The Complexity of Counting Problems in Equational Matching

Miki Hermann[1,*] Phokion G. Kolaitis[2,†]

[1] CRIN (CNRS) and INRIA-Lorraine, BP 239, 54506 Vandœuvre-lès-Nancy, France. (e-mail: hermann@loria.fr)

[2] Computer and Information Sciences, University of California, Santa Cruz, CA 95064, U.S.A. (e-mail: kolaitis@cse.ucsc.edu)

Abstract

We introduce a class of counting problems that arise naturally in equational matching and study their computational complexity. If E is an equational theory, then #E-Matching is the problem of counting the number of complete minimal E-matchers of two given terms. #E-Matching is a well-defined algorithmic problem for every finitary equational theory. Moreover, it captures more accurately the computational difficulties associated with finding complete sets of minimal E-matchers than the corresponding decision problem for E-matching does.

In 1979, L. Valiant developed a computational model for measuring the complexity of counting problems and demonstrated the existence of #P-*complete* problems, i.e., counting problems that are complete for counting non-deterministic Turing machines of polynomial-time complexity. Using the theory of #P-completeness, we analyze the computational complexity of #E-matching for several important equational theories E. We establish that if E is one of the equational theories A, C, AC, I, U, ACI, Set, ACU, or ACIU, then #E-Matching is a #P-complete problem. We also show that there are equational theories, such as the restriction of AC-matching to linear terms, for which the underlying decision matching problem is solvable in polynomial time, while the associated counting matching problem is #P-complete.

1 Introduction and Summary of Results

Since the pioneering work of Plotkin [Plo72] over twenty years ago, the study of matching and unification modulo a fixed equational theory E has occupied a central place in automated deduction and has found numerous applications to several other branches of computer science, including logic programming, program verification, and database query languages. Researchers in this area have investigated a variety of equational theories and have examined in depth certain algorithmic aspects of matching and unification modulo an equational theory E.

*Partially supported by *Institut National Polytechnique de Lorraine* grant 910 0146 R1.

†Part of the research reported here was carried out while this author was visiting CRIN & INRIA-Lorraine supported by the University of Nancy 1 and INRIA-Lorraine. Research of this author is also supported by a 1993 John Simon Guggenheim Fellowship and by NSF Grant CCR-9108631.

There are two main algorithmic problems arising in the study of E-matching and E-unification. The first is a decision problem, namely, given two terms s and t, decide whether or not there is an E-matcher (or an E-unifier) of s and t. The second problem is to design matching and unification algorithms such that, given two terms s and t, the algorithm terminates and returns a set which is empty, if s and t are not E-matchable (respectively, not E-unifiable), or, otherwise, is a complete set of E-matchers of s and t (respectively, a complete set of E-unifiers). The second problem is, of course, meaningful only for theories for which the first problem is solvable and which, moreover, are *finitary*, i.e., for every term s and t there is a finite set of complete E-matchers (E-unifiers). For such theories, algorithms for the second problem should preferably return a complete set of *minimal* E-matchers (*minimal* E-unifiers).

Benanav, Kapur, Narendran [BKN87] and Kapur, Narendran [KN86] established that the decision problem for E-matching is NP-complete for many important equational theories E, including associativity A, commutativity C, associativity-commutativity AC, and extensions of AC with idempotency I or existence of unit U. Benanav, Kapur, and Narendran [BKN87] discovered also one exception to these NP-completeness phenomena, namely they proved that the decision problem for AC1-matching is solvable in polynomial time, where AC1 is AC restricted to *linear terms*, i.e., every variable occurs at most once in a term being matched. Concerning E-unification, Kapur and Narendran [KN92a] showed that the decision problem for AC-unification is NP-complete, which came as a surprise, since the prevailing intuition is that AC-unification is harder than AC-matching and, thus, this decision problem ought to have complexity higher than NP.

Although it is undoubtedly useful to pinpoint the computational complexity of the underlying decision problem, in practice it is far more important to analyze the complexity of E-matching and E-unification algorithms that return complete sets of (minimal) E-matchers or (minimal) E-unifiers. So far, relatively little progress has been made in deriving tight upper and lower bounds for the complexity of such algorithms. A notable exception is the case of AC-unification for which Kapur and Narendran [KN92b] found an algorithm that runs in doubly exponential time and returns a complete set of AC-unifiers, albeit not necessarily a minimal one. This upper bound is quite tight, since Domenjoud [Dom92] produced a set of AC-unification problems with n variables whose complete set of minimal AC-unifiers has $O(2^{2^n})$ elements.

Assume that E is some finitary equational theory and $\mathcal{A}$ is an algorithm such that, given two terms s and t as input, it returns a complete set of minimal E-matchers of s and t, if s and t can be matched. In this case, the algorithm $\mathcal{A}$ can also be used to compute the cardinality of a complete set of minimal E-matchers. Thus, we are able to solve at the same time a *counting problem* associated with E-matching, namely the problem of counting the number of complete minimal E-matchers. Notice that this problem is always well defined, since it is known (cf. [FH86]) that for every two terms s and t all sets of complete minimal E-matchers of s and t are of the same cardinality. Our goal in this paper is to initiate a systematic study of the computational complexity of *counting problems* in equational matching. We believe that these counting problems are quite natural and that they deserve to be studied in their own right. Moreover, we feel that counting problems reflect more accurately the computational difficulties of equational matching than the corresponding decision problems do.

Counting problems arise naturally in many areas of computer science and combinatorial mathematics. In 1979, Valiant [Val79a] developed a computational model for classi-

fying the complexity of counting problems and introduced the class #P of functions that are computed by a *counting* Turing machine in polynomial time, i.e., a non-deterministic Turing machine that runs in polynomial time and has an auxiliary output device on which it prints in binary notation the number of its accepting computations on a given input. Valiant [Val79a] showed that the class #P has *complete* problems under certain restricted type of reductions that either preserve the number of solutions (*parsimonious* reductions) or, at least, make it possible to compute the number of solutions of one problem from the number of solutions of another problem (*counting* reductions). Quite often, NP-completeness proofs for decision problems can be translated to #P-completeness proofs for the corresponding counting problems by observing that the polynomial transformation in the proof of NP-hardness preserves the number of solutions. In particular, this is the case for #3-SAT, the prototypical #P-complete problem, which asks for the number of satisfying assignments of a 3CNF Boolean formula. On the other hand, Valiant [Val79a] demonstrated the existence of polynomial-time decision problems, such as perfect matching in bipartite graphs, whose associated counting problem is #P-complete. Several other problems were subsequently shown to exhibit this behavior in Valiant [Val79b].

In this paper, we apply the theory of #P-completeness to the study of counting problems in equational matching. If E is a finitary equational theory, then the #E-*Matching Problem* is the problem of computing the cardinality of a complete set of minimal E-matchers of two given terms s and t. We examine several important equational theories E and first show that their #E-Matching problem is a member of the class #P. Usually, membership of a counting problem in #P follows more or less directly from the definition of the problem. This, however, turns out not to be the case with counting problems in equational matching. In fact, proving that a particular #E-Matching problem is in #P often requires extensive use of different syntactic and structural properties of the underlying equational theory E. After deriving upper bounds for the complexity of counting problems in equational matching, we obtain tight lower bounds and, thus, establish that #E-Matching is a #P-complete problem for several equational theories E. In particular, we show the #P-completeness of #A-Matching, #C-Matching, and #AC-Matching. Similar #P-completeness results are obtained for the equational theories of idempotency I, existence of unit U, their AC extensions, and the restriction of ACI to Set matching. We also examine AC1-matching, the restriction of AC-matching to linear terms and establish that #AC1-Matching is #P-complete. This is achieved by showing that the problem of counting the number of perfect matchings in bipartite graphs can be reduced in a parsimonious way to #AC1-Matching. Since AC1-matching has a polynomial-time decision problem (cf. [BKN87]), we have a new manifestation of the phenomenon that a counting problem can be harder than its associated decision problem.

The results reported here on the one hand give a rather complete picture of the complexity of counting problems in equational matching and on the other yield a new family of #P-complete problems of different character than the counting problems studied thus far by researchers in computational complexity.

2 Counting Problems in Equational Matching

In this section, we will define the basic concepts and introduce the family of counting problem arising in equational matching. We also present here a minimum amount of the necessary background material from computational complexity and unification.

2.1 Counting Problems and the Class #P

A *counting Turing machine* is a non-deterministic Turing machine equipped with an auxiliary output device on which it prints in binary notation the number of its accepting computations on a given input. A counting Turing machine has *time complexity* $t(n)$ if the longest accepting computation of the machine over all inputs of size n is at most $t(n)$. By varying the functions $t(n)$, we obtain different complexity classes of counting functions. Thus, #P is the class of functions that are computable by counting Turing machines of polynomial-time complexity. Similarly, #EXPTIME is the class of functions that are computable by counting Turing machines of exponential-time complexity. Counting Turing machines and the complexity class #P were introduced and studied in depth by Valiant [Val79a, Val79b]. Here, we will work with a slightly different, but essentially equivalent, description of the class #P that appears in Kozen [Koz92].

Assume that Σ and Γ are nonempty alphabets and let $w: \Sigma^* \to \mathcal{P}(\Gamma^*)$ be a function from the set Σ^* of strings over Σ to the power set $\mathcal{P}(\Gamma^*)$ of Γ^*. If x is a string in Σ^*, then we refer to $w(x)$ as the *witness set* for x and to the elements of $w(x)$ as *witnesses* for x. Every such *witness function* w can be identified with the following *counting problem* w: given a string x in Σ^*, find the number of witnesses for x in the set $w(x)$. In what follows, $|x|$ is the length of a string x, and $|S|$ is the cardinality of a set S.

Definition 2.1 ([Koz92]) The class #P is the class of counting problems w such that:

(1) There is a polynomial-time algorithm to determine, for given strings x and y, if $y \in w(x)$;

(2) There exists a natural number $k \geq 1$ (which can depend on the counting problem w) such that $|y| \leq |x|^k$ for all $y \in w(x)$.

A typical member of #P is the counting problem #SAT: given a string x encoding a Boolean formula, find the number of truth assignment satisfying x. In this case, the witness set $w(x)$ consists of all truth assignments (encoded as strings) satisfying x.

Counting problems relate to each other via *counting reductions* and *parsimonious reductions*, which are stronger than the polynomial-time reductions between NP-problems.

Definition 2.2 Let $w: \Sigma^* \to \mathcal{P}(\Gamma^*)$ and $v: \Pi^* \to \mathcal{P}(\Delta^*)$ be two counting problems.

A *polynomial many-one counting* (or *weakly parsimonious*) *reduction* from w to v consists of a pair of polynomial-time computable functions $\sigma: \Sigma^* \to \Pi^*$ and $\tau: N \to N$ such that $|w(x)| = \tau(|v(\sigma(x))|)$. A *parsimonious reduction* from w to v is a counting reduction σ, τ from w to v such that τ is the identity function.

A counting problem w is #P-*hard* if there are counting reductions from it to all problems in #P. If in addition w is a member of #P, then we say that w is #P-*complete*.

Proving that a counting problem is #P-hard is viewed as evidence that this problem is truly intractable. Actually, in complexity theory it is generally believed that #P-hard problems are not members of the class FPH, the functional analog of the polynomial hierarchy PH. In particular, no #P-hard problem is known to belong to the class $\mathrm{FP}^{\mathrm{NP}}$ of all functions that are computable in polynomial time using NP oracles (cf. Johnson [Joh90, section 4.1]). Thus, to the extent of course that one can compare decision problems with counting problems, a #P-completeness result suggests a higher level of intractability than an NP-completeness result.

The following #P-complete problems will be of particular use to us in the sequel. Notice that the underlying decision problem for #3-SAT is NP-complete, while for the other two it is solvable in polynomial time. In fact, the decision problem for Positive 2-SAT is trivial, since every positive 2-SAT formula is satisfiable.

#**3-SAT** [Val79b]
Input: Set V of variables and Boolean formula F over V in conjunctive normal form with exactly three literals in each clause.
Output: Number of truth assignments for the variables in V that satisfy F.

#**POSITIVE 2-SAT** (appeared in [Val79b] as #**MONOTONE 2-SAT**)
Input: Set V of Boolean variables and Boolean formula F over V in conjunctive normal form such that each clause of F consists of exactly two positive literals.
Output: Number of truth assignments for the variables in V that satisfy F.

#**PERFECT MATCHINGS** [Val79a]
Input: Bipartite graph G with $2n$ nodes.
Output: Number of *perfect matchings* in G , i.e., sets of n edges such that no pair of edges shares a common node.

2.2 Equational Theories

A *signature* $\mathcal{F}$ is a set of function symbols of designated arities. If $\mathcal{F}$ is a signature and $\mathcal{X}$ is a set of variables, we let $\mathcal{T}(\mathcal{F},\mathcal{X})$ denote the set of all terms over the signature $\mathcal{F}$ and the variables in $\mathcal{X}$. We also write $V(t)$ for the set of variables occurring in a term t. As usual, a *ground term* is a term without variables. A *substitution* is a mapping $\rho{:}\,\mathcal{X} \longrightarrow \mathcal{T}(\mathcal{F},\mathcal{X})$ such that $x\rho = x$ for all but finitely many variables x. Thus, a substitution ρ can be identified with its restriction on the finite set $\mathcal{D}om(\rho) = \{x \in \mathcal{X} \mid x\rho \neq x\}$, which is called the *domain* of ρ. A substitution ρ is *ground* if $x\rho$ is a ground term for all $x \in \mathcal{D}om(\rho)$. Every substitution can be extended to an endomorphism on the algebra of terms.

An *equation* is a pair of terms $l = r$. Each equation is viewed as an *equational axiom*, namely as the first-order sentence $(\forall x_1)\ldots(\forall x_m)(l = r)$ obtained from the equation by universal quantification over all variables occurring in the terms l and r. If E is a set of equational axioms, then the *equational theory* Th(E) *presented by* E is the smallest congruence relation over $\mathcal{T}(\mathcal{F},\mathcal{X})$ containing E and closed under substitutions, i.e., Th(E) is the smallest congruence containing all pairs $l\rho = r\rho$, where $l = r$ is in E and ρ is a substitution. By an abuse of terminology, we will often say "the equational theory E" instead of the correct "the equational theory Th(E) presented by E". We write $s =_{\mathrm{E}} t$ to denote that the pair (s,t) of terms is a member of Th(E).

E-equality on terms can be extended to substitutions by setting $\rho =_{\mathrm{E}} \sigma$ if and only if $(\forall x \in \mathcal{X})(x\rho =_{\mathrm{E}} x\sigma)$. If V is a set of variables and ρ,σ are substitutions, we put $\rho =^V_{\mathrm{E}} \sigma$ if and only if $(\forall x \in V)(x\rho =^V_{\mathrm{E}} x\sigma)$. We also consider the preorder $\leq^V_{\mathrm{E}}$ on substitutions defined by the condition: $\sigma \leq^V_{\mathrm{E}} \rho$ if and only if $(\exists\eta)(\sigma\eta =^V_{\mathrm{E}} \rho)$. In turn, this preorder gives rise to the following equivalence relation $\equiv^V_{\mathrm{E}}$ on substitutions:

$$\rho \equiv^V_{\mathrm{E}} \sigma \quad \Longleftrightarrow \quad \rho \leq^V_{\mathrm{E}} \sigma \quad \text{and} \quad \sigma \leq^V_{\mathrm{E}} \rho.$$

Notice that, in general, $\rho \leq^V_{\mathrm{E}} \sigma$ does not imply that $\rho \equiv^V_{\mathrm{E}} \sigma$, and, by the same token, $\rho \equiv^V_{\mathrm{E}} \sigma$ does not imply that $\rho =_{\mathrm{E}} \sigma$. It is easy to see, however, that these three relations coincide on ground substitutions with the same domain.

In the sequel, we will be concerned with equational theories presented by finite sets E whose axioms are among the following:

Associativity	A(f):	$f(x, f(y, z)) = f(f(x, y), z)$
Commutativity	C(g):	$g(x, y) = g(y, x)$
Idempotency	I(f):	$f(x, x) = x$
Existence of Unit	U(f):	$f(x, 1) = x$
Homomorphism	H(f, g, h):	$f(g(x, y)) = h(f(x), f(y))$
Endomorphism	End(f, g):	$f(g(x, y)) = g(f(x), f(y))$

We will also consider AC1-Matching, which is the restriction of AC-matching to linear terms, and *Set Matching*, i.e., the special case of ACI-matching in which there is only one ACI-symbol and it occurs on the top of the matched terms.

2.3 Counting Matching Problems in Equational Theories

In what follows, let s be a term, let $V = V(s)$ be the set of variables of s, and let t be a ground term. An E-*matcher of* s *and* t is a substitution ρ such that $s\rho =_{\mathrm{E}} t$. If such an E-matcher exists, then we say that the term s E-*matches* the ground term t. The E-*matching problem* is the decision problem to determine, given a term s and a ground term t, whether s E-matches t.

A *complete set of* E-*matchers of* s *and* t is a set S of substitutions such that the following hold: (1) each substitution $\rho \in S$ is an E-matcher of s and t, and, moreover, $\mathcal{D}om(\rho) \subseteq V$; (2) for every E-matcher σ of s and t, there is a substitution $\rho \in S$ such that $\rho \leq^{V}_{\mathrm{E}} s$. We say that S is a *complete set of minimal* E-*matchers of* s *and* t if, in addition, every two distinct members of S are $\leq^{V}_{\mathrm{E}}$-incomparable.

In general, it may be the case that s E-matches t, but there is no complete set of minimal E-matchers of s and t. On the other hand, it is well known that if a complete set of minimal E-unifiers of s and t exists, then it is unique up to $\equiv^{V}_{\mathrm{E}}$.

Proposition 2.3 ([FH86]) *Let s be a term, let t be a ground term, and assume that S_1 and S_2 are two complete sets of minimal* E-*unifiers of s and t. Then there is a one-to-one and onto mapping $f{:}U_1 \to U_2$ such that $f(\rho) \equiv^{V}_{\mathrm{E}} \rho$ for every substitution ρ in U_1, where V is the set of variables of s. As a result, all complete sets of minimal E-matchers of s and t are of the same cardinality.*

From now on, we assume that E is a set of equational axioms such that if s E-matches t, then there exists a complete set of minimal E-matchers of s and t. We let $\mu\mathrm{CSM}_{\mathrm{E}}(s, t)$ denote the (unique up to $\equiv^{V}_{\mathrm{E}}$) complete set of minimal E-matchers of s and t, if s E-matches t, or the empty set, otherwise.

Siekmann and Szabó [SS82] initiated a study of equational theories based on the cardinalities of the sets $\mu\mathrm{CSM}_{\mathrm{E}}(s, t)$. In particular, E-matching is said to be *unitary* if for every term s and every ground term t we have that $|\mu\mathrm{CSM}_{\mathrm{E}}(s, t)| \leq 1$. E-matching is said to be *finitary* if for every term s and every ground term t the set $\mu\mathrm{CSM}_{\mathrm{E}}(s, t)$ is finite. It is well known that AC-matching is finitary, but not unitary.

One might contemplate studying finitary equational theories by classifying them into a hierarchy as follows: the first level of the hierarchy consists of all unitary theories, while the k-th level of it, $k \geq 2$, consists of all theories E such that for every term s and

every ground term t we have that $|\mu\mathrm{CSM}_{\mathrm{E}}(s,t)| \leq k$. This, however, turns out not to be meaningful, because Book and Siekmann [BS86] showed that if E is a set of equational axioms such that E-matching is not unitary, then E-matching is *unbounded*, which means that for every natural number $k \geq 2$ there is a term s and a ground term t such that $|\mu\mathrm{CSM}_{\mathrm{E}}(s,t)| > k$.

If E is a set of equational axioms such that E-matching is finitary, then we associate with E the following *counting problem*:

#**E-MATCHING**
Input: A term s and a ground term t
Output: Cardinality of the set $\mu\mathrm{CSM}_{\mathrm{E}}(s,t)$.

Our goal in this paper is to study the computational complexity of #E-Matching problems for finitary equational theories E. In view of the aforementioned result by Book and Siekmann's [BS86], this appears to be the only reasonable approach to analyzing finitary theories according to the cardinalities of the sets $\mu\mathrm{CSM}_{\mathrm{E}}(s,t)$. Notice that if E-matching is unitary, then the complexity of #E-Matching is essentially the same as the complexity of the decision problem for E-matching. In particular, if BR is the set of equational axioms for Boolean rings, then #BR-Matching can be computed in polynomial time using NP oracles (i.e., it belongs to the class $\mathrm{FP}^{\mathrm{NP}}$), since BR-matching is unitary and its decision problem is in NP (cf. Martin and Nipkow [MN89]).

We investigate the computational complexity of the #E-Matching problem for several finitary, non-unitary equational theories E and in each case we establish that #E-Matching is a #P-complete problem. A proof of #P-completeness has two distinct parts, namely one must first show that the problem at hand is indeed a member of the class #P and then establish that every problem in #P has a counting reduction to it. We undertake each of these tasks separately in the next two sections.

3 Membership of #E-Matching Problems in #P

Quite often, membership of a counting problem in #P follows immediately from the definition of the problem. This is, for example, the case with most #P-complete problems considered by Valiant [Val79a, Val79b], including #SAT, #3-SAT, #Positive 2-SAT, and #Perfect Matchings. In contrast, if E is an arbitrary finitary equational theory, then it is not at all obvious that the associated #E-Matching problem is a member of #P. Actually, as we will soon see, in order to prove that such problems are in #P we must reflect on particular properties of the underlying equational theory E.

In the sequel, we will make systematic use of the description of #P given in Definition 2.1. Thus, for each #E-Matching problem considered here we must find a function w defined on pairs (s,t), where s is a term and t is a ground term, such that the two conditions of Definition 2.1 are fulfilled. At first sight, a natural unambiguous choice for the witness set $w(s,t)$ appears to be the set

$$w(s,t) = \{[\rho]_{\equiv^V_{\mathrm{E}}} \mid \rho \text{ is a member of a complete set of minimal E-matchers for } s \text{ and } t\},$$

where $[\rho]_{\equiv^V_{\mathrm{E}}}$ is the equivalence class of the substitution ρ with respect to the congruence $\equiv^V_{\mathrm{E}}$. However, since the members of a witness set must be strings over some alphabet, this raises the question of how to represent $\equiv^V_{\mathrm{E}}$-equivalence classes by strings. It is clear

that if we take a string consisting of the entire equivalence class (by concatenating all its members), then we may not be able to fulfill the second condition, because for many equational theories E, including AC, some $\equiv_{\mathrm{E}}^{V}$-equivalence classes of matchers of s and t may have exponentially many members (in the size of s and t). Thus, our only alternative is to represent each $\equiv_{\mathrm{E}}^{V}$-equivalence class by a unique *canonical representative* of it and then take as witness set the complete set of minimal E-matchers of s and t that are the canonical representatives of their $\equiv_{\mathrm{E}}^{V}$-equivalence class. In what follows, we show that for many important equational theories it is possible to find canonical representatives such that both conditions of Definition 2.1 are satisfied. These canonical representatives will be defined in a uniform way for the class of *regular theories*. For each specific theory E, however, we will have to use particular properties of E in order to establish that the canonical representatives satisfy the desired conditions.

An equational theory E is *regular* if for every axiom $(l = r) \in E$ we have $V(l) = V(r)$. As Fages and Huet [FH86] put it, "in regular theories variables cannot disappear". All equational theories considered here are regular. In regular theories each matcher must be a ground substitution. Thus, as explained in 2.2, if ρ and σ are E-matchers of the terms s and t, then $(\rho \leq_{\mathrm{E}}^{V} \sigma \iff \rho \equiv_{\mathrm{E}}^{V} \sigma \iff \rho =_{\mathrm{E}} \sigma)$. Using this, one can easily derive the following result, which appeared first in Fages and Huet [FH86, Proposition 4.1].

Proposition 3.1 *Let E be a regular equational theory, let s be a term, and let t be a ground term such that s E-matches t. Then the following are true:*

(1) There exists a complete sets of minimal E-matchers of s and t.

(2) A set S is a complete set of minimal E-matchers of s and t if and only if S is a complete set of E-matchers of s and t such that no two distinct members of S are equal with respect to $=_{\mathrm{E}}^{V}$, where V is the set of variables of s.

(3) A substitution ρ is a member of a complete set of minimal E-matchers of s and t if and only if ρ is an E-matcher of s and t.

Let E be a regular equational theory and assume that for every pair of terms s and t such that s E-matches t we have selected a unique representative from each $\equiv_{\mathrm{E}}^{V}$-equivalence class of E-matchers of s and t, where V is the set of variables of s. If $[\rho]_{\equiv_{\mathrm{E}}^{V}} = [\rho]_{=_{\mathrm{E}}^{V}}$ is such an equivalence class, then we let ρ^* denote its selected representative and we say that ρ^* is the *canonical representative* of $[\rho]_{\equiv_{\mathrm{E}}^{V}}$. Once the canonical representatives have been selected, we can define a *canonical witness function* w^* for #E-Matching such that the witness set $w^*(s,t)$ consists of the canonical representatives of all equivalence classes of E-matchers of s and t. By combining Definition 2.1 with Proposition 3.1, we obtain the following useful sufficient criterion for membership in #P.

Proposition 3.2 *Let E be a regular equational theory and let w^* be a canonical witness function for #E-Matching such that the following conditions hold:*

Condition (1a). *There is a polynomial-time algorithm to determine, given a term s, a ground term t, and a ground substitution ρ, whether ρ is an E-matcher of s and t.*

Condition (1b). *There is a polynomial time algorithm to determine, given a term s, a ground term t, and an E-matcher ρ of s and t, whether ρ is the canonical representative of its $\equiv_{\mathrm{E}}^{V}$-equivalence class $[\rho]_{\equiv_{\mathrm{E}}^{V}}$.*

Condition (2). *There is a natural number $k \geq 1$ such that $|\rho^*| \leq (|s| + |t|)^k$ for all canonical representatives ρ^* in $w^*(s,t)$.*

Then the #E-Matching problem is a member of the class #P.

We now examine specific equational theories and for each of them we show that it is possible to find canonical witness functions such that the above conditions are satisfied.

3.1 Verifying Conditions (1a) and (1b)

Notice that Condition (1a) amounts to having a polynomial-time algorithm to determine, given a term s, a ground term t, and a ground substitution ρ, whether $s\rho =^{V}_{\mathrm{E}} t$. Thus, Condition (1a) is automatically satisfied by every equational theory E for which E-equality of terms (i.e., $t_1 \stackrel{?}{=}_{\mathrm{E}} t_2$) can be tested in polynomial time.

Benanav, Kapur, and Narendran [BKN87] showed that AC-equality can be tested in polynomial time by reducing this problem to the existence of a matching of given size in bipartite graphs. As pointed out in Kapur and Narendran [KN86], ACI-equality can also be tested in polynomial time by using the same algorithm and removing identical arguments of an idempotent function after flattening terms. It follows that Set-equality can be checked in polynomial time as well. By removing units instead of identical arguments in flattened terms, we can check ACU-equality in polynomial time. ACIU-equality can be checked in polynomial time by combining ACI-equality testing with ACU-equality testing. Another modification of the AC-equality method can be used to test for C-equality in polynomial time. For this, instead of testing for the existence of a given size matching, we test for the existence of perfect matchings in a graph.

A-equality can be tested in polynomial time by first reducing the terms to the right-associative form and then checking for syntactic equivalence. I-equality can be tested in polynomial time by first reducing each term to its I-normal form using the leftmost innermost strategy and then checking the reduced terms in I-normal form for syntactic equivalence. The same approach works also for U-equality and, hence, can be extended to IU-equality. H-equality (assuming $g \neq h$) can be tested in polynomial time by a similar method with the leftmost outermost strategy. This extends also to ACH.

We now focus on Condition (1b). We will first define the canonical representatives of the equivalence classes $[\rho]_{\equiv^{V}_{\mathrm{E}}}$ for an arbitrary regular theory E and then show that for each specific equational theory studied here there is a polynomial-time algorithm to find the canonical representative ρ^* of the class $[\rho]_{\equiv^{V}_{\mathrm{E}}}$ from a given member ρ of it.

Recall that if E is a regular theory, s is a term, and t is a ground term, then every E-matcher of s and t is a ground substitution. If we have a *total precedence* $\succ$ on the signature $\mathcal{F}$, then the *lexicographic path ordering* $\succ_{lpo}$ induced by it is a well-founded ordering that is total on ground terms [Der87]. Thus, we can compare two E-matchers $\rho = [x_1 \mapsto t_1, \ldots, x_n \mapsto t_n]$ and $\rho' = [x_1 \mapsto t'_1, \ldots, x_n \mapsto t'_n]$ by comparing them lexicographically through the terms t_i and t'_i, provided the sequence of variables x_1, $\ldots$, x_n is fixed. Hence, there exists a well-ordering relation $\succ^{lex}_{lpo}$ on E-matchers and so we can choose the $\succ^{lex}_{lpo}$-smallest E-matcher in each equivalence class as its *canonical representative*. The above definition of canonical representatives is uniform for all regular theories. What turns out, however, to be different for each specific theory considered here is the algorithm for computing the canonical representatives in polynomial time.

For certain equational theories E it is possible to orient appropriately their axioms and obtain a *convergent rewrite system*, i.e., a rewrite system that is both *confluent* and *terminating*. More specifically, if we introduce the rewrite rules

$$\begin{array}{llcllllcl} A: & f(f(x,y),z) & \to & f(x,f(y,z)) & \qquad & I: & f(x,x) & \to & x \\ H: & f(g(x,y)) & \to & h(f(x),f(y)) & & U: & f(x,1) & \to & x, \end{array}$$

then the corresponding rewrite systems for A, I, U, IU, and H are convergent. Thus, if E is one of A, I, U, IU, or H, and the substitution $\rho = [x_1 \mapsto t_1, \ldots, x_n \mapsto t_n]$ is an E-matcher of the terms s and t, then for each term t_i we can find its *normal form* $t_i{\downarrow_R}$, where R is the corresponding convergent rewrite system. This normal form exists, because R is terminating, and is unique, because R is confluent. Thus, the substitution $[x_1 \mapsto t_1{\downarrow_R}, \ldots, x_n \mapsto t_n{\downarrow_R}]$ coincides with the canonical representative of $[\rho]_{\equiv_{\mathrm{E}}^V}$.

The algorithms for determining E-equality for A, I, U, IU, and H can be also used to find the canonical representatives of ground substitutions in polynomial time. The leftmost innermost strategy is complete for I, U, and IU, since we cannot create new redexes below a position at which we apply a rewrite rule and each application of a rewrite rule eliminates one occurrence of an E-symbol. The leftmost outermost strategy is complete for H, provided $g \neq h$, since each application of the rewrite rule eliminates one occurrence of the symbol g. Moreover, in each case the number of steps in the derivation is linear in the size of the input substitution.

The above method does not work for AC or for C. For the case of AC, we use the following polynomial-time algorithm to find canonical representatives. Given an AC-matcher $\rho = [x_1 \mapsto t_1, \ldots, x_n \mapsto t_n]$, transform each term t_i to its right-associative form and flatten it, obtaining this way the flattened term $\bar{t}_i$. The flattening is done from left to right, i.e., if f is an AC-symbol, then a subterm $f(s_1, f(s_2, f(\ldots f(s_{k-1}, s_k))))$ in the right-associative form is flattened to $f(s_1, \ldots, s_k)$. After this, for each subterm of $\bar{t}_i$ headed by an AC-symbol we sort the immediate subterms in a bottom-up way. This means that a flattened subterm $f(s_1, \ldots, s_k)$ is permuted to $f(s_{\pi(1)}, \ldots, s_{\pi(k)})$, where $s_{\pi(k)} \succ_{lpo} s_{\pi(k-1)} \succ_{lpo} \cdots \succ_{lpo} s_{\pi(2)} \succ_{lpo} s_{\pi(1)}$. This way, we obtain the flattened and sorted term t_i^*, $1 \leq i \leq n$, and, hence, the representative ρ^*. The same method, without transformation to the right-associative form and without flattening, works for C. Transformation to the right-associative form is polynomial, flattening is polynomial (cf. [BKN87, KN92a]), sorting of terms is polynomial, the lexicographic path ordering is computed in polynomial time [Sny93], and the number of AC-symbols (C-symbols) is limited by the size of the input ρ. Thus, the algorithm runs in polynomial time.

The AC extensions of the rewrite systems I, U, IU, and H are confluent and terminating modulo AC (although the termination must be proved by an AC-compatible ordering). Thus, we can proceed hierarchically. First, we rewrite an E-matcher ρ to the AC-normal form ρ^+ with respect to the confluent and terminating rewrite system R modulo AC. The system R corresponds to one for I, U, IU, or H. Then the E-matcher ρ^+ is transformed to ρ^* using the previously described method for AC. This hierarchical method is correct, since rewriting modulo AC means rewriting AC-equivalence classes. Thus, we have R-equivalence classes and within them AC-equivalence classes.

3.2 Verifying Condition (2)

An equational theory E is *permutative* if for each axiom $(l = r) \in E$ the multisets of symbols in l and r are equal. The theories A, C, AC are permutative, whereas if a theory contains idempotency I, unit U, homomorphism H, or endomorphism End as one of its axioms, then it is not permutative. If E is a permutative theory and s, t are terms such that $s =_{\mathrm{E}} t$, then $|s| = |t|$. It follows that if E is a permutative theory, s is a term, and t is a ground term such that s E-matches t, then for every E-matcher ρ of s and t we have that $|\rho| \leq |t|$, since $s\rho =_{\mathrm{E}} t$. Thus, Condition (2) holds for every permutative theory. By

combining the above remarks with our findings in the preceding Section 3.1, we obtain our first result establishing membership of several #E-Matching problems in #P.

Theorem 3.3 *Let E be one of the equational theories* A, C, AC. *Then the #E-Matching problem is in the class #P.*

In order to analyze non-permutative theories, we turn again to the rewrite systems used in the previous Section 3.1. We say that an equational theory E is *simplifying* if for every axiom $(l = r) \in$ E, the term r is a proper subterm of the term l. By the same token, we say that a rewrite system R is *simplifying* if for every rewrite rule $(l \to r) \in R$, the term r is a proper subterm of the term l. If E and Q are two equational theories, then E is Q-*simplifying* if for every axiom $(l = r) \in$ E $-$ Q there exists a proper subterm l' of l, such that $l' =_Q r$. A rewrite system R is Q-*simplifying* if for every rewrite rule $(l \to r) \in R$ there exists a proper subterm l' of l such that $l' =_Q r$.

Clearly, the theories I, U, and IU are simplifying, while the theories ACI, ACU, and ACIU are AC-simplifying. It is also clear that if a rewrite step $l \longrightarrow_R r$ is carried out in a simplifying rewrite system R, then $|r| < |l|$. Using these facts, it is not hard to verify that Condition (2) holds for each of the equational theories I, U, IU, and their AC extensions. This completes the proof of our second result about membership in #P.

Theorem 3.4 *Let E be one of the equational theories I, U, IU, Set, ACI, ACU, and ACIU. Then the #E-Matching problem is in the class #P.*

Finally, we consider the homomorphism axiom H and its AC extension. Although the homomorphism rewrite rule $f(g(x, y)) \to h(f(x), f(y))$ increases the size of terms, each application of it eliminates one occurrence of the symbol g. This makes it possible to derive a bound on the size of canonical representatives, namely one can easily show that if ρ is an H-matcher of s and t, then $|\rho^*| \leq 2|t|$. Moreover, a similar bound can be obtained for ACH-matchers. Thus, we have established the following result.

Theorem 3.5 *#H-Matching and #ACH-Matching are both in the class #P.*

The rewrite rule $f(g(x, y)) \to g(f(x), f(y))$ for Endomorphism does not eliminate the occurrence of g. It is an open problem whether #End-Matching is in the class #P.

4 #P-Hardness of #E-Matching Problems

In this section, we derive lower bounds for the complexity of #E-Matching problems.

Theorem 4.1 *Let E be one of the equational theories A, C, AC. Then #E-Matching is a #P-complete problem.*

Proof: (*Sketch*) As shown in Theorem 3.3, each of these counting problems is a member of #P. Benanav, Kapur, and Narendran [BKN87] showed that A-matching and C-matching are NP-hard decision problems by reducing 3-SAT to these problems. These reductions turn out to be parsimonious and, thus, #A-Matching and #C-Matching are #P-hard problems. Benanav, Kapur, and Narendran [BKN87] also showed that AC-matching is an NP-hard decision problem by reducing Monotone 3-SAT to AC-matching,

where Monotone 3-SAT is the satisfiability decision problem for 3CNF Boolean formulas in which each clause consists either of only positive literals or of only negative literals. The reduction given in [BKN87] is not parsimonious. We show that #AC-Matching is a #P-hard problem by producing a parsimonious reduction of #Positive 2-SAT to it.

Consider an instance of #Positive 2-SAT with variables $X = \{x_{11}, x_{12}, \ldots, x_{n1}, x_{n2}\}$ and clauses $C = \{c_1, \ldots, c_n\}$, where $c_i = x_{i1} \vee x_{i2}$. Let f be an AC-symbol, let g be a n-ary function symbol that is neither associative nor commutative, and let 1 and 0 be two constant symbols that will be used to simulate the truth or falsity of a Boolean variable. We also let $Y = \{y_1, \ldots, y_n\}$ be a set of new variables, one for each clause, such that $X \cap Y = \emptyset$. With each clause $c_i = x_{i1} \vee x_{i2}$, we associate the term $f(x_{i1}, x_{i2}, y_i)$ and the ground term $f(1, 1, 0)$. We let s denote the term $g(f(x_{11}, x_{12}, y_1), \ldots, f(x_{n1}, x_{n2}, y_n))$ and let t denote the ground term $g(f(1, 1, 0), \ldots, f(1, 1, 0))$. In the full paper we show that the number of truth assignments satisfying $c_1 \wedge \cdots \wedge c_n$ is equal to the cardinality of the set $\mu\mathrm{CSM}_{\mathrm{AC}}(s, t)$. $\Box$

Recall that AC1-matching is the restriction of AC-matching to terms in which each variable occurs at most once. Benanav, Kapur, and Narendran [BKN87] showed that AC1-matching can be decided in polynomial time. In contrast, we show next that the counting problem associated with AC1-matching is harder than its decision problem

Theorem 4.2 *#AC1-Matching is a #P-complete problem.*

Proof: (*Sketch*) Membership of #AC1-matching in #P is a direct consequence of membership in #P of #AC-matching. We will show that #AC1-matching is #P-hard by producing a parsimonious reduction from #Perfect Matchings to #AC1-Matching.

Suppose that we are given a bipartite graph $G = (S, T, E)$ with $2n$ nodes, where $S = \{s_1, \ldots, s_n\}$ and $T = \{t_1, \ldots, t_n\}$ is the partition of the nodes. Let a be a constant symbol, f a unary function symbol, g a (n+1)-ary function symbol that is neither associative nor commutative, and h an AC-symbol. We also consider the sets of variables $X = \{x_{ij} \mid i, j = 1, \ldots, n\}$ and $Y = \{y_1, \ldots, y_n\}$, where $X \cap Y = \emptyset$.

With each node s_i in the set S we associate the term $s_i^* = g(s_i^1, \ldots, s_i^n, s_i^{n+1})$, where

$$s_i^j \quad = \quad \begin{cases} f(x_{ii}) & \text{if } 1 \leq i, j \leq n \text{ and } i = j \\ x_{ij} & \text{if } 1 \leq i, j \leq n \text{ and } i \neq j \\ y_i & \text{if } 1 \leq i \leq n \text{ and } j = n + 1 \end{cases}$$

Intuitively, we view the nodes $s_1, \ldots, s_n$ in S as vectors of a "matrix":

$$\begin{aligned} s_1^* &= g(f(x_{11}), x_{12}, x_{13}, \ldots, x_{1n}, y_1) \\ s_2^* &= g(x_{21}, f(x_{22}), x_{23}, \ldots, x_{2n}, y_2) \\ &\vdots \\ s_n^* &= g(x_{n1}, x_{n2}, \ldots, x_{n,n-1}, f(x_{nn}), y_n) \end{aligned}$$

in which the subterms $f(x_{ii})$ occupy the main diagonal, while the variables $y_1, \ldots, y_n$ are along the last column. With each node t_i in T we associate the ground term $t_i^* = g(t_i^1, \ldots, t_i^n, t_i^{n+1})$, where

$$t_i^j \quad = \quad \begin{cases} f(a) & \text{if } (s_j, t_i) \in E \\ a & \text{otherwise} \end{cases}$$

Thus, we view the nodes $t_1, \ldots, t_n$ in T as vectors of another "matrix"

$$\begin{aligned} t_1^* &= g(t_1^1, \ldots, t_1^n, f(a)) \\ t_2^* &= g(t_2^1, \ldots, t_2^n, f^2(a)) \\ &\vdots \\ t_n^* &= g(t_n^1, \ldots, t_n^n, f^n(a)). \end{aligned}$$

The intuition behind the second matrix is that it represents the adjacency matrix (extended by the column of $f^k(a)$'s) of the edge relation E of the graph G, where the terms $f(a)$ and a are used to simulate the values 1 and 0, respectively.

In the full paper we show that for each i and j, $1 \leq i, j \leq n$, there is an edge $(s_i, t_j) \in E$ if and only if the term s_i^* AC1-matches the ground term t_j^*. As a result, the above is a parsimonious reduction of #Perfect Matchings to #AC1-Matching. □

We obtain next lower bounds for #E-Matching, where E is idempotency I, or unit U, or one of the AC extensions of I and U. For this, we need to consider yet another counting problem, whose #P-completeness was proved in Creignou and Hermann [CH93].

#**1-in-3-SAT** [CH93]
Input: Set V of variables and Boolean formula F over V in conjunctive normal form with exactly three literals in each clause.
Output: Number of truth assignments for the variables in V such that they make true exactly one literal in each clause.

A parsimonious reduction from #3-SAT to #1-in-3-SAT is obtained by replacing each 3-clause $c_i = l_{i1} \vee l_{i2} \vee l_{i3}$ by the clauses $c_{i1} = l_{i1} \vee x_{i2} \vee x_{i3}$, $c_{i2} = \bar{l}_{i2} \vee x_{i2} \vee y_{i2}$, $c_{i3} = \bar{l}_{i3} \vee x_{i3} \vee y_{i3}$, and $c_{i4} = x_{i3} \vee y_{i2} \vee z_i$.

Theorem 4.3 *Let E be one of the equational theories I, U, IU, Set, ACI, ACU, and ACIU. Then #E-Matching is a #P-complete problem.*

Proof: (*Hint*) By Theorem 3.4, each of these counting problems is a member of #P. The #P-hardness of #I-Matching, #U-Matching, and #ACU-Matching is proved by producing parsimonious reductions from #1-in-3-SAT. The reduction to #U-Matching and to #ACU-Matching is an adaptation of the NP-hardness reduction of U-unification and ACU-unification, given in [TA87]. The #P-hardness of IU-Matching and #ACIU-Matching is established by exhibiting parsimonious reductions from Positive 2-SAT to each of these problems. Finally, the reduction from 3-SAT to Set-matching in [KN86] is linear and, hence, weakly parsimonious. Thus, #ACI-Matching is #P-hard as well. □

Finally, we obtain #P-hardness results for the counting problems associated with the AC extensions of homomorphism H and Endomorphism End. For this, we produce parsimonious reductions from #Positive 2-SAT. Details are given in the full appear.

Theorem 4.4 *#ACH-Matching is a #P-complete problem and #ACEnd-Matching is a #P-hard problem.*

5 Concluding Remarks

In this paper we introduced and studied the class of #E-Matching problems, which are the counting problems that ask for the cardinalities of complete sets of minimal E-matchers in some finitary equational theory E. Using the theory of #P-completeness, we identified the complexity of #E-Matching problems for several equational theories E. The table below summarizes our findings and compares the complexity of counting problems in equational matching with the complexity of the corresponding decision problems.

Theory	Decision	Counting	Theory	Decision	Counting
A	NP-complete	#P-complete	I	NP-complete	#P-complete
C	NP-complete	#P-complete	U	NP-complete	#P-complete
AC	NP-complete	#P-complete	IU		#P-complete
AC1	P	#P-complete	ACI	NP-complete	#P-complete
ACH		#P-complete	Set	NP-complete	#P-complete
ACEnd		#P-hard	ACU	NP-complete	#P-complete
BR	NP-complete	FP^{NP}	ACIU		#P-complete

Although in most cases the NP-completeness of the decision problem is accompanied by the #P-completeness of the associated counting problem, it should be emphasized that in general there is no relation between the complexities of these two problems, as manifested by the results about AC1 and BR.

The work presented here suggests that a similar investigation should be carried out for #E-Unification problems, i.e., counting problems that ask for the cardinalities of the complete sets of minimal E-unifiers in some finitary equational theory E. Although our results imply that several #E-unification problems are #P-hard, we already know that there are equational theories E, such as AC, for which #E-Unification is not a member of #P. Indeed, Domenjoud [Dom92] found AC-unification problems with n variables whose complete set of minimal AC-unifiers has $O(2^{2^n})$ elements. Since a counting problem in #P takes values that are bounded by a single exponential in the size of the input, it follows that #AC-Unification is not in #P. It is an interesting open problem to analyze the computational complexity of #AC-Unification and to determine whether it is complete for some higher counting complexity class, such as #EXPTIME or #PSPACE. Results along these lines will delineate the computational difference between matching and unification in a precise manner and will confirm the intuition that unification is harder than matching.

References

[BKN87] D. Benanav, D. Kapur, and P. Narendran. Complexity of matching problems. *Journal of Symbolic Computation*, 3:203–216, 1987.

[BS86] R. Book and J. Siekmann. On unification: Equational theories are not bounded. *Journal of Symbolic Computation*, 2:317–324, 1986.

[CH93] N. Creignou and M. Hermann. On #P-completeness of some counting problems. Research report 93-R-188, Centre de Recherche en Informatique de Nancy, 1993.

[Der87] N. Dershowitz. Termination of rewriting. *Journal of Symbolic Computation*, 3(1 & 2):69–116, 1987. Special issue on Rewriting Techniques and Applications.

[Dom92] E. Domenjoud. Number of minimal unifiers of the equation $\alpha x_1 + \cdots + \alpha x_b =_{AC} \beta y_1 + \cdots + \beta y_q$. *Journal of Automated Reasoning*, 8:39–44, 1992.

[FH86] F. Fages and G. Huet. Complete sets of unifiers and matchers in equational theories. *Theoretical Computer Science*, 43(1):189–200, 1986.

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

[KN86] D. Kapur and P. Narendran. NP-completeness of the set unification and matching problems. In J.H. Siekmann, editor, *Proceedings 8th International Conference on Automated Deduction (CADE'86), Oxford (England)*, volume 230 of *Lecture Notes in Computer Science*, pages 489–495. Springer-Verlag, July 1986.

[KN92a] D. Kapur and P. Narendran. Complexity of unification problems with associative-commutative operators. *Journal of Automated Reasoning*, 9:261–288, 1992.

[KN92b] D. Kapur and P. Narendran. Double-exponential complexity of computing a complete set of AC-unifiers. In *Proceedings 7th IEEE Symposium on Logic in Computer Science (LICS'92), Santa Cruz (California, USA)*, pages 11–21, 1992.

[Koz92] D.C. Kozen. *The design and analysis of algorithms*, chapter 26: Counting problems and #P, pages 138–143. Springer-Verlag, 1992.

[MN89] U. Martin and T. Nipkow. Boolean unification — the story so far. *Journal of Symbolic Computation*, 7(3 & 4):275–294, 1989.

[Plo72] G.D. Plotkin. Building-in equational theories. In B. Meltzer and D. Mitchie, editors, *Machine Intelligence*, volume 7, pages 73–90. Edinburgh University Press, Edinburgh, UK, 1972.

[Sny93] W. Snyder. On the complexity of recursive path orderings. *Information Processing Letters*, 46:257–262, 1993.

[SS82] J. Siekmann and P. Szabó. Universal unification and classification of equational theories. In D.W. Loveland, editor, *Proceedings 6th International Conference on Automated Deduction (CADE'82), New York (New York, USA)*, volume 138 of *Lecture Notes in Computer Science*, pages 369–389. Springer-Verlag, June 1982.

[TA87] E. Tidén and S. Arnborg. Unification problems with one-sided distributivity. *Journal of Symbolic Computation*, 3(1 & 2):183–202, 1987.

[Val79a] L.G. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8(2):189–201, 1979.

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