2009.