Knowledge Problems vs Unification and Matching: Dichotomy Results

Serdar Erbatur
University of Texas at Dallas, Richardson, TX, USA

Andrew M. Marshall
University of Mary Washington, Fredericksburg, VA, USA

Paliath Narendran
University at Albany, SUNY, Albany, NY, USA

Christophe Ringeissen
Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France

Abstract

The research area of cryptographic protocol analysis contains a number of innovative algorithms and procedures for checking various security properties of protocols. Most of these procedures focus on solving one of several "knowledge problems" that model intruder knowledge. Solving these problems can demonstrate the ability of the intruder to obtain some forbidden information of the protocol, such as secret keys. Two important examples of these problems are the deduction problem and the static equivalence problem.

Deduction is concerned with the ability to derive a term from a set of terms (or knowledge) obtained from the observation of a protocol instance. Static equivalence, on the other hand, is concerned with distinguishing between two runs of a protocol based on two sets of knowledge. These two knowledge problems at first inspection appear to be very close to the older automated reasoning problems of matching and unification. However, this first impression is wrong, and there have been a few results that have shown theories where one problem, such as unification, is undecidable but another problem, such as deduction, is decidable. These existing dichotomy results were, however, incomplete, and not all cases had been examined, thus leaving the possibility of some connection between the problems for those unexamined cases.

In this paper, we consider the missing dichotomy cases. For each of the remaining cases, we demonstrate a theory that separates the two problems. In addition, once the dichotomy results are completed, it leaves open the question of the existence of non-trivial classes of theories for which all four of the problems are decidable. One example for which this is true is the well-known class of subterm convergent term rewrite systems. In this paper, we develop another example, a class of restrictive permutative theories for which all problems are likewise decidable.

2012 ACM Subject Classification Theory of computation → Equational logic and rewriting

Keywords and phrases Knowledge Problems, Unification, Matching, Decidability

Digital Object Identifier 10.4230/LIPIcs.FSCD.2025.18

1 Introduction

The research area of cryptographic protocol analysis contains a number of innovative algorithms and procedures for checking various security properties of protocols, see, for example, [2, 8, 11, 12, 14]. These procedures consider protocols modeled in a symbolic way, typically via a rewrite system or equational theory. Most of these procedures focus on solving one of several knowledge problems which model adversary knowledge. Solving these problems can demonstrate the ability of the adversary to obtain some forbidden knowledge of the protocol such as secret keys. Two of the most important problems are deduction and static equivalence [2, 12]. The problem of deduction considers the ability of an adversary to observe one or more protocol sessions and decide if a specific term is derivable from this

observed knowledge. More specifically, given a substitution, σ, representing knowledge of an adversary, the problem asks if there exists a term, ζ, such that $\zeta\sigma =_E t$, where t is a ground "target" term. The second problem is concerned with deciding whether two runs of a protocol can be distinguished. This is critical in cases such as voting protocols, where the ability to distinguish between, for example, *yes* and *no* votes could violate the security of the protocol. Here, given two substitutions, σ_1 and σ_2, representing two runs of a protocol and two terms t and s, we would like to know if $t\sigma_1 =_E s\sigma_1$ but $t\sigma_2 \neq_E s\sigma_2$. That is, the terms distinguish the runs of the protocol.

Interestingly, these knowledge problems seem to be closely related to the older problems of unification and matching. For example, matching seems closely related to deduction. In matching we are given the terms and must find the substitution whereas in deduction you are given the substitution and must find the term. Likewise, unification seems to be closely related to static equivalence. However, recent results [15, 10, 21, 5] have shown that the problems, although related, are not equivalent. For example, they have demonstrated theories for which one problem is decidable, such as deduction, but another is undecidable, such as unification. Furthermore, there are still some cases that remain open: in particular, the unification problem vs. the static equivalence problem, and matching vs. static-equivalence. We consider those missing cases in this paper showing that they are not equivalent either and thus completing the dichotomy results separating the knowledge problems from unification and matching.

In addition, we consider the classes of theories for which all the problems (i.e., knowledge, unification, and matching) are decidable. We know of some classes of theories for which this is the case, such as subterm convergent term rewrite systems. Here we demonstrate a new class of equational theories for which all these problems are also decidable.

Paper Outline

The remainder of the paper is organized as follows. Section 2 contains the preliminaries and background material, including the definitions of the knowledge problems. Section 3 reviews the previous results and then completes the final dichotomy results comparing the knowledge problems to unification and matching. Section 4 develops a class of equational theories for which all the considered problems are decidable. Finally, Section 5 includes the conclusions, remaining open problems, and future work.

2 Preliminaries

We use the standard notation of equational unification [7] and term rewriting systems [6]. Given a first-order signature Σ and a (countable) set of variables V, the Σ-terms over variables V are built in the usual way by taking into account the arity of each function symbol in Σ. Each Σ-term is well-formed: if it is rooted by a n-ary function symbol in Σ, then it has necessarily n direct subterms. For any term t, $|t|$ denotes the number of symbols occurring in t. The set of Σ-terms over variables V is denoted by $T(\Sigma, V)$. The set of variables from V occurring in a term $t \in T(\Sigma, V)$ is denoted by $Var(t)$. A term t is *ground* if $Var(t) = \emptyset$. A term t is *linear* if each variable in $Var(t)$ occurs only once in t. For any position p in a term t (including the root position ε), $t(p)$ is the symbol at position p, $t|_p$ is the subterm of t at position p, and $t[u]_p$ is the term t in which $t|_p$ is replaced by u. A substitution is an endomorphism of $T(\Sigma, V)$ with only finitely many variables not mapped to themselves. A substitution is denoted by $\sigma = \{x_1 \mapsto t_1, \ldots, x_m \mapsto t_m\}$, where the domain

of σ is $Dom(\sigma) = \{x_1, \ldots, x_m\}$ and the range of σ is $Ran(\sigma) = \{t_1, \ldots, t_m\}$. Application of a substitution σ to t is written $t\sigma$. A Σ-equation is a pair of Σ-terms denoted by $s =^? t$ or simply $s = t$ when it is clear from the context that we do not refer to an axiom.

Let V_C denote a finite set of context variables (sometimes called holes) such that $V_C \cap V = \emptyset$. We use the notation $\diamond_i$ to represent the context variable $\diamond_i \in V_C$. A *context* is a *linear* term containing only context variables. More formally, a context is a term, $C \in T(\Sigma, V_C)$, where each variable (context hole) occurs at most once. Thus, the size of a context follows from the size of a term, where any hole occurrence counts for 1. Given a context C with m variables and m terms $S_1, \ldots, S_m \in T(\Sigma, V)$, $C[S_1, \ldots, S_m]$ denotes the term $C\{\diamond_1 \mapsto S_1, \ldots, \diamond_m \mapsto S_m\}$ where for $i = 1, \ldots, m$, $\diamond_i$ denotes the i-th context variable occurring in the term C, and C is called the *context part* of $C[S_1, \ldots, S_m]$. When $S_1, \ldots, S_m \in \mathcal{S}$, $C[S_1, \ldots, S_m]$ is said to be a *context instantiated with terms in* $\mathcal{S}$.

2.1 Equational Theories

Given a set E of Σ-axioms (i.e., pairs of terms in $T(\Sigma, V)$, denoted by $l = r$), the *equational theory* $=_E$ is the congruence closure of E under the law of substitutivity (by a slight abuse of terminology, E is often called an equational theory). Equivalently, $=_E$ can be defined as the reflexive transitive closure $\leftrightarrow_E^*$ of an equational step $\leftrightarrow_E$ defined as follows: $s \leftrightarrow_E t$ if there exist a position p of s, $l = r$ (or $r = l$) in E, and substitution σ such that $s|_p = l\sigma$ and $t = s[r\sigma]_p$. An axiom $l = r$ is *regular* if $Var(l) = Var(r)$. An axiom $l = r$ is *collapse-free* if l and r are non-variable terms. An equational theory is *regular* (resp., collapse-free) if all its axioms are regular (resp., collapse-free). An equational theory E is said to be *permutative* if for any $l = r$ in E, the number of occurrences of any (function or variable) symbol in l is equal to the number of occurrences of that symbol in r. Well-known theories such as Associativity ($A = \{(x + y) + z = x + (y + z)\}$), Commutativity ($C = \{x + y = y + x\}$), and Associativity-Commutativity ($AC = A \cup C$) are permutative theories. A theory E is said to be *shallow* if variables can only occur at a depth at most 1 in axioms of E. For example, C is shallow but A and AC are not. Any constant in Σ not occurring in E is said to be *free*.

▶ **Definition 1** (Unification). *Given a signature Σ, an equational theory E, and a set of Σ-equations, $\Gamma = \{s_1 =^? t_1, \ldots, s_n =^? t_n\}$. The E-unification decision problem asks if there exists a substitution σ such that $s_i\sigma =_E t_i\sigma$ for all $1 \leq i \leq n$. If E is presented as a rewrite relation R, then we ask if there exists a substitution σ such that $s_i\sigma \leftrightarrow_R^* t_i\sigma$ for all $1 \leq i \leq n$.*

Another important algorithmic problem is *matching*, which can be seen as a restricted form of the unification problem.

▶ **Definition 2** (Matching). *Given a signature Σ, an equational theory E, and a set of Σ-equations, $\Gamma = \{s_1 =^? t_1, \ldots, s_n =^? t_n\}$. The E-matching decision problem asks if there exists a substitution σ such that $s_i\sigma =_E t_i$ for all $1 \leq i \leq n$. If E is presented as a rewrite relation R, then we ask if there exists a substitution σ such that $s_i\sigma \leftrightarrow_R^* t_i$ for all $1 \leq i \leq n$.*

2.2 Rewrite Relations

Given a signature Σ, an oriented Σ-axiom is called a rewrite rule of the form $l \to r$ such that $l, r \in T(\Sigma, V)$, l is not a variable and $Var(r) \subseteq Var(l)$. A finite set of rewrite rules is called a *term rewriting system* (TRS, for short). Let R be any TRS. For any Σ-terms s and t, s *R-rewrites* to t, denoted by $s \to_R t$, if there exist a position p of s, $l \to r \in R$, and substitution σ such that $s_{|p} = l\sigma$ and $t = s[r\sigma]_p$. The term s is said to be *R-reducible*, $s_{|p}$ is called a *redex*, and in the particular case where $s_{|p} = l\sigma$, s *R-rewrites* to t, denoted by $s \to_R t$.

A term is an *innermost redex* if none of its proper subterms is a redex. The symmetric relation $\leftarrow_R \cup \rightarrow_R$ is denoted by $\longleftrightarrow_R$. The rewrite relation $\rightarrow_R$ is confluent if $\longleftrightarrow_R^*$ is included in $\rightarrow_R^* \circ \leftarrow_R^*$. The rewrite relation $\rightarrow_R$ is *convergent* if $\rightarrow_R$ is both terminating and confluent. When $\rightarrow_R$ is convergent, we have that for any terms t, t', $t \longleftrightarrow_R^* t'$ iff $t{\downarrow_R} = t'{\downarrow_R}$, where $t{\downarrow_R}$ (resp., $t'{\downarrow_R}$) denotes the unique normal form of t (resp., t') w.r.t $\rightarrow_R$. A TRS R is said to be *optimally reducing* if for any $l \rightarrow r$ in R and any substitution θ, the following holds: ($t\theta$ is R-irreducible for any strict subterm t of l) implies that $r\theta$ is R-irreducible. A TRS is *linear* if for any $l \rightarrow r$ in R, both l and r are linear. A convergent TRS R is said to be *subterm convergent* if for any $l \rightarrow r \in R$, r is either a strict subterm of l or a constant. We refer to [6] for the classical notion of overlap and critical pair. Hence, a trivial critical pair is obtained by an overlap at the root position of a rule with a renaming of the same rule. A TRS with no non-trivial critical pairs is *non-overlapping*.

2.3 Knowledge Problems

The applied pi calculus and frames are used to model attacker knowledge [3]. In this model, the set of messages or terms which the attacker knows, and which could have been obtained from observing one or more protocol sessions, are the set of terms in $Ran(\sigma)$ of the frame $\phi = \nu\tilde{n}.\sigma$, where σ is a substitution ranging over ground terms. We also need to model cryptographic concepts such as nonces, keys, and publicly known values. We do this by using *names*, which are essentially free constants. Here also, we need to track the names which the attacker knows, such as public values, and the names which the attacker does not know a priori, such as freshly generated nonces. $\tilde{n}$ consists of a finite set of restricted names; these names represent freshly generated names which remain secret from the attacker. The set of names occurring in a term t is denoted by $fn(t)$. For any frame $\phi = \nu\tilde{n}.\sigma$, let $fn(\phi)$ be the set of names $fn(\sigma)\backslash\tilde{n}$ where $fn(\sigma) = \bigcup_{t \in Ran(\sigma)} fn(t)$; and for any term t, let $t\phi$ denote – by a slight abuse of notation – the term $t\sigma$. For any term t, we say that t satisfies the name restriction of ϕ if $fn(t) \cap \tilde{n} = \emptyset$.

▶ **Definition 3** (Deduction). *Let $\phi = \nu\tilde{n}.\sigma$ be a frame, and t a ground term. We say that t is* deduced from ϕ modulo E, *denoted by $\phi \vdash_E t$, if there exists a term ζ such that $\zeta\sigma =_E t$ and $fn(\zeta) \cap \tilde{n} = \emptyset$. The term ζ is called a* recipe *of t in ϕ modulo E.*

Another form of knowledge is the ability to tell if two frames are *statically equivalent* modulo E.

▶ **Definition 4** (Static Equivalence). *Two terms s and t are* equal *in a frame $\phi = \nu\tilde{n}.\sigma$ modulo an equational theory E, denoted $(s =_E t)\phi$, if $s\sigma =_E t\sigma$, and $\tilde{n} \cap (fn(s) \cup fn(t)) = \emptyset$. The set of all equalities $s = t$ such that $(s =_E t)\phi$ is denoted by $Eq(\phi)$. Given a set of equalities Eq, the fact that $(s =_E t)\phi$ for all $s = t \in Eq$ is denoted by $\phi \models Eq$. Two frames $\phi = \nu\tilde{n}.\sigma$ and $\psi = \nu\tilde{n}.\tau$ are* statically equivalent modulo E, *denoted as $\phi \approx_E \psi$, if $Dom(\sigma) = Dom(\tau)$, $\phi \models Eq(\psi)$ and $\psi \models Eq(\phi)$.*

Deduction and static equivalence are known to be decidable for subterm convergent rewrite systems [2].

2.4 Graph Embedded TRS

It will be useful in one of the dichotomy results that follows to use a restricted class of term rewrite systems for which it is already known that the static equivalence problem is decidable. These systems were introduced in [25] and then simplified in [10], where they are

used to expand on the class of subterm convergent term rewrite systems. Actually, we don't need the full class here for the proofs that follow. Therefore, we only introduce the needed components below, which keeps the definition simpler and more compact. In particular, we introduce a subset of contracting TRSs below that omit the permutative component. One can find the full definition in [10].

The class of contracting TRS first starts by modeling the graph minor relation in the TRS setting. This is done by a set of rewrite *schemas*. This set of rewrite schemata then induces a graph-embedded term rewrite system. Notice that this is very similar to what is often done when considering the homeomorphic embedding relation in TRSs.

▶ **Definition 5** (Graph Embedding). *Consider the following reduction relation, $\rightarrow^*_{R_{gemb}}$, where R_{gemb} is the set of rules given by the instantiation of the following rule schema:*

$$\left\{\begin{array}{ll} & \text{for any } f \in \Sigma \\ (1) & f(x_1,\ldots,x_n) \rightarrow x_i \\ (2) & f(x_1,\ldots,\ x_{i-1},\ x_i,\ x_{i+1}\ \ldots,\ x_n) \rightarrow f(x_1,\ldots,\ x_{i-1},\ x_{i+1},\ \ldots,\ x_n) \\ & \text{and for any } f,g \in \Sigma \\ (3) & f(x_1,\ldots,x_{i-1},g(\bar{z}),x_{i+1},\ \ldots,x_m) \rightarrow g(x_1,\ldots,\ x_{i-1},\bar{z},x_{i+1},\ \ldots,x_m) \\ (4) & f(x_1,\ldots,x_{i-1},g(\bar{z}),x_{i+1},\ \ldots,x_m) \rightarrow f(x_1,\ldots,\ x_{i-1},\bar{z},x_{i+1},\ \ldots,x_m) \end{array}\right\}$$

We say a term t' is graph-embedded *in a term t, denoted $t \succcurlyeq_{gemb} t'$, if t' is a well formed term such that $t \rightarrow^*_{R_{gemb}} t'$.*

A TRS R is graph-embedded *if for any $l \rightarrow r \in R$, $l \succcurlyeq_{gemb} r$ or r is a constant.*

The graph embedded class is not sufficient to obtain decidability of static equivalence, so we next need to introduce a type of projection rule that ensures access to subterms and thus decidability of static equivalence.

▶ **Definition 6** (Projecting Rule). *Let R be a TRS, x a variable and t a term with a single occurrence of x. A* projecting rule in R over t leading to x *is a rule in R which is a variant of $t' \rightarrow x$ where t' is a superterm of t with no additional occurrences of x.*

How these projecting rules relate to the entire term-rewrite system can now be defined.

▶ **Definition 7** (Projection-Closed Derivation). *Let R be a TRS. A non-empty R_{gemb}-derivation is said to be* projection-closed with respect to R *if it has the form*

$$s \rightarrow^{l\rightarrow r,\gamma}_{R_{gemb}} s' \rightarrow^*_{R_{gemb}} t$$

where

- *s is a linear term and t is a well formed term,*
- *$l \rightarrow r$ is a R_{gemb} rule among (1), (2) and (4),*
- *γ is a substitution ranging over variables,*
- *if $l \rightarrow r$ is rule (1), $f(x_1,\ldots,x_n) \rightarrow x_i$, then for any $x_i\gamma$ occurring in t there exists a projecting rule in R over $f(x_1,\ldots,x_n)\gamma$ leading to $x_i\gamma$,*
- *if $l \rightarrow r$ is rule (4), $f(x_1,\ldots,x_{i-1},g(\bar{z}),x_{i+1},\ldots,x_m) \rightarrow f(x_1,\ldots,x_{i-1},\bar{z},x_{i+1},\ldots,x_m)$, then for any $z_i\gamma$ occurring in t there exists a projecting rule in R over $g(\bar{z})\gamma$ leading to $z_i\gamma$,*
- *$s' \rightarrow^*_{R_{gemb}} t$ is either projection-closed with respect to R or empty.*

These rule can be combined into a subset of the graph-embedded TRS, called contracting, for which the knowledge problems are decidable. However, the contracting definition also allows for a type of restricted permutation. For the results in this paper we don't need to

include the permutation component of the contracting definition and by leaving it out we obtain a strict subset of contracting, introduced below, which is simpler and for which the knowledge problems are decidable.

▶ **Definition 8** (Permutation-free Contracting TRS). *A TRS R is said to be* permutation-free contracting, *$\approx$-free contracting for short, if for any non-subterm rule $l \to r$ in R, there exist a position p of l, a substitution σ ranging over variables, and a projection-closed derivation $g \to^{+}_{R_{gemb}} d$ with respect to R such that $l|_p = g\sigma$ and $r = d\sigma$ is of depth 1.*

▶ **Lemma 9** ([25, 10]). *Deduction and static equivalence are decidable for $\approx$-free contracting TRSs.*

Proof. Follows from [25] where it is shown that the knowledge problems are decidable for contracting TRS, and the fact that $\approx$-free contracting TRS are a strict subset of contracting. ◀

3 Dichotomy Results

It is known that Deduction and Static Equivalence are undecidable in general, see [1, 9]. It was also shown in [1] that static equivalence is not reducible to deduction. Interestingly, it is shown in [1] that deduction can be reduced to static equivalence. However, the proof in [1] requires the addition of a free function symbol. It is unknown if the reduction can be done without the addition. Furthermore, there exist theories for which deduction is decidable but static equivalence is not, see, for example, [9].

In this section we want to compare the knowledge problems to the well-known unification and matching problems. Actually, this work has already been started, motivated by the close similarities in the problems. Previous work has provided dichotomy results that compare a knowledge problem to either unification or matching, demonstrating a theory for which one is decidable but the other is not. Thus, before beginning the new results, let us quickly review these existing results. After the review, we then complete the picture by developing the final needed results comparing the problems.

Considering each of the possible cases:

1. Unification vs deduction and static equivalence.
   a. *There exist theories for which unification is undecidable but deduction is decidable.* It's shown in [10] that deduction is decidable in permutative theories but in [21] unification is shown to be undecidable for permutative theories.
   b. *There exist theories for which unification is decidable but deduction is undecidable.* In [5] a theory is developed for which unification is decidable but deduction is undecidable.
   c. *There exist theories for which unification is decidable but static equivalence is undecidable.* **Shown in this paper.**
   d. *There exist theories for which unification is undecidable but static equivalence is decidable.* **Shown in this paper.**
2. Matching vs deduction and static equivalence.
   a. *There exist theories for which matching is undecidable but deduction is decidable.* It is shown in [10] that deduction is decidable in permutative theories but in [21] unification is shown to be undecidable for permutative theories. Actually, the problem used in the reduction is a matching problem, so this also demonstrates a theory for which matching is undecidable.
   b. *There exist theories for which matching is decidable but deduction is undecidable.* This is due to the unification case (1b) above.

c. *There exist theories for which matching is decidable but static equivalence is undecidable.* Due to the unification case (1c) above.

d. *There exist theories for which matching is undecidable but static equivalence is decidable.* **Shown in this paper.**

We now proceed to consider each of the remaining dichotomy problems in turn.

3.1 Remaining Dichotomy Problems

Problem: Unification Decidable and Static Equivalence Undecidable

For this problem, we are comparing the unification problem to the problem of static equivalence, showing that the problems are not comparable. That is, we show that there exist theories for which unification is decidable and static equivalence is not. In particular, we will be using the theory of optimally reducing, linear, and convergent TRS.

In [5] it is shown that there exists an optimally reducing, linear, and convergent TRS for which the cap problem is undecidable. Since the cap problem is a special case of the deduction problem [10], the result developed in [5] provides the following:

▶ **Lemma 10** ([5]). *Deduction is undecidable in general for optimally reducing, linear, non-overlapping and convergent TRS.*

The result is obtained by using a reduction from the halting problem for 2-counter Minsky machines [5]. We modify that reduction here to extend the result from the deduction problem to the static equivalence problem.

A 2-counter Minsky machine is a counter machine that is equipped with 2 non-negative counters that can be incremented, decremented and checked for equality with zero. Since these machines are Turing complete, the *zero halting problem* – whether the machine will halt in an accepting state with both counters at 0 – is undecidable. A configuration of a 2-counter machine can be represented as a triple (C_1, q_l, C_2), where q_l is the label of the next instruction to be executed and C_i, $i = 1, 2$, are the current counter values. As shown in [5] the operations on such a machine can be modeled by a TRS. We modify their construction a little. For any of the finite many states, let q_l be a unary function symbol and represent state l. Let f and s be unary function symbols, and let c be a rank 3 function symbol. Let m be a nonce and 0 be a constant representing zero. s is used to represent the successor function, while c is used to encode configurations. For any 2-counter machine, we can encode the initial configuration as $c(s^k(0), q_0(m), s^p(0))$ and the final configuration as $c(0, q_F(m), 0)$. We also assume that the machine makes no moves once it is in the final state F. Then the state transitions of the counter machine can be modeled by the following convergent, linear, and optimally reducing TRS R:

$$
\begin{aligned}
f(c(x, q_l(z), y)) &\to c(s(x), q_{l+1}(z), y)\\
f(c(x, q_l(z), y)) &\to c(x, q_{l+1}(z), s(y))\\
f(c(s(x), q_l(z), y)) &\to c(x, q_{l+1}(z), y)\\
f(c(0, q_l(z), y)) &\to c(0, q_{l'}(z), y))\\
f(c(x, q_l(z), s(y))) &\to c(x, q_{l+1}(z), y))\\
f(c(x, q_l(z), 0)) &\to c(x, q_{l'}(z), 0))
\end{aligned}
$$

The f symbol in the above TRS is used to ensure the optimally reducing requirement. In addition, notice that once the final state F is reached the rewrite derivation comes to an end, and this exactly corresponds to the termination of the counter machine. Let CT stand

for "configuration terms," i.e., ground terms of the form $c(s^i(0), q, s^j(0))$ where q is a state symbol and $S = c(s^k(0), q_0(m), s^p(0))$, $T = c(0, q_F(m), 0)$ respectively be the initial and final configuration terms. It can be shown that T can be deduced from S modulo R if and only if $\exists n : f^n(S) \to_R^! T$ if and only if the Minsky machine reaches the final configuration.

We now show that the static equivalence problem is undecidable for this theory.

▶ **Lemma 11.** *The static equivalence problem is undecidable in general for optimally reducing, linear, non-overlapping and convergent TRSs.*

Proof. Let R be the the optimally reducing, linear, and convergent TRS outlined above. Construct two frames as follows:

1. $\phi_1 = \nu\tilde{n}.\sigma_1$, where $\sigma_1 = \{x_1 \mapsto c(s^k(0), q_0(m), s^p(0)),\ x_2 \mapsto T\}$
2. $\phi_2 = \nu\tilde{n}.\sigma_2$, where $\sigma_2 = \{x_1 \mapsto c(s^k(0), q_0(m), s^p(0)),\ x_2 \mapsto d\}$

where d is a new private name (nonce) in $\tilde{n}$. Thus $\tilde{n} = \{m, d\}$.

Notice that the encoding of the initial configuration of the Minsky machine, that the variable x_1 maps to, is the same in both frames. The only difference between the frames is in the mappings on x_2. Clearly if $f^n(c(s^k(0), q_0(m), s^p(0))) \to_R^! T$ then $x_2 \approx^? f^n(x_1)$ is unified by σ_1, but not by σ_2.

The proof in the other direction – that if there is an equation that is unified by σ_1 but not by σ_2 then T can be deduced from $c(s^k(0), q_0(m), s^p(0))$ – can be proved in a way similar to the proof of Proposition 4 (pages 8–9) of [2].

▶ Claim 1. Let t be a ground term and T be a subterm of t at position p. If t is reducible, then $t[b]_p$ is reducible where b is a free constant.

Proof. Without loss of generality suppose $t = l\theta$ for left-hand side of R. Since there are no moves after the machine reaches the final state, the symbol q_F does not appear on any left-hand side. Thus no nonvariable subterm of t is unifiable with T. ◁

It is also worth noting here that the term rewriting system $R \cup \{T \to b\}$, where b is a free constant, is convergent.

We can make use of the constant abstraction ("cutting") function $\mathbf{cut}_{U,d}$ as defined in [2]

$$\mathbf{cut}_{T,d}(a) = a \quad \text{if } a \text{ is a name or constant}$$

$$\mathbf{cut}_{T,d}(g(T_1, \ldots, T_k)) = \begin{cases} d & \text{if } g = c,\ c(T_1, T_2, T_3) =_R T \\ g(\mathbf{cut}_{T,d}(T_1), \ldots, \mathbf{cut}_{T,d}(T_k)) & \text{otherwise} \end{cases}$$

By extension, we also define for any substitution σ, the substitution $\mathbf{cut}_{T,d}(\sigma) = \{x \mapsto \mathbf{cut}_{T,d}(x\sigma)\}_{x \in Dom(\sigma)}$. With $\mathbf{cut}_{T,d}$, we can prove the following:

▶ Claim 2. $M \downarrow_R N$ if and only if $\mathbf{cut}_{T,d}(M) \downarrow_R \mathbf{cut}_{T,d}(N)$.

Proof. The "if" part is straightforward. For the "only if" part, suppose

$$\mathbf{cut}_{T,d}(M) \not\downarrow_R \mathbf{cut}_{T,d}(N) \quad \text{and} \quad [T/d](\mathbf{cut}_{T,d}(M) \downarrow_R [T/d](\mathbf{cut}_{T,d}(N).$$

Let $M' = \mathbf{cut}_{T,d}(M)\downarrow_R$ and let $N' = \mathbf{cut}_{T,d}(N)\downarrow_R$. Now if T is a subterm of M then d occurs in M' since every rule in R is variable-preserving. By the previous claim, if $[T/d](M')$ is reducible then so is M' which leads to a contradiction. ◁

▶ Claim 3. Suppose T cannot be deduced from $S = c(s^k(0), q_0(m), s^p(0))$. For any terms M and N such that $(fn(M) \cup fn(N)) \cap \tilde{n} = \emptyset$, we have that $(M =_R N)\phi_1$ implies $(M =_R N)\phi_2$.

Proof. Assume $M\sigma_1 \downarrow_R N\sigma_1$. By the earlier claim, we have $\mathbf{cut}_{T,d}(M\sigma_1) \downarrow_R \mathbf{cut}_{T,d}(N\sigma_1)$. Let us show that $\mathbf{cut}_{T,d}(M\sigma_1) = M\,\mathbf{cut}_{T,d}(\sigma_1)$ and $\mathbf{cut}_{T,d}(N\sigma_1) = N\,\mathbf{cut}_{T,d}(\sigma_1)$. Suppose by contradiction, say for M, that $\mathbf{cut}_{T,d}(M\sigma_1) \neq M\,\mathbf{cut}_{T,d}(\sigma_1)$. Then there must be a nonvariable subterm M' of M such that $M'\sigma_1 \downarrow_R T$. But this means that T can be deduced from S, which is impossible by assumption. Thus, we have $\mathbf{cut}_{T,d}(M\sigma_1) = M\,\mathbf{cut}_{T,d}(\sigma_1)$ and $\mathbf{cut}_{T,d}(N\sigma_1) = N\,\mathbf{cut}_{T,d}(\sigma_1)$. So, $M\,\mathbf{cut}_{T,d}(\sigma_1) \downarrow_R N\,\mathbf{cut}_{T,d}(\sigma_1)$, equivalently $M\sigma_2 \downarrow_R N\sigma_2$. ◁

Thus ϕ_1 and ϕ_2 are statically equivalent if and only if T cannot be deduced from S. ◀

We need now the second component of the result, that unification is decidable for this TRS. We can rely on the following known result:

▶ **Lemma 12** ([22]). *Unification is decidable for optimally reducing and convergent TRSs.*

Thus we obtain the desired dichotomy result for this section.

▶ **Theorem 13.** *There exist theories for which unification is decidable but static equivalence is undecidable.*

Proof. By Lemma 11 and Lemma 12. ◀

Problem: Unification Undecidable and Static Equivalence Decidable

For this problem we again compare unification to static equivalence. However, now we show that there are theories for which unification is undecidable but static equivalence is decidable. We follow a similar strategy to that in [4], which shows the undecidability of unification for Δ-strong TRSs. We need to modify the reduction so that it models not Δ-strong theories but $\approx$-free contracting . This modification does not impact the unification problem but since the static equivalence problem is decidable for $\approx$-free contracting TRS, we obtain the desired result.

We will use the modified Post Correspondence Problem (MPCP). Let $\Sigma = \{a, b\}$, and let $P = \{(\alpha_i, \beta_i) \mid i = 1, 2, \ldots, n\} \subseteq \Sigma^+ \times \Sigma^+$. Then an instance of the MPCP is defined as follows:

- Given a non-empty string $\alpha_0 \in \Sigma^+$.
- Does there exist indices $j_1, j_2, \ldots, j_m \in \{1, 2, \ldots, n\}$ such that:

$$\alpha_0\alpha_{j_1}, \alpha_{j_2} \ldots \alpha_{j_m} = \beta_{j_1}\beta_{j_2} \ldots \beta_{j_m}$$

To start we need to convert the MPCP into a rewriting problem by interpreting the symbols in Σ of the MPCP as unary function symbols, for each $a \in \Sigma$ we have a function $a^1()$. Next, for any string $w \in \Sigma^*$ we will convert the concatenation operation into function composition, denoted as $\widetilde{w}$ such that:

$$\widetilde{\lambda}(x) = x,\ \widetilde{aw}(x) = a(\widetilde{w}(x)),\ \widetilde{bw}(x) = b(\widetilde{w}(x)).$$

Now, introduce new function symbols, f^3, and n distinct unary symbols, $g_1, \ldots, g_n$.

Let P be an instance of the MPCP, from P construct the following TRS, R_P, For each pair $(\alpha_i,\ \beta_i) \in P$:

$$R_P = R_P \cup \{f(\widetilde{\alpha_i}(x), g_i(y), \widetilde{\beta_i}(z)) \to f(x, y, z)\}$$

These rules can be seen as checking that the strings correspond to a sequence of MPCP blocks. Notice that since the unique g_i symbols prevent critical pairs between the rules of R_P, the TRS is confluent. However, this current system, while convergent, is not a $\approx$-free contracting TRS. Notice that the projection rules of Definition 8 are missing.

Now we need to add a second set of rules, R_C, that ensure the TRS is $\approx$-free contracting. The key requirement is that for any of the rules in R_P that remove the function symbols $a()$, $b()$, and $g_i()$, there must exists another rule in R of the form $l[a(x)] \to x \in R$. Therefore, we introduce $n+2$ new unary function symbols h_i for each $i \in \{1, \ldots, n\}$, and h_a and h_b. Define the TRS R_C as follows:

$$R_C = \{h_i(g_i(y)) \to y \mid f(\widetilde{\alpha_i}(x), g_i(y), \widetilde{\beta_i}(z)) \to f(x,y,z) \in R_P\} \bigcup \\ \{h_a(a(x)) \to x,\ h_b(b(x)) \to x\}$$

Again notice that due to the unique symbols h_i, h_a, and h_b, there are no critical pairs between the rules of R_C.

Finally, form the TRS R_{PC} by combining R_P and R_C: $R_{PC} = R_P \cup R_C$.

▶ **Lemma 14.** *The TRS R_{PC} is $\approx$-free contracting convergent.*

Proof. The convergence follows from the absence of critical pairs between the rules. R_{PC} is $\approx$-free contracting due to the additional rules in R_C. ◀

▶ **Lemma 15.** *Static equivalence is decidable for R_{PC}.*

Proof. From Lemma 14 and Lemma 9. ◀

▶ **Theorem 16.** *There exist theories for which unification is undecidable but static equivalence is decidable.*

Proof. Let P, α_0 be an arbitrary instance of the MPCP. Let R_{PC} be the TRS constructed as above. From Lemma 15 the static equivalence problem is decidable.

Now consider the unification problem

$$f(x, y, \widetilde{\alpha_0}(x)) =_{R_{PC}} f(c, c, c)$$

where c is a new constant. Notice that due to the string α_0, we cannot use the rules contained in R_C for reducing the left-hand side of this problem to the right, since there are no h_{i_j} symbols contained in $\widetilde{\alpha_0}$. Thus, the reduction is done via the rules in R_P. However, as shown in [4] the unification problem above is decidable iff the corresponding MPCP is. ◀

▶ Remark 17. It is interesting to note that there is an interesting parallel to the above result already in the literature. As already cited, in [4] it is shown that the unification problem is undecidable in general for Δ-strong theories. In addition, in [19] a procedure is developed for solving the static equivalence problem for Δ-strong theories. However, there is a difference in the definition of Δ-strong from [4] to [19], for example in the first paper the rules in Δ are required to be dwindling, while in the latter they are not dwindling. Thus, it's not immediately clear if these two definitions are equivalent.

Problem: Matching Undecidable and Static Equivalence Decidable

Next we consider the final dichotomy results that remain, that of matching compared to static equivalence. First, we can note that since we have shown a case for which unification is decidable but static equivalence is undecidable, Theorem 13, we already have the following result due to the fact that if unification is decidable so is matching.

▶ **Corollary 18.** *There exist theories for which matching is decidable but static equivalence is undecidable.*

Next, notice that in the proof of Theorem 16, the right-hand side of the problem, $f(x, y, \widetilde{\alpha_0}(x)) =_{R_{PC}} f(c, c, c)$ is ground. Therefore, what we have is actually a matching problem and thus we also obtain the following result via the same proof.

▶ **Corollary 19.** *There exist theories for which matching is undecidable but static equivalence is decidable.*

4 Decidable Theories

With the completion of the dichotomy results we next want to consider the question of non-trivial classes of theories for which all the above problems, knowledge, matching, and unification, decidable. One such example is the class of subterm convergent TRSs. These TRSs are very useful in the formal analysis of cryptographic protocols since they can model a number of cryptographic primitives. However, they require that the theory can be oriented into a restricted type of TRS. We introduce in this section another non-trivial example, the separate variable-permuting class of theories which are a restricted form of equational theory. We show that all of the above problems are decidable for this particular class of permutative theories.

4.1 Separate Variable-Permuting Theories

Let us begin by introducing the class of variable-permuting theories and then the class of separate variable-permuting (SVP) theories.

▶ **Definition 20** (Variable-Permuting Theory). *An axiom $l = r$ is said to be* variable-permuting *[21] if all the following conditions are satisfied:*

1. *the set of positions of l is identical to the set of positions of r,*
2. *for any non-variable position p of l, $l(p) = r(p)$,*
3. *for any $x \in Var(l) \cup Var(r)$, the number of positions of x in l is identical to the number of positions of x in r.*

An equational theory E is said to be separate variable-permuting *(SVP, for short) if for any $l = r \in E$ we have:*

- *$l = r$ is a variable-permuting axiom,*
- *l and r are rooted by the same function symbol that does not occur elsewhere in E.*

An inner constructor *symbol of E is a function symbol from E that never occurs at the root of any equality in E.*

▶ **Example 21** (Intruder Theory with a Permutative Axiom). Let us consider a theory used in practice to model a group messaging protocol [13]. For this protocol, the theory modeling the intruder can be defined [23] as a combination $R \cup K$ where

$$K = \{keyexch(x, pk(x'), y, pk(y')) = keyexch(x', pk(x), y', pk(y))\}$$

and

$$R = \left\{ \begin{array}{llll} adec(aenc(m, pk(sk)), sk) & \to & m & getmsg(sign(m, sk)) \to m \\ checksign(sign(m, sk), m, pk(sk)) & \to & ok & sdec(senc(m, k), k) \to m \end{array} \right\}.$$

R is a subterm convergent TRS and K is a SVP theory sharing with R the constructor symbol pk.

The class of SVP theories could be sufficient to specify intruder theories where one needs only a permutation of variables expressed via a single axiom, as in Example 21. Compared to a shallow theory like Commutativity, a SVP theory allows us to express a permutation involving terms with constructor symbols. Thus, SVP theories provide good examples for applying the combination methods developed in [16] to the deduction and static equivalence problems in the union of theories sharing only constructor symbols. We believe that SVP theories could be useful to approximate AC, and so to provide an alternative to AC in some cases. The AC theory is clearly very interesting in protocol analysis, for instance to specify the exclusive-or theory, but it is difficult to implement.

Contrary to AC, the decision procedures discussed in the following for the deduction and static equivalence problems in SVP theories are easy to implement in existing systems since they are very similar to the ones known in the class of subterm convergent theories, which is ubiquitous in protocol analysis.

4.2 Knowledge Problems in Separate Variable-Permuting Theories

The deduction problem is known to be decidable for the class of permutative theories [10], and so it is decidable for the class of SVP theories since SVP theories are permutative. The case of static equivalence is more complicated since it is known to be undecidable in permutative theories [15]. Therefore, we need to show that static equivalence becomes decidable when considering the particular case of SVP theories. Static equivalence has been shown to be decidable in shallow permutative theories [17]. However, the SVP theory K is not shallow. As shown below, the approach followed in [17] for the case of shallow permutative theories can be reused for the case of SVP theories. Actually, a decision procedure for static equivalence in any SVP theory E can obtained by computing a finite set of equalities bounded by $|E|$, where $|E| = \max_{\{l \mid l=r \in E\}} |l|$. To define this particular finite set of equalities, we first construct a frame saturation dedicated to SVP theories. Then, the recipes of the (deducible) terms included in that saturation are used to build an appropriate set of bounded equalities.

Let $St(t)$ be the set of subterms of a term t and let $St(\sigma)$ be the set of subterms of a substitution σ defined by $St(\sigma) = \bigcup_{x \in Dom(\sigma)} St(x\sigma)$.

▶ **Definition 22** (Frame Saturation for SVP Theories). *Let E be any SVP theory. For any frame $\phi = \nu\tilde{n}.\sigma$, let $ESt(\sigma) = St(\sigma) \cup \bigcup_{t \in ET} St(t)$ where ET is the set of terms $s[\vec{d}]$ such that $|s| \leq |E|$, $fn(s) \cap \tilde{n} = \emptyset$, $\vec{d} \in St(\sigma)$ and $s[\vec{d}] \leftrightarrow_E^{\epsilon} t$ for some term t.*

The set of terms $sat_E(\phi)$ is the smallest set D such that:

(1) *$Ran(\sigma) \subseteq D$,*

(2) *if $t_1, \ldots, t_n \in D$ and $f(t_1, \ldots, t_n) \in ESt(\sigma)$ then $f(t_1, \ldots, t_n) \in D$,*

(3) *if $t \in D$, $t' \in ESt(\sigma)$, $t =_E t'$, then $t' \in D$,*

For any frame $\phi = \nu\tilde{n}.\sigma$, σ_* is the substitution $\sigma\{\chi_u \mapsto u \mid u \in sat_E(\phi) \backslash Ran(\sigma)\}$ where χ_u is a fresh variable for any $u \in sat_E(\phi) \backslash Ran(\sigma)$, and ϕ_* denotes the frame $\nu\tilde{n}.\sigma_*$. Given a recipe ζ_u for each $u \in sat_E(\phi) \backslash Ran(\sigma)$, the substitution $\{\chi_u \mapsto \zeta_u \mid u \in sat_E(\phi) \backslash Ran(\sigma)\}$ is called a recipe substitution *of ϕ and is denoted by ζ_ϕ. Let $Bt = \{t \mid |t| \leq |E|\}$. The set $Eq^B(\phi)$ is the set of equalities $t\zeta_\phi = t'\zeta_\phi$ such that $(t\zeta_\phi =_E t'\zeta_\phi)\phi$ and $t, t' \in Bt$.*

The two following technical lemmas are similar to the ones developed in [17] for the case of shallow permutative theories.

▶ **Lemma 23.** *Let E be any SVP theory. For any terms s and t satisfying the name restriction, if $s\phi_* =_E t\phi_*$ and $\psi \models Eq^B(\phi)$ then $(s\zeta_\phi)\psi =_E (t\zeta_\phi)\psi$.*

Proof. By induction on $\max(|s|,|t|,|E|) - |E|$.

- Base case: if $\max(|s|,|t|,|E|) = |E|$, then it is true by definition of Eq^B.
- Inductive step: consider s and t are terms such that $\max(|s|,|t|,|E|) > |E|$, $s = f(s_1,\ldots,s_n)$, $t = f(t_1,\ldots,t_n)$, and $s\phi_* =_E t\phi_*$. Then, there exist $f(l_1,\ldots,l_n) = f(r_1,\ldots,r_n) \in E$ and substitutions φ, μ such that
  - for $i = 1,\ldots,n$, $s_i\phi_* =_E l_i\varphi$ and $t_i\phi_* =_E r_i\varphi$. In all these matching problems, note that l_i and r_i are only built over inner constructors. Thus, these matching problems correspond to syntactic matching. For any variable x in $Var(l_i) = Var(r_i)$, only two cases can occur.
    1. $x\varphi$ is a term $u[w_1,\ldots,w_m]$ where $w_j \in Ran(\phi_*)$ for $j = 1,\ldots,m$. In that case, we define $x\mu = u[v_1,\ldots,v_m]$ where $v_j\phi_* = w_j$ for $j = 1,\ldots,m$.
    2. $x\varphi$ is a subterm $u|_p$ $(p \neq \epsilon)$ of a term $u \in Ran(\phi_*)$ such that any subterm $u|_q$ for $\epsilon \leq q < p$ is rooted by a inner constructor. Thus, $u|_p \in Ran(\phi_*)$ and so there exists a variable y such that $y\phi_* = u|_p$. In that case, we define $x\mu = y$.

    With the above definition of μ, we have $\varphi = \mu\phi_*$.
  - Let $i = 1,\ldots,n$. By definition of μ, all the function symbol occurrences in $l_i\mu$ are already in s or in l_i. Thus, $|l_i\mu| \leq \max(|s_i|,|l_i|)$. Since $|s_i| < |s|$ and $|l_i| < |E|$, we have $|l_i\mu| < \max(|s|,|E|)$. Using the same argument, for $i = 1,\ldots,n$, $|r_i\mu| < \max(|t|,|E|)$.

  Hence the induction hypothesis applies to $s_i\phi_* =_E (l_i\mu)\phi_*$ and $t_i\phi_* =_E (r_i\mu)\phi_*$, implying that
  - $(s_i\zeta_\phi)\psi =_E (l_i\mu\zeta_\phi)\psi$ for $i = 1,\ldots,n$,
  - $(t_i\zeta_\phi)\psi =_E (r_i\mu\zeta_\phi)\psi$ for $i = 1,\ldots,n$.

  Thus we have:

$$\begin{aligned}
(s\zeta_\phi)\psi &= f((s_1\zeta_\phi)\psi,\ldots,(s_n\zeta_\phi)\psi)\\
&=_E f((l_1\mu\zeta_\phi)\psi,\ldots,(l_n\mu\zeta_\phi)\psi)\\
&=_E f((r_1\mu\zeta_\phi)\psi,\ldots,(r_n\mu\zeta_\phi)\psi)\\
&=_E f((t_1\zeta_\phi)\psi,\ldots,(t_n\zeta_\phi)\psi)\\
&= (t\zeta_\phi)\psi
\end{aligned}$$

  which completes the result. ◀

▶ **Lemma 24.** *Let E be any SVP theory. For any frames ϕ and ψ, $\phi \approx_E \psi$ iff $\psi \models Eq^B(\phi)$ and $\phi \models Eq^B(\psi)$.*

Proof. (If direction) Let $Eq(\phi)$ be the set of all equalities $s\zeta_\phi = t\zeta_\phi$ such that $(s\zeta_\phi =_E t\zeta_\phi)\phi$.

Consider any $s\zeta_\phi = t\zeta_\phi \in Eq(\phi)$. By definition of ζ_ϕ and ϕ_*, we have $(s\zeta_\phi)\phi =_E s\phi_*$ and $(t\zeta_\phi)\phi =_E t\phi_*$. Since $(s\zeta_\phi)\phi =_E (t\zeta_\phi)\phi$, we obtain $s\phi_* =_E t\phi_*$. Then, by Lemma 23, we get $(s\zeta_\phi)\psi =_E (t\zeta_\phi)\psi$, which means that $\psi \models Eq(\phi)$. In a symmetric way, we show that $\phi \models Eq(\psi)$. Then, we can conclude since $\phi \approx_E \psi$ iff $\psi \models Eq(\phi)$ and $\phi \models Eq(\psi)$. ◀

▶ **Theorem 25.** *Deduction and static equivalence are decidable in any SVP theory.*

Proof. Decidability of deduction follows from the decidability of deduction in permutative theories [10], *SVP* theories being permutative. Static equivalence is decidable thanks to Lemma 24. ◀

The combination method developed in [16] can be directly applied to a union $R \cup E$ where R is a convergent TRS with decidable deduction and static equivalence, and E is a SVP theory. In that case, the function symbols shared by R and E must be constructors for R and inner constructors for E. Notice that with a shallow permutative theory E, constructor symbols can only occur in the ground terms appearing in E, and this is clearly a limitation to the application of the combination method. Thus, considering SVP theories instead of shallow ones allows us to get more significant applications.

▶ **Corollary 26.** *Let R be any convergent TRS where both deduction and static equivalence are decidable, and E any SVP theory such that the function symbols shared by R and E are constructors for R and inner constructors for E. Then, both deduction and static equivalence are decidable in $R \cup E$.*

4.3 Unification and Matching in Separate Variable-Permuting Theories

Finally, let us consider the unification and matching problems in SVP theories. Similar to the static equivalence case, we cannot rely on the decidability of unification in the whole class of permutative theories, since unification have been shown to be undecidable in permutative theories [26] and even in variable permutative theories [15]. Fortunately, SVP theories are examples of theories with the property of being closed by paramodulation, a notion formally introduced in [24, 20, 18]. Actually, the closure by paramodulation follows from the absence of non-variable overlap between two axioms of any SVP theory. Since unification is decidable and finitary in theories closed by paramodulation [24, 20], we have that unification is finitary in SVP theories. Since SVP theories are regular and collapse-free, any unification algorithm in SVP theories leads to a unification algorithm with free symbols, thanks to the combination method known for unification algorithms in regular collapse-free theories [27], the empty (aka free) theory being regular collapse-free. Consequently, unification with free constants, and so matching is also finitary in SVP theories.

▶ **Theorem 27.** *Unification and matching are decidable in any SVP theory.*

Finally, we can put all the above results together to obtain the following.

▶ **Theorem 28.** *Unification, matching, deduction, and static equivalence are decidable for any SVP theory.*

5 Conclusions

In this paper we have completed the dichotomy results comparing the knowledge problems of static equivalence and deduction to matching and unification. In particular, we have completed the final four missing results in this comparison. We summarize the results in Table 1. The fields at the intersection of each possibility either cite the paper in the literature where the entire (or part of) the dichotomy result can be found. Otherwise the cell in the table contains the result *in this paper*.

In addition to completing the above dichotomy results we have developed a non-trivial class of theories, the SVP theories, which are capable of symbolically modeling some cryptographic primitives and for which deduction, static equivalence, unification, and matching are all decidable.

There is still the interesting question of deduction compared to static equivalence. There is a reduction from deduction to static equivalence contained in [1]. However, this requires the addition of a new function symbol to the signature. Thus, this leaves open the question

Table 1 Summary of Dichotomy Results.

		Deduction		Static Equivalence	
		Decidable	Undecidable	Decidable	Undecidable
Unification	Decidable	Theorem 28	[5]	Theorem 28	Theorem 13
	Undecidable	[10] and [21]	[1]	Theorem 16	[1]
Matching	Decidable	Theorem 28	[5]	Theorem 28	Theorem 13
	Undecidable	[10] and [21]	[1]	Corollary 19	[1]

of whether this reduction can be done without the additional symbol or if there it can be shown that there are theories for which deduction can be separated from static equivalence. We plan to investigate these questions in future work.

References

1 Martín Abadi, Mathieu Baudet, and Bogdan Warinschi. Guessing attacks and the computational soundness of static equivalence. In Luca Aceto and Anna Ingólfsdóttir, editors, *Foundations of Software Science and Computation Structures, 9th International Conference, FOSSACS 2006, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2006, Vienna, Austria, March 25-31, 2006, Proceedings*, volume 3921 of *Lecture Notes in Computer Science*, pages 398–412, Berlin, Heidelberg, 2006. Springer. doi:10.1007/11690634_27.

2 Martín Abadi and Véronique Cortier. Deciding knowledge in security protocols under equational theories. *Theor. Comput. Sci.*, 367(1-2):2–32, 2006. doi:10.1016/j.tcs.2006.08.032.

3 Martín Abadi and Cédric Fournet. Mobile values, new names, and secure communication. In Chris Hankin and Dave Schmidt, editors, *Conference Record of POPL 2001: The 28th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, London, UK, January 17-19, 2001*, pages 104–115. ACM, 2001. doi:10.1145/360204.360213.

4 Siva Anantharaman, Hai Lin, Christopher Lynch, Paliath Narendran, and Michaël Rusinowitch. Unification modulo homomorphic encryption. *J. Autom. Reason.*, 48(2):135–158, 2012. doi:10.1007/s10817-010-9205-y.

5 Siva Anantharaman, Paliath Narendran, and Michaël Rusinowitch. Intruders with caps. In Franz Baader, editor, *Term Rewriting and Applications, 18th International Conference, RTA 2007, Paris, France, June 26-28, 2007, Proceedings*, volume 4533 of *Lecture Notes in Computer Science*, pages 20–35. Springer, 2007. doi:10.1007/978-3-540-73449-9_4.

6 Franz Baader and Tobias Nipkow. *Term rewriting and all that.* Cambridge University Press, New York, NY, USA, 1998.

7 Franz Baader and Wayne Snyder. Unification theory. In John Alan Robinson and Andrei Voronkov, editors, *Handbook of Automated Reasoning (in 2 volumes)*, pages 445–532. Elsevier and MIT Press, 2001. doi:10.1016/B978-044450813-3/50010-2.

8 Mathieu Baudet, Véronique Cortier, and Stéphanie Delaune. YAPA: A generic tool for computing intruder knowledge. *ACM Trans. Comput. Log.*, 14(1):4:1–4:32, 2013. doi:10.1145/2422085.2422089.

9 Johannes Borgström. Static equivalence *is* harder than knowledge. *Electronic Notes in Theoretical Computer Science*, 154(3):45–57, 2005. doi:10.1016/j.entcs.2006.05.006.

10 Carter Bunch, Saraid Dwyer Satterfield, Serdar Erbatur, Andrew M. Marshall, and Christophe Ringeissen. Knowledge problems in protocol analysis: Extending the notion of subterm convergent. *CoRR*, abs/2401.17226, 2024. doi:10.48550/arXiv.2401.17226.

11 Rohit Chadha, Vincent Cheval, Stefan Ciobaca, and Steve Kremer. Automated verification of equivalence properties of cryptographic protocols. *ACM Trans. Comput. Log.*, 17(4):23, 2016. doi:10.1145/2926715.

12 Stefan Ciobaca, Stéphanie Delaune, and Steve Kremer. Computing knowledge in security protocols under convergent equational theories. *J. Autom. Reason.*, 48(2):219–262, 2012. doi:10.1007/s10817-010-9197-7.

13 Katriel Cohn-Gordon, Cas Cremers, Luke Garratt, Jon Millican, and Kevin Milner. On ends-to-ends encryption: Asynchronous group messaging with strong security guarantees. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, pages 1802–1819. ACM, 2018. doi:10.1145/3243734.3243747.

14 Jannik Dreier, Charles Duménil, Steve Kremer, and Ralf Sasse. Beyond subterm-convergent equational theories in automated verification of stateful protocols. In Matteo Maffei and Mark Ryan, editors, *Principles of Security and Trust - 6th International Conference, POST 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22-29, 2017, Proceedings*, volume 10204 of *Lecture Notes in Computer Science*, pages 117–140, Berlin, Heidelberg, 2017. Springer. doi:10.1007/978-3-662-54455-6_6.

15 Serdar Erbatur, Andrew M. Marshall, Paliath Narendran, and Christophe Ringeissen. Deciding knowledge problems modulo classes of permutative theories. In Juliana Bowles and Harald Søndergaard, editors, *Logic-Based Program Synthesis and Transformation - 34th International Symposium, LOPSTR 2024, Milan, Italy, September 9-10, 2024, Proceedings*, volume 14919 of *Lecture Notes in Computer Science*, pages 47–63. Springer, 2024. doi:10.1007/978-3-031-71294-4_3.

16 Serdar Erbatur, Andrew M. Marshall, and Christophe Ringeissen. Notions of knowledge in combinations of theories sharing constructors. In Leonardo de Moura, editor, *Automated Deduction - CADE 26 - 26th International Conference on Automated Deduction, Gothenburg, Sweden, August 6-11, 2017, Proceedings*, volume 10395 of *Lecture Notes in Computer Science*, pages 60–76. Springer, 2017. doi:10.1007/978-3-319-63046-5_5.

17 Serdar Erbatur, Andrew M. Marshall, and Christophe Ringeissen. Computing knowledge in equational extensions of subterm convergent theories. *Math. Struct. Comput. Sci.*, 30(6):683–709, 2020. doi:10.1017/S0960129520000031.

18 Serdar Erbatur, Andrew M. Marshall, and Christophe Ringeissen. Non-disjoint combined unification and closure by equational paramodulation. In Boris Konev and Giles Reger, editors, *Frontiers of Combining Systems - 13th International Symposium, FroCoS 2021, Birmingham, UK, September 8-10, 2021, Proceedings*, volume 12941 of *Lecture Notes in Computer Science*, pages 25–42. Springer, 2021. doi:10.1007/978-3-030-86205-3_2.

19 Kimberly A. Gero. *Deciding Static Inclusion for Delta-strong and Omega Delta-strong Intruder Theories: Applications to Cryptographic Protocol Analysis*. PhD thesis, University at Albany–SUNY, 2015.

20 Christopher Lynch and Barbara Morawska. Basic syntactic mutation. In Andrei Voronkov, editor, *Automated Deduction - CADE-18, 18th International Conference on Automated Deduction, Copenhagen, Denmark, July 27-30, 2002, Proceedings*, volume 2392 of *Lecture Notes in Computer Science*, pages 471–485. Springer, 2002. doi:10.1007/3-540-45620-1_37.

21 Paliath Narendran and Friedrich Otto. Single versus simultaneous equational unification and equational unification for variable-permuting theories. *J. Autom. Reason.*, 19(1):87–115, 1997. doi:10.1023/A:1005764526878.

22 Paliath Narendran, Frank Pfenning, and Richard Statman. On the unification problem for cartesian closed categories. *J. Symb. Log.*, 62(2):636–647, 1997. doi:10.2307/2275552.

23 Ky Nguyen. Formal verification of a messaging protocol. Internship report, 2019. Work done under the supervision of Vincent Cheval and Véronique Cortier.

24 Robert Nieuwenhuis. Decidability and complexity analysis by basic paramodulation. *Inf. Comput.*, 147(1):1–21, 1998. doi:10.1006/inco.1998.2730.

25 Saraid Dwyer Satterfield, Serdar Erbatur, Andrew M. Marshall, and Christophe Ringeissen. Knowledge problems in security protocols: Going beyond subterm convergent theories. In Marco Gaboardi and Femke van Raamsdonk, editors, *8th International Conference on Formal Structures for Computation and Deduction, FSCD 2023, July 3-6, 2023, Rome, Italy*, volume 260 of *LIPIcs*, pages 30:1–30:19, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.FSCD.2023.30.

26 Manfred Schmidt-Schauß. Unification in permutative equational theories is undecidable. *J. Symb. Comput.*, 8(4):415–421, 1989. doi:10.1016/S0747-7171(89)80037-9.

27 Katherine A. Yelick. Unification in combinations of collapse-free regular theories. *J. Symb. Comput.*, 3(1/2):153–181, 1987. doi:10.1016/S0747-7171(87)80025-1.