On the Complexity of Unification and Disunification in Commutative Idempotent Semigroups

Miki Hermann[1] and Phokion G. Kolaitis[2] *

[1] LORIA (CNRS), BP 239, 54506 Vandœuvre-lès-Nancy, France, hermann@loria.fr
[2] Computer Science Department, University of California, Santa Cruz, CA 95064, U.S.A., kolaitis@cse.ucsc.edu

Abstract. We analyze the computational complexity of elementary unification and disunification problems for the equational theory ACI of commutative idempotent semigroups. From earlier work, it was known that the decision problem for elementary ACI-unification is solvable in polynomial time. We show that this problem is inherently sequential by establishing that it is complete for polynomial time (P-complete) via logarithmic-space reductions. We also investigate the decision problem and the counting problem for elementary ACI-matching and observe that the former is solvable in logarithmic space, but the latter is #P-complete. After this, we analyze the computational complexity of the decision problem for elementary ground ACI-disunification. Finally, we study the computational complexity of a restricted version of elementary ACI-matching, which arises naturally as a set-term matching problem in the context of the logic data language LDL. In both cases, we delineate the boundary between polynomial-time solvability and NP-hardness by taking into account two parameters, the number of free constants and the number of disequations or equations.

1 Introduction and Summary of Results

Among all equational theories studied in the context of unification and automated deduction, the equational theory AC of commutative semigroups is undoubtedly the one that has attracted the most attention. There are many applications, however, in which the operators at hand may satisfy other equational axioms in addition to associativity A and commutativity C. For this reason, various extensions of AC have also been examined. A particularly natural and useful such extension is the equational theory ACI of commutative idempotent semigroups, which augments AC with the idempotence axiom I: $(\forall x)(x * x = x)$. ACI provides perhaps the simplest way to consider finite sets of objects that are represented by free constants, without being hampered by the drawbacks of a full Set datatype specification. Moreover, simpler set constraints can be treated by considering unification problems in commutative idempotent semigroups.

* Research of this author was partially supported by NSF Grant CCR-9610257.

The computational complexity of general ACI-matching and general ACI-unification (that is, the terms to be unified or matched may contain both free function and free constant symbols) was investigated by Kapur and Narendran [KN86,KN92], who established that these decision problems are NP-complete. In contrast, they also proved that elementary ACI-unification with a finite number of free constants is solvable in polynomial time [KN92]. More recently, Narendran [Nar96] showed that ground elementary ACIU-disunification is NP-hard, where ACIU is the extension of ACI with a unit element.

In this paper, we investigate further the computational complexity of elementary ACI-unification and ACI-disunification with a finite number of free constants. First, we establish that elementary ACI-unification with at least two free constants is a P-hard problem, which means that every decision problem solvable in polynomial time can be reduced to elementary ACI-unification with two free constants via some logarithmic-space reduction. This complements the aforementioned result of Kapur and Narendran [KN92] stating that elementary ACI-unification with a finite number of free constants is solvable in polynomial time. Moreover, it suggests strongly that elementary ACI-unification is inherently sequential and, thus, lacks "fast parallel" algorithms (see [GHR95]). We also investigate the decision problem and the *counting* problem for elementary ACI-matching, where the latter is the problem of finding the number of minimal complete ACI-matchers of a given finite system of equations between terms. In [HK95a], we introduced counting problems in equational matching and embarked on a study of their computational complexity as a way to obtain lower bounds on the performance of algorithms for finding minimal complete sets of matchers. Here, we observe that the decision problem for ACI-matching is solvable in LOGSPACE, but the counting problem for ACI-matching is #P-complete. Since #P-complete problems are considered to be highly intractable (see [Joh90,Pap94]), this shows a dramatic difference in computational complexity between a decision problem in equational matching and its corresponding counting problem. It should be noted that Baader and Büttner [BB88] designed an algorithm for finding a minimal complete set of elementary ACI-unifiers of a *single* equation between two terms. They also computed explicitly the cardinality of this minimal set and pointed out that it can be an "enormous number".

After this, we analyze the computational complexity of the decision problem for elementary ground ACI-disunification. We delineate the boundary between polynomial-time solvability and NP-hardness by taking into account two parameters, the number of free constants and the number of disequations. Specifically, we show that, when the number of disequations is fixed, the decision problem for elementary ground ACI-disunification with any number of free constants is solvable in polynomial time. In contrast, when the number of disequations is unbounded, the decision problem for elementary ground ACI-disunification is NP-hard, as long as at least two free constants are available (the latter result was implicit in Narendran [Nar96]).

Finally, we investigate the computational complexity of a restricted version of elementary ACI-matching, which arises naturally as a set-term matching prob-

lem in the context of the logic data language LDL. This problem asks: given a system of elementary ACI-equations, does there exist an ACI-matcher such that every variable is instantiated by a single constant? This restricted ACI-matching problem has been introduced by Shmueli, Tsur, and Zaniolo [STZ92], and also studied by Arni, Greco, and Saccà [AGS96] under the name *bounded set-term matching*. Here, we show that restricted ACI-matching with two free constants and an unbounded number of equations is NP-complete, but restricted ACI-matching with a fixed number of free constants and a fixed number of equations is solvable in polynomial time.

2 Preliminaries

A *signature* $\mathcal{F}$ is a countable set of function and constant symbols. If $\mathcal{X}$ is a countable set of variables, then $\mathcal{T}(\mathcal{F},\mathcal{X})$ denotes the set of all terms over the signature $\mathcal{F}$ and the variables in $\mathcal{X}$. A *ground term* is a term without variables.

An *identity* over $\mathcal{F}$ is a first-order sentence of the form $(\forall x_1)\ldots(\forall x_n)(l = r)$, where l and r are terms in $\mathcal{T}(\mathcal{F},\mathcal{X})$ with variables among $x_1,\ldots,x_n$. Every set E of identities can be viewed as the set of *equational axioms* of an *equational theory* Th(E) consisting of all identities over $\mathcal{F}$ that are logically implied by E. By an abuse of terminology, we will often say the "equational theory E", instead of the "equational theory Th(E)". The notation $s =_{\mathrm{E}} t$ denotes that the identity $(\forall x_1)\ldots(\forall x_n)(s = t)$ is a member of Th(E). We write $\mathcal{T}(\mathcal{F},\mathcal{X})/=_{\mathrm{E}}$ to denote the term algebra modulo the equational theory E. Similarly, $\mathcal{T}(\mathcal{F})/=_{\mathrm{E}}$ denotes the *ground* term algebra modulo E, which is also the initial algebra of E.

An E-*unification* problem is a finite set Γ of equations $s = t$ between terms from $\mathcal{T}(\mathcal{F},\mathcal{X})$. A *solution* (or a *unifier*) of a unification problem Γ is a substitution ρ such that $s\rho =_{\mathrm{E}} t\rho$ for every equation $s = t$ in Γ, which means that the system of equations in Γ has a solution in the term algebra $\mathcal{T}(\mathcal{F},\mathcal{X})/=_{\mathrm{E}}$. Since solutions are closed under instantiations of variables by arbitrary terms, this is also equivalent to having a solution in the ground term algebra $\mathcal{T}(\mathcal{F})/=_{\mathrm{E}}$ (and, consequently, equivalent to having a solution in every model of E). An E-matching problem is an E-unification problem Γ such that for every equation $s = t$ in Γ the term t is ground.

An E-*disunification* problem is a finite set Δ of equations $s = t$ and disequations $s' \neq t'$ between terms from $\mathcal{T}(\mathcal{F},\mathcal{X})$. A *solution* of a disunification problem Δ is a substitution ρ such that $s\rho =_{\mathrm{E}} t\rho$ for every equation $s = t$ in Δ, and $s'\rho \neq_{\mathrm{E}} t'\rho$ for every disequation $s' \neq t'$ in Δ. As before, this means that the system of equations and disequations in Δ has a solution in the term algebra $\mathcal{T}(\mathcal{F},\mathcal{X})/=_{\mathrm{E}}$. It should be emphasized, however, that this is *not* always equivalent to having a solution in the ground term algebra $\mathcal{T}(\mathcal{F})/=_{\mathrm{E}}$, as solutions to systems of disequations may not be closed under substitutions of variables by ground terms. A *ground solution* of a disunification problem Δ is a solution ρ of Δ in the ground term algebra $\mathcal{T}(\mathcal{F})/=_{\mathrm{E}}$, that is, ρ is both a ground substitution and a solution of Δ. A *ground disunification problem* is a disunification problem in which only ground solutions are sought.

If E is a set of identities, then $sig(\mathrm{E})$ is the set of all function and constant symbols occurring in some member of E. From now on we assume that $\mathcal{F} \setminus sig(\mathrm{E})$ consists of constants symbols *only*. Thus, the only symbols of the signature $\mathcal{F}$ that do not occur in some member of E are *free* constants. In this case, we speak of *elementary* E-*unification* and *elementary* E-*disunification*.

In the sequel, we will analyze the computational complexity of elementary unification and disunification problems by taking into account the number of equations, the number of disequations, and the number of free constants. For this reason, we introduce the following notation. If k and m are two positive integers, then $\mathrm{E}(k;m)$ is the collection of all elementary E-unification problems Γ with k equations such that $\mathcal{F} \setminus sig(\mathrm{E})$ consists of m free constants. We put

$$\mathrm{E}(\omega;m) = \bigcup_{k\geq 1} \mathrm{E}(k;m) \qquad \text{and} \qquad \mathrm{E}(\omega;\omega) = \bigcup_{k,m\geq 1} \mathrm{E}(k;m).$$

If k is a non-negative integer, and l and m are two positive integers, then $\mathrm{E}(k,l;m)$ is the collection of all elementary E-disunification problems Γ with k equations, l disequations and such that $\mathcal{F} \setminus sig(\mathrm{E})$ consists of m free constants. We also put

$$\mathrm{E}(k,\omega;m) = \bigcup_{l\geq 1} \mathrm{E}(k,l;m) \qquad \text{and} \qquad \mathrm{E}(\omega,l;\omega) = \bigcup_{k,m\geq 1} \mathrm{E}(k,l;m),$$

Our main focus will be on the equational theory ACI of commutative idempotent semigroups. For this theory, the signature $\mathcal{F}$ consists of free constants and a binary function symbol $*$ that is assumed to be associative, commutative, and idempotent. Thus, the equational axioms of ACI are the identities

$$\mathrm{A}\colon (\forall x)(\forall y)(\forall z)(x*(y*z) = (x*y)*z),\ \mathrm{C}\colon (\forall x)(\forall y)(x*y = y*x),\ \mathrm{I}\colon (\forall x)(x*x = x)$$

Note that if $\mathcal{F} = \{*, c_1, \ldots, c_m\}$, then the ground term algebra $\mathcal{T}(\mathcal{F})/{=_{\mathrm{ACI}}}$ is the commutative idempotent semigroup freely generated by $c_1, \ldots, c_m$.

We will also encounter briefly the equational theory ACIU of commutative idempotent monoids. For this theory, the signature $\mathcal{F}$ contains also a constant symbol 1, which is the *unit* element for $*$. Thus, ACIU satisfies also the identity U: $(\forall x)(x*1 = x)$. If $\mathcal{F} = \{*, c_1, \ldots, c_m\}$, then the ground algebra $\mathcal{T}(\mathcal{F})/{=_{\mathrm{ACIU}}}$ is the commutative idempotent monoid freely generated by $c_1, \ldots, c_m$.

3 Elementary ACI-unification and ACI-matching

Kapur and Narendran [KN92] showed that the decision problem for elementary ACI-unification with a finite number of free constants is solvable in polynomial time, even when the signature $\mathcal{F}$ is part of the input. More precisely, consider the following decision problem.

ELEMENTARY ACI-UNIFICATION: Given a finite set $\{c_1, \ldots, c_m\}$ of free constants and an elementary ACI$(\omega;m)$-unification problem Γ over the signature $\mathcal{F} = \{*, c_1, \ldots, c_m\}$, does Γ have a solution?

Kapur and Narendran [KN92] showed that ELEMENTARY ACI-UNIFICATION can be reduced in polynomial time to PROPOSITIONAL HORN SATISFIABILITY, that is, to the problem: given a Horn[1] formula, does it have a satisfying truth assignment? It is well known that PROPOSITIONAL HORN SATISFIABILITY is solvable in polynomial time; as a matter of fact, it has a linear-time algorithm [DG84]). Next, we describe Kapur and Naredran's [KN92] reduction of ELEMENTARY ACI-UNIFICATION to PROPOSITIONAL HORN SATISFIABILITY in some detail, since it will be of interest to us in the sequel.

Assume that we are given a set C of m free constants and an elementary ACI($\omega\,;m$)-unification problem Γ. Let $\mathcal{X}_s$ and $\mathcal{X}_t$ be the sets of variables occurring in s and t respectively. For every given free constant c and every variable $x \in \mathcal{X}_s \cup \mathcal{X}_t$, we introduce a propositional variable $P_{x,c}$; intuitively, $P_{x,c}$ expresses the fact that the free constant c does not occur in the value of the solution for the variable x. For every equation $s = t$ in Γ, we form a set $\Theta(s,t)$ of propositional Horn clauses asserting that a free constant occurs in the left-hand side of the solved equation if and only if it occurs in the right-hand side of the solved equation. Formally, the Horn clauses in the set $\Theta(s,t)$ are obtained as follows.

- Let c be a free constant that occurs in s, but does not occur in t. If $\mathcal{X}_t \neq \emptyset$, we introduce the Horn clause $(\bigvee_{x \in \mathcal{X}_t} \neg P_{x,c})$. If $\mathcal{X}_t = \emptyset$, we conclude immediately that Γ is not ACI-unifiable, since $s = t$ is not ACI-unifiable.
- Let c be a free constant that occurs in t, but does not occur in s. If $\mathcal{X}_s \neq \emptyset$, we introduce the Horn clause $(\bigvee_{x \in \mathcal{X}_s} \neg P_{x,c})$. If $\mathcal{X}_s = \emptyset$, we conclude immediately that Γ is not ACI-unifiable, since $s = t$ is not ACI-unifiable.
- Let c be a free constant that does not occur in the equation $s = t$. For every variable $x \in \mathcal{X}_s$, we introduce the Horn clause $(\bigwedge_{y \in \mathcal{X}_t} P_{y,c} \supset P_{x,c})$ and, for every variable $y \in \mathcal{X}_t$, we introduce the Horn clause $(\bigwedge_{x \in \mathcal{X}_s} P_{x,c} \supset P_{y,c})$. If $\mathcal{X}_t = \emptyset$, then the first clause is the unit clause $P_{x,c}$. Similarly, if $\mathcal{X}_s = \emptyset$, then the second clause is the unit clause $P_{y,c}$.
- Finally, for every variable x occurring in some equation in Γ, we introduce the Horn clause $(\bigvee_{c \in C} \neg P_{x,c})$ to ensure that the value of the solution for x contains at least one free constant.

Let $\Theta(\Gamma) = \bigcup_{(s=t) \in \star} \Theta(s,t)$. It is now quite straightforward to verify that the ACI($\omega\,;m$)-unification problem Γ has a solution if and only if the set $\Theta(\Gamma)$ of Horn clauses is satisfiable. Note that $\Theta(\Gamma)$ has size polynomial in m and the size of Γ; in fact, the size of $\Theta(\Gamma)$ is $\mathrm{O}(mkr^2)$, where k is the number of equations in Γ and r is the maximum size of an equation in Γ. Thus, the decision problem for elementary ACI-unification is solvable in polynomial time.

Once an algorithmic problem has been shown to be solvable in polynomial time (and, hence, tractable from the point of view of sequential computation), it is natural to ask whether it can also be solved "fast in parallel". To formalize this concept, researchers in computational complexity introduced and studied in depth the class NC of all decision problems that can be solved in polylog-

[1] A *Horn* formula is a conjunction of propositional clauses each of which has at most one positive literal.

arithmic time using polynomially many processors. It is easy to see that NC contains LOGSPACE and, in turn, is contained in P, where LOGSPACE is the class of problems solvable by a deterministic Turing machine using a logarithmic amount of space in its work tape, and P is the class of problems solvable by a deterministic Turing machine in polynomial time. Although it is widely believed that NC is properly contained in P, the question NC $\stackrel{?}{=}$ P remains one of the outstanding open problems of computational complexity. Starting with the work of Cook [Coo74], researchers identified numerous decision problems that are candidates to manifesting the separation between NC and P. These problems, which are known as P-*complete* problems, are the "hardest" members of P, in the sense that every problem in P can be reduced to them via some logarithmic-space reduction. Establishing that a certain problem is P-complete is viewed as providing strong evidence that this problem is not in NC and, hence, it is inherently sequential. Examples of P-complete problems of relevance to automated deduction include UNIT RESOLUTION [JL76] and SYNTACTIC UNIFICATION [DKM84] (a comprehensive treatment of the theory of P-completeness and a catalogue of P-complete problems can be found in the monograph [GHR95]). Our first result in this paper shows that ELEMENTARY ACI-UNIFICATION yields a new paradigm of a P-complete problem.

Theorem 1. ACI(ω;2)-*unification is* P-*complete. In words, the decision problem for elementary ACI-unification with two free constants is* P-*complete.*

Proof. Plaisted [Pla84] showed that PROPOSITIONAL HORN SATISFIABILITY is a P-complete problem; in fact, he showed that the problem remains P-complete even when restricted to inputs in which each clause has at most three literals. We now present a logarithmic-space reduction of this restricted problem to the decision problem for ACI(ω;2)-unification.

Let a and b be two free constants. Given a set S of propositional Horn clauses each of which has at most three literals, we generate the following elementary ACI-unification problem $\Gamma(S)$ in the free constants a and b.

1. For every propositional variable X occurring in some clause in S, we introduce a variable $x \in \mathcal{X}$ and the equation $x * a = a * b$.
2. For every unit clause X in S, we introduce the equation $x = b$.
3. For every clause in S of the form $(\neg X \vee \neg Y \vee \neg Z)$, we introduce the equation $x * y * z = a * b$.
4. For every clause of the form $(\neg X \vee \neg Y \vee Z)$, we introduce the equation $x * y * z = x * y$.
5. For every clause of the form $(\neg X \vee Z)$, we introduce the equation $x * z = x$.

We now claim that the set S has a satisfying assignment if and only of the elementary ACI-unification problem $\Gamma(S)$ has a solution. If h is a truth assignment that satisfies every clause in S, then it is easy to see that the substitution

$$x\sigma = \begin{cases} b & \text{if } h(X) = \text{TRUE} \\ a * b & \text{if } h(X) = \text{FALSE} \end{cases}$$

is a solution of the elementary ACI-unification problem $\Gamma(S)$. Conversely, suppose that σ is a solution of $\Gamma(S)$. For every variable x occurring in $\Gamma(S)$, it must be the case that $x\sigma = b$ or $x\sigma = a * b$, since the equations in group (1) above imply that $x\sigma * a =_{\mathrm{ACI}} a * b$. It is now easy to verify that the truth assignment

$$h(X) = \begin{cases} \text{TRUE} & \text{if } x\sigma = b \\ \text{FALSE} & \text{if } x\sigma = a * b \end{cases}$$

satisfies every clause in S. For example, a clause of the form $(\neg X \vee \neg Y \vee Z)$ must be satisfied by h, since, otherwise, it would be the case that $x\sigma = b$, $y\sigma = b$ and $z\sigma = a * b$, which implies that σ is not a solution of the equation $x * y * z = x * y$ associated with this clause. □

The preceding reduction of PROPOSITIONAL HORN SATISFIABILITY to ELEMENTARY ACI-UNIFICATION made use of two free constants in the signature. It should be noted that the presence of at least two free constants is indispensable in obtaining this P-hardness result. Indeed, the decision problem for ACI(ω; 1)-unification is trivial, as the constant substitution $\lambda x.a$ is a solution of every elementary ACI-unification problem in which a is the only free constant.

Let AC be the equational theory of commutative semigroups. It is well known that at the level of the decision problem elementary AC-matching and elementary AC-unification have the same computational complexity, namely each of these two problems is NP-complete (see [BKN87,KN92]). Moreover, the same holds true for the equational theory ACU of commutative monoids (see [KN92,HK95b]). In contrast to the above, we observe next that the decision problem for elementary ACI-matching is in LOGSPACE and, thus, of lower computational complexity than the decision problem for elementary ACI-unification (unless P collapses to LOGSPACE, which is considered extremely unlikely). We also examine the computational complexity of elementary #ACI-matching, which is the following *counting* problem: given an elementary ACI-matching problem Γ, find the cardinality of the the minimal complete set of ACI-matchers of Γ. Valiant [Val79a,Val79b] developed a complexity theory of counting and enumeration problems by introducing the class #P and identifying problems that are complete for this class under parsimonious[2] reductions. In our earlier papers [HK95a,HK95b], we initiated a systematic investigation of the computational complexity of counting problems in equational matching using concepts and results from the theory of #P-completeness. In particular, we showed that the counting problem for elementary AC-matching, as well as the counting problem for elementary ACU-matching, is #P-complete. It should be noted that a #P-completeness result indicates that the counting problem at hand is highly intractable, in fact it suggests a higher level of intractability than an NP-completeness result for the corresponding decision problem (for more on this point see Johnson [Joh90] and Papadimitriou [Pap94]).

[2] A *parsimonious* reduction is a polynomial-time reduction between two counting problems that preserves the number of the solutions of each instance.

Theorem 2. *The decision problem for elementary* ACI*-matching is solvable in logarithmic space, but the counting problem for elementary* ACI*-matching is* #P*-complete, even when restricted to instances with just two free constants. Thus,*

- ACI$(\omega;\omega)$*-matching is in* LOGSPACE*;*
- #ACI$(\omega;2)$*-matching is* #P*-complete.*

Proof. Let Γ be an arbitrary elementary ACI-matching problem, that is, for every equation $s = t$ in Γ, the term t is ground. Let $S(\Gamma)$ be the set of Horn clauses that results from the reduction of elementary ACI-unification to PROPOSITIONAL HORN SATISFIABILITY, as described in the beginning of this section. An inspection of this reduction reveals that each clause in $S(\Gamma)$ is either a unit clause or a disjunction of negated propositional variables, since the set $\mathcal{X}_t$ of variables of t is empty. This special case of PROPOSITIONAL HORN SATISFIABILITY is solvable in logarithmic space, as one needs only to verify that, for every clause consisting of negated propositional variables, at least one of its propositional variables does not appear as a unit clause.

Next, we focus on elementary #ACI-matching. It is easy to see that this counting problem is in #P (this follows also from more general results in [HK95a]). For the lower bound, we have to show that #ACI$(\omega;2)$-matching is a #P-hard problem. Consider the following counting problem:

#POSITIVE 2SAT: Given a propositional CNF formula φ such that each clause is a disjunction of two propositional variables, find the number of satisfying truth assignments of φ.

Although the underlying decision problem POSITIVE 2SAT is trivial, Valiant showed in [Val79b] that the corresponding counting problem #POSITIVE 2SAT is #P-hard. This is an extreme instance of an interesting phenomenon, first observed by Valiant [Val79a], in which the counting version of an "easy" decision problem may be "hard". We now exhibit a parsimonious reduction of #POSITIVE 2SAT to #ACI$(\omega;2)$-matching. Assume that a and b are two free constants. Given a a positive 2SAT propositional formula φ, construct the following ACI$(\omega;2)$-matching problem $\Gamma(\varphi)$:

- for every propositional variable X occurring in φ, introduce a propositional variable x and the equation $x * a = a * b$;
- for every clause $(X \vee Y)$ of φ, introduce the equation $x * y = a * b$.

It is easy to check that there is a one-to-one correspondence between satisfying truth assignments of φ and solutions of $\Gamma(\varphi)$, since for every solution σ of the equation $x * a = a * b$ it must be the case that either $x\sigma = b$ or $x\sigma = a * b$. □

Two remarks are in order now.

1. The preceding Theorem 2 shows that elementary ACI-matching is an algorithmic problem whose decision version is "easy", but its counting version is "hard". Compare this with elementary AC-matching and elementary ACU-matching for which the decision problem is NP-complete [BKN87] and the counting problem is #P-complete [HK95a].

2. In proving that #ACI$(\omega;2)$-matching is a #P-hard problem, the number of equations used varied with the input. In the full paper, we show that

if the number of equations is fixed, then the counting problem for elementary ACI-matching is solvable in polynomial time using a dynamic programming algorithm. More formally, we can show that, for every two positive integers k and m, the counting problem $\#\mathrm{ACI}(k;m)$ is in the class FP of functions computable in polynomial time. It should also be pointed out that Baader and Büttner [BB88] obtained explicit expressions for the number of most general complete unifiers of elementary ACI-unification problems consisting of a *single* equation.

The proofs of Theorems 1 and 2 can be adapted easily to obtain similar results for elementary ACIU-unification and ACIU-matching. In fact, since a unit element of $*$ is available, a single free constant suffices to obtain the hardness results in each case.

Theorem 3. *The following statements are true for the equational theory* ACIU *of commutative idempotent monoids.*

- $\mathrm{ACIU}(\omega;1)$*-unification is* P*-complete.*
- $\mathrm{ACIU}(\omega;\omega)$*-matching is in* LOGSPACE*.*
- $\#\mathrm{ACIU}(\omega;1)$*-matching is* #P*-complete.*

4 Ground elementary ACI-disunification

Narendran [Nar96] studied the computational complexity of the equational theory ACIUZ, which is the extension of ACIU with the equational axiom Z: $(\forall x)(x * 0 = 0)$ asserting that $*$ has a *zero* element 0. In particular, he showed that ground elementary ACIUZ-disunification is NP-hard by exhibiting a reduction from 3SAT. An inspection of that proof shows that actually a stronger result is established, namely that ground elementary ACIU-disunification with a single free constant a is NP-complete. Moreover, Narendran [Nar96] commented in a footnote that *"A similar reduction will work also for the* ACI *case. There we have to use two constants, say a and b, since we do not have the unit* 1*."* The following result confirms that ground elementary ACI-disunification with two free constants is NP-complete, although the reduction we use is not from 3SAT, but from the problem NOT-ALL-EQUAL 3SAT, which asks: given a 3CNF formula φ, is there a truth assignment such that each clause has at least one true literal and at least one false literal? (see [GJ79, page 259]).

Theorem 4. *For every positive integer $m \geq 2$, ground* $\mathrm{ACI}(0,\omega;m)$*-disunification is* NP*-complete. In words, the decision problem for ground elementary* ACI*-disunification with no equations and at least two free constants is* NP*-complete.*

Proof. Membership in NP is obvious; in fact, even ground $\mathrm{ACI}(\omega,\omega;\omega)$-disunification is in NP, as it suffices to guess a ground substitution σ such that $|x\sigma| \leq m$ for every variable x occurring in the given instance, where m is the number of free constants occurring in the instance.

Let C be a set of m free constants, $m \geq 2$, and let a and b two free constants in C. Given a 3CNF formula φ to be tested for NOT-ALL-EQUAL 3SAT, we generate the following $\mathrm{ACI}(0,\omega;m)$-disunification problem $\Delta(\varphi)$.

- For every propositional variable X_i occurring in φ, we introduce two variables x_i and y_i, and the following disequations. For every ground term t different than $a * b$ and such that each free constant occurs at most once in t, we introduce the disequation $x_i * y_i \neq t$. For every ground term t' different than a and b, and such that each free constant occurs at most once in t', we introduce the disequations $x_i \neq t'$ and $y_i \neq t'$.
- For every clause of φ of the form $(X_i \vee \neg X_j \vee X_r)$ and for every term t different than $a * b$ and such that each free constant occurs at most once in t, we introduce the disequation $x_i * y_j * x_r \neq t$. In a similar manner, we introduce disequations for clauses of the other possible forms. For example, if a clause is of the form $(\neg X_i \vee \neg X_j \vee X_r)$, then we introduce the disequations $y_i * y_j * x_r \neq t$.

The first group of disequations enforces the following property on every ground substitution σ that is a solution of $\Delta(\varphi)$ in the commutative idempotent semigroup freely generated by the constants in C: for every propositional variable X_i occurring in φ, either $x_i\sigma = a \wedge y_i\sigma = b$ or $x_i\sigma = b \wedge y_i\sigma = a$. In turn, this property implies that for every disequation in the second group at least one variable takes value a and at least one variable takes value b. Thus, there is a truth assignment such that every clause of φ has at least one true and at least one false literal if and only if $\Delta(\varphi)$ has a solution in the commutative idempotent semigroup freely generated by the constants in C. □

In the above NP-hardness proof, both free constants a and b were used; moreover, the number of disequations that were introduced varied with the size of input. It turns out that if either of these conditions is relaxed, then ground elementary ACI-disunification becomes tractable. First, note that ground elementary ACI-disunification with a single free constant a is trivial, since in this case the ground term algebra $\mathcal{T}(\{*, a\})/{=_{\mathrm{ACI}}}$ is the singleton $\{a\}$. Next, we will show that ground elementary ACI-disunification with a fixed number of disequations is solvable in polynomial time, even when an arbitrary number of equations is present and an arbitrary number of free constants is available. For this, we need to establish an auxiliary result first.

Lemma 5. *For every positive integer k, there is a polynomial-time algorithm for solving the following decision problem: given a propositional Horn formula θ and k propositional formulas $\psi_1, \ldots, \psi_k$ each in disjunctive normal form, is the formula $\theta \wedge \psi_1 \wedge \cdots \wedge \psi_k$ satisfiable?*

Proof. Let $\theta, \psi_1, \ldots, \psi_k$ be an instance of this problem of size s. Without loss of generality, we may assume that there is a positive integer $n \leq s$ such that each ψ_i consists of exactly n disjuncts. Thus, for every $i \leq k$, $\psi_i \equiv \chi_{i1} \vee \cdots \vee \chi_{in}$, where each χ_{ij} is a conjunction of at most s literals. By distributing conjunctions over disjunctions, we have that

$$\psi_1 \wedge \cdots \wedge \psi_k \quad \equiv \bigvee_{(j_1,\ldots,j_k)\in\{1,\ldots,n\}^k} \chi_{1j_1} \wedge \cdots \wedge \chi_{kj_k}.$$

It follows that the formula $\theta \wedge \psi_1 \wedge \cdots \wedge \psi_k$ is satisfiable if and only if there is a k-tuple $(j_1, \ldots, j_k) \in \{1, \ldots, n\}^k$ such that the formula $\theta \wedge \chi_{1j_1} \wedge \cdots \wedge \chi_{kj_k}$ is satisfiable. Since each of the n^k formulas $\theta \wedge \chi_{1j_1} \wedge \cdots \wedge \chi_{kj_k}$ is a propositional Horn formula, we can apply the polynomial-time algorithm for propositional Horn satisfiability n^k times and, thus, determine in polynomial time whether the formula $\theta \wedge \psi_1 \wedge \cdots \wedge \psi_k$ is satisfiable. □

Theorem 6. *For every positive integer k, ground* ACI$(\omega, k; \omega)$*-disunification is in P. In words, ground elementary disunification with k disequations and an arbitrary number of free constants is solvable in polynomial time.*

Proof. Fix a positive integer k. Let Δ be a given ACI$(\omega, k; \omega)$-disunification problem. Thus, $\Delta = \Gamma \cup \{p_1 \neq q_1, \ldots, p_k \neq q_k\}$, where Γ is an arbitrary elementary ACI-unification problem. Recall Kapur and Narendran's [KN92] reduction of elementary ACI-unification to Propositional Horn Satisfiability, which was described in Section 3. In particular, recall that for every equation $l = r$ this reduction generates a set $\Theta(l, r)$ of Horn clauses. Let θ be the Horn formula $\bigwedge_{(s=t)\in\star} \Theta(s, t)$. For every disequation $p_i \neq q_i$, $1 \leq i \leq k$, consider the set $\Theta(p_i, q_i)$ of Horn clauses generated when the reduction is applied to the equation $p_i = q_i$. For every $i \leq k$, let ϕ_i be the conjunction of all Horn clauses in $\Theta(p_i, q_i)$, and let ψ_i be the formula in disjunctive normal form that is obtained from $\neg\phi_i$ using de Morgan's laws. It is now easy to verify that the ACI$(\omega, k; \omega)$-disunification problem Δ has a ground solution if and only if the formula $\theta \wedge \psi_1 \wedge \cdots \wedge \psi_k$ is satisfiable. Note that the size of this formula is polynomial in the size of Δ. We can now apply Lemma 5 and obtain the desired polynomial-time algorithm. □

5 Restricted ACI-matching

Set-matching is an important special case of general ACI-matching; its computational complexity has been investigated by Kapur and Narendran [KN86], who showed that this problem is NP-complete. Set-matching, as well as certain variants of it, arise naturally in deductive database systems and in logic-based languages that support complex objects. Shmueli et al. [STZ92] studied set matching problems in the context of LDL, a Horn-clause programming language for deductive database systems. The semantics of LDL require that set-terms consisting of variables and constants be matched in such a way that variables are instantiated only by individual constants. In essence, the problems examined by Shmueli et al. [STZ92] can be formalized as follows: given an elementary ACI-matching problem Γ, is there a a solution ρ of Γ such that, for every variable x occuring in Γ, the value $x\rho$ is equal to one of the constant symbols occurring in Γ? In what follows, we call such problems *restricted* ACI-*matching* problems. Let *restricted* ACI(k, m)-*matching* be the class of restricted ACI-matching problems with k equations and m free constants. The classes *restricted* ACI(ω, m)-*matching*, *restricted* ACI(k, ω)-*matching*, and *restricted* ACI(ω, ω)-*matching* are defined in an analogous way. Shmueli et al. [STZ92] gave an exponential-time algorithm for restricted ACI(ω, ω)-matching. After this, Arni et al. [AGS96] considered *bounded set-term matching*, which, in our terminology, is the same as

restricted ACI$(1,\omega)$-matching, that is, restricted ACI-matching with a single equation, but no a priori bound on the number of free constants. An instance of this problem can be written as $XD = C$, where X is a set of variables, C and D are sets of free constants, each set stands for the "product" of its members, and the concatenation XD denotes the union $X \cup D$. Arni et al. [AGS96] pointed out that $XC = D$ has a restricted ACI-matcher if and only if $C \subseteq D$ and $|X| \geq |D \setminus C|$. This gives a simple polynomial-time test for the decision problem for restricted ACI$(1,\omega)$-matching. Moreover, Arni et al. [AGS96] gave an explicit formula for the number of restricted ACI-matchers of a single equation, from which it follows that restricted #ACI$(1,\omega)$-matching is in the class FP of functions computable in polynomial time.

In the sequel, we analyze the computational complexity of restricted ACI-matching by considering once again the interplay between the number of equations and the number of constants. We first observe that a slight modification of the proof of Theorem 4 shows that restricted ACI-matching with no a priori bound on the number of equations is intractable, even if only two free constants are available. This should be contrasted with the low complexity of the decision problem for ACI-matching (cf. Theorem 2).

Theorem 7. *Restricted* ACI$(\omega;2)$*-matching is a* NP*-complete problem and restricted* #ACI$(\omega;2)$*-matching is a* #P*-complete problem.*

Proof. Since the upper bounds are obvious, we focus on establishing NP-hardness and #P-hardness. For this, we give a parsimonious reduction of NOT-ALL-EQUAL 3SAT to restricted ACI$(\omega,2)$-matching. For every propositional variable X_i in a given 3CNF formula ϕ, we introduce two variables x_i and y_i, and the equation $x_i * y_i = a * b$, where a and b are free constants. For each clause of ϕ of the form $(X_i \vee \neg X_j \vee X_r)$, we introduce the equation $x_i * y_j * x_r = a * b$. We introduce equations for clauses of the other possible forms in a similar manner. Then ϕ has an assignment that falsifies at least one literal in every clause if and only if the associated system of equations has a restricted ACI-matcher. □

Next, we consider restricted ACI-matching with a fixed number of equations. The main result of this section is that for any two positive integers k and m, the decision problem for restricted ACI(k,m)-matching is solvable in polynomial time. Note that restricted ACI(k,m)-matching can be reduced easily to ground ACI-disunification, but at the expense of introducing an unbounded number of disequations (and so Theorem 6 can not be used). Indeed, for every variable x of a given restricted ACI(k,m)-matching problem with constants $c_1,\ldots,c_m$, we introduce $2^m - m - 1$ new disequations of the form $x \neq c_{i_1} * \cdots * c_{i_r}$, where $2 \leq r \leq m$, and $c_{i_j} \neq c_{i_l}$ for all $j \neq l$; these disequations capture the restriction $x \in \{c_1,\ldots,c_m\}$. It follows that every restricted ACI(k,m)-matching problem with n variables can be reduced to a ground ACI$(k, n(2^m - m - 1); m)$-disunification problem. This approach, however, reduces restricted ACI(k,m)-matching to ground ACI$(k,\omega;m)$-disunification, which is NP-complete (recall Theorem 4). Thus, a different method is needed to establish the tractability of restricted ACI(k,m)-matching.

Fix two positive integers k and m. Let Γ be a restricted ACI(k,m)-matching problem with free constants $c_1,\ldots,c_m$ and variables $x_1,\ldots,x_n$. Each equation of Γ can be written as $XC = D$, where X is a subset of $\{x_1,\ldots,x_n\}$, and C and D are subsets of $\{c_1,\ldots,c_m\}$. For every $i \leq n$, let $C(x_i)$ be the set of all constants that occur in the right-hand side of every equation of Γ in which the variable x_i occurs (that is, $C(x_i)$ is the intersection of the sets D such that x_i occurs in some equation $XC = D$ of Γ). Our polynomial-time algorithm for restricted ACI(k,m)-matching will examine the cardinalities of the sets $C(x_i)$, $1 \leq i \leq n$. Note that if there is a variable x_i such that $C(x_i) = \emptyset$, then Γ has no restricted ACI-matchers. On the other hand, if $C(x_i)$ is a singleton for some variable x_i, then we can eliminate x_i from Γ by replacing it with the unique member of $C(x_i)$. Finally, it may be the case that $|C(x_i)| \geq 2$, for every variable x_i. The next lemma shows that in this case restricted ACI$(k.m)$-matching can be reduced to restricted ACI$(k,m-1)$-matching. In what follows, we will use the notation $XD = c_jC$ to indicate that the right-hand side of this equation is the union $\{c_j\} \cup C$, where $c_j \notin C$.

Lemma 8. *Assume that Γ is a restricted* ACI(k,m)*-matching problem with free constants $c_1,\ldots,c_m$ and variables $x_1,\ldots,x_n$ such that each equation of Γ has a restricted* ACI*-matcher and $|C(x_i)| \geq 2$ for every $i \leq n$. Fix a free constant c_j and assume that $X_1C_1 = c_jD_1,\ldots,X_rC_r = c_jD_r$ is a list of all equations of Γ in which c_j occurs in the right-hand side, but not in the left-hand side of the equation. Then Γ has a restricted* ACI*-matcher if and only if there is a sequence $z_1,\ldots,z_r$ of (not necessarily distinct) variables with the following properties:*

1. *For every $i \leq r$, the variable z_i occurs in the equation $X_iD_i = c_jD_i$; moreover z_i can occur only in equations in which c_j occurs.*
2. *The system Γ^* obtained from Γ by eliminating all occurrences of the free constant c_j and of the variables $z_1,\ldots,z_r$ has a restricted* ACI*-matcher.*

Proof. If: Take such a sequence $z_1,\ldots,z_r$ of variables and a restricted ACI-matcher ρ^* of Γ^*. Extend $\rho*$ to a substitution ρ on the variables of Γ by assigning c_j as the value of $z_i\rho$, $1 \leq i \leq r$. Then ρ is a restricted ACI-matcher of Γ.

Only If: Suppose that ρ is a restricted ACI-matcher of Γ. Then, for every $i \leq r$, there exists a variable z_i occurring in the equation $X_iC_i = c_jD_i$ and such that $z_i\rho = c_j$. It is clear that z_i can not occur in any equation of Γ in which c_j does not occur. Consider now the system Γ^* obtained from Γ by eliminating all occurrences of the free constant c_j and the variables $z_1,\ldots,z_r$. Define a substitution ρ^* on the variables of Γ^* as follows. If $z\rho \neq c_j$, then $z\rho^* = z\rho$. If $z\rho = c_j$, then put $z\rho^* = c_l$, where c_l is a free constant that is different than c_j and occurs in the right-hand side of every equation of Γ in which z occurs. Note that such a free constant exists, because $|C(z)| \geq 2$. It is now easy to verify that ρ^* is a restricted ACI-matcher of Γ^*. □

Theorem 9. *Let k and m be two positive integers. The decision problem for restricted* ACI(k,m)*-matching is solvable in polynomial time.*

Proof. Let `Propagate-and-Split`(Δ) be the following procedure, where Δ is a system with at most k equations and at most m free constants $c_1, \ldots, c_m$.

1. If only one free constant occurs in Δ, then stop and report that Δ has a restricted ACI-matcher.

2. If one of the equations of Δ does not have a restricted ACI-matcher or if $C(x_i) = \emptyset$ for one of the variables x_i of Δ, then stop and report that Δ does not have a restricted ACI-matcher.

3. Replace every variable x_i such that $|C(x_i)| = 1$ by the unique member of the singleton $C(x_i)$. Let Δ' be the resulting system.

4. Let j be the smallest integer such that the free constant c_j occurs in Δ'. Consider all equations $X_1 C_1 = c_j D_1, \ldots, X_r C_r = c_j D_r$ of Δ' in which c_j occurs in the right-hand side, but not in the left-hand side of the equation. For every sequence $z_1, \ldots, z_l$ of variables such that each z_i occurs in the equation $X_i C_i = c_j D_i$, but does not occur in any equation in which c_j does not occur, generate the system $\Delta'(c_j, z_1, \ldots, z_l)$ obtained from Δ' by eliminating all occurrences of the free constant c_j and the variables $z_1, \ldots, z_l$.

Note that each of the first three steps takes time $O(n)$, where n is the number of variables of Δ. Note also that, after the last step has been completed, at most n^k systems of the form $\Delta'(c_j, z_1, \ldots, z_l)$ are generated; moreover, the number of constants of each such system is one less than the number of constants of Δ.

The above procedure `Propagate-and-Split` gives rise to a polynomial-time algorithm for restricted ACI(k, m)-matching. Let Γ be a restricted ACI$(k.m)$-matching problem. Apply first the procedure to Γ, then to each system generated in the last step, and continue this way until either a restricted ACI-matcher has been found or there are no systems to which the procedure can be applied. Since each application of the last step eliminates a free constant, at most n^{km} systems have to be considered, where n is the number of variables of Γ. Thus, the running time of the algorithm is $O(n^{km+1})$. The correctness of the algorithm follows from the preceding Lemma 8 and the fact that the systems generated by the third step of the procedure satisfy the hypotheses of Lemma 8. □

Note that the running time of the above algorithm for restricted ACI(k, m)-matching depends exponentially on the number m of free constants. It remains an open problem to determine whether restricted ACI(k, ω)-matching can be solved in polynomial time.

6 Concluding Remarks

Combined with the work of Kapur and Narendran [KN92] and Narendran [Nar96], the results presented here provide a fairly complete picture of the computational complexity of elementary ACI-unification and elementary ACI-disunification. Moreover, they shed light on the computational complexity of restricted ACI-matching, a class of constrained set-term matching problems that arose in the development of deductive database systems [STZ92,AGS96].

We conclude by pointing out that in the full version of the paper we venture beyond ACI-unification and ACI-disunification by considering the full first-

order theory of free commutative idempotent semigroups with m generators, $m \geq 2$. We analyze the computational complexity of this theory and show it to be PSPACE-complete; moreover, we study the complexity of fragments of this theory obtained by restricting the quantifier alternation or the quantifier-free part of first-order sentences.

References

[AGS96] N. Arni, S. Greco, and D. Saccà. Matching of bounded set terms in the logic language LDL^{++}. *J. Logic Prog.*, 27(1):73–87, 1996.

[BB88] F. Baader and W. Büttner. Unification in commutative idempotent monoids. *Theoretical Comp. Sci.*, 56(3):345–353, 1988.

[BKN87] D. Benanav, D. Kapur, and P. Narendran. Complexity of matching problems. *J. Symb. Comp.*, 3:203–216, 1987.

[Coo74] S.A. Cook. An observation on time-storage trade off. *J. of Comp. and System Sci.*, 9(3):308–316, 1974.

[DG84] W.F. Dowling and J.H. Gallier. Linear-time algorithms for testing the satisfiability of propositional Horn formulae. *J. Logic Prog.*, 1(3):267–284, 1984.

[DKM84] C. Dwork, P.C. Kanellakis, and J.C. Mitchell. On the sequential nature of unification. *J. Logic Prog.*, 1:35–50, 1984.

[GHR95] R. Greenlaw, H.J. Hoover, and W.L. Ruzzo. *Limits to parallel computation: P-completeness theory*. Oxford University Press, New York, 1995.

[GJ79] M.R. Garey and D.S. Johnson. *Computers and intractability: A guide to the theory of NP-completeness*. W.H. Freeman and Co, 1979.

[HK95a] M. Hermann and P.G. Kolaitis. The complexity of counting problems in equational matching. *J. Symb. Comp.*, 20(3):343–362, 1995.

[HK95b] M. Hermann and P.G. Kolaitis. Computational complexity of simultaneous elementary matching problems. In J. Wiedermann and P. Hájek, eds, *Proc. 20th MFCS, Prague (Czech Republic)*, LNCS 969, pp 359–370. Springer, 1995.

[JL76] N.D. Jones and W.T. Laaser. Complete problems for deterministic polynomial time. *Theoretical Comp. Sci.*, 3(1):105–117, 1976.

[Joh90] D.S. Johnson. A catalog of complexity classes. In J. van Leeuwen, ed, *Handbook of Theoretical Computer Science, Volume A*, chapter 2, pp 67–161. North-Holland, Amsterdam, 1990.

[KN86] D. Kapur and P. Narendran. NP-completeness of the set unification and matching problems. In J.H. Siekmann, ed, *Proc. 8th CADE, Oxford (England)*, LNCS 230, pp 489–495. Springer, 1986.

[KN92] D. Kapur and P. Narendran. Complexity of unification problems with associative-commutative operators. *J. of Autom. Reasoning*, 9:261–288, 1992.

[Nar96] P. Narendran. Unification modulo ACI+1+0. *Fundamenta Informaticae*, 25(1):49–57, 1996.

[Pap94] C.H. Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.

[Pla84] D.A. Plaisted. Complete problems in the first-order predicate calculus. *J. of Comp. and System Sci.*, 29(1):8–35, 1984.

[STZ92] O. Shmueli, S. Tsur, and C. Zaniolo. Compilation of set terms in the logic data language (LDL). *J. Logic Prog.*, 12(1 & 2):89–119, 1992.

[Val79a] L.G. Valiant. The complexity of computing the permanent. *Theoretical Comp. Sci.*, 8:189–201, 1979.

[Val79b] L.G. Valiant. The complexity of enumeration and reliability problems. *SIAM J. on Comp.*, 8(3):410–421, 1979.