UNIVERSITE D'ORLEANS
Faculté des Sciences
LIFO

Laboratoire d'Informatique Fondamentale d'Orléans
4, rue Léonard de Vinci, BP 6759
F-45067 Orléans Cedex 2
FRANCE

Rapport de Recherche

www : http://www.univ-orleans.fr/SCIENCES/LIFO/

Unification modulo $ACUI$ plus Homomorphisms/Distributivity

S. ANANTHARAMAN, LIFO, Orléans (Fr.)
P. NARENDRAN, SUNY at Albany-NY (USA)
M. RUSINOWITCH, LORIA, Nancy (Fr.)

Rapport N^o **2002-11**

Unification modulo $ACUI$ plus Homomorphisms/Distributivity

Siva Anantharaman, Paliath Narendran*, Michael Rusinowitch

LIFO - Orléans (France), University at Albany–SUNY (USA), LORIA - Nancy (France)

Abstract. E-unification problems are central in automated deduction. In this paper, we consider theories that are extensions of the well-known ACI or $ACUI$, obtained by adding finitely many homomorphism symbols, or a symbol '$*$' that distributes over the $ACUI$-symbol denoted '$+$'. We first show that when we adjoin a set of *commuting* homomorphisms to $ACUI$, unification is undecidable. We then consider the $ACUID_l$-unification problem, i.e., unification modulo $ACUI$ plus left-distributivity of a given '$*$' w.r.t. '$+$', and prove its NEXPTIME-decidability. When we assume the symbol '$*$' to be 2-sided distributive w.r.t. '$+$', we get the theory $ACUID$, for which the unification problem remains decidable. But if equations of associativity-commutativity, or just of associativity, on '$*$' are added on to $ACUID$, then the unification problem becomes undecidable.

Keywords: E-Unification, Complexity, Rewrite reachability, Minsky machine, Post correspondence problem, Set constraints.

1 Introduction

The unification problem for the theories AC ("Associativity-Commutativity"), ACI ("AC plus Idempotence") and $ACUI$ ("ACI with Unit element") have been studied in great detail in the past. Natural extensions are the theories that one obtains by adjoining a binary operator '$*$', which is assumed two-sided or just left-distributive over the basic $ACUI$-symbol denoted '$+$'; we shall denote these theories respectively by $ACUID$ and $ACUID_l$. From a practical point of view, the theory $ACUID$ can be used in program specifications based on set-constraints; and the theory $ACUID_l$ can be used: i) in analyzing processes for *non-interference* via trace equivalence, e.g. as in [9] (note: the concatenation of action symbols is left-distributive over the choice operator '$+$' on processes); ii) for the analysis of the temporal properties of state machines: the temporal connective '*until*' is left-distributive w.r.t. the boolean '*or*'. To our knowledge the unification problems over $ACUID$ or $ACUID_l$ have not been studied so far. (The theory $ACUID_r$ – where only right-distributivity of '$*$' over '$+$' is assumed – is similar to $ACUID_l$.)

Now any ground term over an $ACUID_l$ signature defines a homomorphism w.r.t. the $ACUI$-operator (because of distributivity). Thus a theory $ACUIH$ obtained by adjoining a set H of homomorphisms over the $ACUI$ symbol '$+$' can be viewed as intermediary between $ACUI$, and $ACUID$ or $ACUID_l$; it is natural then to consider first the unification problem over such theories. Now, in [4] the case where H is a set of *non-commuting* homomorphisms over the $ACUI$-symbol has been studied in detail, and its unification problem

* Research supported in part by NSF grant CCR-9712396 and ONR grant N00014-01-1-0430.

has been proved DEXPTIME-complete. So we begin by considering the unification problem over the theory $ACUIH^C$, i.e., $ACUI$ plus a family of *commuting* homomorphisms; and show that $ACUIH^C$-unification is undecidable. Our proof goes via reducing a reachability problem for commutative word rewriting (with constraints) to $ACUIH^C$-unification. We then show that the halting problem of Minsky counter machines can be reduced to this reachability problem for commutative word rewriting (with constraints). These results constitute Section 2.

As for the theory $ACUID_l$, its unification problem turns out to be related to $ACUIH$-unification in the non-commuting case: as such, $ACUID_l$-unification turns out to be NEXPTIME-decidable. Section 3 is devoted to this proof.

Section 4 considers the $ACUID$-unification problem. We first show, as a rather immediate consequence of the results of Section 2, that $ACUID$-unification is undecidable if the symbol '$*$' – assumed 2-sided distributive w.r.t. the $ACUI$-symbol '$+$' – is in addition assumed also AC. We show next that, when '$*$' is assumed only associative besides being two-sided distributive over '$+$', $ACUID$-unification still remains undecidable: this is established by reduction from the modified Post correspondence problem. When no further laws other than 2-sided distributivity over '$+$' are assumed on '$*$', the $ACUID$-unification problem can be formulated as a problem of solvability *in terms of finite sets* for a particular class of set constraints. Its decidability can as such be deduced for instance via the results of [10]. We deduce a DEXPTIME lower bound for this problem, as well as for $ACUID_l$-unification, via a reduction from the results of [4].

2 *ACUI* with Commuting Homomorphisms

An $ACUIH^C$-signature consists of an AC-symbol denoted by '$+$' and assumed idempotent, a unit element for this '$+$' (denoted '0'), a finite family of unary symbols with additive homomorphism relations on each symbol, and commutativity relations between these unary symbols. It is generally assumed that a homomorphism maps 0 to 0, so to have more than just one ground term, we also assume given finitely many free constants. The idempotent AC symbol '$+$' can be viewed in this section as the set-union operation. The set of ground terms over such a given signature $ACUIH^C$ will be denoted by $\mathcal{G}$; a ground term is thus a sum (or set) of *commutative* strings over the symbols representing the commuting homomorphisms each applied itself to a ground term. The unification problem over such a signature reduces to solving linear equations over unknowns, with commutative strings over the unary homomorphism symbols as coefficients. For instance, if f, g are homomorphism symbols and a a free constant, then the unification problem $f(f^2ga + gX) =^? gf(fa + fX)$ reduces to solving the "linear equation"

$$f^3ga + fgX = gf^2a + gf^2X$$

for the 'unknown' X in $\mathcal{G}$. (Note: parentheses for the arguments of unary symbols are sometimes omitted.) The substitution $\{X \leftarrow fa\}$ is a solution here. For a more systematic presentation on how to formulate such E-unification problems in terms of solving linear equations over appropriate semi-rings, the interested reader may consult [3], [12].

We show in this section that the $ACUIH^C$-unification problem with free constants is undecidable. The proof is from a reduction of the reachability problem for configurations of Minsky's counter machines. We briefly outline the ideas behind. As we just saw, the problem is that of solving linear equations of the form:

$$l_1X_1 + \ldots + l_nX_n + V \;=\; r_1X_1 + \ldots + r_nX_n + U$$

where the $X_i, 1 \leq i \leq n$ are the 'unknowns', U, V are sets of ground terms, and l_i, r_i are commutative strings over the homomorphism symbols, that we will also call 'ground terms' or 'ground strings' with no risk of confusion. Without any loss of generality, we shall *assume that the unit element '0' for the ACI-symbol '+' is neither in the set U, nor in V*. To this unification problem we associate a set of rewrite rules $l_i \to r_i,\ 1 \leq i \leq n$ and utilise the usual notion of rewrite steps (denoted '$\Longrightarrow$') on ground terms via these rules. The first step in our proof is to show that - provided the transitive closure of '$\Longrightarrow$' has no cycles - the linear equation of the type above has a solution if and only if V is a set of elements reachable from elements of the set U for the relation '$\Longrightarrow^*$'. The second step will be a reduction of the halting problem of Minsky machines with 2 counters to this reachability problem.

2.1 $ACUIH^C$-Unification Problem vs Rewrite Reachability

Before establishing our first step, a notational convention: for any binary relation '$\Longrightarrow$' between terms, we denote as usual by '$\Longrightarrow^+$' and '$\Longrightarrow^*$' its transitive closure and reflexive transitive closure respectively. If u, v are terms such that $u \Longrightarrow^* v$, we will say that v is reachable or derivable from u (or that u derives v) for the relation '$\Longrightarrow$'. For any given pair of sets of terms $S1, S2$, we will write $S1 \models\!\!\Longrightarrow^* S2$ iff:

i) for any element $v \in S2$, there exists a $u \in S1$ such that $u \Longrightarrow^* v$;

ii) for any element $u \in S1$, there exists a $v \in S2$ such that $u \Longrightarrow^* v$.

We will then say: the set $S2$ is a set of *reachable* (or *derivable*) elements from the set $S1$, for the relation '$\Longrightarrow$' that we omit to mention if clear from the context.

Proposition 1 *With the above notation, let*

$$(\&) \quad l_1X_1 + \ldots + l_nX_n + V \;=\; r_1X_1 + \ldots + r_nX_n + U$$

be an $ACUIH^C$-unification problem, $\{l_i \to r_i, i = 1..n\}$ the associated ground rewrite system, and '$\Longrightarrow$' the corresponding rewrite relation. If the problem (&) is solvable, **and** *if the relation $\Longrightarrow$ is acyclic* (i.e., $\Longrightarrow^+$ is irreflexive)*, then $U \models\!\!\Longrightarrow^* V$.*

Proof. We reason by induction on the minimal 'measure' $|X_1| + \ldots + |X_n|$ w.r.t. solutions $(X_1, ..., X_n)$ of the problem (&), for any unification problem with the same associated rewrite system (here, as usual, $|S|$ denotes the cardinality of the set S). If $|X_1|+...+|X_n| = 0$ then $U = V$, and we have the trivial base case. So suppose $U \neq V$.

Case (a): $U \not\subset V$.

Let w be a string in $U \setminus V$. Without loss of generality we assume $w \in l_1 X_1$, so $w = l_1 w_1$ for some $w_1 \in X_1$; then write $X_1 = w_1 + Y_1$ with $w_1 \notin Y_1$ (and $w_1 \neq 0$).

Thus $r_1 X_1 = r_1 w_1 + r_1 Y_1$ and $(Y_1, X_2, ..., X_n)$ has a strictly smaller measure than $(X_1, ..., X_n)$ and satisfies the unification problem:

$$l_1 Y_1 + ... + l_n X_n + (V + l_1 w_1) \;=\; r_1 Y_1 + ... + r_n X_n + (U + r_1 w_1)$$

By induction hypothesis therefore we have: $(U + r_1 w_1) \models\!\Longrightarrow^* (V + l_1 w_1)$.

This means that (i) every term in V is reachable from some term in $U + r_1 w_1$ and hence from some term in $U + l_1 w_1 = U$, and (ii) every term in U can derive some term in $V + l_1 w_1$. Furthermore, $r_1 w_1$ must derive elements in V, since $\Longrightarrow$ is assumed acyclic. Thus every term in U derives some term in V.

<u>Case (b):</u> $U \subset V$.

Let x be a term in V not in U; without loss of generality we assume $x \in r_1 X_1$, i.e., $x = r_1 x_1$ for some $x_1 \in X_1$. Let then $X_1 = x_1 + Z_1$ with $x_1 \notin Z_1$ (and $x_1 \neq 0$).

Thus $l_1 X_1 = l_1 x_1 + l_1 Z_1$; now $(Z_1, X_2, ..., X_n)$ has a strictly smaller measure than $(X_1, ..., X_n)$ and satisfies the unification problem:

$$l_1 Z_1 + ... + l_n X_n + (V + l_1 x_1) \;=\; r_1 Z_1 + ... + r_n X_n + (U + r_1 x_1)$$

Again, by induction hypothesis we have $(U + r_1 x_1) \models\!\Longrightarrow^* (V + l_1 x_1)$. Now, since the relation $\Longrightarrow$ is assumed acyclic, it cannot be that $r_1 x_1 \Longrightarrow^* l_1 x_1$, so there must be some string in U *other than* $r_1 x_1$ which reduces to $l_1 x_1$, and hence to $r_1 x_1$ as well. This implies then that any element of V is reachable from elements of U. Since U is a subset of V, the result follows. $\square$

Remark 1. The above proposition ceases to be true if the rewrite relation is not acyclic. Here is a counter-example. We suppose given two commuting homomorphisms, and consider the following unification problem:

$$l_1 X_1 + l_2 X_2 + a^3 b^3 + a^4 b^2 + a^3 b^2 \;= r_1 X_1 + r_2 X_2 + a^3 b^3$$

where a, b are two homomorphism symbols, $l_1 = a^2 b, r_1 = ab$ and $l_2 = a^2 b^2, r_2 = a^3 b^2$; the strings on a, b with no following variables are all assumed evaluated at some fixed constant. This problem does have a solution, namely: $X_1 = a^2 b, X_2 = a$ (since '+' is idempotent). But neither $a^4 b^2$ nor $a^3 b^2$ can be reached from $a^3 b^3$ via the rewrite rules. This is because each rule consuming a power of b brings back exactly the same power of b.

Proposition 2 *Let* (&): $l_1 X_1 + \ldots + l_n X_n + V \;=\; r_1 X_1 + \ldots + r_n X_n + U$ *be an* $ACUIH^C$*-unification problem,* $\{l_i \rightarrow r_i, i = 1..n\}$ *the associated ground rewrite system, '*$\Longrightarrow$*' the corresponding rewrite relation. Then the problem* (&) *is solvable* **if** $U \models\!\Longrightarrow^* V$.

Proof. Let us define the *'distance'* between U and V (w.r.t. the relation $\Longrightarrow$) as the sum of the (minimal) number of rewrite steps needed for deriving the elements of V from the elements of U. We shall reason by induction on this distance. The case of distance 0 corresponds to the trivial base case $U = V$. So assume that this distance d is non-zero;

then there exist $u \in U$ and $v \in V$ such that $u \Longrightarrow^+ v$; without loss of generality we assume that $u = l_1 u'$ and $r_1 u' \Longrightarrow^* v$.

Now write $U = u' + U'$ with $u' \notin U'$; then the sets $(r_1 u' + U')$ and V are such that $(r_1 u' + U') \models\!\Longrightarrow^* V$, and the distance (w.r.t. $\Longrightarrow$) from $(r_1 u' + U')$ to V is strictly smaller than d. So, by induction hypothesis there must exist sets $X'_1, \ldots, X'_n$ such that

$$l_1 X'_1 + \ldots + l_n X'_n + V = \ r_1 X'_1 + \ldots + r_n X'_n + (r_1 u' + U')$$

Now, setting $X_1 = X'_1 + u'$ and $X_j = X'_j$ for all $j > 1$, we get a solution for (&). $\square$

Putting these two propositions together, we get

Theorem 1. *Let* (&): $l_1 X_1 + \ldots + l_n X_n + V \ = \ r_1 X_1 + \ldots + r_n X_n + U$ *be a given* $ACUIH^C$*-unification problem,* $\{l_i \to r_i, i = 1..n\}$ *the associated ground rewrite system, and '*$\Longrightarrow$*' the corresponding rewrite relation such that the relation* $\Longrightarrow$ *is* acyclic. *Then the problem (&) is solvable* if and only if $U \models\!\Longrightarrow^* V$.

Corollary 1.1: Let $a_1, \ldots, a_m$ be commuting homomorphisms. Then the equation

$$a_1 X_1 + \ldots + a_m X_m + \epsilon \ = \ X_1 + \ldots + X_m + W$$

is solvable if and only if W consists of strings over the $a_i, i = 1..m$, evaluated on some constants. $\square$

In particular if a, b, c are commuting homomorphisms, then the equation: $a(x_1) + b(x_2) + \epsilon = x_1 + x_2 + W$ forces W to consist entirely of terms not containing c.

2.2 $ACUIH^C$-Unification in Terms of Admissible Reachability

Definition 1 *A marked rewrite rule over the set* $\mathcal{G}$ *is a rule of the form* $l \to r\ [a]$*, where the* mark a *is a given ground term, and* l, r *are strings such that* l *does* not contain the *mark* a as a factor. *A ground term not having the mark* a *as a factor is said to be* compatible *with the marked rewrite rule* $l \to r\ [a]$. *A rewrite step using the rewrite rule* $l \to r\ [a]$ *is said to be* admissible *iff the string rewritten under this step is compatible with the rule.*

A rewrite rule with no attached mark may be seen as marked with the empty string; any ground string is compatible with such a rewrite rule. Given a set of marked rewrite rules over $\mathcal{G}$, we shall be henceforth denoting by $\Longrightarrow$ the associated relation of admissible reduction, and by $\models\!\Longrightarrow^*$ the relation of *admissible reachability* (or derivability) between sets of terms, defined as in the previous subsection but here w.r.t. this restricted notion $\Longrightarrow$ of *admissible reduction.*

Theorem 2. *Let* (&): $l_1 X_1 + \ldots + l_n X_n + V \ = \ r_1 X_1 + \ldots + r_n X_n + U$ *be a given* $ACUIH^C$*-unification problem. Let* $\{l_i \to r_i\}$ *be the associated rewrite system, each rule respectively* marked *with either a constant* $a_i, i = 1..n$*, or the empty string; and let '*$\Longrightarrow$*'*

be the associated admissible rewrite *relation, that we assume to be acyclic. We consider the following constraint on the* X_i *:*

$$(\mathcal{C}) \qquad \text{if } l_i \to r_i \text{ is marked with } a_i, \text{ then } X_i \text{ does not contain terms in which the mark } a_i \text{ occurs.}$$

Then the the problem (&) is solvable under constraint $\mathcal{C}$ if and only if *we have* $U \models\!\Longrightarrow^* V$.

Proof. The proof of Theorem 1 based on Propositions 1 and 2, goes through verbatim; we mention here the points to check, for completeness. (We are assuming each rewrite rule $l_i \to r_r$ is marked with an a_i, which is a constant or the empty string.)

Proposition 1, case (a): if $(X_1, \ldots, X_n)$ is a solution satisfying constraint $\mathcal{C}$, $w = l_1 w_1$, and $X_1 = w_1 + Y_1$, then $(Y_1, X_2, \ldots, X_n)$ continues to satisfy the constraint $\mathcal{C}$, so induction goes through. The same reasoning holds also for case (b).

Proposition 2: the distance from U to V is defined now over admissible rewrite steps. If $u = l_1 u' \Longrightarrow r_1 u'$ admissibly, then u' doesn't contain the mark a_1 on the rule $l_1 \to r_1\,[a_1]$; so if $(X_1', X_2', \ldots, X_n')$ is a solution satisfying constraint $\mathcal{C}$ for the linear equation at the inductive step, then the solution $(X_1' + u', X_2', \ldots, X_n')$ for the initial equation also satisfies the constraint $\mathcal{C}$. □

We will show in the next section that *admissible reachability is undecidable.* From this and the above theorem, it will follow that solving an $ACUIH^C$-linear equation (&) with marked associated rewrite rules, and under constraint $\mathcal{C}$, is undecidable. But it follows from Corollary 1.1 that the constraint $\mathcal{C}$ is itself expressible in terms of linear equations over the $ACUIH^C$-signature: indeed the equation

$$a_1 X_1 + \ldots + a_{j-1} X_{j-1} + \epsilon + a_{j+1} X_{j+1} + \ldots + a_n X_n = X_1 + X_2 + \ldots + X_n$$

forces the component X_j of the solution to be free from terms containing a_j, if the marks a_i are all distinct. In other words, we will be able to deduce the undecidability of the $ACUIH^C$-unification problem.

2.3 Minsky Counter Machines and Admissible Reachability

A Minsky machine with two counters C_1, C_2 storing non-negative integer values, executes *programs* which are finite lists of *instructions* labeled with the natural numbers from 1 to L, each of which has one of the following forms (where $l \in \{1, \ldots, L-1\}$, $l' \in \{1, \ldots, L\}$, $l \neq l'$, and i is 1 *or* 2):

(i) l: ADD 1 to C_i and GOTO $l+1$;

(ii) l: If $C_i \neq 0$
then SUBTRACT 1 from C_i and GOTO $l+1$;
else GOTO l';

(iii) L: STOP.

We shall agree that in any given program $\mathcal{P}$ there is exactly one instruction 'STOP' labeled L. A *configuration* of such a 2-counter machine, at any given stage of a computation following such a program, can then be defined as a triple (l, C_1, C_2) where l is the (label of the) next instruction to execute, and $C_i, i = 1, 2$, are the current integer values of the two counters. A configuration of the form (L, xxx, yyy) (resp. $(1, xxx, yyy)$) will be labeled *final* (resp. *start* or *initial*). A program by definition *starts* on an initial configuration, and *halts*, by definition again, if it reaches a configuration which is final (in which case, it has no more instructions to execute). The following result on Minsky's 2-counter machines is classical, and will well serve our purposes, after some minor adaptations.

Theorem 3. *(Minsky[11]) For every partial recursive function f on natural numbers there exists a program $\mathcal{Q}$ such that applied with $(1, 2^d, 0)$ as starting configuration, $\mathcal{Q}$ halts with the final configuration $(L, 2^{f(d)}, 0)$ if $f(d)$ is defined on d, and does* not *halt otherwise.*

In particular it is undecidable in general if an arbitrarily chosen program $\mathcal{P}$ will halt, when applied with an arbitrarily given starting configuration $(1, 2^d, 0)$.

We introduce a homomorphism symbol h_l for every instruction $l \in 1..L$; add two more homomorphisms $h_{c_i}, i = 1, 2$, for the two counters. Any given machine configuration can then be seen as a (commutative) string $h_l\, h_{c_1}^{m_1}\, h_{c_2}^{m_2}$, if l is the current instruction to execute and $m_i, i = 1, 2$, are the current (non-negative) integer values of the two counters. We then associate the following system of *marked* rewrite rules to each type of machine instructions:

i) (increment C_1 or C_2) For any $j \in 1..(L-1)$,
$h_j \to h_{c_1} h_{j+1}$, $\quad h_j \to h_{j+1}\, h_{c_2}$

ii) (conditional subtraction on C_i) For any $j \in 1..(L-1)$,
$h_{c_1} h_j \to h_{j+1}$, $\quad h_j\, h_{c_2} \to h_{j+1}$

ii') (counter C_1 or C_2 must be empty) For any $j \in 1..(L-1)$, $k \in 1..L$, with $j \neq k$,
$h_j \to h_k\ [h_{c_1}]$, $\quad h_j \to h_k\ [h_{c_2}]$

Any given instruction transforms the current configuration-string by applying a unique well-determined rewrite rule from this set. Note that the last two sets of rewrite rules are *marked*, and therefore can be applied for reduction only on strings which are compatible: the first among these can rewrite only strings with no first counter, and the second only the strings with no second counter. Also note that any sequence of admissible rewrite steps starting from the configuration string $h_1\, h_{c_1}^{2^d}$ corresponds to a (correct) sequence of machine instructions and vice versa. The final statement of the above theorem of Minsky on 2-counter machines can then be formulated as: Given two arbitrary strings $s = h_1\, h_{c_1}^{2^d}$ and $f = h_L\, h_{c_1}^{2^m}$ over our $ACUIH^C$-signature, it is undecidable in general if the latter is reachable from the former via admissible rewrite steps using the marked rules of the above rewrite system.

The proof that $ACUIH^C$-unification is undecidable is almost complete now, thanks to the result of our previous section; remains the hitch that this result made an assumption of acyclicity on the rewrite relation, and this does not hold for the above rewrite system. In order to get that property, we introduce a *third* counter C_3. The idea is that initially counter C_3 has value 1, that every instruction of type (i) or (ii) also increments C_3 by 2, instruction of type (iii) increments C_3 by 1 and goes to a new instruction where the only thing done is decrementing C_3 by 2 as long as possible, until it stops. Following the same lines of thinking as above, such a 3-counter machine program $\mathcal{P}'$ extending an earlier 2-counter program $\mathcal{P}$ can be visualized as executions of rewrite steps applying rules from the following system:

i) (increment C_1 or C_2) For any $j \in 1..(L-1)$,
$$h_j \to h_{c_1} h_{j+1}\, h_{c_3}^2, \qquad h_j \to h_{j+1}\, h_{c_2}\, h_{c_3}^2$$
ii) (conditional subtraction on C_i) For any $j \in 1..(L-1)$,
$$h_{c_1} h_j \to h_{j+1}\, h_{c_3}^2, \qquad h_j\, h_{c_2} \to h_{j+1}\, h_{c_3}^2$$
ii') (counter C_1 or C_2 must be empty) For any $j \in 1..(L-1)$, $k \in 1..L$, with $j \neq k$,
$$h_j \to h_k h_{c_3}^2\ [h_{c_1}], \qquad h_j \to h_k\, h_{c_3}^2\ [h_{c_2}]$$
iii) (pause) $h_L \to h_{L+1}\, h_{c_3}$

iv) (conditional decrement on C_3, or STOP)
$$h_{L+1}\, h_{c_3}^2 \to h_{L+1}, \qquad h_{L+1} \to h_{L+2}\ [h_{c_3}]$$

It is immediate that this rewrite system is acyclic, and that no right-hand-side is a substring of any of the marks (note: the marks here are h_{c_1}, h_{c_2} and h_{c_3}). The following fact is equally obvious: Our earlier 2-counter machine run on a program $\mathcal{P}$ starting from an initial configuration $h_1\, h_{c_1}^{2^d}$ halts, with as final configuration a string $h_L\, h_{c_1}^{2^m}$, **if and only if** our current 3-counter machine run on the extended program $\mathcal{P}'$ from the initial configuration $h_1\, h_{c_1}^{2^d}\, h_{c_3}$ halts, with final configuration $h_{L+2}\, h_{c_1}^{2^m}$.

We can now formulate our undecidability result, whose proof is a direct consequence of our second theorem 2 combined with the above reformulation of Minsky's theorem in terms of a 3-counter machine.

Theorem 4. *The $ACUIH^C$-unification problem is undecidable.*

Remark 2. It is immediate from the above lines of proof that the unit element '0' for the AC-symbol '+' has played no special role in the reasoning. In other words, we may also conclude that: *the $ACIH^C$-unification problem is undecidable.*

3 $\boldsymbol{ACUID_l}$-unification is decidable

The equational theory we are considering in this section is

$$\begin{array}{rcll} x + (y + z) &=& (x + y) + z & \text{(A)} \\ x + y &=& y + x & \text{(C)} \\ x + 0 &=& x & \text{(U)} \\ x + x &=& x & \text{(I)} \\ x * (y + z) &=& (x * y) + (x * z) & (D_l) \\ x * 0 &=& 0 & \text{(Z)} \end{array}$$

Any $ACUID_l$-unification problem can be transformed into one where each equation is in one of the following four forms:

$$(i)\ x =^? 0, \quad (ii)\ x =^? a, \quad (iii)\ x =^? y + z, \quad (iv)\ x =^? y * z$$

where x, y, z are variables, 0 is the unit element for the $ACUI$-symbol '+'and a is a ground constant. If equations of type (i) do not occur then the problem is called a *simple* unification problem.

Definition 1. *i) A substitution is* deciduous *if each variable in its domain is replaced either by 0 or by another variable.*

ii) A unifier θ for an $\mathcal{E}$-unification problem S is said to be a discriminating *unifier if and only if the following hold for all variables in $Var(S)$:*

$$\theta(u) \neq_{\mathcal{E}} 0 \qquad \textit{and} \qquad \theta(v) =_{\mathcal{E}} \theta(w) \textit{ iff } v = w$$

It is not hard to see that an $ACID_l$-unification problem S has a unifier if and only there is a deciduous substitution η such that: i) either η is a unifier for S, ii) or $\eta(S)$ has a discriminating unifier. Since such an η can be nondeterministically chosen, we can reduce our unification problem to the following problem:

Instance: A set of equations S, each of the form (ii), (iii) or (iv) defined above.
Question: Does S have a discriminating $ACID_l$-unifier?

Now given such a unification problem S, we transform each equation of the form $x =^? y * z$ (case (iv) above) into the equation

$$x =^? h_y(z)$$

where h_y is a homomorphism we introduce. Let $\mathcal{H}(S)$ denote the set of all homomorphisms introduced in this way and let $\mathcal{V}_h(S)$ denote the set of variables for which homomorphisms were created. We thus get rid of the $*$ symbol, but our new equational theory consists of the $ACUI$ axioms plus

$$\{h(u_1 + u_2) = h(u_1) + h(u_2),\ h(0) = 0\}$$

for all $h \in \mathcal{H}(S)$. We call the transformed $\mathcal{S}$-unification problem as the *h-image* of S, and denote it by T. We now make this notion precise. Let t be any term over the signature consisting of $*$, $+$ and additional (free) constants. We introduce distinct homomorphisms for the equivalence classes of subterms of t, and define the following transformation ξ:

$$\begin{aligned} \xi(r) &= r && \text{if } r \text{ is a variable or a constant} \\ \xi(x+y) &= \xi(x) + \xi(y) && \\ \xi(s * t) &= h_{[s]}(\xi(t)) && \text{where } [s] \text{ is the equivalence class of } s \text{ w.r.t. } ACUI. \end{aligned}$$

It is not difficult to see that $\xi(t)$ is unique (up to $=_{ACUI}$) for every term over $\{*, +\}$ and additional constants. This definition of ξ can now be extended to sets of equations of terms. Thus, for the unification problem S, h-image$(S) = \xi(S)$. In what follows, we will drop the square brackets for the indices of homomorphisms, and consider them as defined modulo $=_{ACUI}$.

The transformed problem T = h-image of S is over an $ACUIH$ signature, where H is the set $\mathcal{H}(S)$ of homomorphisms we introduced above for the variables in $\mathcal{V}_h(S)$. However, it is not hard to see that unifiability of the T (modulo $ACUIH$) does not automatically ensure unifiability of the original $ACUID_l$ problem. We need to impose additional constraints on the $ACUIH$ problem, which are reformulations of the usual occur-check condition on the variables: for instance, the substitution for a variable x in $\mathcal{V}_h(S)$ must not contain h_x, the homomorphism introduced for x. Cycles should not occur either, i.e., x contains h_y and y contains h_x.

These occur-check constraints are enforced by requiring that the unifiability problem of T w.r.t. $ACUIH$ be solved subject to *linear* constraints (specifying conditions like) $x \succ h_x$ for every homomorphism $h_x \in \mathcal{H}(S)$, i.e. to say, a unifier θ of T should satisfy the condition that for every $x \in Var(T)$, $\theta(x)$ *does not contain* any occurrence of h_x.

Definition 2. *i) Given a simple $ACUID_l$-unification problem S and its h-image T modulo $ACUIH$, a* linear constraint *C is a total order $\succ_C$ over $Var(T) \cup \mathcal{H}(S)$ such that $x \succ_C h_x$ for all variables x in $\mathcal{V}_h(S)$.*

ii) Let T be an $ACUIH$-unification problem. A substitution β whose domain is $Var(T)$ is said to satisfy a linear constraint $\succ_C$ if and only if the following holds: for every $x \in Var(T)$, $\beta(x)$ does not contain *any of the function symbols below x in C. In other words, if $x \succ_C h_j$, then $\beta(x)$ does not contain any occurrence of h_j.*

The following two theorems relate a unification problem S over $ACUID_l$ to its h-image T over $ACUIH$, where H denotes the set of homomorphisms $\mathcal{H}(S)$.

Theorem 5. *If a simple $ACUID_l$-unification problem S has a discriminating unifier, then its h-image T is solvable as an $ACUIH$-unification problem. Furthermore, there is a linear constraint $\succ_C$ that the unifier satisfies.*

Proof. Let θ be a discriminating ground unifier of S. One could define a substitution α on T as $\alpha(x) = \xi\ (\theta(x))$. But α is not (yet) a unifier for T. Consider an equation $x =^? y * z$ in S. Its h-image is $x =^? h_y(z)$. But $\xi(\theta(y * z))$ is $h_{\theta(y)}(\xi(\theta(z)))$ and not $h_y(\xi\theta(z)))$.

The idea is to transform α into a unifier β by replacing, for every variable v in $\mathcal{V}_h(S)$, $h_{\theta(v)}$ by the corresponding h_v in every term. This mapping is *one-to-one* since θ is a discriminating unifier. □

Example. Consider the $AUID_l$-unification problem

$$V =^? X + Z, \quad Y =^? W + U, \quad W =^? X * Y$$

whose h-image is the problem: $V =^? X + Z, \quad Y =^? W + U, \quad W =^? h_X(Y)$. The former admits as a unifier: $X \mapsto a,\ Y \mapsto a,\ U \mapsto a,\ Z \mapsto a * a,\ W \mapsto a * a,\ V \mapsto a + (a * a)$; from which the above reasoning would derive the following $ACUIH$-unifier for the latter: $X \mapsto a,\ Y \mapsto a,\ U \mapsto a,\ Z \mapsto h_X(a),\ W \mapsto h_X(a),\ V \mapsto a + h_X(a)$ □

Theorem 6. *Let S be a simple $ACUID_l$-unification problem, and let T be its h-image which is an $ACUIH$-unification problem. If T has a solution which satisfies a linear constraint, then S is solvable.*

Proof. Let β be a ground $ACUIH$-unifier of T which satisfies a linear constraint C. From C, we get a subconstraint C' on the variables in $Var(T)$. Assume without loss of generality that $C' = x_n \succ \cdots \succ x_i \succ \cdots \succ x_1$. Now we will use induction on C' to form θ.

Let us first consider the variable x_n in C'. Since x_n is the first variable, and $\beta(x_n)$ should not contain any item below x_n in C, it must be that $\beta(x_n)$ does not contain any of the homomorphisms that were introduced, and we define $\theta(x_n) := \beta(x_n)$.

Assume that we have already constructed all $\theta(x_{j'})$ $(j \leq j' \leq n)$. For variable x_{j-1}, $\beta(x_{j-1})$ could contain constants and some $h_{x_{w_i}}$ for $x_{w_i} \in \mathcal{V}_h(S)$ where each $h_{x_{w_i}} \succ x_{j-1}$. Since $x_{w_i} \succ h_{x_{w_i}}$, we have $x_{w_i} \succ h_{x_{w_i}} \succ x_{j-1}$. By the induction hypothesis, we have already constructed these $\theta(x_{w_i})$'s. Therefore, we can define $\theta(x_{j-1}) := \boldsymbol{rep}(\beta(x_{j-1}))$ where the function $\boldsymbol{rep}$ is defined as:

$$\boldsymbol{rep}(a) \;=\; a \text{ where } a \text{ is a constant.}$$

$$\boldsymbol{rep}(A+B) \;= \boldsymbol{rep}(A) + \boldsymbol{rep}(B)$$

$$\boldsymbol{rep}(h_{x_{w_i}}(A)) \;=\; \theta(x_{w_i}) * \boldsymbol{rep}(A)$$

where A, B stand for any terms. It can be shown that θ is a solution for S. Indeed consider each equation in T of the form $x_{u_i} =^? h_{x_{w_i}}(x_{v_i})$. Since $\beta(x_{u_i}) =_{ACUIH} h_{x_{w_i}}(\beta(x_{v_i}))$, we have $\theta(x_{u_i}) =_{ACUID_l} \theta(x_{w_i}) * (\theta(x_{v_i}))$ by our definition of θ. □

We now prove that the linear constraints over the $ACUIH$-signature can themselves be formulated as instances of $ACUIH$-unification problems. This is done by using the following lemma, similar to the one we used earlier in Section 2.1; its proof is obtained along entirely similar lines.

Lemma 1. *Let $a_1, \ldots, a_m$ be homomorphisms and c be a free constant. Then the equation*

$$a_1(X_1) + \ldots + a_m(X_m) + c \;=^?_{ACUIH}\; X_1 + \ldots + X_m + W$$

is solvable if and only if $W \in Terms(\{+, a_1, \ldots, a_m, c\})$ □

We may now use the results of [4], and get the desired result:

Theorem 7. *$ACUID_l$-unification is NEXPTIME-decidable.*

4 The *ACUID*-Unification Problem

We are interested here in the unification problem w.r.t. the following theory that we shall denote $ACUID$:

$$\begin{aligned} x + 0 &\approx x \\ x + x &\approx x \\ x * 0 &\approx 0 \\ x * (y + z) &\approx (x * y) + (x * z) \\ (u + v) * w &\approx (u * w) + (v * w) \end{aligned}$$

This set of equations can be converted naturally to a convergent rewrite system; every ground term in normal form (over any given set of free constants) w.r.t. this system can be viewed as a *finite set* of terms over '$*$' and the constants: indeed '$+$' can be viewed as set union. An ACUID-unification problem with free constants is that of solving modulo the above equational theory, a family of equations of the form: $\{s_1 = t_1, \ldots, s_k = t_k\}$, where the terms in the equations and/or the solutions can involve the given constants.

4.1 Undecidability if '*' is in addition AC or Associative

In this section we prove that the $ACUID$-unification problem is undecidable if we assume that the symbol '$*$' is in addition either AC, or associative. The AC-case follows easily from our previous results on $ACUIH^C$-unification:

Theorem 8. *If the symbol '$*$' above is also assumed AC – i.e., if the equations of associativity and commutativity on '$*$' are added on to $ACUID$ – , then the $ACUID$-unification problem is undecidable.*

Proof. Given finitely many free constants $\{a, b, c, ...\}$, let $h_a, h_b, h_c, ...$ denote the homomorphisms they define in a natural manner over the term algebra: $h_a(x) = a * x$ etc.. Since we are assuming the symbol '$*$' to be AC, these homomorphisms *commute*. (*Note*: it does **not** suffice to have only the commutativity of '$*$' here.) Consider then any arbitrarily given family of equations of the form:

$$l_1\, X_1 + \ldots + l_p\, X_p + V = r_1\, X_1 + \ldots + r_p\, X_p + U$$

where the l_i, r_i are '*'-commutative strings over these homomorphisms, and V, U are sets of ground terms over the given constants. Solving this family of equations amounts to solving the $ACUIH^C$-unification problem expressed in terms of such linear equations. And we saw in the previous section that $ACUIH^C$-unification is undecidable. □

Theorem 9. *If the symbol '$*$' is also assumed associative – i.e., if the equations of associativity on '$*$' are added on to $ACUID$ – , then the $ACUID$-unification problem is undecidable.*

Proof. The proof is via reduction from the so-called modified Post correspondence problem (MPCP). Recall that the formulation of MPCP goes as follows: Given a list of pairs $\{(w_i, w_i')\}_{i=0..n}$ of non-empty (finite) strings over some alphabet Σ, it is undecidable in general to determine whether there exist indices $i_1, \ldots, i_k$ with $0 \leq i_j \leq n$ for all $j = 1..k$, such that: $w_0 w_{i_1}\, w_{i_2}\, \ldots\, w_{i_k} \;=\; w_0' w_{i_1}'\, w_{i_2}'\, \ldots\, w_{i_k}'$.

Let $(w_0, w_0'), \ldots (w_n, w_n')$ be any given instance of MPCP; we may assume without loss of generality, that $w_0 \neq w_0'$. Consider then the $ACUID$-unification problem defined by the following two equations:

(1) $\quad w_1 * X_1 * \overline{1} + \ldots + w_n * X_n * \overline{n} + w_0 * \# \;=\; X_1 + X_2 + ... + X_n + V$

(2) $\quad w_1' * Z_1 * \overline{1} + \ldots + w_n' * Z_n * \overline{n} + w_0' * \# \;=\; Z_1 + Z_2 + ... + Z_n + V$

where the unknowns are the X_i, Z_i and V, the $\#, \overline{1}, \overline{2}, \ldots, \overline{n}$ are some fixed new symbols. Assume this $ACUID$-unification problem to be solvable.

We observe first that not all of the X_i can be 0 in the solution of (1): otherwise V will have to be $w_0 * \#$, so V contains none of the new symbols $\overline{1}, \overline{2}, \ldots, \overline{n}$; but then, all the Z_i in the second equation have to be 0, and V would have to be be $w'_0 * \#$, which would contradict the assumption we made above.

Consequently, V must contain words ending with symbols from $\{\overline{1}, \overline{2}, \ldots, \overline{n}\}$. We then look at the solution for equation (1), and observe that: words of maximal length over the rhs (right-hand side) of (the solution for) this equation must come from the set V, and can be in no instance of any X_j over the rhs: indeed, if v is a word from an X_j, then $w_j * v * \overline{j}$ will be an element of some set over the lhs (left-hand side), so must also be an element over the rhs, and necessarily of bigger length than the word v.

Let then $v \in V$ be a word of maximal length over the rhs of (1). If v is not equal to $w_0 * \#$, then it has to be of the form $w_i * w * \overline{i}$ for some unique i. We may suppose $i = 1$, write $v = w_1 * w * \overline{1}$ and set $X_1 = X'_1 + w$, for a $w \in X_1$, $w \notin X'_1$. Then v has a unique occurrence in both the lhs and rhs of the equation, so we can cancel it from each side. After such a cancellation, we get an equality of the form:

$$w_1 * X'_1 * \overline{1} + \ldots + w_n * X_n * \overline{n} + w_0 * \# \quad = \quad X'_1 + X_2 + ... + X_n + V'$$

where $V' = (V \setminus \{w_1 * w * \overline{1}\}) + w$.

Now (the solution for) V' is smaller than (that of) V for the multiset ordering over the lengths of words. Thus we can apply an inductive argument to deduce that each element of V has to be a word of the form: $w_{i_1} * \ldots * w_{i_n} * w_0 * \# * s$, where s is some string over the symbols $\overline{1}, \ldots, \overline{n}$. A similar reasoning with equation (2) leads us then to the assertion that each such element of V must also be, at the same time, of the form: $w'_{i_1} * \ldots * w'_{i_n} * w'_0 * \# * s$, with the same sequence of indices for the w'_i as for the w_i, because of the new symbols introduced $\overline{i}$.

In other words, if our above $ACUID$-unification problem were solvable (with a non-zero value for at least one unknown), then the solution for V contains a solution for the instance of MPCP we started with. Conversely from a solution of MPCP we can build a solution to the $ACUID$-unification problem defined by the above two equations (1), (2), in terms of sets of terms over '$*$' and the constants. $\Box$

4.2 The General Case

We shall be assuming henceforth that our $ACUID$-unification problem (after some normalization and decomposition steps) is in *standard form*, i.e., to say, every equation in our problem has one of the following forms (respectively referred to as of type *'product'*, *'sum'*, or *'constant'*):

$$x = y * z, \qquad u = v + w, \qquad u = c$$

where u, v, w, x, y, z are variables and c is any constant or 0. A given $ACUID$-unification problem can be reduced to a standard form in more than one manner. Since '$+$' is idempotent and '$*$' distributes left and right over '$+$', we may view this $ACUID$-unification problem as a set constraint problem; e.g. in the first case, if y and z are interpreted as sets of terms over $*$ and the constants, then $y * z = \{s * t \mid s \in y,\, t \in z\}$. Actually the set constraints in this context are *with union* only, following the terminology introduced in [7]; with the additional restriction that *all sets must be finite*.

The problem of satisfiability of set constraints in general, i.e., allowing arbitrary sets in the solutions, has been studied intensively over the past decade, in particular in [1, 2, 6, 8, 10, 7], although not all known positive results give a complexity estimate. However very few results seem to be known for solvability in terms of finite sets, or finite non-empty sets. The only result we actually know of is very general, and is based on the Σ-graph automata of Gilleron, Tison and Tommasi (cf. Proposition 14, [10]); by appealing to this result, we may formulate our next result:

Proposition 3 *The $ACUID$-unification problem is decidable.*

Actually the results of [4] give a lower bound for the complexity of $ACUID$-unification:

Theorem 10. *$ACUID$-unification is DEXPTIME-hard.*

Proof. We proceed as follows: Consider any $ACUIh$-unification problem – where $ACUIh$ is the theory with an AC-symbol '+', assumed idempotent with a unit $U = 0$, and h is any given finite family of non-commuting homomorphisms w.r.t. '+'. First, the homomorphisms in the family h can be seen as defined by finitely many free constants $\{a, b, c, \ldots\}$ in an $ACUID$-signature; next, the $ACUIh$-problem itself can be seen as solving the following $ACUID$-unification problem:

$$l_1(X_1) + l_2(X_2) + \ldots + l_n(X_n) + T \ =^? \ r_1(X_1) + r_2(X_2) + \ldots + r_n(X_n) + S$$

where the unknowns are the X_i, the $\{l_i, r_i\}_{i=1..n}$ are *right-parenthesized terms* over '$*$' and the constants, and S, T are two finite sets of such terms. Indeed, it is easily checked that in any $ACUID$-solution of this problem, the X_i have *got to be* right-parenthesized terms over '$*$' and the constants; i.e. to say, terms representing (right-)compositions of the homomorphisms defined by the constants. This shows that $ACUIh$-unification can be seen as an instance of $ACUID$-unification; and proves the DEXPTIME-hardness of the latter, since the former has been proved DEXPTIME-complete in [4]. $\square$

Remark 3. The above reasoning applies obviously also to $ACUID_l$-unification, so we also deduce that $ACUID_l$-unification is DEXPTIME-hard.

5 The Unification problem over $ACIH^C$, $ACID_l$ and $ACID$

It is a natural question to ask how much of the results, or reasonings presented above go through if we do *not* assume the presence of the unit element for the ACI-symbol '+', null w.r.t. the distributing '$*$'; let us denote the corresponding theories respectively by $ACIH^C, ACID_l, ACID$. We observed already that our proof for $ACUIH^C$ holds as such for $ACIH^C$. The $ACID$-unification problem, once transformed into a standard form, can still be seen as one of solving a (special class of) set constraints problem, with the restriction that the *sets should be not only finite but also non-empty.* The work [10] mentioned above still holds here, so we have the decidability of $ACID$-unification as well. However, the NEXPTIME upper bound and the DEXPTIME lower bound deduced above for $ACUID_l$-unification both rely on [4], where the proofs depend on the presence of the unit element a priori, and therefore cannot be carried through as they stand for $ACID_l$ and $ACID$. We present below an argument which adapts the lines of proof of the needed result from [4], in order to deduce a DEXPTIME lower bound for the unification problems for the two theories $ACID_l$ or $ACID$.

Theorem 11. *The unification problem over ACID or $ACID_l$ is DEXPTIME-hard.*

Proof. Thanks to the lines of proof of our previous theorem, it suffices to show that the proof of the DEXPTIME-hardness of $ACUIh$-unification, as given in [4], can be adapted so as to get rid of the (possible) role played by the unit element U of the theory.

Recall that this proof relies on Seidl's result [13] that if $A_i, i = 1..k$ are given finitely many DTTA's (deterministic topdown tree automata), then deciding the emptiness of their intersection is DEXPTIME-hard. In [4], an $ACUIH$-equation E_i is constructed for each of the A_i; these equations share exactly one common variable X. Solutions to any equation E_i are assignments of finite unions of 'run tree sets' constructed from finite trees accepted by the corresponding automaton A_i; the unique common variable X serves to glue these different assignments together into a single one, whenever there is a common solution for *all* the equations $E_i, i = 1..k$. A single global equation combining all these equations is defined, such that it is solvable iff there is a *common tree* t accepted by *all* the $A_i, i = 1..k$, in which case the run tree set derived from t is assigned to this unique common variable X; if the acceptance of this common tree t by any of the A_i does not involve all the transitions in the DTTA A_i, then the value assigned to this variable X is $U = 0$. We propose to restructure this reasoning in such a manner that the solvability of the global equation amounts to the existence of **one run tree set** common to the assignments for the different variables (which is weaker than asking for a common accepted tree t).

First we show that Seidl's result holds even if we restrict to signatures with *one* binary symbol g, *one* unary symbol s, and *one* constant '#'. The key idea is this: suppose have a set of binary function symbols $\{g_1, ..., g_n\}$. Then we can replace each g_i with $\lambda(x, y).s(s(...s(g(x, y))))$, with i occurrences of the symbol s. (In other words, the term is $s^i(g(x, y))$.) A DTTA on the original alphabet can thus be converted into a DTTA on the new alphabet in polynomial time.

Next, we form the equations E_i for each DTTA A_i like in [4] and following the notation therein, except that the variables are all *disjoint*:

$$\{q_f\}\cdot X_i \;\cup\; \bigcup_{(q,g)\in Q\times\Sigma}\{q\}\cdot {X^i}_{(q,g)} \;=\; \{q_0\} \;\cup\; \bigcup_{\delta(q,g)=(q_1,\ldots,q_k)}\{q_1 1g, \ldots, q_k kg\}\cdot {X^i}_{(q,g)}$$

(Recall that the $\{q_f\}\cdot X_i$ in each of these equations E_i will be 'storing' finite unions of run tree sets.) We then form an additional DTTA A with q_f as its **only** state, and with the transitions: $\delta(q_f, g) := (q_f, q_f)$, $\delta(q_f, s) := q_f$. We form an equation for this DTTA too:

$$\{q_f\} \cdot Z \cup q_f \cdot X_{(q_f,g)} \cup q_f \cdot X_{(q_f,s)} = \{q_f\} \cup \{q_f 1g, q_f 2g\} \cdot X_{(q_f,g)} \cup \{q_f 1s\} \cdot X_{(q_f,s)}$$

Then we add the following equations which express the conditions that $\{q_f\} \cdot Z$ is a subset of all the other $\{q_f\} \cdot X_i$, for the variables X_i coming from the various equations $E_i, i = 1..k$:

$$\{q_f\} \cdot X_i = \{q_f\} \cdot Z \cup \{q_f\} \cdot X_i$$

The rest of the reasoning in [4] can now be carried through: we can solve all these equations simultaneously with non-empty run tree sets for the variables, if and only if there is a common run tree set that can be assigned to the variable Z, i.e., iff the DTTAs $A_i, i = 1 \ldots k$ and A have a non-empty intersection. $\square$

Remark 5. It seems possible to relate the $ACID$-unification problem to the emptiness problem for (a class of) labeled dag automata; and thereby deduce a NEXPTIME upper bound for $ACID$-unification, and subsequently also for the $ACUID$-unification problem. This is a work under way and will be presented elsewhere.

6 Conclusion

Our concern in this paper has been to address the unification problem w.r.t. theories "above AC" which might be useful in practical situations, such as $ACUID$ and $ACUID_l$ which seem well-suited for the analysis of programs specified via set constraints. As we mentioned in the introduction, the notion of a family of homomorphisms lies hidden under the distributive operator in $ACUID$ or $ACUID_l$. Thus the theory $ACUIH^C$ can be situated "somewhere between" $ACUI$, and $ACUID$ or $ACUID_l$; it turns out that $ACUIH^C$-unification is undecidable since it encodes counter-machine configurations reachability. From the proof of this undecidability we deduce that unification modulo $ACUID$ is undecidable if the distributing symbol '$*$' is in addition associative-commutative; if this distributing symbol is only associative, again unification modulo $ACUID$ is undecidable by reduction from the Modified Post Correspondence Problem.

When no further theory on the distributing symbol '$*$' is added on to $ACUID$, the unification modulo $ACUID$ is decidable; and a DEXPTIME lower bound can be obtained. We have also deduced the same lower bound for $ACUID_l$-unification. On the other hand, a reformulation of the $ACUID_l$-unification problem as one modulo $ACUIH$, for a set H of non-commuting homomorphisms, helped us to show that it is NEXPTIME-decidable.

References

1. A. Aiken, E. Wimmers. *Solving Systems of Set Constraints.* In Proc. of the 7th IEEE Symposium on Logic in Computer Science, 1992 (LICS'92), pp. 329 – 340.

2. A. Aiken, D. Kozen,, M. Vardi, E. Wimmers. *The Complexity of Set Constraints.* In Proc. Conf. CSL'93, EACSL, September 1993, pp. 1–18.
3. F. Baader. *Unification in Commutative Theories.* Journal of Symbolic Computation 8:479–497, 1989.
4. F. Baader, P. Narendran. *Unification of Concept Terms in Description Logics.* Journal of Symbolic Computation 31 (3): 277–305, 2001.
5. F. Baader, W. Snyder. *Unification Theory.* In: J.A. Robinson and A. Voronkov, editors, Handbook of Automated Reasoning. Elsevier Science Publishers, 2001.
6. L. Bachmair, H. Ganzinger, U. Waldemann. *Set Constraints are the Monadic Class.* In Proc. of the 8th IEEE Symposium on Logic in Computer Science, 1993 (LICS'93), pp. 75 – 83.
7. W. Charatonik, A. Podelski. *Set Constraints with Intersection.* In Proc. of the 12th IEEE Symposium on Logic in Computer Science, Warsaw 1997 (LICS'97), pp 362 – 372. (To appear in: *Information and Computation*).
8. H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, M. Tommasi. *Tree Automata Techniques and Applications.* http://www.grappa.univ-lille3.fr/tata/
9. R. Focardi *Analysis and Automatic Detection of Information Flows and Network Systems.* (Doctoral Thesis) Technical Report UBLCS-99-16, July 1999, University of Bologna.
10. R. Gilleron, S. Tison, M. Tommasi. *Set Constraints and Tree Automata* Information and Computation 149, pp. 1–41, 1999. (cf. also Technical Report IT 292, Laboratoire-LIFL, Lille, 1996.)
11. M. Minsky. *Computation: Finite and Infinite Machines. Prentice-Hall International*, London, (1972)
12. W. Nutt. *Unification in Monoidal Theories.* In Stickel (Ed.), Proc. CADE-10, Springer-Verlag, LNAI n^o449, pp. 618–632.
13. H. Seidl. *Haskell Overloading is DEXPTIME-complete.* In Information Processing Letters, 52(2):pp. 57–60.