Unification modulo a partial theory of exponentiation

Deepak Kapur*
University of New Mexico
Department of Computer Science
kapur@cs.unm.edu

Andrew Marshall†
University at Albany–SUNY
Computer Science Department
marshall@cs.albany.edu

Paliath Narendran‡
University at Albany–SUNY
Computer Science Department
dran@cs.albany.edu

Modular exponentiation is a common mathematical operation in modern cryptography. This, along with modular multiplication at the base and exponent levels (to different moduli) plays an important role in a large number of key agreement protocols. In our earlier work [5, 6] we gave many decidability as well as undecidability results for multiple equational theories, involving various properties of modular exponentiation. Here, we consider a partial subtheory focussing only on exponentiation and multiplication operators. Two main results are proved. The first result is positive, namely, that the unification problem for the above theory (in which no additional property is assumed of the multiplication operators) is decidable. The second result is negative: if we assume that the two multiplication operators belong to two different abelian groups, then the unification problem becomes undecidable. This result is established using a construction patterned after those employed in [5, 9] by reducing Hilbert's 10^{th} problem to the unification problem.

1 Introduction

With network use and online transactions becoming all pervasive in many applications, especially online shopping, social networking, video-conferencing, group conferencing, and e-voting etc, multi-party and group protocols need to be employed. These protocols are often complex, rich and sophisticated, built as a collection of protocols, whose interaction is often quite complex. Their reliability and security thus become a critical issue, especially in case the protocols use arithmetic operators, such as modular multiplication and exponentiation and boolean operators such as *exclusive-or* [11]. In collaboration with the Maude-NPA team [3], we have developed an approach for analyzing whether a given protocol is vulnerable to specific attacks by modeling the protocol as a state machine and an execution of the protocol as a sequence of state transitions. The search space is explored using unification and narrowing techniques to handle equational properties of the operators used in a protocol.

Modular exponentiation is a common mathematical operation in modern cryptography. This, along with modular multiplication at the base and exponent levels (to different moduli) plays an important role in the El Gamal signature scheme, the Nyberg-Rueppel key agreement protocol (Protocol 5.3 in [2]), and the MTI and Yacobi-Shmuely protocols for public key distribution (Protocols 5.7 and 5.33 in [2]). In our earlier work [5, 6] we gave many decidability as well as undecidability results for multiple equational theories, involving various properties of modular exponentiation. Here, consider a partial subtheory focussing only on exponentiation and multiplication operators.

The axioms of the theory are

$$\begin{aligned} exp(g(X),Y) &= g(X \circledast Y) \\ exp(X * Y,Z) &= exp(X,Z) * exp(Y,Z) \end{aligned}$$

*Partially supported by the NSF grants CNS-0831462 and CNS-0905222
†Partially supported by the NSF grants CNS-0831209 and CNS-0905286
‡Partially supported by the NSF grants CNS-0831209 and CNS-0905286

Maribel Fernandez (Ed.): 24th International Workshop on
Unification (UNIF2010).
EPTCS 42, 2010, pp. 12–23, doi:10.4204/EPTCS.42.2

Here exp is the exponentiation operator and g is exponentiation over a *fixed base,* such as 2^n. The multiplication operators $*$ and $\circledast$ are often modulo a prime p and $p-1$, respectively. The reason for modeling two different exponentiation operators is that in a large majority of protocols, many operations are done using a fixed base. In addition, when specifying such protocols in Maude, as in Maude-NPA, the use of the subsort mechanism can make unification more efficient if the first argument in exp is fixed.

Two main results are proved. The first result is positive, namely, that the unification problem for the above theory (in which no additional property is assumed of the multiplication operators) is decidable. The second result is negative: if we assume that the multiplication operators $*$ and $\circledast$ belong to two different abelian groups, then the unification problem becomes undecidable. This result is established using a construction patterned after those employed in [5, 9] by reducing Hilbert's 10^{th} problem to the theory.

The decidability result uses a novel construction and is discussed in the next three sections. The next section models the equational properties of the above two axioms as an inference system. Section 3 analyzes possible reasons when the unification fails, corresponding to the function clashes, occur-check, and an infinite application of one of the inference rules. Section 4 gives the unification algorithm along with a termination proof. The final section sketches the undecidability proof for the equational theory in which, along with the above two axioms, the multiplication operators come from abelian groups.

2 Inference Rules

Below we present a set of inference rules for unification. Termination of these rules is proved later.

(a) $$\frac{\{U =^? V\} \uplus \mathscr{E}\mathscr{Q}}{\{U =^? V\} \cup [V/U](\mathscr{E}\mathscr{Q})} \qquad \text{if } U \text{ occurs in } \mathscr{E}\mathscr{Q}$$

(b) $$\frac{\mathscr{E}\mathscr{Q} \uplus \{U =^? V*W,\ U =^? X*Y\}}{\mathscr{E}\mathscr{Q} \uplus \{U =^? V*W,\ V =^? X,\ W =^? Y\}}$$

(c) $$\frac{\mathscr{E}\mathscr{Q} \uplus \{U =^? V \circledast W,\ U =^? X \circledast Y\}}{\mathscr{E}\mathscr{Q} \uplus \{U =^? V \circledast W,\ V =^? X,\ W =^? Y\}}$$

(d) $$\frac{\mathscr{E}\mathscr{Q} \uplus \{U =^? exp(V,W),\ U =^? exp(X,Y)\}}{\mathscr{E}\mathscr{Q} \uplus \{U =^? exp(V,W),\ V =^? X,\ W =^? Y\}}$$

(e) $$\frac{\mathscr{E}\mathscr{Q} \uplus \{U =^? g(V),\ U =^? g(W)\}}{\mathscr{E}\mathscr{Q} \cup \{U =^? g(V),\ V =^? W\}}$$

(f) $$\frac{\mathscr{E}\mathscr{Q} \uplus \{U =^? exp(V,W),\ U =^? g(X)\}}{\mathscr{E}\mathscr{Q} \cup \{U =^? g(X),\ V =^? g(V'),\ X =^? V' \circledast W\}}$$

(g) $$\frac{\mathscr{E}\mathscr{Q} \uplus \{U =^? exp(V,W),\ U =^? X*Y\}}{\mathscr{E}\mathscr{Q} \cup \{U =^? X*Y,\ V =^? V_1*V_2,\ X =^? exp(V_1,W),\ Y =^? exp(V_2,W)\}}$$

The variable V' in rule (f) is a fresh variable. Similarly V_1, V_2 in rule (g) are fresh variables. The symbol $\uplus$ stands for *disjoint union*. Furthermore, rules (f) and (g) are applied only when the other rules cannot be applied. The variable U in rule (f) (resp. rule (g)) is called an *(f)-peak* (*(g)-peak.*

A set of equations is said to be *reduced* if none of the inference rules (a) thru (e) are applicable. Thus only rules (f) and (g) are applicable to a reduced system. We define relations $\longrightarrow_f$ and $\longrightarrow_g$ between reduced sets of equations as follows: for reduced sets of equations S_1 and S_2, $S_1 \longrightarrow_f S_2$ (resp., $S_1 \longrightarrow_g S_2$) if and only if S_2 can be obtained from S_1 by applying rule (f) (resp., rule (g)) *once* and then eagerly applying rules (a) thru (e). Clearly, rule (f) decreases the number of *exp* symbols. But (g) introduces new *exp* symbols. Thus termination of the algorithm is not obvious. For simplicity, we assume that the equations deleted while applying the inference are actually put into "cold storage" by a marking strategy.

Before proceeding we will need to define several relations over the variables in terms of equations both marked and unmarked. These will be needed later in this paper:

- $U \succ_b V$ iff there is an equation $U =^? exp(V,W)$.
- $U \succ_e W$ iff there is an equation $U =^? exp(V,W)$.
- $U \succ_{l_*} V$ iff there is an equation $U =^? V * W$. Likewise, $U \succ_{l_\circledast} V$ iff $U =^? V \circledast W$.
- $U \succ_{r_*} W$ iff there is an equation $U =^? V * W$. Likewise, $U \succ_{r_\circledast} W$ iff $U =^? V \circledast W$.
- $U \succ_m V$ iff $U \succ_{l_*} V$ or $U \succ_{r_*} V$.
- $U \succ_g V$ iff there is an equation $U =^? g(V)$.
- $U \succ V$ iff there is an equation $U =^? t$ such that t is a non-variable term that contains V.

Clearly all other relations are sub-relations of $\succ$. For a relation p, let p^+ denote its transitive closure. Let $\sim$ be the reflexive, symmetric, transitive closure of $\succ_b$.

We can also view these relations in terms of graphs, where the nodes are the variables and the edges correspond to the various relations between them[1]. These graphs will be useful in checking for failure conditions during unification. Figure 1 and Figure 2 are example graphs and the resulting transformation after applying an inference rule.

3 Failure Conditions

Detection of failure involves several cases. Some cases are caused by function clashes and can be detected using the following rules:

[1] This method is developed by Tiden and Arnborg in [10].

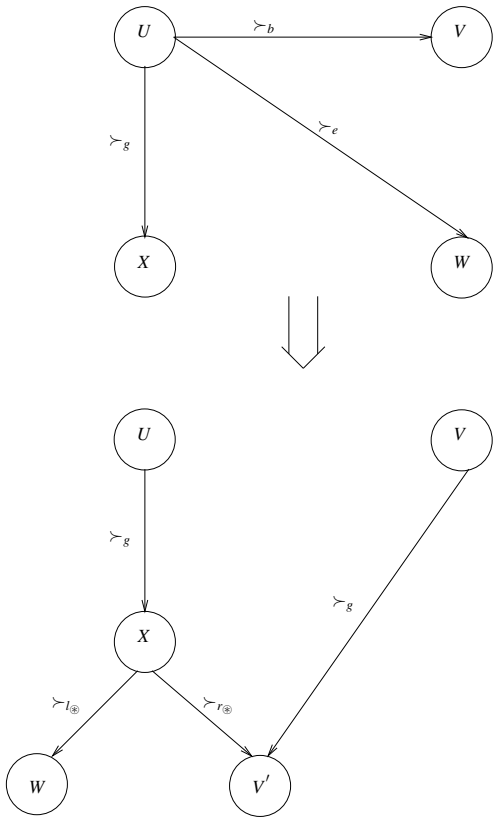

Figure 1: Rule (f)

$$\text{(F1)} \qquad \frac{\mathscr{E}\mathscr{Q} \uplus \{U =^? exp(V,W),\ U =^? X \circledast Y\}}{FAIL}$$

$$\text{(F2)} \qquad \frac{\mathscr{E}\mathscr{Q} \uplus \{U =^? g(V),\ U =^? X \circledast Y\}}{FAIL}$$

$$\text{(F3)} \qquad \frac{\mathscr{E}\mathscr{Q} \uplus \{U =^? g(V),\ U =^? X * Y\}}{FAIL}$$

$$\text{(F4)} \qquad \frac{\mathscr{E}\mathscr{Q} \uplus \{U =^? V \circledast W,\ U =^? X * Y\}}{FAIL}$$

Two other failure cases must be addressed. The first is similar to the "occur check" condition in standard unification. The second is a special case when infinite applications of a rule can happen. Here we use congruence classes over the ground terms, i.e. if t_1 and t_2 are ground terms and $t_1 = t_2$ then they are in the same class.

Lemma 3.1. *Every congruence class over the ground terms is finite. Hence, a term cannot be equivalent to a proper-subterm of it.*

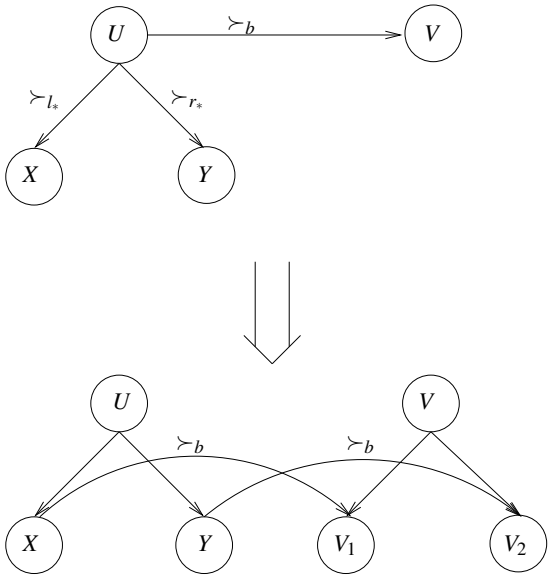

Figure 2: Rule (g): relevant parts

Proof. The fact that the congruence classes are finite is due to the initial system of equations. If a term was equivalent to a proper-subterm this would create infinite congruence classes by allowing continual replacement of the subterm. □

Lemma 3.2. *If there is a variable X such that $X \succ^+ X$ then there is no solution.*

Proof. Follows from Lemma 3.1, this indicates the attempt to unify terms in which one is a proper-subterm of the other, resulting in an occur check failure. □

Next we need to identify cycles between the equivelant classes.

Lemma 3.3. *If there are two variables X and Y such that $X \succ_m Y$ and $Y\ (\sim \cup \succ_m)^+ X$, then there is no solution.*

Proof. We consider the reduction that follows when the *exp* and $\circledast$ functions are interpreted as a projections onto the first argument. The reduction will enable a simpler proof of the result.

Definition 3.4. Let *exp* and $\circledast$ be interpreted as a projection onto the first argument. We define the term $\hat{t}_i$ for any term t_i such that if $t_i = exp(t_{i1}, t_{i2})$ then $\hat{t}_i = \hat{t}_{i1}$. Also, if $t_i = t_{i1} \circledast t_{i2}$ then $\hat{t}_i = \hat{t}_{i1}$.

Consider the Lemma under the interpretation of Definition 3.4. Then any variables related along a $\sim$ edge will become equivalent. Now consider paths along $\succ_m$ edges from equivalent classes formed from $\sim$. By definition there is at least one $\succ_m$ edge from X to Y. We then proceed by induction on the length of the $\succ_m$ path. If no additional $\succ_m$ edges exist we have failure due to $X \succ_m Y$ and $Y \sim X$ $(X = Y)$. Now we can see that adding $\sim$ edges will not effect the unification of the system. we then can assume that we have a cycle of $k\,(0 \leq k)$ $\succ_m$ edges that do not form a unifiable system. That is, a cycle of the form $E_1 \succ_m E_2 \succ_m \cdots \succ_m E_{k+1}$, where each E_i is an equivalence class, $Y \in E_1$, $X \in E_{k+1}$ and $X \succ_m Y$. Then because adding another $\succ_m$ edge would only move X into a lower class we can see the cycle is not unifiable. □

Lemma 3.5. *If* $\{U =^? X*Y,\ V =^? g(Z)\} \subset \mathscr{E}\mathscr{Q}$ *and* $U \sim V$ *then there is no solution.*

Proof. Because of the bi-directional nature of $\sim$ we prove both directions.

First: let $u = x*y$, $v = g(z)$ and $u \succ_b^+ v$. If $u \succ_b v$ we must unify the equations $x*y$ and $exp(v,w)$ but this immediately leads to a function clash due to the need to unify $v = g(z)$ and $v = v_1 * v_2$. We can see that for any path along $\succ_b^+$ we can continue to move the $*$ along the path until eventually we will need to unify $v = g(z)$ and $v = v_1 * v_2$.

Second: Let $u = x*y$, $v = g(z)$, and $v \succ_b^+ u$. Just as in the first direction then we can move the g function along the $\succ_b$ path eventually we will be required to unify $u = x*y$ and $u = g(v')$, a function clash. □

4 Unification Algorithm

First we need a method for detecting "occur check" failure conditions. To accomplish this, we use the methods developed in Tiden and Arnborg [10], building two special graphs to check for failure conditions.

Definition 4.1. Let D be a graph defined on a reduced system of equations. The nodes in the graph correspond to variables in the system. The edges correspond to the parameters of each equation type. See Figure 1.

Lemma 4.2. *If there exists a cycle in D, the set of equations represented by D is not unifiable.*

Proof. Directly from Lemma 3.2. □

We will also need to detect cases requiring an infinite unifier. An example of this is the set of equations comprising $U =^? exp(X,W)$, and $U =^? X*Y$. This example (g)-peak would cause a new (g)-peak creation after each application of Rule (g) (See Figure 3). We will use a propagation graph P to check for these conditions.

Definition 4.3. Let P be a directed simple graph defined on a set of equations as follows: Each vertex in P is a $\sim$-equivalence class. There is an edge between the vertex containing v to the vertex containing w in P, if there is a $\succ_m$ labeled edge from v to w in D.

Lemma 4.4. *If there exists a cycle in P, the set of equations represented by P is not unifiable.*

Proof. Follows from Lemma 3.3. □

We now give a general unification algorithm for unification modulo the partial theory of exponentiation.

Algorithm 1 Unification modulo partial exponentiation

Require: EQ, the set of equations
  while An inference rule can be applied **do**
    Build graphs D and P; if a cycle is found exit with failure.
    If any of rules (F1) through (F4) apply exit with failure.
    Eagerly apply rule (a).
    Eagerly apply rules (b) through (e).
    Apply rules (f) and (g) if possible.
  end while

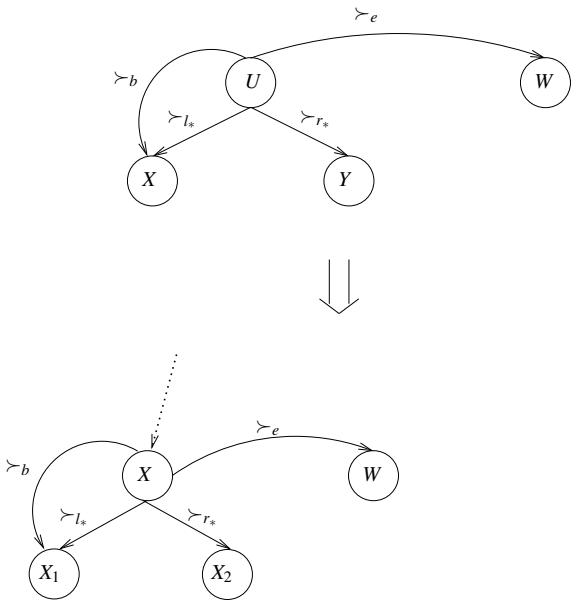

Figure 3: $U = exp(X,W), U = X * Y$

Lemma 4.5. *Rule (f) commutes with rule (g). (See Figure 4)*

Proof. No variable can be an (f)-peak and a (g)-peak at the same time because this would cause failure. Thus, application of rule (g) first does not affect the applicability of rule (f).

□

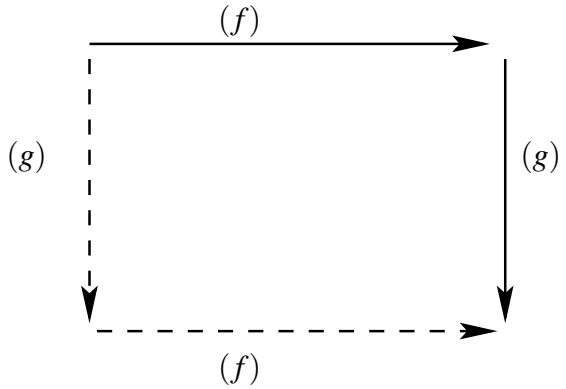

Figure 4: Rule (f) commutes with Rule (g)

Theorem 4.6. *Algorithm 1 always terminates.*

Proof. If a failure condition or cycle in one of the graphs is found, Algorithm 1 will clearly halt. Assume none of these conditions occur. Then some observations can be made: Every $\sim$-congruence class has to have a unique sink (wrt $\succ_b$). Also, applying rule (g) does not increase the number of congruence classes — the new variables V_1 and V_2 are $\sim$-equivalent to X and Y respectively. Now $\succ$ can be used to define a

well-founded partial order on the $\sim$-*congruence classes.* Thus the new *exp* equations created in rule (g) are on congruence classes lower than the earlier one. Applications of rule (g) will thus always terminate under the above assumptions. Since rule (f) can potentially increase the number of congruence classes, we need Lemma 4.5. Since one cannot get an infinite sequence of (g)-steps or (f)-steps, the algorithm terminates. □

5 Undecidability of unification of partial exponentiation with two Abelian group operators

Let us now consider the expanded theory where both $*$ and $\circledast$ are Abelian group operations. That is, we let $*$ represent multiplication modulo a prime p and $\circledast$ represent multiplication modulo $p-1$. We denote this equational theory as $\mathscr{E}_1$ and the resulting AC-convergent system as $\mathscr{R}_1$:

$$
\begin{array}{ll}
X * X^{-1} \to 1 & \\
X * 1 \to X & exp(X,1) \to x \\
(X * Y)^{-1} \to X^{-1} * Y^{-1} & exp(1,Z) \to 1 \\
\left((Z)^{-1}\right)^{-1} \to Z & exp\left(Z^{-1},X\right) \to (exp(Z,X))^{-1} \\
1^{-1} \to 1 & exp(g(X),Y) \to g(X \circledast Y) \\
X \circledast 1 \to X & exp((X * Y),Z) \to exp(X,Z) * exp(Y,Z) \\
X \circledast i(X) \to 1 & \\
i(i(X)) \to X & \\
i(X \circledast Y) \to i(X) \circledast i(Y) &
\end{array}
$$

where $<*,^{-1},1>$ forms the first Abelian group and $<\circledast, i(), 1>$ the second. The unification problem for this system is undecidable. The proof is by reduction from Hilbert's 10^{th} problem (solvability of polynomial equations over the integers). It will be shown that multiplication and addition of a number can be simulated in the above system. We make the assumption for the first part of the proof that we are allowed the distinct free constants b and c. The following proof is a modification of the proof given in [9].

Definition 5.1. Let $\bigcirc_i(u)$ denote

- $\underbrace{u \circledast u \circledast \ldots \circledast u}_{i}$, if $i > 0$.
- $\bigcirc_i(u) = 1$ if $i = 0$ and
- $\underbrace{i(u) \circledast i(u) \circledast \ldots \circledast i(u)}_{i}$, if $i < 0$.

Lemma 5.2. $g(s) =_{\mathscr{E}_1} g(t) \Rightarrow s =_{\mathscr{E}_1} t$.

Lemma 5.3. *For every m, n* $\in \mathbb{Z}$*, the equation:*

$$x * g(\bigcirc_n(b)) =_{\mathscr{E}_1} exp(x,b) * g(\bigcirc_m(b))$$

is solvable.

Proof.

(a) If $n > m$, then $x = g(\bigcirc_{n-1}(b)) * \ldots * g(\bigcirc_m(b))$ is a solution.

(b) If $n < m$, then $x = (g(\bigcirc_n(b)) * \ldots * g(\bigcirc_{m-1}(b)))^{-1}$ is a solution.

(c) If $n = m$, then $x = 1$ is a solution. □

Lemma 5.4. *Let b be a free constant and m be an integer. Then, every solution to*

$$x * g(y) =_{\mathscr{E}_1} exp(x,b) * g(\bigcirc_m(b))$$

is of one of the following forms:

(a) $n > m$, $y = \bigcirc_n(b)$, $x = g(\bigcirc_{n-1}(b)) * \ldots * g(\bigcirc_m(b))$

(b) $n < m$, $y = \bigcirc_n(b)$, $x = (g(\bigcirc_n(b)) * \ldots * g(\bigcirc_{m-1}(b)))^{-1}$

Proof. The proof is by contradiction. Suppose that there exist an integer m and terms t_x and t_y, in normal form modulo $\mathscr{R}_1$, such that

$$t_x * g(t_y) =_{\mathscr{E}_1} exp(t_x,b) * g(\bigcirc_m(b))$$

where $t_y \neq \bigcirc_n(b)$ for any n. Without loss of generality assume also that t_x is a minimal (by size) counterexample, i.e., a minimal term such that $\exists m \exists t_y : t_x * g(t_y) =_{\mathscr{E}_1} exp(t_x,b) * g(\bigcirc_m(b))$.

First of all note that since $\mathscr{R}_1$ is AC-convergent, it must be that

$$t_x * exp(t_x^{-1},b) * g(t_y) \rightarrow^!_{\mathscr{R}_1} g(\bigcirc_m(b)).$$

Then t_x can have two possible forms:

Case 1: $t_x = g(\bigcirc_m(b)) * t_x'$. Then,

$$\begin{array}{rcll} g(\bigcirc_m(b)) * t_x' * g(t_y) & =_{\mathscr{E}_1} & exp(g(\bigcirc_m(b)),b) * exp(t_x',b) * g(\bigcirc_m(b)) & \text{and thus} \\ t_x' * g(t_y) & =_{\mathscr{E}_1} & g(\bigcirc_{m+1}(b)) * exp(t_x',b) & \end{array}$$

Thus t_x' is a smaller counterexample.

Case 2: $t_x = g(\bigcirc_{m-1}(b))^{-1} * t_x'$. Then,

$$\begin{array}{rcll} g(\bigcirc_{m-1}(b))^{-1} * t_x' * g(t_y) & =_{\mathscr{E}_1} & exp(g(\bigcirc_{m-1}(b)),b)^{-1} * exp(t_x',b) * g(\bigcirc_m(b)) & \text{and thus} \\ t_x' * g(t_y) & =_{\mathscr{E}_1} & g(\bigcirc_{m-1}(b)) * exp(t_x',b) & \end{array}$$

Thus t_x' is a smaller counterexample. □

Lemma 5.5. *Let b and c be free constants. Then, the equations*

$$\begin{aligned} exp(x,c) * g(\bigcirc_j(b)) &=_{\mathscr{E}_1} exp(x,b) * g(u) \\ z * g(u) &=_{\mathscr{E}_1} exp(z,c) * g(1) \end{aligned}$$

force u to be equal to $\bigcirc_j(c)$.

Proof. By Lemma 5.4 the second equation, $z * g(u) =_{\mathscr{E}_1} exp(z,c) * g(1)$, forces $u = \bigcirc_n(c)$. Now replacing b with c everywhere in the first equation we get

$$exp(x,c) * g(\bigcirc_j(c)) = exp(x,c) * g(\bigcirc_n(c)).$$

By Lemma 5.2 $\bigcirc_j(c) = \bigcirc_n(c)$ and $j = n$. □

Lemma 5.6. *Let b and c be free constants. Then the equations:*

$$\begin{aligned} exp(x, \bigcirc_k(c)) * g(\bigcirc_j(b)) &=_{\mathscr{E}_1} exp(x,b) * g(u) \\ z * g(u) &=_{\mathscr{E}_1} exp(z,c) * g(1) \end{aligned}$$

force u to be equal to $\bigcirc_{jk}(c)$

Proof. By Lemma 5.4 $u = \bigcirc_n(c)$ as before. Now replacing b by $\bigcirc_k(c)$ we get

$$exp(x, \bigcirc_k(c)) * g(\bigcirc_{jk}(c)) = exp(x, \bigcirc_k(c)) * g(\bigcirc_n(c)).$$

By Lemma 5.2 $\bigcirc_{jk}(c) = \bigcirc_n(c)$ and $n = jk$. □

With Lemma 5.6 we can now simulate multiplication with the natural numbers. To see how this can be done consider $z = x * y$ and let $x = \bigcirc_i(b)$ and $y = \bigcirc_j(b)$. We force $z = \bigcirc_{ij}(b)$ as follows:

$$\begin{aligned} exp(w_1,c) * g(\bigcirc_i(b)) &=_{\mathscr{E}_1} exp(w_1,b) * g(x_2) \qquad \text{and} \\ w_2 * g(x_2) &=_{\mathscr{E}_1} exp(w_2,c) * g(1) \end{aligned}$$

force $x_2 = \bigcirc_i(c)$ by Lemma 5.5.

$$\begin{aligned} exp(w_3,x_2) * g(\bigcirc_j(b)) &=_{\mathscr{E}_1} exp(w_3,b) * g(z_2) \qquad \text{and} \\ w_4 * g(z_2) &=_{\mathscr{E}_1} exp(w_4,c) * g(1) \end{aligned}$$

force $z_2 = \bigcirc_{ij}(c)$ by Lemma 5.6. Finally we copy z_2 to z with the equation

$$exp(w_5,c) * g(z) =_{\mathscr{E}_1} exp(w_5,b) * g(z_2).$$

Lemma 5.7. *Addition of natural numbers can be simulated in* $\mathscr{E}_1$.

Proof. Let $x = \bigcirc_i(b)$ and $y = \bigcirc_j(b)$, where b is a free constant. Then $x \circledast y =_{\mathscr{E}_1} \bigcirc_{i+j}(b)$ □

Theorem 5.8. *Unification over* $\mathscr{E}_1$ *with free constants is undecidable.*

Proof. Following the above outline a unification problem can be constructed that simulates a system of diophantine equations. □

6 Extension and Limitations

In this paper we examined a partial theory of exponentiation, a critical component in several cryptographic protocols. Many of the protocols based on modular exponentiation also contain additional algebraic properties and axioms that could correspond to extensions of this partial exponentiation theory. Therefore, an important question that naturally arises is, how far we can extend the theory and maintain decidability. Unfortunately, additional extensions can quickly result in undecidable unification problems. This was demonstrated when the operations of $\circledast$ and $*$ were allowed to form abelian groups. Therefore, ideally, extensions should maintain decidability while adding additional axioms useful in modeling additional cryptographic protocols. We are currently examining two different possible extensions. The first is allowing just one of either the $\circledast$ or $*$ operations to be abelian. The second is extending the axiom set to include additional algebraic operators such as *modular addition.* Several other papers, including [8, 6, 5, 7], have also considered the unification problem for equational systems that contain some type of exponentiation. For convenience, we give a condensed overview of a selection of these results in Table 1.

Ref	Equational Theory	Unification Problem: Results
[8]	Abelian group with the axioms $exp(x,1) = 1$ and $exp(exp(x,y)z) = exp(x,y*z)$	NP-complete
[6]	Two theories, denoted $\mathscr{E}_1$ and $\mathscr{E}_2$. $\mathscr{E}_1$ consists of an abelian group with operator, $\cdot$, and a monoid with operator $\circ$ with the addition of the axioms: $x^1 = x$, $1^x = 1$, $(x\cdot y)^z = (x^z)\cdot(y^z)$, and $(x^y)^z = x^{y\circ z}$. $\mathscr{E}_2$ adds the axiom $x\circ i(x) = 1$, $i(x)$ being the inverse, to the theory $\mathscr{E}_1$.	Undecidable for both $\mathscr{E}_1$ and $\mathscr{E}_2$
[5]	Two main results: Theory $\mathscr{E}_3$ consists of an abelian group for operator $\cdot$ along with the axioms, $x^1 = x$, $1^x = 1$, and $(x\cdot y)^z = (x^z)\cdot(y^z)$. Theory $\mathscr{E}_4$ consists of $\mathscr{E}_3$ with the addition of a monoid operator $\circ$ and the axiom $(x^y)^z = x^{y\circ z}$.	$\mathscr{E}_3$ is decidable and $\mathscr{E}_4$ is undecidable.
[7]	Two theories, denoted $\mathscr{E}$ and $\mathscr{E}_0$. $\mathscr{E}$ consists of an abelian group with operator, $\cdot$, and the axioms $x^1 = x$, $1^x = 1$, $(x\cdot y)^z = (x^z)\cdot(y^z)$, and $(x^y)^z = x^{y\cdot z}$. $\mathscr{E}_0$ is the same as $\mathscr{E}$ but the axiom $(x^y)^z = x^{y\cdot z}$ is replaced with the axiom $x^{y^z} = x^{z^y}$	$\mathscr{E}$ is undecidable and $\mathscr{E}_0$ is decidable.

Table 1: Results for E-unification with exponentiation.

Most of these results are of high complexity. Therefore, we are also exploring heuristic methods of implementation to enable their integration into the automated protocol analysis system Maude-NPA [4].

References

[1] Franz Baader & Wayne Snyder (2001): *Unification Theory*. In: John Alan Robinson & Andrei Voronkov, editors: *Handbook of Automated Reasoning*, Elsevier and MIT Press, pp. 445–532.

[2] Colin Boyd & Anish Mathuria (2002): *Protocols For Key Establishment And Authentication*. Springer.

[3] S. Escobar, C. Meadows & J. Meseguer (2007): *Equational Cryptographic Reasoning in the Maude-NRL Protocol Analyzer*. In: *Proc. 1st International Workshop on Security and Rewriting Techniques (SecReT 2006)*, ENTCS 171(4), Elsevier, pp. 23–36.

[4] Santiago Escobar, Catherine Meadows & José Meseguer (2009): *Maude-NPA: Cryptographic Protocol Analysis Modulo Equational Properties*. In: *Foundations of Security Analysis and Design V, FOSAD 2007/2008/2009 Tutorial Lectures*, *Lecture Notes in Computer Science* 5705, Springer, pp. 1–50.

[5] Deepak Kapur, Paliath Narendran & Lida Wang (2003): *An E-unification Algorithm for Analyzing Protocols That Use Modular Exponentiation*. In: Robert Nieuwenhuis, editor: *RTA*, *Lecture Notes in Computer Science* 2706, Springer, pp. 165–179. Available at `http://link.springer.de/link/service/series/0558/bibs/2706/27060165.htm`.

[6] Deepak Kapur, Paliath Narendran & Lida Wang (2003): *Undecidability of unification over two theories of modular exponentiation.* In: *Seventeenth International Workshop on Unification (UNIF-2003), Valencia, Spain.*

[7] Deepak Kapur, Paliath Narendran & Lida Wang (2005): *A Unification Algorithm for Analysis of Protocols with Blinded Signatures*. In: *Mechanizing Mathematical Reasoning*, *Lecture Notes in Computer Science* 2605, Springer Berlin / Heidelberg, pp. 433–451.

[8] Catherine Meadows & Paliath Narendran (2002): *A Unification Algorithm for the Group Diffie-Hellman Protocol*. In: *IN PROC. OF WITS 2002*, pp. 14–15.

[9] P. Narendran, F. Pfenning & R. Statman (1993): *On the Unification Problem for Cartesian Closed Categories*. In: *In Proceedings, Eighth Annual IEEE Symposium on Logic in Computer Science*, IEEE Computer Society Press, pp. 57–63.

[10] Erik Tidén & Stefan Arnborg (1987): *Unification Problems with One-Sided Distributivity*. *J. Symb. Comput.* 3(1/2), pp. 183–202.

[11] Max Tuengerthal, Ralf Küsters & Mathieu Turuani (2006): *Implementing a Unification Algorithm for Protocol Analysis with XOR*. *CoRR* abs/cs/0610014. Available at `http://arxiv.org/abs/cs/0610014`.