Mathematical Structures in Computer Science (2020), **30**, pp. 597–626
doi:10.1017/S0960129519000185

CAMBRIDGE UNIVERSITY PRESS

PAPER

Extensions of unification modulo ACUI

Franz Baader[1], Pavlos Marantidis[1,*,†], Antoine Mottet[2,‡] and Alexander Okhotin[3]

[1]Theoretical Computer Science, Technische Universität Dresden, Germany, [2]Department of Algebra, Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic and [3]St. Petersburg State University, St. Petersburg, Russia
*Corresponding author. Email: pavlos.marantidis@tu-dresden.de

(Received 20 February 2019; revised 18 September 2019; accepted 28 September 2019; first published online 11 November 2019)

Abstract

The theory ACUI of an associative, commutative, and idempotent binary function symbol + with unit **0** was one of the first equational theories for which the complexity of testing solvability of unification problems was investigated in detail. In this paper, we investigate two extensions of ACUI. On one hand, we consider approximate ACUI-unification, where we use appropriate measures to express how close a substitution is to being a unifier. On the other hand, we extend ACUI-unification to ACUIG-unification, that is, unification in equational theories that are obtained from ACUI by adding a finite set G of ground identities. Finally, we combine the two extensions, that is, consider approximate ACUIG-unification. For all cases we are able to determine the exact worst-case complexity of the unification problem.

Keywords: Unification theory; ACUI; approximate unification; ground identities

1. Introduction

An important topic in unification theory (Siekmann, 1989; Baader and Siekmann, 1994; Baader and Snyder, 2001) is investigating the complexity of deciding solvability of unification problems Γ w.r.t. an equational theory E, that is, for a fixed equational theory E one considers as input a set of equations between terms of the form $\Gamma = \{s_1 =^? t_1, \ldots, s_k =^? t_k\}$, and asks whether there exists a substitution σ such that $\sigma(s_i) =_E \sigma(t_i)$ holds for $i = 1, \ldots, k$. The complexity is then measured in the s of the unification problem Γ. The theory ACUI of an associative, commutative, and idempotent binary function symbol + with unit **0** was one of the first equational theories for which the complexity of testing solvability of unification problems was investigated in detail (Kapur and Narendran, 1992). Interestingly, it turned out that this complexity depends on which symbols are allowed to occur in Γ. In elementary E-unification, the terms in Γ may only contain variables and the constant and function symbols occurring in E, that is, for $E = \mathsf{ACUI}$ these terms are built using variables and +, **0**. In E-unification with constants, additional free constants (i.e., constants not occurring in E) may be used, whereas in general E-unification both free constants and free function symbols may occur in Γ. Kapur and Narendran (1992) have shown that elementary ACUI-unification and ACUI-unification with constants are polynomial, whereas general ACUI-unification is NP-complete.

Our renewed interest in ACUI-unification stems from the fact that the theory ACUI is a common subtheory of the equational theories corresponding to the Description Logics $\mathcal{FL}_0$ (Baader, 1996)

†Supported by DFG Graduiertenkolleg 1763 (QuantLA).
‡Partially supported by DFG Graduiertenkolleg 1763 (QuantLA).

and $\mathcal{EL}$ (Baader et al., 1999): for $\mathcal{FL}_0$, ACUI is extended with unary function symbols that behave like homomorphisms and for $\mathcal{EL}$ the additional unary function symbols behave like monotone operators. In both cases, the binary symbol $+$ corresponds to concept conjunction, and $\mathbf{0}$ to the top concept, which, in turn, are the description logic equivalents of "logical and" and "logical true". Unification with constants in (the equational theory corresponding to) $\mathcal{FL}_0$ is known to be ExpTime-complete (Baader and Narendran, 2001) and NP-complete in (the equational theory corresponding to) $\mathcal{EL}$ (Baader and Morawska, 2009). We believe that ACUI is thus a good testing ground when trying to extend these results to more general settings. Here, we consider extensions of ACUI-unification in two orthogonal directions, which we then bring together: generalizing unification to approximate unification and adding a finite ground theory to ACUI.

In approximate E-unification, one does not require that the left- and right-hand sides of the equations become equal modulo E, but only almost equal. The formal meaning of when a substitution approximately (i.e., "almost") solves a unification problem is formalized using distance measures between terms that are tailored towards the equational theory E in question. Approximate unification with constants in $\mathcal{FL}_0$ was investigated in Baader et al. (2016) w.r.t. two such measures. In contrast to approaches that try to speed up reasoning by employing approximate inference techniques (Pan et al., 2016), this work (Baader et al., 2016) uses approximation as a way to extend the range of admissible solutions to a unification problem. In the present paper, we consider approximate unification with constants in ACUI w.r.t. three different measures: the first considers the number of equations that are violated, the second the number of constants violating at least one equation, and the third counts the overall number of violations (see Section 3 for exact definitions of these measures). It turns out that in the first and the third cases, the complexity increases from P to NP-complete when going from exact ACUI-unification with constants to the approximate case, whereas it stays in P for the second case.

The results for unification in the Description Logics $\mathcal{FL}_0$ and $\mathcal{EL}$ respectively shown in Baader and Narendran (2001) and Baader and Morawska (2009) are restricted to the case where equivalence of concept descriptions (corresponding to equality modulo the respective equational theories mentioned above) is considered without a background knowledge base. It is not known how to extend these decidability and complexity results to unification in the presence of so-called general TBoxes, though for $\mathcal{EL}$ there are some positive results for a restricted form of TBoxes (Baader et al., 2012). Since, from an equational theory point of view, general TBoxes correspond to finite sets of ground identities, we are interested in equational theories for which decidability of unification is stable under adding finite sets of ground identities. We will show in Section 4 that ACUI is such a theory. For the word problem, Marché proved in Marché (1996) that ACUI remains decidable if it is extended with a finite set of ground identities, but no complexity bounds are given. This result actually holds for a signature possibly containing several ACUI symbols and free function symbols. In Section 4 we consider a restricted setting where the ground theory G is built using only one ACUI symbol and free constants. In this case we can show that both the word problem and unification with constants remain solvable in P. Actually, the result for unification holds for unification problems with so-called constant restrictions, which allows us to employ the combination results from (Baader and Schulz, 1996) to show that general unification in ACUI $\cup$ G is NP-complete for any finite set of ground identities G satisfying the restrictions mentioned above. The complexity upper bounds shown in Section 4 (in P for unification with constant restrictions and in NP for general unification) are actually somewhat stronger: these upper bounds hold even if we view G to be part of the input, that is, measure the complexity in the size of Γ and G. NP-hardness already holds for any fixed finite set of ground identities G.

In the last part of this paper, we are combining the two extensions, that is, we consider approximate unification modulo ACUIG, that is, ACUI extended with a finite set of ground identities G. For the cases where one considers the number of violated equations or the overall number of violations, the NP-completeness results transfer from ACUI to ACUIG. For the remaining case where one considers the number of constants violating at least one equation, things become more

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

interesting: we will show that there is a finite set of ground identities G such that approximate ACUIG-unification becomes NP-complete, but we will also exhibit a class of ground identities G for which this problem remains in P.

In the following, we assume that the reader is familiar with the basic notions regarding equational theories and unification modulo equational theories, as they can be found, for example, in Baader and Nipkow (1998), Baader and Snyder (2001).

2. Unification Modulo ACUI

In this section, we introduce the equational theory ACUI, characterize the word problem in ACUI using sets of constants, and recall the polynomial-time decision procedure for ACUI-unification by Kapur and Narendran (1992), which is based on a translation of ACUI-unification problems into propositional Horn formulae. This translation will then be extended in the next section to deal with approximate unification.

Let $\Sigma = \{+, \mathbf{0}\}$ be the signature consisting of a binary function symbol $+$ and a constant symbol $\mathbf{0}$. We denote the equational theory that states that $+$ is an associative, commutative, and idempotent symbol with unit $\mathbf{0}$ by ACUI:

$$\mathsf{ACUI} := \{(x + y) + z = x + (y + z), x + y = y + x, x + \mathbf{0} = x, x + x = x\}.$$

Furthermore, let F be a countably infinite set of constants and V a countably infinite set of variables, where we assume that F, V, and Σ are pairwise disjoint. We call the elements of F *free constants* since they are not constrained by the identities of ACUI. We denote the set of terms built from Σ, F, and V as $T_\Sigma(F, V)$, and the subset of ground terms, that is, terms in $T_\Sigma(F, V)$ that do not contain variables, as $T_\Sigma(F)$. For example, if $a, b \in F$ and $x, y \in V$, then $x + y$ and $a + x$ belong to $T_\Sigma(F, V)$, and $a + b + b$ and $b + a + a$ are elements of $T_\Sigma(F)$. The latter two terms are actually equivalent modulo ACUI. More generally, two ground terms are equivalent modulo ACUI iff they contain the same free constants. To be more precise, given a ground term $t \in T_\Sigma(F)$, we denote the set of free constants occurring in t with $S(t)$. For example, $S(a + b + b) = \{a, b\} = S(b + a + a)$, and $S(a + \mathbf{0}) = \{a\} = S(a + a)$. The following result is well known.

Lemma 2.1. *Let $s, t \in T_\Sigma(F)$. Then $s =_{ACUI} t$ iff $S(s) = S(t)$.*

Before we can define ACUI-unification, we need to introduce the notion of a substitution. A *substitution* is a mapping $\sigma : V \to T_\Sigma(F, V)$ that is the identity for all but finitely many variables. It can be homomorphically extended to a mapping from $T_\Sigma(F, V)$ to $T_\Sigma(F, V)$ in the obvious way, that is, $\sigma(c) = c$ for every $c \in F \cup \{\mathbf{0}\}$, and $\sigma(t_1 + t_2) = \sigma(t_1) + \sigma(t_2)$ for every $t_1, t_2 \in T_\Sigma(F, V)$.

Definition 2.2. *ACUI-unification problem with constants.*

Input: *A finite system $\Gamma = \{s_1 =^? t_1, \ldots, s_k =^? t_k\}$ of equations between terms of $T_\Sigma(F, V)$.*

Question: *Is there a substitution σ such that $\sigma(s_i) =_{ACUI} \sigma(t_i)$ for every $i = 1, \ldots, k$?*

Such a substitution is called an ACUI-unifier of Γ.

The ACUI-unification problems introduced in the above definition are called unification problems *with constants* since they may contain additional free constants (i.e., the elements of F, which do not belong to Σ), but no additional free function symbols of arity > 0. Now, let Γ be such an ACUI-unification problem with constants and C and X be the finite sets of constants and variables respectively occurring in Γ. In order to check whether Γ has an ACUI-unifier or not, it is sufficient to consider substitutions that are the identity on $V \setminus X$ and replace every $x \in X$ by a term in $T_\Sigma(C)$, that is, a ground term containing (in addition to $\mathbf{0}$) only constants from C. In fact, any

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

ACUI-unifier of Γ can be turned into one satisfying these properties by making it the identity on $V \setminus X$ and replacing variables and constants in $F \setminus C$ occurring in $\sigma(x)$ for $x \in X$ with $\mathbf{0}$. If we apply such a substitution σ to the terms occurring in Γ, then we obtain terms in $T_\Sigma(C)$. According to Lemma 2.1, σ is an ACUI-unifier of Γ iff $S(\sigma(s)) = S(\sigma(t))$ for every equation $s \stackrel{?}{=} t \in \Gamma$, that is, every constant $c \in C$ occurring on the left-hand side $\sigma(s)$ also occurs on the right-hand side $\sigma(t)$ and vice versa.

Example 2.3. Consider the following ACUI-unification problem with constants:

$$\begin{aligned}\Gamma = \{&x_1 + x_2 \stackrel{?}{=} a + b + c,\\ &a + x_2 \stackrel{?}{=} b + x_3,\\ &x_1 + x_3 \stackrel{?}{=} a + c\}.\end{aligned}$$

We have $C = \{a, b, c\}$ and $X = \{x_1, x_2, x_3\}$. If we define $\sigma(x_1) := a + c$ and $\sigma(x_2) := b$, then σ solves the first equation, since then all the constants in C occur on both sides. In order to solve the second equation as well, we then must define $\sigma(x_3)$ such that a occurs in it, and c does not occur in it. This is satisfied by setting $\sigma(x_3) := a$, which then also satisfies the third equation. Note that setting $\sigma(x_3) := a + b$ would have satisfied the second equation, but not the third.

We will now recall Kapur and Narendran's (1992) reduction of solvability of ACUI-unification problems to satisfiability of propositional Horn formulae. Each equation in the unification problem Γ is translated into several Horn clauses, and the overall Horn formula is the conjunction of all clauses for all equations. In this reduction, to which from now on we will refer to as the KN-reduction, we use propositional variables $p(a, x)$ for every $a \in C$ and $x \in X$. The intuitive semantics of these variables is that $p(a, x)$ is true iff a is *not* in $S(\sigma(x))$ for the given substitution σ.

It is easy to see that each equation $s \stackrel{?}{=} t \in \Gamma$ can be written in the form

$$s_0 + x_1 + \ldots + x_m \stackrel{?}{=} t_0 + y_1 + \ldots + y_n, \tag{1}$$

where $s_0, t_0 \in T_\Sigma(F)$, $m, n \geq 0$, $x_1, \ldots, x_m \in X$ are distinct variables, and $y_1, \ldots, y_n \in X$ are distinct variables.

Now, for each equation of the form (1) and each $a \in S(s_0) \setminus S(t_0)$, we generate the Horn clause

$$p(a, y_1) \wedge \ldots \wedge p(a, y_n) \rightarrow \bot.$$

Indeed, whenever an element $a \in C$ is in $S(s_0)$ but not in $S(t_0)$, for the equation to hold true, a must occur in the image of some y_j. The symmetric Horn clauses are also produced, that is, for each $a \in S(t_0) \setminus S(s_0)$

$$p(a, x_1) \wedge \ldots \wedge p(a, x_m) \rightarrow \bot.$$

Constants $a \in S(s_0) \cap S(t_0)$ are obviously harmless since they automatically occur on both sides of the equation. Thus, it remains to deal with the constants $a \in C$ that are *not* in $S(s_0) \cup S(t_0)$. First, if such a constant a does not occur in the image of any of the variables on the right-hand side, then it should not occur in the image of any of the variables on the left-hand side, which is expressed by the Horn clauses

$$p(a, y_1) \wedge \ldots \wedge p(a, y_n) \rightarrow p(a, x_j) \qquad \text{for all } j = 1, \ldots, m.$$

Again, we also need the symmetric clauses, that is, for each $a \notin C \setminus (S(s_0) \cup S(t_0))$ we produce

$$p(a, x_1) \wedge \ldots \wedge p(a, x_m) \rightarrow p(a, y_j) \qquad \text{for all } j = 1, \ldots, n.$$

It is easy to see that the Horn formula obtained by conjoining all the Horn clauses derived from the unification problem Γ is satisfiable iff Γ has a solution (see Kapur and Narendran (1992) for details). The number of derived Horn clauses and their sizes are obviously polynomial in the size

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

of the given ACUI-unification problem Γ. Since satisfiability of propositional Horn formulae can be tested in linear time (Dowling and Gallier, 1984), this yields the following upper bound for deciding solvability of ACUI-unification problems with constants.

Proposition 2.4. (Kapur and Narendran, 1992). *ACUI-unification with constants is decidable in polynomial time.*

3. Approximate Unification Modulo ACUI

Intuitively, in approximate unification we consider the case where a given unification problem is not solvable, and we ask what is the "best we can do" towards solving it. De facto, we will also consider solvable unification problems as input for approximate unification, but then producing an exact unifier should be the best we can do. Before introducing formal approaches for how to rank the quality of approximate unifiers, we illustrate the underlying ideas using an example.

Example 3.1. Consider the following ACUI-unification problem with constants:

$$\begin{aligned}\Gamma = \{x_1 + x_2 &\stackrel{?}{=} a,\\ x_1 + x_3 &\stackrel{?}{=} c,\\ b + c + x_2 &\stackrel{?}{=} b + x_3,\\ b + x_2 + x_3 &\stackrel{?}{=} c + x_1\}.\end{aligned}$$

Trying to solve the first two equations already fully determines the substitution $\sigma = \{x_1 \mapsto \mathbf{0}, x_2 \mapsto a, x_3 \mapsto c\}$. Even though σ solves the first two equations, it does not solve the remaining two. Under σ, the set of free constants occurring in the left-hand side of the third equation becomes $\{a, b, c\}$, while the right-hand side yields the set $\{b, c\}$. Likewise for the fourth equation we obtain $\{a, b, c\}$ on the left-hand side and $\{c\}$ on the right-hand side.

How far away is this substitution from being an exact solution? One idea is to count the number of equations that are not satisfied by it. In our example, this would yield the number 2 since σ does not solve the third and fourth equations. It is easy to see that, w.r.t. this measure, σ is actually the best we can do. In fact, in any solution of the fourth equation, b must occur in the image of x_1, and thus this substitution violates at least the first two equations. In addition, we have already seen that a solution of the first two equations cannot satisfy the third and fourth equations. Thus, it is not possible to satisfy more than two of the four equations.

The above measure fails, however, to assess how far from being solved are the violated equations. Another possibility is to count how many of the elements of C take part in at least one violation, that is, occur on one side but not on the other when the substitution has been applied. In our example, there are two such constants, namely a and b. In fact, after applying σ, both a and b occur on the left-hand side of the fourth equation, but not on the right-hand side. In contrast, c does not take part in any violation. Again, it is not hard to show that this is the best we can do w.r.t. this measure.

Still, this new measure ignores for how many equations a given element of C takes part in a violation. To take this aspect into account, we will also consider a measure that counts the overall number of violations, that is, sums up the number of equations each element of C violates. Returning to our example, this would be 3 violations for the substitution σ: 2 for a, 1 for b, and 0 for c. For this measure, we can actually do better than σ. In fact, if we define $\theta(x_1) := \theta(x_2) := \mathbf{0}$ and $\theta(x_3) := c$, then a violates only the first equation, b violates only the fourth equation, and c violates no equation. Consequently, θ gets assigned the value 2, which is better than 3, which was the value for σ.

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

In this example, and also in our general definitions, we consider only substitutions that are the identity on $V \setminus X$ and assign terms in $T_\Sigma(C)$ to the variables in X. The reason is that the assignment to variables in $V \setminus X$ has no influence on the images of the left- and right-hand sides of equations in Γ, and that free constants and variables in the images may only introduce additional violations, but cannot remove violations caused by constants in C.

In the following, we will investigate all three of the measures sketched in the example, and determine the computational complexity of the corresponding decision problems, that is, given a unification problem Γ and a natural number ℓ, is there a substitution whose value w.r.t. the given measure is $\leq \ell$? It should be clear that these decision problems are in NP. Any substitution basically assigns subsets of C to the variables, and thus all such substitutions can be guessed in nondeterministic polynomial time. For a given substitution, the value assigned to it by the respective measure can obviously be computed in polynomial time. However, for the cases that are actually NP-complete, we also provide a reduction to Max-HSAT (Jaumard and Simeone, 1987), which is known to be NP-complete. This allows us to use existing optimized solvers for Max-HSAT (see e.g., (Marques-Silva et al., 2017)) to solve approximate ACUI-unification problems. NP-hardness will be shown by a reduction from Max-HSAT.

For later reference, we now define this problem formally.

Definition 3.2. (Max-HSAT).

Input: *A nonnegative integer ℓ and a formula $\varphi = \bigwedge_{i=1}^{n} C_i$ over a finite set of propositional variables P, where C_i is a* Horn clause*, that is, of the form $p_1 \wedge \cdots \wedge p_m \rightarrow q$, with $p_1, \ldots, p_m$ being propositional variables, and q either a propositional variable or $\bot$. Note that $m = 0$ is possible, where the empty conjunction is interpreted as $\top$.*

Question: *Does there exist a set of indices $I \subseteq \{1, \ldots, n\}$ of cardinality at least ℓ, and a valuation $\nu : P \rightarrow \{0, 1\}$ such that $\nu(C_i) = 1$ for all $i \in I$.*

We call this decision problem Max-HSAT. *For a given ℓ, Max-HSAT(ℓ) consists of all Horn formulae $\varphi = \bigwedge_{i=1}^{n} C_i$ for which there is a valuation that satisfies at least ℓ clauses C_i.*

3.1 Minimizing the number of violated equations

As mentioned above, we consider only substitutions that are the identity on $V \setminus X$ and assign terms in $T_\Sigma(C)$ to the variables in X. We say that such a substitution σ *violates* an equation of the form (1) if

$$S(s_0 + \sigma(x_1) + \ldots + \sigma(x_m)) \neq S(t_0 + \sigma(y_1) + \ldots + \sigma(y_n)).$$

Definition 3.3. (MinVEq-ACUI). *Given an ACUI-unification problem with constants Γ and a nonnegative integer ℓ, we now ask whether there exists a substitution σ such that at most ℓ of the equations of the system are violated by σ. We call this decision problem* MinVEq-ACUI. *For a given ℓ, MinVEq-ACUI (ℓ) consists of all ACUI-unification problems with constants for which there is a substitution that violates at most ℓ equations of the system.*

We will show that MinVEq-ACUI is NP-complete using reductions to and from Max-HSAT.

3.1.1 Reducing MinVEq-ACUI to Max-HSAT

For this purpose, we introduce new propositional variables *good*(i), whose rôle is to determine whether the ith equation is to be satisfied or not. We conjoin *good*(i) to the left-hand side of each

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

of the Horn clauses derived from the ith equation, that is, if the ith equation is of the form (1), then we generate the following Horn clauses:

- For each $a \in S(s_0) \setminus S(t_0)$: $good(i) \wedge p(a, y_1) \wedge \ldots \wedge p(a, y_n) \to \bot$;
- For each $a \in S(t_0) \setminus S(s_0)$: $good(i) \wedge p(a, x_1) \wedge \ldots \wedge p(a, x_m) \to \bot$;
- For each $a \notin S(s_0) \cup S(t_0)$:

$$\begin{aligned} good(i) \wedge p(a, y_1) \wedge \ldots \wedge p(a, y_n) &\to p(a, x_j) \quad \text{for all } j = 1, \ldots, m; \\ good(i) \wedge p(a, x_1) \wedge \ldots \wedge p(a, x_m) &\to p(a, y_j) \quad \text{for all } j = 1, \ldots, n. \end{aligned}$$

- Furthermore, we add the Horn clause $\top \to good(i)$.

If k' is the number of clauses generated by the original KN-reduction (see Section 2) and k is the number of equations in the unification problem Γ, then we obtain $k' + k$ Horn clauses in this modified reduction. Let $\varphi_\Gamma = C_1 \wedge \cdots \wedge C_{k'+k}$ denote the Horn formula obtained by conjoining these Horn clauses.

Before proving soundness and completeness of this modified reduction, we illustrate the above construction with an example.

Example 3.4. Consider the following ACUI-unification problem with constants, which is not solvable:

$$\begin{aligned} \Gamma = \{x_1 + x_2 &\stackrel{?}{=} a, \\ b + c + x_2 &\stackrel{?}{=} b + x_3, \\ x_1 \ + x_3 &\stackrel{?}{=} c\}. \end{aligned}$$

Then the modified reduction yields the following Horn clauses:

$$\begin{array}{ll} good(1) \wedge p(a, x_1) \wedge p(a, x_2) \to \bot & good(3) \to p(a, x_1) \\ good(1) \to p(b, x_1) & good(3) \to p(a, x_3) \\ good(1) \to p(b, x_2) & good(3) \to p(b, x_1) \\ good(1) \to p(c, x_1) & good(3) \to p(b, x_3) \\ good(1) \to p(c, x_2) & good(3) \wedge p(c, x_1) \wedge p(c, x_3) \to \bot \\ good(2) \wedge p(a, x_2) \to p(a, x_3) & \top \to good(1) \\ good(2) \wedge p(a, x_3) \to p(a, x_2) & \top \to good(2) \\ good(2) \wedge p(c, x_3) \to \bot & \top \to good(3) \end{array}$$

The system contains $k = 3$ equations and the original KN-reduction would produce $k' = 13$ clauses. Thus, φ_Γ contains $k' + k = 16$ Horn clauses.

The valuation that sets $p(a, x_1)$, $p(a, x_2)$, $p(a, x_3)$, $p(c, x_3)$, and $good(3)$ to false and all other variables to true satisfies all but the last clause. This corresponds to the substitution σ with $\sigma(x_1) = \sigma(x_2) = a$ and $\sigma(x_3) = a + c$, which satisfies the first two equations and violates the third one.

Intuitively, setting the propositional variable $good(i)$ to false "switches off" the Horn clauses induced by the ith equation in the original KN-reduction. Consequently, the satisfaction of these clauses is no longer enforced, which means that the ith equation may be violated. By maximizing satisfaction of the clauses $\top \to good(i)$, we thus minimize the number of violated equations from Γ. More precisely, we can show the following lemma.

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

Lemma 3.5. *Let Γ be an ACUI-unification problem with constants consisting of k equations and generating k' clauses in the KN-reduction introduced in Section 2. Then we have, for all $0 \leq \ell \leq k$:*

$$\Gamma \in \mathit{MinVEq\text{-}ACUI}(\ell) \text{ iff } \varphi_\Gamma \in \mathit{Max\text{-}HSAT}((k' + k) - \ell).$$

Proof. Let $\Gamma \in$ MinVEq-ACUI(ℓ). This means that there exists a substitution σ such that at most ℓ of the equations in Γ are violated by σ. We use σ to define a valuation of the propositional variables occurring in φ_Γ. For every $a \in C$ and every variable $x \in X$ we set $p(a, x)$ to true iff $a \notin S(\sigma(x))$. In addition, we set $good(i)$ to true iff σ does not violate the ith equation of Γ.

Now, suppose that the ith equation of Γ is not violated by σ. Then our valuation makes all clauses corresponding to the ith equation evaluate to true, and in addition it also satisfies $\top \rightarrow good(i)$. If the jth equation of Γ is violated, then $good(j)$ is false. Consequently, all clauses of φ_Γ corresponding to the jth equation evaluate to true, but $\top \rightarrow good(j)$ evaluates to false. Summing up, the valuation induced by σ satisfies all the clauses of φ_Γ, with the exception of the clauses $\top \rightarrow good(j)$ if the jth equation is violated by σ. Since σ violates $\leq \ell$ equations and there are $k' + k$ clauses in φ_Γ, the valuation satisfies $\geq (k' + k) - \ell$ clauses, which shows that $\varphi_\Gamma \in$ Max-HSAT$((k' + k) - \ell)$.

For the opposite direction, let $\varphi_\Gamma \in$ Max-HSAT$((k' + k) - \ell)$. This means that there is a valuation ν and a set of indices $I \subseteq \{1, \ldots, k' + k\}$ with $|I| \geqslant k' + k - \ell$ such that $\nu(C_i) = 1$ for all $i \in I$. Since $k \geq \ell$, we know that $\geq k - \ell$ of the clauses $\top \rightarrow good(i)$ must evaluate to true, that is, $\geq k - \ell$ of the propositional variables $good(i)$ evaluate to true. If i is an index for which $\nu(good(i)) = 1$, then all the clauses produced by the KN-reduction of Kapur and Narendran (see Section 2) for the ith equation evaluate to true. By the correctness of the KN-reduction, the ACUI-unification problem consisting of the equations for which $\nu(good(i)) = 1$ has a solution. Thus, there is a substitution that solves $\geq k - \ell$ equations of Γ, that is, violates $\leq \ell$ equations. This shows that $\Gamma \in$ MinVEq-ACUI(ℓ). □

Since Max-HSAT is in NP, this lemma implies that MinVEq-ACUI also belongs to NP.

3.1.2 Reducing Max-HSAT to MinVEq-ACUI

Consider the Horn formula $\varphi = C_1 \wedge \ldots \wedge C_k$, where C_i is a Horn clause for $i = 1, \ldots, k$. To construct a corresponding ACUI-unification problem with constants Γ_φ, it is sufficient to use a single free constant a, that is, we will have $C = \{a\}$. For every propositional variable p appearing in φ, we introduce a variable x_p. Intuitively, a occurs in x_p iff p is set to false. Now, each Horn clause in φ yields the following equations:

- If C_i is of the form $p_1 \wedge \ldots \wedge p_n \rightarrow p$, then the corresponding equation is
$$x_{p_1} + \ldots + x_{p_n} + x_p =^? x_{p_1} + \ldots + x_{p_n}.$$
Obviously, this equation enforces that a cannot occur in x_p if it does not occur in any of the variables x_{p_i}.
- If C_i is of the form $p_1 \wedge \ldots \wedge p_n \rightarrow \bot$, then the corresponding equation is
$$x_{p_1} + \ldots + x_{p_n} =^? a.$$
This equation enforces that a must occur in one of the variables x_{p_i}.
- If C_i is of the form $\top \rightarrow p$, then the corresponding equation is
$$x_p =^? \mathbf{0}.$$
This equation ensures that a cannot occur in x_p.

The following example illustrates the construction of Γ_φ.

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

Example 3.6. Consider the Horn formula

$$\varphi = (p_1 \wedge p_2 \to p_3) \wedge (p_1 \wedge p_3 \to \bot) \wedge (p_2 \wedge p_3 \to \bot) \wedge (\top \to p_1) \wedge (\top \to p_2).$$

The corresponding ACUI unification problem consists of the following equations:

$$\begin{aligned} &x_{p_1} + x_{p_2} + x_{p_3} \stackrel{?}{=} x_{p_1} + x_{p_2}, \\ &x_{p_1} + x_{p_3} \stackrel{?}{=} a, \quad x_{p_2} + x_{p_3} \stackrel{?}{=} a, \\ &x_{p_1} \stackrel{?}{=} \mathbf{0}, \quad x_{p_2} \stackrel{?}{=} \mathbf{0}. \end{aligned}$$

It is easy to see that φ is not satisfiable. The valuation ν that makes p_1 and p_2 true, and p_3 false satisfies all clauses except for the first one. Given the intuition that a occurs in x_{p_i} iff p_i is set to false, this valuation induces the following substitution σ:

$$\sigma := \{x_{p_1} \mapsto \mathbf{0}, x_{p_2} \mapsto \mathbf{0}, x_{p_3} \mapsto a\},$$

which solves all equations in the ACUI-unification problem except for the first one.

More generally, we will show that there is a 1–1-relationship between valuations satisfying certain clauses and substitutions satisfying the corresponding equations. It is worth noting, however, that in Max-HSAT the number of satisfied clauses is maximized, whereas in MinVEq-ACUI the number of violated equations is minimized.

Lemma 3.7. *Let $\varphi = C_1 \wedge \ldots \wedge C_k$ be a Horn formula and Γ_φ the corresponding ACUI unification problem. Then we have, for all $0 \leq \ell \leq k$:*

$$\varphi \in \textit{Max-HSAT}(\ell) \textit{ iff } \Gamma_\varphi \in \textit{MinVEq-ACUI}(k - \ell).$$

Proof. Suppose that $\varphi \in \text{Max-HSAT}(\ell)$. This means that there exists a valuation ν and a set of indices $I \subseteq \{1, \ldots, k\}$, $|I| \geq \ell$ such that $\nu(C_i) = 1$ for every $i \in I$. Given such a valuation ν, we define the substitution σ as follows:

$$\sigma(x_{p_i}) := a \text{ if } \nu(p_i) = 0 \text{ and } \sigma(x_{p_i}) := \mathbf{0} \text{ if } \nu(p_i) = 1.$$

We show that, for every $i \in I$, the ith equation is solved by this substitution. Indeed, if the ith equation is of the form:

- $x_{p_1} + \ldots + x_{p_n} + x_p \stackrel{?}{=} x_{p_1} + \ldots + x_{p_n}$, then $C_i = p_1 \wedge \cdots \wedge p_n \to p$ evaluates to true under ν. This means that either $\nu(p_j) = 0$ for some $j \in I$ or $\nu(p) = 1$. In the first case, a then occurs on both sides of the equation, and thus the equation is solved. In the second case, $\sigma(x_p) = \mathbf{0}$, and again the equation is solved by σ.
- $x_{p_1} + \cdots + x_{p_n} \stackrel{?}{=} a$, then $C_i = p_1 \wedge \cdots \wedge p_n \to \bot$ evaluates to true under ν. This means that $\nu(p_j) = 0$ for some $j \in I$, and thus $\sigma(x_{p_j}) = a$ for some $j \in I$. Consequently, $\sigma(x_{p_1} + \cdots + x_{p_n})$ is a sum of as and $\mathbf{0}$s, which implies that the ith equation is solved by σ.
- $x_p \stackrel{?}{=} \mathbf{0}$, then $C_i = \top \to p$ evaluates to true under ν. This means that $\nu(p) = 1$, and thus $\sigma(x_p) = \mathbf{0}$. This shows that the ith equation is solved by σ.

Consequently, the substitution σ solves at least ℓ equations of Γ_φ, and thus violates at most $k - \ell$ equations, which implies $\Gamma_\varphi \in \text{MinVEq-ACUI}(k - \ell)$.

For the opposite direction, if $\Gamma_\varphi \in \text{MinVEq-ACUI}(k - \ell)$, then there is a substitution σ such that at least ℓ equations of Γ_φ are *not* violated. We can assume without loss of generality that σ uses a as the only free constant. If we define $\nu(p) = 0$ iff a occurs in $\sigma(x_p)$, then we can show (in the same way as above) that ν satisfies at least ℓ clauses of φ, which implies $\varphi \in \text{Max-HSAT}(\ell)$. □

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

Since Max-HSAT is NP-hard, this lemma implies that MinVEq-ACUI is also NP-hard. Put together, the two lemmas yield the exact complexity of MinVEq-ACUI.

Theorem 3.8. *MinVEq-ACUI is NP-complete. The NP-hardness result holds even for ACUI-unification problems with only one free constant.*

3.2 Minimizing the number of violating elements

For the second measure, instead of minimizing the number of violated equations, we will minimize the number of violating elements of C, where as before C is the set of free constants occurring in the unification problem.

Given a substitution σ, we say that $a \in C$ *violates* an equation of the form (1) *w.r.t.* σ if

$$a \in S(s_0 + \sigma(x_1) + \ldots + \sigma(x_m)) \,\Delta\, S(t_0 + \sigma(y_1) + \ldots + \sigma(y_n)),$$

where Δ denotes the symmetric difference of two sets. We say that $a \in C$ *violates* Γ *w.r.t.* σ if it violates at least one equation in Γ w.r.t. σ.

Definition 3.9. *Given an ACUI-unification problem with constants* Γ *and a nonnegative integer* ℓ*, we now ask whether there exists a substitution* σ *such that at most* ℓ *constants violate* Γ *w.r.t.* σ*. We call this decision problem* MinVEl-ACUI*. For a given* ℓ*, MinVEl-ACUI* (ℓ) *consists of all ACUI-unification problems* Γ *with constants for which there is a substitution w.r.t. which at most* ℓ *constants violate* Γ*.*

In contrast to the problem MinVEq-ACUI considered in the previous section, MinVEl-ACUI can be solved in polynomial time. In order to show this, we use projections of equations to free constants. As noted earlier, any term $t \in T_\Sigma(F, V)$ can be written in the form $t = t_0 + x_1 + \ldots + x_n$, where $t_0 \in T_\Sigma(F)$ is a ground term and $x_1, \ldots, x_n$ are variables in V. Given a constant $a \in F$, the *projection* of such a term onto a is defined to be $t^a = \pi_a(t_0) + x_1 + \ldots + x_n$, where for a ground term $t_0 \in T_\Sigma(F)$ we set $\pi_a(t_0) = a$ if a occurs in t_0, and $\mathbf{0}$ otherwise. Then the projection of an equation $s =^? t$ to a is $s^a =^? t^a$, and the projection of an ACUI-unification problem with constants Γ to a, denoted by Γ^a, is the system of the projections of the equations in Γ to a. Finally, the projection of a ground substitution σ to a is the substitution $\sigma^a : V \to T_\Sigma(\{a\})$ defined as $\sigma^a(x) := \sigma(x)^a$.

Consider the unification problem Γ and the substitution σ introduced in Example 3.4. The constant a violates Γ w.r.t. σ, while b, c do not. For the elements of $C = \{a, b, c\}$, we obtain the following projections of Γ:

Γ^a	Γ^b	Γ^c
$x_1 + x_2 =^? a$	$x_1 + x_2 =^? \mathbf{0}$	$x_1 + x_2 =^? \mathbf{0}$
$x_2 =^? x_3$	$b + x_2 =^? b + x_3$	$c + x_2 =^? x_3$
$x_1 + x_3 =^? \mathbf{0}$	$x_1 + x_3 =^? \mathbf{0}$	$x_1 + x_3 =^? c$

Likewise, for the substitution σ we obtain the projections:

σ^a	σ^b	σ^c
$\sigma^a(x_1) = a$	$\sigma^b(x_1) = \mathbf{0}$	$\sigma^c(x_1) = \mathbf{0}$
$\sigma^a(x_2) = a$	$\sigma^b(x_2) = \mathbf{0}$	$\sigma^c(x_2) = \mathbf{0}$
$\sigma^a(x_3) = a$	$\sigma^b(x_3) = \mathbf{0}$	$\sigma^c(x_3) = c$

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

One can easily check that σ^b and σ^c solve Γ^b and Γ^c, respectively, whereas σ^a does not solve Γ^a. This is closely related to the fact that a violates Γ w.r.t. σ, but b and c do not.

Lemma 3.10. *Let Γ be an ACUI-unification problem with constants. Then the following holds:*

(1) *The constant $a \in C$ violates Γ w.r.t. σ iff σ^a does not solve Γ^a.*
(2) *Given substitutions $\sigma_a : V \to T_\Sigma(\{a\})$ for all $a \in C$, we define the substitution $\sigma : V \to T_\Sigma(C)$ as*

$$\sigma(x) = \sum_{a \in C} \sigma_a(x) \text{ for all } x \in V.$$

Then we have $\sigma^a = \sigma_a$ for all $a \in C$.
(3) *There is a substitution $\sigma : V \to T_\Sigma(C)$ such that at most ℓ of the elements of C violate Γ w.r.t. σ iff at most ℓ of the ACUI-unification problems Γ^a ($a \in C$) are not solvable.*

Proof. We will show the first two facts stated in the lemma, and then use them to prove the third.

(1) This is an easy consequence of the following equivalences. The constant a violates the equation $s \stackrel{?}{=} t \in \Gamma$ w.r.t. σ iff $a \in S(\sigma(s)) \,\Delta\, S(\sigma(t))$ iff $a \in S(\sigma(s)^a) \,\Delta\, S(\sigma(t)^a)$ iff $a \in S(\sigma^a(s^a)) \,\Delta\, S(\sigma^a(t^a))$ iff σ^a does not solve $s^a \stackrel{?}{=} t^a \in \Gamma^a$.
(2) $\sigma^a(x) = (\sum_{a' \in C} \sigma_{a'}(x))^a = \sum_{a' \in C} (\sigma_{a'}(x)^a) = \sigma_a(x)^a = \sigma_a(x)$.
(3) Suppose that there is a substitution $\sigma : V \to T_\Sigma(C)$ such that at most ℓ elements of C violate Γ w.r.t. σ. Because of the first fact this implies that at least $k - \ell$ of the projected unification problems Γ^a are solved by the projected substitutions σ^a. Consequently, at most ℓ of the projected problems Γ^a are not solvable.
For the opposite direction, suppose that at most ℓ of the systems Γ^a are not solvable. For every $a \in C$, if Γ^a is solvable, let $\sigma_a : V \to T_\Sigma(\{a\})$ be a substitution that solves it, and an arbitrary substitution $V \to T_\Sigma(\{a\})$ otherwise. Define the substitution $\sigma : V \to T_\Sigma(C)$ as $\sigma(x) := \sum_{a \in C} \sigma_a(x)$ for all $x \in V$. Then the constant $a \in C$ violates Γ w.r.t. σ iff $\sigma^a = \sigma_a$ solves Γ^a iff Γ^a is solvable. Consequently, at most ℓ of the elements of C violate Γ w.r.t. σ.

This completes the proof of the lemma. □

Due to the third fact stated in the above lemma, to check whether $\Gamma \in \text{MinVEl-ACUI}(\ell)$, it is sufficient to check which of the ACUI-unification problems Γ^a for $a \in C$ are solvable. This can obviously be done in polynomial time.

Theorem 3.11. *The problem MinVEl-ACUI is in P.*

3.3 Minimizing the number of violations

A disadvantage of the violated elements measure used in the previous section is that it does not distinguish between constants that violate only one equation and those violating many equations. To overcome this problem, we count for each violating constant how many equations it actually violates. We say that $a \in C$ *violates* Γ *p times w.r.t.* σ if it violates p equations in Γ w.r.t. σ. Further, we say that σ *violates* Γ *q times* if $q = \sum_{a \in C} p_a$ where, for each $a \in C$, the element a violates Γ p_a times w.r.t. σ.

Definition 3.12. (MinV-ACUI). *Given an ACUI-unification problem with constants Γ and a non-negative integer ℓ, we now ask whether there exists a substitution σ that violates Γ at most ℓ times.*

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

$$good(1,a) \wedge p(a,x_1) \wedge p(a,x_2) \to \bot$$

$$good(1,b) \to p(b,x_1) \qquad good(3,a) \to p(a,x_1)$$
$$good(1,b) \to p(b,x_2) \qquad good(3,a) \to p(a,x_3)$$
$$good(1,c) \to p(c,x_1) \qquad good(3,b) \to p(b,x_1)$$
$$good(1,c) \to p(c,x_2) \qquad good(3,b) \to p(b,x_3)$$
$$good(2,a) \wedge p(a,x_2) \to p(a,x_3) \qquad good(3,c) \wedge p(c,x_1) \wedge p(c,x_3) \to \bot$$
$$good(2,a) \wedge p(a,x_3) \to p(a,x_2)$$
$$good(2,c) \wedge p(c,x_3) \to \bot$$

$$\top \to good(1,a) \qquad \top \to good(2,a) \qquad \top \to good(3,a)$$
$$\top \to good(1,b) \qquad \top \to good(2,b) \qquad \top \to good(3,b)$$
$$\top \to good(1,c) \qquad \top \to good(2,c) \qquad \top \to good(3,c)$$

Figure 1. The Horn clauses obtained by applying the modified reduction to Γ from Example 3.4.

We call this decision problem MinV-ACUI. *For a given threshold value ℓ, MinV-ACUI (ℓ) consists of those ACUI-unification problems with constants Γ for which there is a substitution that violates Γ at most ℓ times.*

The approach used in Section 3.1 to solve MinVEq-ACUI can easily be adapted to solve this new problem. Basically, we now introduce propositional variables $good(i,a)$ (instead of simply $good(i)$) to characterize whether the element $a \in C$ violates the ith equation. We conjoin $good(i,a)$ to the left-hand side of each of the Horn clauses derived from the ith equation for a. Furthermore, we add the Horn clauses $\top \to good(i,a)$ instead of $\top \to good(i)$. Following the earlier notation, we obtain $k' + k|C|$ Horn clauses in this modified reduction, and again use φ_Γ to denote the Horn formula obtained this way.

If we apply this modified reduction to the ACUI-unification problems of Example 3.4, then we obtain the Horn clauses depicted in Figure 1. Consider the substitution θ with $\theta(x_1) = \theta(x_2) = a$ and $\theta(x_3) = c$. Then a violates the second and the third equations, whereas b and c do not violate any equation w.r.t. θ. We can use θ to define a valuation ν, which sets $p(a,x_1)$, $p(a,x_2)$, $p(c,x_3)$, $good(2,a)$, $good(3,a)$ to false and all other propositional variables to true. This valuation satisfies all clauses in Figure 1, except for $\top \to good(2,a)$ and $\top \to good(3,a)$.

The following lemma states correctness of the modified reduction. Since its proof is very similar to the proof of Lemma 3.5, we leave it to the reader.

Lemma 3.13. *Let Γ be an ACUI-unification problem consisting of k equations, and generating k' clauses in the KN-reduction introduced in Section 2, and let C be the set of free constants occurring in Γ. Denote with $\varphi_\Gamma = C_1 \wedge \cdots \wedge C_{k'+k|C|}$ the Horn formula obtained by applying the the modified reduction to Γ. Then we have*

$$\Gamma \in \textit{MinV-ACUI}(\ell) \;\; \textit{iff} \;\; \varphi_\Gamma \in \textit{Max-HSAT}((k' + k|C|) - \ell).$$

Since Max-HSAT is in NP, this lemma implies that MinV-ACUI is also in NP. NP-hardness of MinV-ACUI actually follows directly from Lemma 3.7. In fact, the reduction considered in this lemma requires only a single constant a. In this setting, counting the number of violated equations is the same as counting the number of all violations, and thus MinV-ACUI coincides with MinVEq-ACUI. This shows that MinV-ACUI is NP-hard.

Theorem 3.14. *The problem MinV-ACUI is NP-complete.*

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

4. Unification Modulo ACUIG

In this section, we consider unification modulo ACUIG, that is, ACUI extended with a finite set of ground identities G. We will prove that, in this setting, we obtain the same complexity bounds as for ACUI-unification. Initially, we will demonstrate that the word problem in ACUIG is decidable in polynomial time. Subsequently, we will use this result to prove that ACUIG-unification with constant restrictions, a notion that generalizes unification with constants, is also decidable in polynomial time. Finally, using previous combination results from Baader and Schulz (1993) and the hardness result for general ACUI-unification by Kapur and Narendran (1992), we will conclude that general ACUIG-unification is NP-complete.

4.1 The word problem for ACUIG

Just as in Section 2, we consider the signature $\Sigma = \{+, \mathbf{0}\}$ and the equational theory ACUI that states that $+$ is an associative, commutative, and idempotent binary function symbol with unit $\mathbf{0}$. But now we extend ACUI with a finite set of ground identities $G \subseteq T_\Sigma(F) \times T_\Sigma(F)$, and denote the equational theory obtained this way by ACUIG. The *word problem* for ACUIG asks whether two given terms $s, t \in T_\Sigma(F)$ are equivalent modulo ACUIG, that is, whether $s =_{\mathsf{ACUIG}} t$ holds or not. As already mentioned in the introduction, we can measure the complexity of this problem in two different ways. On one hand, we can assume that G is fixed beforehand, and then consider the word problem for the fixed equational theory $\mathsf{ACUIG} = \mathsf{ACUI} \cup G$. The complexity of the word problem is then measured in the size of the input terms s, t. On the other hand, we can view G to be part of the input and then measure the complexity in the combined size of s, t, and G. If the complexity is measured in terms of s, t only, we will call this *term complexity*, and otherwise *combined complexity*. We will actually show that the word problem for ACUIG is in P for combined complexity, which obviously implies that it is also in P for term complexity.

Recall that modulo ACUI, two ground terms $s, t \in T_\Sigma(F)$ are equivalent iff they contain the same constants. However, in the presence of ground identities G, the latter condition is sufficient, but not necessary for two terms to be equivalent. In fact, $\mathsf{ACUI} \subseteq \mathsf{ACUIG}$ obviously yields that $S(s) = S(t)$ implies $s =_{\mathsf{ACUIG}} t$. However, the opposite direction need no longer hold, as shown by the following example. Consider the terms $s = b + a + a$ and $t = a + b + c$, with corresponding sets $S(s) = \{a, b\}$ and $S(t) = \{a, b, c\}$, and the ground theory $G = \{a + b = c\}$. We have $s = b + a + a =_{\mathsf{ACUI}} a + b + a + b =_G a + b + c = t$, and thus $s =_{\mathsf{ACUIG}} t$, even though $S(s) \neq S(t)$. Intuitively, the identity in G can be used to add c to the set $\{a, b\}$.

We will now show how to decide whether two terms are equivalent modulo ACUIG. For this purpose, we saturate the sets of constants occurring in the terms using the identities in G to add constants, as we have done with c in our example.

Definition 4.1. *Given a finite set of constants $A \subseteq F$, its* saturation A^G *is obtained by iteratively applying the identities of G as follows:*

- *begin with setting $A^G := A$;*
- *as long as there is an identity $g_i = h_i$ in G such that $S(g_i) \subseteq A^G$ and $S(h_i) \not\subseteq A^G$ (or $S(h_i) \subseteq A^G$ and $S(g_i) \not\subseteq A^G$), extend A^G by setting $A^G := A^G \cup S(h_i)$ (respectively, by setting $A^G := A^G \cup S(g_i)$).*

This saturation process terminates after a number of iterations that is bounded by the cardinality of G. In fact, once an identity $g_i = h_i$ is applied in the saturation process, it is no longer applicable since the set A^G then contains $S(g_i) \cup S(h_i)$. It is also easy to see that the result of the saturation does not depend on the order in which rules are applied. Thus, each finite set $A \subseteq F$ has a unique saturation A^G, which can be computed in time polynomial in the cardinality of A and the size of G.

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

Example 4.2. Consider the set of ground identities

$$G = \{a + b + c = d,\ b + c + e = f\}$$

and the term $s = a + f$, which yields the start set $A = \{a, f\}$. The saturation process for A starts with setting $A^G := \{a, f\}$. For the second identity, we have that $S(f) = \{f\} \subseteq A^G$, but $S(b + c + e) = \{b, c, e\} \not\subseteq A^G$. Hence, we can extend A^G to the new set $A^G := A^G \cup S(b + c + e) = \{a, b, c, e, f\}$. Now, for the first identity, we have that $S(a + b + c) = \{a, b, c\} \subseteq A^G$, but $S(d) = \{d\} \not\subseteq A^G$, and thus we obtain $A^G := A^G \cup S(d) = \{a, b, c, d, e, f\}$. This is the final saturated set since it cannot be further extended using the identities in G.

The following lemma is an easy consequence of the definition of saturation.

Lemma 4.3. *Let A, B be finite subsets of F. Then the following holds:*

$$A \subseteq A^G,\ \ A^{GG} = A^G,\ \ A \subseteq B \Rightarrow A^G \subseteq B^G,\ \ A^G \cup B^G \subseteq (A \cup B)^G.$$

Proposition 4.4. *Let $s, t \in T_\Sigma(F)$. Then $s =_{\mathsf{ACUIG}} t$ iff $S(s)^G = S(t)^G$. In particular, the combined and thus also the term complexity for the word problem for ACUIG is in P.*

Proof. Decidability in polynomial time obviously follows from the equivalence in the first statement since the saturation A^G of a finite set $A \subseteq F$ can be computed in polynomial time, and the cardinality of $S(s), S(t)$ is bounded by the size of s, t.

To show the equivalence, first assume that $S(s)^G = S(t)^G$. To conclude from this that $s =_{\mathsf{ACUIG}} t$, it is sufficient to show that saturation steps correspond to rewrite steps in ACUIG. Thus, assume that $l \in T_\Sigma(F)$, and that $g_i = h_i$ is an identity in G such that $S(g_i) \subseteq S(l)$. Then l is of the form $l =_{\mathsf{ACUI}} g_i + l'$. We now have $l =_{\mathsf{ACUI}} g_i + l' =_{\mathsf{ACUI}} g_i + g_i + l' =_G h_i + g_i + l' =_{\mathsf{ACUI}} h_i + l$, and $S(h_i + l) = S(l) \cup S(h_i)$. This shows that there are terms $s^G, t^G \in T_\Sigma(F)$ such that $u =_{\mathsf{ACUIG}} u^G$ and $S(u^G) = S(u)^G$ for $u \in \{s, t\}$. By Lemma 2.1, we thus know that $S(s)^G = S(t)^G$ implies $s^G =_{\mathsf{ACUI}} t^G$, and thus we have $s =_{\mathsf{ACUIG}} s^G =_{\mathsf{ACUI}} t^G =_{\mathsf{ACUIG}} t$.

Second, assume that $S(s)^G \neq S(t)^G$. To show that this implies $s \neq_{\mathsf{ACUIG}} t$, we construct a model $\mathcal{A}$ of ACUIG in which the identity $s = t$ does not hold. As interpretation domain, we use all saturated sets over the constants occurring in s, t, or G, that is, $\Delta := \{A^G \mid A \subseteq C\}$, where C consists of the elements of F that occur in s, t, or G. Since saturation adds only constants occurring in G, we know that $A \subseteq C$ implies $A^G \subseteq C$. The binary symbol $+$ is interpreted as union followed by saturation, that is, $A^G + B^G := (A^G \cup B^G)^G$, $\mathbf{0}$ as $\emptyset^G$, and $c \in C$ as $\{c\}^G$. Given a term $u \in T_\Sigma(C)$, its interpretation in this algebra is $S(u)^G$. This can easily be shown by induction on the structure of u, where the induction step uses the fact that

$$(A^G \cup B^G)^G = (A \cup B)^G, \tag{2}$$

which is an easy consequence of Lemma 4.3. Thus, $S(s)^G \neq S(t)^G$ implies that the terms s, t have different interpretations in $\mathcal{A}$. To show $s \neq_{\mathsf{ACUIG}} t$, it is thus sufficient to show that $\mathcal{A}$ satisfies all identities of ACUIG. For the identities in ACUI, this is an easy consequence of (2) and the fact that set union is associative, commutative, and idempotent and has $\emptyset$ as unit. Now consider an identity $g_i = h_i \in G$. When saturating the corresponding sets $S(g_i)$ and $S(h_i)$, one can in a first step go both from $S(g_i)$ and from $S(h_i)$ to $S(g_i) \cup S(h_i)$ (unless this step is void due to an inclusion). Saturating further, one thus obtains identical saturated sets, which shows that g_i and h_i are interpreted in $\mathcal{A}$ by the same saturated set. □

Continuing Example 4.2, recall that the term $s = a + f$ has the saturated set $S(s)^G = \{a, b, c, d, e, f\}$. It is easy to see that, for $t = b + d + e$, saturation produces the sequence of sets

$$S(t) = \{b, d, e\} \to \{a, b, c, d, e\} \to \{a, b, c, d, e, f\} = S(t)^G,$$

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

where in the first step the identity $d = a + b + c$ is applied, and in the second the identity $b + c + e = f$. Thus, we have $S(s)^G = S(t)^G$, which shows that $s = a + f =_{\mathsf{ACUIG}} b + d + e = t$.

4.2 ACUIG-unification with constant restriction

As in the previous subsection, let $\Sigma = \{+, \mathbf{0}\}$, F a countably infinite set of constants, and V a countably infinite set of variables. Given a finite set of ground identities $G \subseteq T_\Sigma(F) \times T_\Sigma(F)$, we now consider unification modulo $\mathsf{ACUIG} = \mathsf{ACUI} \cup G$. Note that in this setting the constants occurring in G are no longer free constants, but theory constants. Thus, an ACUIG-unification problem with constants may contain the constant $\mathbf{0}$, the constants from G, and additional free constants, that is, elements of F that do not occur in G. For a given unification problem, a constant restriction prohibits the occurrence of certain free constants in the image of certain variables.

Definition 4.5. ACUIG-unification problem with constant restriction.

Input: *A finite system* $\Gamma = \{s_1 =^? t_1, \ldots, s_k =^? t_k\}$ *of equations between terms in* $T_\Sigma(V, F)$*, a finite set of ground identities* $G = \{g_1 = h_1, \ldots, g_m = h_m\}$ *between terms in* $T_\Sigma(F)$*, and a mapping* $\mathfrak{r} : D \to 2^X$ *where* $X \subseteq V$ *is the set of variables occurring in* Γ *and* $D \subseteq F$ *is the set of free constants occurring in* Γ*.*

Question: *Is there a substitution* σ *such that* $\sigma(s_i) =_{\mathsf{ACUIG}} \sigma(t_i)$ *for every* $i = 1, \ldots, k$ *and for every* $x \in X$ *and* $d \in D$ *we have that* d *does not occur in* $\sigma(x)$ *if* $x \in \mathfrak{r}(d)$*? Such a substitution is called an ACUIG-unifier of* Γ *w.r.t.* $\mathfrak{r}$*.*

Note that we consider the set of ground identities G to be part of the input. Thus, the complexity upper bound of P shown below for deciding ACUIG-unification with constant restriction holds for combined complexity, which implies the same upper bound also for term complexity.

In the following, we assume that $C \subseteq F$ is the set of constants occurring in Γ or G, and $X \subseteq V$ is the set of variables occurring in Γ. As before, in order to check whether Γ has a unifier w.r.t. $\mathfrak{r}$, it is sufficient to consider substitutions that are the identity on $V \setminus X$ and replace every $x \in X$ by a term in $T_\Sigma(C)$. In fact, any ACUIG-unifier of Γ w.r.t. $\mathfrak{r}$ can be turned into one satisfying these properties by making it the identity on $V \setminus X$ and replacing variables and constants in $F \setminus C$ occurring in $\sigma(x)$ for $x \in X$ with $\mathbf{0}$.

Intuitively, our algorithm for solving ACUIG-unification with constant restriction starts with a maximal substitution σ that respects the constant restriction. More precisely, maximal means that $\sigma(x)$ contains all theory constants (constants occurring in G) and all free constants not disallowed by the constant restriction. Using the set notation, we have that $S(\sigma(x)) = \{c \in C \mid c \text{ occurs in } G \text{ or } c \in D \text{ and } x \notin \mathfrak{r}(c)\}$. Note that this is an overapproximation of any solution θ (i.e., $S(\theta(x)) \subseteq S(\sigma(x))$ holds for all variables x), if a solution exists at all.

Next, whenever an equation $s =^? t$ is not satisfied by the current substitution σ modulo ACUIG, we know that $S(\sigma(s))^G \neq S(\sigma(t))^G$. Assume for example that $S(\sigma(s))^G \not\subseteq S(\sigma(t))^G$. By Lemma 4.3 we obtain that $S(\sigma(s)) \not\subseteq S(\sigma(t))^G$, that is, there occurs a constant a in $\sigma(s)$ such that $a \notin S(\sigma(t))^G$. This constant is either introduced to $\sigma(s)$ by the substitution, or it already occurs in s. In the latter case, since the substitution σ is an overapproximation of any solution, we cannot remedy this situation by adding the constant a to $\sigma(t)$. Thus, the algorithm terminates in this case and returns **Fail**. Otherwise, there exists a variable x occurring in s such that $S(\sigma(x)) \not\subseteq S(\sigma(t))^G$. In this case, we trim the substitution, so that it no longer introduces this violation. This trimming process is such that the obtained substitution remains an overapproximation of any solution. The algorithm terminates successfully if all the equations are satisfied modulo ACUIG, in which case the current substitution σ is a solution. This happens after a polynomial number of steps since

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

Algorithm 1

Input: An ACUIG-unification problem with constant restriction $\mathfrak{r}$, as introduced in Definition 4.5.
Output: A unifier σ w.r.t. $\mathfrak{r}$ or **Fail**.
Let σ be the maximal substitution satisfying the constant restriction, i.e., for all $x \in X$ we set

$$\sigma(x) := \sum(\{c \in C \mid c \text{ occurs in } G \text{ or } c \in D \text{ and } x \notin \mathfrak{r}(c)\}).$$

while *some equation* $s =^? t$ *in* Γ *is not satisfied by* σ *modulo* ACUIG **do**
  if *there is a variable* x *in* s *such that* $S(\sigma(x)) \not\subseteq S(\sigma(t))^G$ *or* y *in* t *such that* $S(\sigma(y)) \not\subseteq S(\sigma(s))^G$ **then**
    Set $\sigma(x) := \sum(S(\sigma(x)) \cap S(\sigma(t))^G)$ for all variables x in s
    Set $\sigma(y) := \sum(S(\sigma(y)) \cap S(\sigma(s))^G)$ for all variables y in t
  else
    return Fail
  end
end
return σ

Figure 2. Algorithm for ACUIG-unification with constant restriction.

in each iteration of the while-loop the set $S(\sigma(x))$ for at least one variable x becomes smaller in the trimming step. Otherwise, the algorithm must terminate with failure at some point due to the reasons explained above.

The formal description of our algorithm, called Algorithm 1, can be found in Figure 2. In this description, we use the following notation for turning a set of constants into the term that sums up these constants: given $A \subseteq C$, we denote with $\sum(A)$ the term $\sum_{a \in A} a$. Note that $S(\sum(A)) = A$ and $\sum(S(s)) =_{\mathsf{ACUI}} s$ for all $s \in T_\Sigma(C)$.

Before proving correctness of this algorithm, we illustrate how it works on two examples, one where the input is a solvable unification problem, and the other where the input problem is not solvable.

Example 4.6. Consider the system of equations

$$\Gamma = \{g + x_2 =^? a + x_1,\ b + x_1 =^? c + f + g,\ c + x_2 =^? a + c + e\},$$

the set of ground identities $G = \{a + b + c = d,\ b + c + e = f\}$ from Example 4.2, and the constant restriction

$$\mathfrak{r}(g) = \{x_2\}.$$

Note that g is the only *free* constant occurring in Γ, and thus it is the only constant occurring in this constant restriction. Also note that if we had $x_1 \in \mathfrak{r}(g)$, then the second equation of Γ would not be solvable. In addition, without G, this second equation would not be solvable either: b belongs to the left-hand side, but could never belong to the right-hand side without additional ground identities.

The algorithm begins by setting

$$\sigma(x_1) := a + b + c + d + e + f + g \text{ and } \sigma(x_2) := a + b + c + d + e + f.$$

Next, the algorithm enters the while-loop and picks in each iteration an equation that is not satisfied modulo ACUIG:

– The second equation is not satisfied by σ modulo ACUIG:
  we have $S(\sigma(c + f + g))^G = \{b, c, e, f, g\}$, and hence $S(\sigma(x_1)) \not\subseteq S(\sigma(c + f + g))^G$. The

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

algorithm then proceeds to set $\sigma(x_1) := \sum(\{a,b,c,d,e,f,g\} \cap \{b,c,e,f,g\}) = b+c+e+f+g$.
- The third equation is not satisfied by σ modulo ACUIG:
  we have that $S(\sigma(a+c+e))^G = \{a,c,e\}$, and hence $S(\sigma(x_2)) \not\subseteq S(\sigma(a+c+e))^G$. The algorithm then proceeds to set $\sigma(x_2) := \sum(\{a,b,c,d,e,f\} \cap \{a,c,e\}) = a+c+e$.
- The first equation is not satisfied by σ modulo ACUIG:
  we have $S(\sigma(x_1)) = \{b,c,e,f,g\} \not\subseteq S(\sigma(g+x_2))^G = \{a,c,e,g\}$.
  The algorithm proceeds to set $\sigma(x_1) := \sum(\{b,c,e,f,g\} \cap \{a,c,e,g\}) = c+e+g$.

The algorithm then terminates since all equations are satisfied modulo ACUIG, and yields the substitution $\sigma = \{x_1 \mapsto c+e+g, x_2 \mapsto a+c+e\}$ as output, which is indeed an ACUIG-unifier of Γ that respects the constant restriction $\mathfrak{r}$.

Next, we provide an instance that admits no solution, and demonstrate how the algorithm reaches this conclusion.

Example 4.7. Consider the system of equations

$$\Gamma = \{a + x_2 =^? x_1,\ b + x_1 =^? c + f + g,\ c + x_2 =^? c + e\},$$

the set of ground identities $G = \{a+b+c=d,\ b+c+e=f\}$ considered in the previous example, and the constant restriction

$$\mathfrak{r}(g) = \emptyset.$$

Note that g is the only free constant occurring in Γ, and that $\mathfrak{r}(g) = \emptyset$ means that, in effect, there is no constant restriction, that is, all substitutions are admissible w.r.t. this constant restriction.

The algorithm begins by setting

$$\sigma(x_1) := a+b+c+d+e+f+g \text{ and } \sigma(x_2) := a+b+c+d+e+f+g.$$

Next, the algorithm enters the while-loop and picks in each iteration an equation that is not satisfied modulo ACUIG:

- The second equation is not satisfied by σ modulo ACUIG:
  we have $S(\sigma(c+f+g))^G = \{b,c,e,f,g\}$, and hence $\sigma(x_1) \not\subseteq S(\sigma(c+f+g))^G$.
  The algorithm then proceeds to set $\sigma(x_1) := \sum(\{a,b,c,d,e,f,g\} \cap \{b,c,e,f,g\}) = b+c+e+f+g$.
- The third equation is not satisfied by σ modulo ACUIG:
  we have that $S(\sigma(c+e))^G = \{c,e\}$, and hence $\sigma(x_2) \not\subseteq S(\sigma(c+e))^G$.
  The algorithm then proceeds to set $\sigma(x_2) := \sum(\{a,b,c,d,e,f,g\} \cap \{c,e\}) = c+e$.
- The first equation is not satisfied by σ modulo ACUIG:
  we have $S(\sigma(x_1)) = \{b,c,e,f,g\} \not\subseteq S(\sigma(a+x_2))^G = \{a,c,e\}$.
  The algorithm proceeds to set $\sigma(x_1) := \sum(\{b,c,e,f,g\} \cap \{a,c,e\}) = c+e$ and $\sigma(x_2) := \sum(\{c,e\} \cap \{b,c,e,f,g\}) = c+e$.
- The first equation is still not satisfied by σ modulo ACUIG, but we have $S(\sigma(x_1)) = \{c,e\} \subseteq S(\sigma(a+x_2))^G = \{a,c,e\}$ and $S(\sigma(x_2)) = \{c,e\} \subseteq S(\sigma(x_1))^G = \{c,e\}$. Hence the if condition is not satisfied and the algorithm returns **Fail**.

It is not hard to see that the problem indeed does not have an ACUIG-unifier. Basically, the first equation implies that, for any unifier σ, a or d must occur in $\sigma(x_1)$ since otherwise a cannot be produced on the right-hand side of the first equation. However, any such σ then violates the second equation.

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

Proposition 4.8. *Algorithm 1 terminates in time polynomial in the size of G and Γ. If Γ has an ACUIG-unifier w.r.t. $\mathfrak{r}$, then it provides such a unifier as output; otherwise it fails.*

Proof. Termination in polynomial time is an easy consequence of the facts that (i) in each iteration of the while-loop, at least one constant is removed from the image of a variable, or the loop is exited; and (ii) saturation can be done in polynomial time.

Since the algorithm only returns a substitution if the while-loop is exited regularly, this substitution satisfies all the equations of Γ modulo ACUIG. It satisfies the constant restriction due to the fact that the original substitution satisfies it and that constants are only removed from, but never added to, the image of variables during the run of the algorithm. Consequently, if the algorithm returns a substitution, then this substitution is an ACUIG-unifier of Γ w.r.t. $\mathfrak{r}$. This shows that the algorithm must return **Fail** in case Γ has no unifier w.r.t. $\mathfrak{r}$.

To prove the completeness of the algorithm, assume that $\widehat{\sigma}$ is a ACUIG-unifier of Γ w.r.t. $\mathfrak{r}$, and that the algorithm terminates during the rth iteration of the while-loop. Let $\sigma^{(0)}$ be the substitution σ before the first iteration of the while-loop. For $i \in \{1, \ldots, r-1\}$, let $\sigma^{(i)}$ be the substitution obtained at the end of the ith iteration of the while-loop.

We extend $\subseteq$ to substitutions in a natural way by using point-wise comparison of constant sets, that is, $\tau \subseteq \tau'$ iff $S(\tau(x)) \subseteq S(\tau'(x))$ for all $x \in X$. We prove by induction on i that $\widehat{\sigma} \subseteq \sigma^{(i)}$, for all $i \in \{0, \ldots, r-1\}$. The base case $i = 0$ is obvious: since $\widehat{\sigma}$ satisfies the constant restriction, we clearly have $\widehat{\sigma} \subseteq \sigma^{(0)}$. Let now $i \in \{0, \ldots, r-2\}$, and assume that we already know that $\widehat{\sigma} \subseteq \sigma^{(i)}$. We must prove $\widehat{\sigma} \subseteq \sigma^{(i+1)}$.

Since the algorithm does not exit the while-loop at this iteration, there is an equation $s =^? t$ in Γ that is not satisfied by $\sigma^{(i)}$ modulo ACUIG. In addition, since the algorithm does not fail at iteration i, there exists a variable x in s such that $S(\sigma(x)) \not\subseteq S(\sigma(t))^G$ or y in t such that $S(\sigma(y)) \not\subseteq S(\sigma(s))^G$. Clearly, for every $x \in X$ that does not appear in this equation, we have $S(\widehat{\sigma}(x)) \subseteq S(\sigma^{(i)}(x)) = S(\sigma^{(i+1)}(x))$. Let now x be a variable occurring in s (variables in t can be treated analogously). To prove that $S(\widehat{\sigma}(x)) \subseteq S(\sigma^{(i+1)}(x))$, it suffices to prove that $S(\widehat{\sigma}(x)) \subseteq S(\sigma^{(i)}(x))$ and that $S(\widehat{\sigma}(x)) \subseteq S(\sigma^{(i)}(t))^G$. The first statement is true by the induction hypothesis. Now, we have

$$S(\widehat{\sigma}(x)) \overset{(1)}{\subseteq} S(\widehat{\sigma}(s)) \overset{(2)}{\subseteq} S(\widehat{\sigma}(s))^G \overset{(3)}{=} S(\widehat{\sigma}(t))^G \overset{(4)}{\subseteq} S(\sigma^{(i)}(t))^G,$$

where (1) holds because x occurs in s, (2) by Lemma 4.3, (3) because $\widehat{\sigma}$ is a unifier of Γ, and (4) by Lemma 4.3 since $\widehat{\sigma} \subseteq \sigma^{(i)}$. This finishes the induction proof.

Therefore, we now know that $\widehat{\sigma} \subseteq \sigma^{(r-1)}$. There are two possible reasons for the algorithm terminating in the rth iteration. Either the while-loop is exited regularly or the algorithm returns **Fail**. In the first case, $\sigma^{(r-1)}$ is a unifier and the algorithm returns this substitution.

It remains to show that the second case cannot occur. In this case, we have that $S(\sigma^{(r-1)}(s))^G \neq S(\sigma^{(r-1)}(t))^G$ for some equation $s =^? t$ in Γ, but $S(\sigma^{(r-1)}(x)) \subseteq S(\sigma^{(r-1)}(t))^G$ for all variables x in s and $S(\sigma^{(r-1)}(y)) \subseteq S(\sigma^{(r-1)}(s))^G$ for all variables y in t. This can only be the case if there is a constant $c \in C$ such that c occurs in s, but $c \notin S(\sigma^{(r-1)}(t))^G$; or c occurs in t, but $c \notin S(\sigma^{(r-1)}(s))^G$. We show that this is impossible.

Thus assume that c occurs in s (the case where c occurs in t can be treated symmetrically). We have

$$c \overset{(1)}{\in} S(\widehat{\sigma}(s)) \overset{(2)}{\subseteq} S(\widehat{\sigma}(s))^G \overset{(3)}{=} S(\widehat{\sigma}(t))^G \overset{(4)}{\subseteq} S(\sigma^{(r-1)}(t))^G,$$

where (1) holds since c occurs in s, (2) by Lemma 4.3, (3) since $\widehat{\sigma}$ is a unifier of Γ, and (4) by Lemma 4.3 since $\widehat{\sigma} \subseteq \sigma^{(r-1)}$. □

The following theorem is an immediate consequence of Proposition 4.8.

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

Theorem 4.9. *The combined complexity and thus also the term complexity of ACUIG-unification with constant restriction is in P.*

Note that this implies that elementary ACUIG-unification and ACUIG-unification with constants are also decidable in polynomial time. The next section deals with general ACUIG-unification.

4.3 General ACUIG-unification

General ACUIG-unification problems Γ differ from the ones we have considered until now in that the terms used in Γ may contain free function symbols, that is, function symbols not occurring in the identities of ACUIG. For example, $\{f(x+a, a+b) =^? f(b+y, x)\}$ is such a general ACUIG-unification problem since it contains the additional function symbol f, which does not occur in the identities of ACUIG. Using techniques for the combination of unification algorithms, one can transfer complexity results for unification with constant restrictions to general unification. In fact, the following was proved by Baader and Schulz (1996) for arbitrary equational theories E, where linear constant restrictions are a special form of constant restrictions.

Theorem 4.10. (Baader and Schulz, 1996). *If solvability of E-unification problems with linear constant restriction is in NP, then solvability of general E-unification problems is also in NP.*

Together with Theorem 4.9 from the previous subsection, this provides us with an NP-upper bound for general ACUIG-unification. The corresponding lower bound can be obtained by adapting the proof of NP-hardness for general ACUI-unification by Kapur and Narendran (1992).

Theorem 4.11. *General ACUIG-unification is NP-complete, both w.r.t. combined complexity and w.r.t. term complexity.*

Proof. Membership in NP (for combined complexity, and thus also for term complexity) is an immediate consequence of Theorem 4.9 together with Theorem 4.10 since P $\subseteq$ NP and any linear constant restriction is a constant restriction.

We show that NP-hardness (w.r.t. term complexity) holds for any fixed finite set of ground identities G. This obviously implies NP-hardness also for combined complexity. This NP-hardness result can be shown by the same reduction from the set-matching problem used in (Kapur and Narendran, 1986) to show that general *ACI*-unification is NP-hard. To be more precise, this reduction yields *ACI*-unification problems of the form

$$\Gamma = \{g(s_1) + \ldots + g(s_m) =^? g(t_1) + \ldots + g(t_n)\},$$

where $+$ is an associative, commutative, and idempotent function symbol, g is a unary free function symbol, and the terms $s_1, \ldots, s_m, t_1, \ldots, t_n$ contain only free function symbols and variables. The presence of a unit and of ground identities in ACUIG does not change solvability of such problems compared to *ACI* since

- in the top-level sum the additional identities cannot be used due to the fact that all terms on this level start with the free function symbol g;
- even if the variables occurring in the terms $g(s_1), \ldots, g(s_m), g(t_1), \ldots, g(t_n)$ are replaced by terms containing $+$ and constant symbols from G (with the corresponding unifier being an ACUIG substitution, and not an *ACI* one), we can obtain an *ACI* unifier by abstracting away these "alien" subterms by free constants (see Lemma 4.1 in (Baader and Schulz, 1996) and Example 4.12 below).

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

This shows that such a problem Γ is solvable modulo *ACI* iff it is solvable modulo ACUIG for an arbitrary finite set of ground identities G, which completes our proof of NP-hardness of general ACUIG-unification w.r.t. term complexity. □

Example 4.12. The above argument of abstracting away alien subterms, that is, subterms whose root symbol is from a different signature, is well known in unification theory. We illustrate it here using a simple example. Consider the general *ACI*-unification problem

$$\Gamma = \{g(x) + g(f(y)) + g(z) =^? g(z) + g(y)\}$$

and the ground theory $G = \{a + b = a + b + c, b + c = a + b + c\}$. If we view Γ as an ACUIG-unification problem, then it is easy to see that the substitution $\sigma = \{x \mapsto a + b,\ y \mapsto b + c, z \mapsto f(a + b)\}$ is an ACUIG-unifier of Γ. Note that σ replaces the variables occurring in Γ by terms containing $+$ and constant symbols from G, and is thus not an *ACI*-unifier of Γ. We obtain an *ACI*-unifier from σ by replacing the terms $a + b$ and $b + c$, which are equivalent modulo ACUIG, by the same free constant d. This yields the substitution $\sigma' = \{x \mapsto d,\ y \mapsto d, z \mapsto f(d)\}$, which clearly solves Γ modulo *ACI*.

5. Approximate Unification Modulo ACUIG

In this section, we investigate how the addition of a ground theory to ACUI affects the complexity of approximate unification. Basically, we will investigate the same three measures that we treated in Section 3, but must adapt their definition to the extended setting.

First, one needs to decide which are the constants that can lead to violations. In Section 3, these were the free constants occurring in the problem since presence or absence of the theory constant **0** cannot contribute to the violation of an equation. For example, given the equation $x + a =^? a$, the substitution $\sigma = \{x \mapsto \mathbf{0}\}$ solves this equation, although syntactically, after applying this substitution, the left-hand side $\mathbf{0} + a$ contains **0** whereas the right-hand side a does not. This is the reason why we did not consider **0** when defining the set of constants $S(t)$ contained in a term t. In contrast, constants occurring in G may lead to a violation. For instance, consider the equation from above and the ground theory $G = \{b + c = b\}$. The substitution $\theta = \{x \mapsto b\}$ does not solve the equation since $\theta(x + a) = b + a \neq_{\mathsf{ACUIG}} a = \theta(a)$. Here, the constant b, which is a theory constant for ACUIG, violates the equation. This motivates to consider both the free constants occurring in Γ and the theory constants different from **0** occurring in G when counting violations. However, when defining under what conditions such a constant violates an equation, one cannot just consider the constants explicitly occurring on the left- and the right-hand side after applying the substitution. In fact, if we consider the equation $x + c =^? b$, then the substitution θ from above solves this equation, although c occurs in $\theta(x + c) = b + c$, but not in $\theta(b) = b$. To see whether a theory constant from G really violates an equation, we first need to apply saturation to the sets of constants occurring on each side of the equation.

In the following, let G be a ground theory, that is, a finite set of ground identities between terms of $T_\Sigma(F)$, and $\Gamma = \{s_1 =^? t_1, \ldots, s_k =^? t_k\}$ an ACUIG-unification problem with constants. We denote the set of constants appearing in Γ or G with C and the set of variables appearing in Γ with X. As before, it is sufficient to consider only substitutions that are the identity on $V \setminus X$ and assign terms in $T_\Sigma(C)$ to variables in X.

Definition 5.1. *The substitution* σ violates *the equation* $s =^? t \in \Gamma$ modulo G *if*

$$S(\sigma(s))^G \neq S(\sigma(t))^G.$$

A constant $a \in C$ violates $s =^? t \in \Gamma$ modulo G w.r.t. σ *if*

$$a \in S(\sigma(s))^G \,\Delta\, S(\sigma(t))^G,$$

and it violates Γ modulo G w.r.t. σ *if it violates at least one equation in* Γ *modulo* G *w.r.t.* σ.

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

A constant $a \in C$ violates Γ modulo G p times w.r.t. σ *if it violates p equations in Γ modulo G w.r.t. σ. Finally, we say that σ* violates Γ q times modulo G *if $q = \sum_{a \in C} p_a$ where, for each $a \in C$, the element a violates Γ modulo G p_a times w.r.t. σ.*

Given this redefined notions of violation, we can now formalize the three different decision problems for approximate ACUIG-unification analogously to how this was done in Section 3 for ACUI.

Definition 5.2. *Let G be a ground theory, that is, a finite set of ground identities between terms of $T_\Sigma(F)$. Given an ACUIG-unification problem with constants Γ and a nonnegative integer ℓ, we ask whether there exists a substitution σ such that:*

- *at most ℓ of the equations of the system are violated modulo G by σ. We call this decision problem* MinVEq-ACUIG.
- *at most ℓ constants violate Γ modulo G w.r.t. σ. We call this decision problem* MinVEl-ACUIG.
- *σ violates Γ modulo G at most ℓ times. We call this decision problem* MinV-ACUIG.

It is easy to see that all three problems belong to NP w.r.t. combined complexity, and thus also w.r.t. term complexity. In fact, as mentioned before, we can restrict the attention to substitutions that assign terms in $T_\Sigma(C)$ to the variables in X. Recall that, modulo ACUI, terms with the same constants are equivalent. Hence, for every $x \in X$, in order to define the assigned term $\sigma(x)$ it suffices to determine the set of constants $S(\sigma(x))$ occurring in it. Consequently, any such substitution can be guessed in nondeterministic polynomial time. For a given substitution, the saturations of the constant sets on the left- and right-hand sides of the equations can be computed in polynomial time due to Proposition 4.4, and thus the violations modulo G can be counted also in polynomial time. The only place where the ground theory plays a role in this straightforward NP-procedure is during the computation of the saturations. Since this computation is polynomial w.r.t. combined complexity, the NP upper bound also holds for combined complexity.

Proposition 5.3. *The problems MinVEq-ACUIG, MinVEl-ACUIG, and MinV-ACUIG are in NP w.r.t. combined complexity, and thus also w.r.t. term complexity.*

In the next subsection, we will show that, for MinVEq-ACUIG and MinV-ACUIG, this NP upper bound is optimal, by showing NP-hardness w.r.t. term complexity for any fixed finite set of ground identities G. Note that the corresponding two problems are also NP-hard for ACUI, and actually MinVEq-ACUI and MinV-ACUI are used in the reduction that establishes these hardness results.

Analyzing the complexity of MinVEl-ACUIG turns out to be more subtle. Recall that the corresponding problem is in P for ACUI. We will show that this upper bound in general does not transfer from ACUI to ACUIG. In fact, we will exhibit a fixed finite ground theory G for which the problem MinVEl-ACUIG is NP-hard w.r.t. term complexity. Obviously, this implies that MinVEl-ACUIG is NP-hard w.r.t. combined complexity. However, there are fixed finite ground theories G for which MinVEl-ACUIG is in P w.r.t. term complexity. An obvious example is $G = \emptyset$ since we know that MinVEl-ACUI is in P, but we will also give some other examples of such theories.

5.1 The problems MinVEq-ACUIG and MinV-ACUIG

We show NP-hardness of MinVEq-ACUIG and MinV-ACUIG by reduction from MinVEq-ACUI and MinV-ACUI, respectively.

Lemma 5.4. *Let G be a finite ground theory. Then MinVEq-ACUIG and MinV-ACUIG are NP-hard w.r.t. term complexity.*

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

Proof. We restrict the attention to MinVEq-ACUIG (since MinV-ACUIG can be treated similarly), and reduce MinVEq-ACUI to MinVEq-ACUIG. Thus, let G be a fixed finite set of ground identities. Given an ACUI-unification problem Γ, we can assume without loss of generality that Γ contains none of the constants occurring in G (otherwise, we just rename the free constants in Γ). We now view Γ as ACUIG-unification problem.

To show correctness of the reduction, first assume that σ is a substitution that uses only constants occurring in Γ, and which violates at most ℓ equations of Γ. Equations $s =^? t \in \Gamma$ that are not violated satisfy $\sigma(s) =_{\mathsf{ACUI}} \sigma(t)$, and thus also $\sigma(s) =_{\mathsf{ACUIG}} \sigma(t)$. By Proposition 4.4, this implies that these equations are also not violated modulo G by σ. Thus, if we view Γ as ACUIG-unification problem, then σ violates at most ℓ equations of Γ modulo G.

Second, assume that σ is a substitution that uses only constants occurring in Γ and G, and which violates at most ℓ equations of Γ modulo G. Consider the substitution σ' that is obtained from σ by replacing every constant from G by $\mathbf{0}$. We claim that all equations not violated modulo G by σ are not violated by σ'. Thus, assume that $s =^? t \in \Gamma$ is an equation not violated by σ, which means that

$$S(\sigma(s))^G = S(\sigma(t))^G. \tag{3}$$

To show that this implies $S(\sigma'(s)) = S(\sigma'(t))$, assume that $c \in S(\sigma'(s))$. By our construction of σ' and the fact that theory constants from G do not occur in Γ, we know that c does not occur in G. In addition, $c \in S(\sigma'(s))$ implies $c \in S(\sigma(s))^G$, and thus (3) yields $c \in S(\sigma(t))^G$. Since saturation w.r.t. G can only add theory constants from G, we obtain $c \in S(\sigma(t))$. Thus, c is contained in s or introduced by σ. The latter implies that c is also introduced by σ' since only theory constants are removed when going from σ to σ'. Thus, we have shown that $S(\sigma'(s)) \subseteq S(\sigma'(t))$. Since the other inclusion can be shown analogously, this shows that $s =^? t$ is not violated by σ'. Consequently, σ' violates at most ℓ equations of Γ. □

Together with Proposition 5.3, this lemma yields the following complexity results.

Theorem 5.5. *The problems MinVEq-ACUIG and MinV-ACUIG are NP-complete both w.r.t. term complexity and w.r.t. combined complexity.*

5.2 The problem MinVEl-ACUIG

For the setting where the number of violating elements is minimized, the situation is less clear. Proposition 5.3 yields an NP upper bound for combined complexity. However, since MinVEl-ACUI is in P, a reduction from it would not yield a matching NP lower bound for MinVEl-ACUIG. One could try to adapt the polynomial-time algorithm for MinVEl-ACUI to MinVEl-ACUIG. Recall that this algorithm is based on considering projections of the equations in Γ to the free constants, and then solving the projected equations separately. While this would still work for the free constants in an ACUIG-unification problem, we now also need to consider the theory constants from G. For these, the separation into different equations does not work since the addition of constants by saturation would not be taken into account. This problem is illustrated in the next example.

Example 5.6. Consider the system of equations

$$\Gamma = \{a + x_2 =^? x_1,\ b + x_1 =^? c + f + g,\ c + x_2 =^? c + e\},$$

and the set of ground identities

$$G = \{a + b + c = d,\ b + c + e = f\}$$

from Example 4.7. As shown in that example, Γ does not have an ACUIG-unifier. Thus, it makes sense to look for approximate unifiers.

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

If we tried to solve the projected systems of equations (as described in Section 3.2) independently for each constant modulo ACUIG, then we would see that 5 out of the 7 systems are not solvable: the systems for a, b, e, f, g are not solvable, whereas the ones for c, d are. This yields the substitution $\theta = \{x_1 \mapsto c, x_2 \mapsto c\}$, w.r.t. which five constants violate Γ (see the proof of Lemma 3.10), and it also happens that only these five constants violate Γ modulo G.

However, by looking at the projections separately, we partially lose the possibility to prevent a violation by using G to add the violating constant to the side where it does not occur syntactically. For this, other constants than the violating one may need to be present. For instance, for the substitution $\{x_1 \mapsto c + e, x_2 \mapsto \mathbf{0}\}$ the constant f is no longer violating (even though the projection of Γ onto f is not solvable). Conversely, constants added by saturation may increase the number of violating constants, and their addition may depend on the presence of several constants. If, for instance, Γ also included the trivially unsatisfiable equation $f \stackrel{?}{=} \mathbf{0}$, f would definitely be a violating element. Due to the second identity, the constants b, c, e would be violating as well. However, when projecting onto these constants, we would obtain the harmless equation $\mathbf{0} \stackrel{?}{=} \mathbf{0}$.

Returning to our original example, we can actually find a substitution w.r.t. which only two constants violate Γ. If we set $\sigma(x_1) := \sigma(x_2) := c + e$, then the first equation is violated by a, and the second by g, while the third is not violated at all. Thus, w.r.t. σ there are only two violating constants. The reason is that b, e, f do not violate the second equation due to saturation w.r.t. G: $\sigma(b + x_1) = b + c + e$ and $\sigma(c + f + g) = c + f + g$, but saturation yields $S(\sigma(b + x_1))^G = \{b, c, e, f\}$ and $S(\sigma(c + f + g))^G = \{b, c, e, f, g\}$, and thus only g violates this equation modulo G.

As illustrated by this example, the approach used for MinVEl-ACUI to get a polynomial-time decision procedure in general does not work if we add ground identities. We will now show that, in fact, there is a fixed finite set of ground identities for which MinVEl-ACUIG is NP-hard w.r.t. term complexity. This obviously implies that the problem is also NP-hard for combined complexity. We will use a reduction from the NP-complete problem of 3-colorability (Garey and Johnson, 1990) to prove the hardness result.

Definition 5.7. (3-colorability).

Given: *A finite graph $H = (X, E)$, where X is the set of vertices and $E \subseteq X \times X$ is the set of edges of H.*

Question: *Does there exist a* proper coloring *of H with three colors (say a, b, c), that is, an assignment $\sigma : X \to \{a, b, c\}$ such that $\sigma(x) \neq \sigma(y)$ for every $(x, y) \in E$?*

The fixed ground theory G_{3c} employed to encode the "nature" of 3-colorability uses, as constants, the colors a, b, c and additionally e and f. Let C be the set of these five constants. The theory G_{3c} then consists of the following identities:

- for every $i, j \in \{a, b, c\}$, $i \neq j$

$$i + j + e = f. \tag{4}$$

We will explain the intuition underlying these identities once we have introduced the ACUIG_{3c}-unification problem. For the moment, just note that G_{3c} in fact does not depend on the input graph, that is, it is fixed in the sense that the same theory G_{3c} is used for every input graph.

The input graph $H = (X, E)$ is represented by the unification problem, where the vertices from X are used as variables. To be more precise, given an input graph $H = (X, E)$, the corresponding ACUIG_{3c}-unification problem Γ_H consists of the following equations:

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

– for every $x \in X$

$$x + e \stackrel{?}{=} \mathbf{0}, \tag{5}$$

– for every $(x, y) \in E$

$$x + y + e \stackrel{?}{=} f, \tag{6}$$

– finally, Γ_H also contains the equations

$$a + b + c \stackrel{?}{=} \mathbf{0}, \tag{7}$$

$$e \stackrel{?}{=} \mathbf{0}. \tag{8}$$

At first sight, it may seem strange to have ground equations as part of a unification problem since they either hold modulo the given equational theory or are violated, independently of what substitution is used. However, in the context of MinVEl-ACUIG$_{3c}$, such equations can have a relevant effect. In fact, the equations (7) and (8) force all elements except f to be violating. Intuitively, the question is then whether we can keep f from being violating. If this is the case, then the identities of G_{3c} together with equation (5) ensure that every $x \in X$ is assigned at most one of a, b, c. Furthermore, together with equation (6) the identities of G_{3c} ensure that x and y are assigned at least two different colors, if $(x, y) \in E$. Combining these two observations, we see that every variable (i.e., vertex) belonging to an edge is assigned a unique color, and that vertices that are connected by an edge are assigned different colors.

Lemma 5.8. *The graph $H = (X, E)$ has a proper coloring iff there is a substitution σ such that at most four constants violate the ACUIG$_{3c}$-unification problem Γ_H modulo G_{3c} w.r.t. σ.*

Proof. First, note that there is a substitution σ such that at most four constants violate the ACUIG$_{3c}$-unification problem Γ_H modulo G_{3c} w.r.t. σ iff the constant f does not violate Γ_H modulo G_{3c}.

Now, assume that $H = (X, E)$ has a proper coloring, that is an assignment $\sigma : X \to \{a, b, c\}$ such that $\sigma(x) \neq \sigma(y)$ for every $(x, y) \in E$. Note that σ can also be viewed as a substitution. We claim that f does not violate Γ_H modulo G_{3c} w.r.t. this substitution. To see this, let us go through the equations of Γ_H:

– Equations of the form (5): such an equation $x + e \stackrel{?}{=} \mathbf{0}$ could only be violated by f modulo G_{3c} w.r.t. σ if an identity of the form (4) produced f on the left-hand side; this is not possible since $\sigma(x)$ contains only one of the constants from $\{a, b, c\}$ and not two different ones.
– Equations of the form (6): for every $(x, y) \in E$, since $\sigma(x), \sigma(y) \in \{a, b, c\}$ and $\sigma(x) \neq \sigma(y)$, one of the identities (4) ensures that f does not violate the equation $x + y + e \stackrel{?}{=} f$ modulo G_{3c} w.r.t. σ.
– f clearly does not violate the equations (7) and (8) modulo G_{3c}.

Conversely, assume that $\sigma : X \to T_\Sigma(C)$ is a substitution such that f does not violate Γ_H modulo G_{3c} w.r.t. σ. Since, for every $x \in X$, f does not violate equation (5), we know that f does not appear in $\sigma(x)$. Furthermore, $\sigma(x)$ contains no more than one of a, b, c; otherwise, an identity of the form (4) would cause f to be violating. Moreover, since for every $(x, y) \in E$, f does not violate equation (6), we know that one of the identities (4) was activated, and hence $\sigma(x)$ and $\sigma(y)$ contain at least two different colors. Paired with the result for equations of the form (5), every variable belonging to an edge is assigned exactly one color. Overall, we can thus define the coloring τ by setting $\tau(x)$ to be the unique element of $\{a, b, c\}$ that appears in $\sigma(x)$ for vertices that belong to an edge, and arbitrarily set $\tau(x)$ for the remaining vertices. Furthermore, by the fact that f does not

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

violate any equation of the form (6), we also obtain that τ is a proper coloring of H. This completes the proof of the lemma. □

This lemma shows that MinVEl-ACUIG$_{3c}$ is NP-hard w.r.t. term complexity. Together with Proposition 5.3, we thus obtain the following theorem.

Theorem 5.9. *There exists a ground theory G_{3c} for which MinVEl-ACUIG$_{3c}$ is NP-complete w.r.t. term complexity. This implies that MinVEl-ACUIG is NP-complete w.r.t. combined complexity.*

Note, however, that there are finite ground theories G for which MinVEl-ACUIG is in P w.r.t. term complexity. In fact, our results in Section 3.2, in particular Theorem 3.11, show that the empty ground theory $G = \emptyset$ is such a theory.

Similarly, if we consider a theory G containing only identities of the form $a = b$ for $a, b \in C$ for a finite set of constants C, then MinVEl-ACUIG is in P w.r.t. term complexity. Basically, the idea is to reduce MinVEl-ACUIG to MinVEl-ACUI by replacing all occurrences of equivalent constants by a single representative of the class. In the resulting system of equations, there is no interaction between the different constants occurring in it, and hence it can be treated as a MinVEl-ACUI problem. An optimal substitution for this instance is also optimal for the original one, although the actual number of violating elements modulo G may be higher in the original problem. In fact, if b is the chosen representative of an equivalence class of constants, and b violates an equation in the MinVEl-ACUI instance, then all the elements of the class violate this equation modulo G.

In order to provide less trivial examples of ground theories G for which MinVEl-ACUIG can be decided in polynomial time, we restrict the syntactic form of the ground identities that may occur in G.

Definition 5.10. *The finite set of ground identities is called* unary *if every identity in G is of the form $a = a + t$ for a constant $a \in F \cup \{\mathbf{0}\}$ and a ground term t.*

Note that unary theories actually also cover the case of identities of the form $a = b$. In fact, it is easy to see that the theory $G_i = \{a = b\}$ is equivalent to the unary theory $G_u = \{a = a + b, b = b + a\}$ in the presence of ACUI:

- we have $a =_{\mathsf{ACUIG}_u} a + b =_{\mathsf{ACUIG}_u} b + a =_{\mathsf{ACUIG}_u} b$,
- and $a =_{\mathsf{ACUIG}_i} a + a =_{\mathsf{ACUIG}_i} a + b$ as well as $b =_{\mathsf{ACUIG}_i} b + b =_{\mathsf{ACUIG}_i} b + a$.

The name "unary" for theories satisfying the above definition stems from the fact that, for such theories, the saturation rules are unary in the sense that their applicability depends on the presence of at most a single constant. Indeed, an identity of the form $a = a + t$ induces a rule that is applicable whenever $a \in F$ occurs in a given set of constants (or every time, if $a = \mathbf{0}$), and its effect is to add the constants occurring in t to this set (unless they are already there).

Lemma 5.11. *Let G be a unary ground theory and $s, t \in T_\Sigma(F)$. Then $S(s + t)^G = S(s)^G \cup S(t)^G$. In particular, this implies that $S(s)^G = \bigcup_{a \in S(s)} \{a\}^G$.*

Proof. The inclusion from right to left follows from Lemma 4.3, and thus holds in general (i.e., also for non-unary theories): $S(s)^G \cup S(t)^G \subseteq (S(s) \cup S(t))^G = S(s + t)^G$.

To show the other direction, assume that $a \in S(s + t)^G$. We show $a \in S(s)^G \cup S(t)^G$ by induction on the number of saturation steps needed to add a to $S(s + t)^G$. In the base case, we have $a \in S(s + t) = S(s) \cup S(t) \subseteq S(s)^G \cup S(t)^G$. Thus, assume that a is added in step n of the saturation process, and that all the constants that have been added previously are contained in $S(s)^G \cup S(t)^G$.

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

By definition of saturation, there is a constant $b \in F \cup \{\mathbf{0}\}$ and an identity $b = b + u \in G$ with $a \in S(u)$. If $b = \mathbf{0}$, then clearly $a \in S(s)^G$ (and also $a \in S(t)^G$), since $S(s)^G$ ($S(t)^G$) is saturated. Otherwise, if $b \in F$, it holds that $b \in S(s)^G \cup S(t)^G$ by the induction hypothesis. If $b \in S(s)^G$, then the existence of the identity $b = b + u$ in G and the fact that $S(s)^G$ is saturated imply that $a \in S(s)^G$. In the same way, $b \in S(t)^G$ implies that $a \in S(t)^G$. □

Note that this lemma need not hold for non-unary ground theories. An easy counterexample is $G = \{a + b = c\}$, where $c \in S(a + b)^G$, but $c \notin S(a)^G \cup S(b)^G$. Also note that the theory G_{3c} used in the proof of Theorem 5.9 does not satisfy this property. In fact, we have $S(a + b + e)^{G_{3c}} = \{a, b, e, f\}$, but $S(a + b)^{G_{3c}} \cup S(e)^{G_{3c}} = \{a, b\} \cup \{e\}$.

Given a unary ground theory G, we will show that MinVEl-ACUIG can be reduced to MinVEl-ACUI. Before we can describe this reduction, we need to extend the notion of saturation from sets of constants to terms, substitutions, and unification problems. Given a ground term $s \in T_\Sigma(F)$, its saturation is $s^G := \sum (S(s)^G)$. Note that we have $S(s^G) = S(s)^G$ for any $s \in T_\Sigma(F)$, and that Lemma 5.11 yields $(s_1 + \ldots + s_n)^G =_{\mathsf{ACUIG}} s_1^G + \ldots + s_n^G$ for $s_1, \ldots, s_n \in T_\Sigma(F)$.

For a ground substitution $\sigma : X \to T_\Sigma(F)$, we define its saturation σ^G as $\sigma^G(x) := \sigma(x)^G$ for every $x \in X$. Recall that a term $t \in T_\Sigma(F, V)$ can be written in the form $t = t_0 + x_1 + \ldots + x_n$, where $t_0 \in T_\Sigma(F)$ and $x_1, \ldots, x_n \in V$. We set $t^G := t_0^G + x_1 + \cdots + x_n$. Finally, given an ACUIG-unification problem with constants Γ, its saturation Γ^G consists of the equations $s^G =^? t^G$ for every $s =^? t$ in Γ.

The following lemma is an easy consequence of Lemma 5.11.

Lemma 5.12. *Let G be a unary ground theory, $t \in T_\Sigma(F)$, and σ a ground substitution. Then $S(\sigma^G(t^G)) = S(\sigma(t))^G$.*

Proof. Let t be of the form $t = t_0 + x_1 + \cdots + x_n$ for a ground term t_0. Then we have $S(\sigma^G(t^G)) = S(t_0^G + \sigma^G(x_1) + \cdots + \sigma^G(x_n)) = S(t_0^G + \sigma(x_1)^G + \cdots + \sigma(x_n)^G) = S(\sigma(t)^G) = S(\sigma(t))^G$. □

The idea is now to reduce MinVEl-ACUIG for Γ to MinVEl-ACUI for Γ^G. The following theorem states correctness of this reduction.

Theorem 5.13. *Let G be a unary ground theory, Γ an ACUIG-unification problem, and $\ell \geq 0$. Then the following are equivalent:*

(1) *There is a substitution σ such that at most ℓ constants violate Γ modulo G w.r.t. σ.*
(2) *There is a substitution θ such that at most ℓ constants violate Γ^G w.r.t. θ.*

Proof. Assume initially that there exists a substitution σ w.r.t. which Γ has at most ℓ violating elements modulo G. It is easy to see that σ^G is a substitution w.r.t. which Γ^G has at most ℓ violating elements. In fact, Lemma 5.12 yields for every $s =^? t \in \Gamma$

$$S(\sigma^G(s^G)) \,\Delta\, S(\sigma^G(t^G)) = S(\sigma(s))^G \,\Delta\, S(\sigma(t))^G,$$

and hence Γ^G (viewed as a MinVEl-ACUI instance) has at most ℓ violating constants w.r.t. σ^G.

For the opposite direction, assume that there exists a substitution θ w.r.t. which Γ^G has at most ℓ violating elements (not considering G). Recall that, in Lemma 3.10.3, when constructing an optimal substitution for a MinVEl-ACUI problem, we only use non-violating elements. Hence, we can assume without loss of generality that θ introduces no violating constants. Under this assumption, we will now show that, also modulo G, Γ has at most ℓ violating elements w.r.t. θ. Indeed, assume that a does not violate Γ^G w.r.t. θ. Then, for every $s =^? t \in \Gamma$ we have that $a \notin S(\theta(s^G)) \,\Delta\, S(\theta(t^G))$.

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

- If $a \in S(\theta(s^G)) \cap S(\theta(t^G))$, we obtain by Lemma 5.12 that $a \in S(\theta(s))^G \cap S(\theta(t))^G$, and hence a does not violate $s =^? t$ modulo G.
- If $a \notin S(\theta(s^G)) \cup S(\theta(t^G))$, it is still possible that $a \in S(\theta(x)^G)$ for some x occurring in s (the case where x occurs in t is treated similarly), and hence $a \in S(\theta(s))^G$. By Lemma 5.11, we obtain that there exists some $b \in S(\theta(x))$ such that $a \in \{b\}^G$. Since we assume without loss of generality that θ does not introduce any violating elements, b is also not violating. Hence, b occurs in $\theta(t^G)$, and thus also in $\theta(t)^G$, which again by Lemma 5.11 implies that $a \in S(\theta(t))^G$. If, on the other hand, $a \notin S(\theta(x))^G$ for all variables x occurring in s or t, then we have $a \notin S(\theta^G(s^G)) \cup S(\theta^G(t^G))$, which by Lemma 5.12 is the same as $a \notin S(\theta(s))^G \cup S(\theta(t))^G$.

In any case, we conclude that $a \notin S(\theta(s))^G \,\Delta\, S(\theta(t))^G$, and hence a does not violate $s =^? t$ modulo G. To sum up, we have shown that every non-violating element of Γ^G w.r.t. θ remains non-violating modulo G for Γ w.r.t. θ, and thus the upper bound of ℓ for violating elements also holds in Γ modulo G. □

Since saturation can be done in polynomial time (even if G is seen as part of the input) and MinVEl-ACUI can be solved in polynomial time, this theorem yields the following complexity result for the class of unary ground theories.

Corollary 5.14. *Restricted to unary ground theories G, MinVEl-ACUIG is in P w.r.t. combined complexity, and thus also w.r.t. term complexity.*

Obviously, all we need for Theorem 5.13 to hold is that the property stated in Lemma 5.11 holds for G. Let us call such a theory "well behaved."

Definition 5.15. *The ground theory G is called* well behaved *if* $S(s+t)^G = S(s)^G \cup S(t)^G$ *holds for all terms* $s, t \in T_\Sigma(F)$.

By Lemma 5.11, every unary theory is well behaved. From a syntactic point of view, the converse need not be true. However, semantically it is true: every well-behaved theory G is equivalent to a unary theory G', where G and G' are *equivalent* if $=_{\mathsf{ACUIG}} = =_{\mathsf{ACUIG}'}$.

Lemma 5.16. *A ground theory is well behaved iff it is equivalent to a unary theory.*

Proof. The if direction is an immediate consequence of Lemma 5.11. To prove the only-if direction, assume that G is a well-behaved theory. We define G' as

$$G' := \{a = a + b \mid a \in F(G) \cup \{\mathbf{0}\}, b \in F(G),\ a \neq b,\ \text{and } a =_{\mathsf{ACUIG}} a + b\},$$

where $F(G)$ denotes the constants from F occurring in G.

Clearly, G' is a unary theory. To show that G and G' are equivalent, it is sufficient to show that $S(s)^G = S(s)^{G'}$ holds for all terms $s \in T_\Sigma(F)$, by Proposition 4.4. Since both G and G' are well behaved (G by assumption and G' since it is unary), this is the case if $S(a)^G = S(a)^{G'}$ holds for all constants $a \in F \cup \{\mathbf{0}\}$.

First, assume that $b \in S(a)^G$. If $b = a$, then $b = a \in S(a)^{G'}$. Otherwise, $b \in S(a)^G$ implies that $b \in F(G)$ since only constants from G can be added by saturation. Now we have $a =_{\mathsf{ACUIG}} \sum (S(a)^G) =_{\mathsf{ACUIG}} \sum (S(a)^G) + b =_{\mathsf{ACUIG}} a + b$. Note that the first identity in the above derivation is an easy consequence of Proposition 4.4 since $S(a)^G = S(a)^{GG} = S(\sum (S(a)^G))^G$. The second identity uses idempotency and the fact that b is a summand in $\sum (S(a)^G)$ since $b \in S(a)^G$. If $a \in F(G) \cup \{\mathbf{0}\}$, this shows that $a = a + b \in G'$, and thus we also have $b \in S(a)^{G'}$. Otherwise, it

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

is easy to see that we must have $\mathbf{0} =_{\mathsf{ACUIG}} b =_{\mathsf{ACUIG}} \mathbf{0} + b$, which yields $\mathbf{0} = \mathbf{0} + b \in G'$, and thus $b \in S(a)^{G'}$.

Conversely, assume that $b \in S(a)^{G'}$. Again, this implies that $a =_{\mathsf{ACUIG}'} a + b$. By the definition of G', we have $G' \subseteq =_{\mathsf{ACUIG}}$ and thus $=_{\mathsf{ACUIG}'} \subseteq =_{\mathsf{ACUIG}}$, which yields $a =_{\mathsf{ACUIG}} a + b$. Clearly, this implies $b \in S(a)^G$. □

6. Conclusion

In this paper, we have extended ACUI-unification in two directions. On one hand, we have considered approximate ACUI-unification w.r.t. three different ways of measuring the degree to which the equations of the unification problem are violated by a given substitution that is not a unifier. For two of these measures, the complexity of the associated decision problem increases from P to NP-complete, whereas for one of them it stays in P. On the other hand, we have extended ACUI-unification to ACUIG-unification, that is, unification in equational theories that are obtained from ACUI by adding a finite set G of ground identities. We were able to show that adding such identities does not change the complexity of the unification problem. Finally, we have combined the two extensions, that is, we have investigated approximate ACUIG-unification. For the measures for which already approximate ACUI-unification is NP-complete, the same holds for approximate ACUIG-unification. For the third measure, which counts the number of violated elements, the situation turns out to be more interesting. We were able to show that there is a finite set G of ground identities such that approximate ACUIG-unification for this fixed theory is NP-complete. But we have also introduced the class of unary ground theories G for which approximate ACUIG-unification is in P. It would be interesting to see whether there exist ground theories that are not equivalent to unary theories, but for which approximate ACUIG-unification is still in P.

These results are of interest for unification theory. In fact, while it was known for quite some time that adding finite sets of ground identities to certain theories such as ACUI leaves the word problem decidable (Marché, 1996; Narendran and Rusinowitch, 1996), no such preservation result was available for unification. In addition, there are only very few results on approximate unification. In (Iranzo and Rubio-Manzano, 2015), the authors introduce what they call proximity-based unification, which basically is an extension of syntactic unification to the approximate case, and use it in the context of fuzzy logic programming. Set unification, which is basically unification modulo ACUI, has drawn considerable attention (Dovier et al., 2006) due to its applications in various areas, such as deductive databases, theorem proving, and static program analysis, and it is conceivable that these applications could also profit from approximate variants. Our results show that the complexity of approximate unification strongly depends on measures used to define the degree to which the equations of the unification problem are violated: while for some measures going from exact unification to approximate unification increases the complexity, it stays the same for other measures. We have no example of nontrivial measures where the complexity decreases in the approximate case, though this is also conceivable. Adding ground identities in the approximate case makes things even more interesting since it may now depend on the ground theory whether the complexity increases or not when going to the approximate case. In the setting of the violated elements measure considered in this paper, where this situation occurs, it would be interesting to see whether one can show a dichotomy result, that is, whether one can prove that, depending on which ground theory G is used, the complexity of approximate ACUIG-unification is either in P or NP-complete. If this is actually the case, the next step would be to attempt a classification of the two cases, that is, come up with conditions that ensure membership in P or NP-completeness.

In addition to the fact that ACUI is a theory investigated extensively in unification theory and in set unification, our interest in this theory is motivated by the fact that this theory is a common subtheory of the equational theories corresponding to the Description Logics $\mathcal{FL}_0$ (Baader, 1996) and

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

$\mathcal{EL}$ (Baader et al., 1999). Unification in DLs (Baader and Morawska, 2009; Baader and Narendran, 2001) was introduced as a novel inference problem that can be used to detect redundancies in knowledge bases, and approximate variants (Baader et al., 2016) were investigated in order to increase the recall of these approaches. The decidability and complexity results for unification in $\mathcal{FL}_0$ and $\mathcal{EL}$ respectively shown in (Baader and Narendran, 2001) and (Baader and Morawska, 2009) are restricted to the setting without a background TBox. The big open problem in this area is whether these results can be extended to unification in the presence of general TBoxes, which corresponds to adding a finite set of ground identities to the corresponding equational theory. While a negative result for ACUIG would have implied negative results for the two DLs, the methods used to show the positive results for ACUIG are too simple to be useful for $\mathcal{FL}_0$ and $\mathcal{EL}$. For $\mathcal{EL}$, there are some positive results for the case where the TBox satisfies certain cycle-restrictions (Baader et al., 2012), while for $\mathcal{FL}_0$ there are only positive results for matching in the presence of TBoxes (Baader et al., 2018), but none for unification.

Approximate unification has been considered for $\mathcal{FL}_0$ (Baader et al., 2016), but with measures that differ from the ones employed in the present paper for ACUI and ACUIG. Basically, instead of constants, the sets that one needs to consider there are sets of words, and the measures employed in (Baader et al., 2016) take the length of these words into account: violations caused by long words are counted as less important than violations caused by shorter words. In addition, in (Baader et al., 2016) only a single equation is considered, and thus measures counting the number of violated equations have not been considered there. It would be interesting to see how approximate unification w.r.t. the measures considered in the present paper behaves for $\mathcal{FL}_0$. For $\mathcal{EL}$, there are, to the best of our knowledge, no results on approximate unification. Here it would be interesting to see whether one can use similarity measures on $\mathcal{EL}$ concepts (Lehmann and Turhan, 2012) to define appropriate measures for approximate unification in $\mathcal{EL}$, and whether approximate $\mathcal{EL}$-unification w.r.t. such measures is decidable. The results of the present paper show that the complexity of approximate unification modulo ACUI and ACUIG strongly depends on which measures are used, and which ground theories are considered. We conjecture that the same is true for unification in DLs.

References

Baader, F. (1996). Using automata theory for characterizing the semantics of terminological cycles. *Ann. of Mathematics and Artificial Intelligence* **18**, 175–219.

Baader, F., Borgwardt, S. and Morawska, B. (2012). Extending unification in $\mathcal{EL}$ towards general TBoxes. In: *Proceedings of the 13th International Conference on Principles of Knowledge Representation and Reasoning (KR 2012)*, AAAI Press/The MIT Press, 568–572.

Baader, F., Fernández Gil, O. and Marantidis, P. (2018). Matching in the description logic $\mathcal{FL}_0$ with respect to general TBoxes. In: Barthe, G., Sutcliffe, G. and Veanes, M. (eds.) *Proceedings of the 22nd International Conference on Logic for Programming, Artificial Intelligence and Reasoning (LPAR 2018)*, vol. 57. EPiC Series in Computing, EasyChair, 76–94.

Baader, F., Küsters, R. and Molitor, R. (1999). Computing least common subsumers in description logics with existential restrictions. In: *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI 1999)*, 96–101.

Baader, F., Marantidis, P. and Okhotin, A. (2016). Approximate unification in the description logic $\mathcal{FL}_0$. In: Michael, L. and Kakas, A. C. (eds.) *Proceedings of the 15th European Conference on Logics in Artificial Intelligence (JELIA 2016)*, vol. 10021. Lecture Notes in Artificial Intelligence. Springer, 49–63.

Baader, F. and Morawska, B. (2009). Unification in the description logic $\mathcal{EL}$. In: Treinen, R. (ed.) *Proceedings of the 20th International Conference on Rewriting Techniques and Applications (RTA 2009)*, vol. 5595. Lecture Notes in Computer Science. Springer, 350–364.

Baader, F. and Narendran, P. (2001). Unification of concept terms in description logics. *Journal of Symbolic Computation* **31** (3), 277–305.

Baader, F. and Nipkow, T. (1998). *Term Rewriting and All That*, Cambridge University Press.

Baader, F. and Schulz, K. U. (1993). General A- and AX-unification via optimized combination procedures. In: Abdulrab, H. and Pécuchet, J.-P. (eds.) *Word Equations and Related Topics*, vol. 677. Lecture Notes in Computer Science. Springer, 23–42. ISBN: 978-3-540-47636-8

https://doi.org/10.1017/S0960129519000185 Published online by Cambridge University Press

Baader, F. and Schulz, K. U. (1996). Unification in the union of disjoint equational theories: Combining decision procedures. *Journal of Symbolic Computation* **21**, 211–243.

Baader, F. and Siekmann, J. H. (1994). Unification theory. In: Gabbay, D. M., Hogger, C. J. and Robinson, J. A. (eds.) *Handbook of Logic in Artificial Intelligence and Logic Programming*, Oxford, UK, Oxford University Press, 41–125.

Baader, F. and Snyder, W. (2001). Unification theory. In: Robinson, J. A. and Voronkov, A. (eds.) *Handbook of Automated Reasoning*, vol. I. Elsevier Science Publishers, 447–533.

Dovier, A., Pontelli, E. and Rossi, G. (2006). Set unification. *TPLP* **6** (6), 645–701. doi:10.1017/S1471068406002730.

Dowling, W. F. and Gallier, J. (1984). Linear-time algorithms for testing the satisfiability of propositional horn formulae. *Journal of Logic Programming* **1** (3), 267–284.

Garey, M. R. and Johnson, D. S. (1990). *Computers and Intractability; A Guide to the Theory of NP-Completeness*. New York, NY, USA, W. H. Freeman & Co. ISBN: 0716710455.

Iranzo, P. J. and Rubio-Manzano, C. (2015). Proximity-based unification theory. *Fuzzy Sets and Systems*, 262, 21–43. doi:10.1016/j.fss.2014.07.006.

Jaumard, B. and Simeone, B. (1987). On the complexity of the maximum satisfiability problem for Horn formulas. *Information Processing Letters* **26** (1), 1–4.

Kapur, D. and Narendran, P. (1992). Complexity of unification problems with associative-commutative operators. *Journal of Automated Reasoning* **9** (2), 261–288. ISSN: 1573-0670. doi:10.1007/BF00245463.

Kapur, D. and Narendran, P. (1986). NP-completeness of the set unification and matching problems. In: Siekmann, J. H. (ed.) *Proceedings of the 8th International Conference on Automated Deduction*, vol. 230. Lecture Notes in Computer Science. Oxford, UK, Springer, 489–495.

Lehmann, K. and Turhan, A.-Y. (2012). A framework for semantic-based similarity measures for $\mathcal{ELH}$-concepts. In: *Proceedings of the 13th European Conference on Logics in Artificial Intelligence (JELIA'2012)*, vol. 7519. Lecture Notes in Computer Science. Springer, 307–319.

Marché, C. (1996). Normalized rewriting: An alternative to rewriting modulo a set of equations. *Journal of Symbolic Computation* **21** (3), 253–288. ISSN: 0747-7171. https://doi.org/10.1006/jsco.1996.0011.

Marques-Silva, J., Ignatiev, A. and Morgado, A. (2017). Horn maximum satisfiability: Reductions, algorithms and applications. In: Oliveira, E. et al. (eds.) *Progress in Artificial Intelligence*, Cham, Springer International Publishing, 681–694.

Narendran, P. and Rusinowitch, M. (1996). Any ground associative-commutative theory has a finite canonical system. *Journal of Automated Reasoning* **17** (1), 131–143. doi: 10.1007/BF00247671.

Pan, J. Z., Ren, Y. and Zhao, Y. (2016). Tractable approximate deduction for OWL. *Artificial Intelligence* **235**, 95–155.

Siekmann, J. H. (1989). Unification theory: A survey. *Journal of Symbolic Computation* **7** (3–4), 207–274.

Cite this article: Baader F, Marantidis P, Mottet A and Okhotin A (2020). Extensions of unification modulo ACUI. *Mathematical Structures in Computer Science* **30**, 597–626. https://doi.org/10.1017/S0960129519000185