An E-unification algorithm for analyzing protocols that use modular exponentiation

Technical Report TR-03-2
Department of Computer Science
University at Albany—SUNY
Albany, NY 12222

August 8, 2003

Deepak Kapur[1]
Department of Computer Science
University of New Mexico
Albuquerque, NM 87131

Paliath Narendran[2]
Lida Wang[3]
Department of Computer Science
University at Albany—SUNY
Albany, NY 12222

Abstract

Modular multiplication and exponentiation are common operations in modern cryptography. Unification problems with respect to some equational theories that these operations satisfy are investigated. Two different but related equational theories are analyzed. A unification algorithm is given for one of the theories which relies on solving syzygies over multivariate integral polynomials with noncommuting indeterminates. For the other theory, in which the distributivity property of exponentiation

[1] Research supported in part by the NSF grant nos. CCR-0098114 and CDA-9503064, the ONR grant no. N00014-01-1-0429, and a grant from the Computer Science Research Institute at Sandia National Labs.
[1] Research supported in part by NSF grant no. CCR-0098095 and ONR grant no. N00014-01-1-0430.
[1] Research supported in part by NSF grant no. CCR-0098095.

over multiplication is assumed, the unifiability problem is shown to be undecidable by adapting a construction developed by one of the authors to reduce Hilbert's 10th problem to the solvability problem for linear equations over semi-rings. A new algorithm for computing strong Gröbner bases of right ideals over the polynomial semiring $\mathsf{Z}<X_1, \ldots, X_n>$ is proposed; unlike earlier algorithms proposed by Baader as well as by Madlener and Reinert which work only for right admissible term orderings with the boundedness property, this algorithm works for *any* right admissible term ordering. The algorithms for some of these unification problems are expected to be integrated into Naval Research Laboratory's Protocol Analyzer (NPA), a tool developed by Catherine Meadows, which has been successfully used to analyze cryptographic protocols, particularly emerging standards such as the Internet Engineering Task Force's (IETF) Internet Key Exchange [11] and Group Domain of Interpretation [12] protocols. Techniques from several different fields – particularly symbolic computation (ideal theory and Gröbner basis algorithms) and unification theory — are thus used to address problems arising in state-based cryptographic protocol analysis.

1 Introduction

Modular arithmetic is the mainstay of many modern cryptographic algorithms. Arithmetical operations such as modular multiplication and exponentiation are part of several algorithms. For instance, the RSA and El Gamal algorithms use modular exponentiation for encryption/decryption, whereas the Diffie-Hellman and group Diffie-Hellman algorithms use it for key exchange. Abstracting equational properties of modular exponentiation and designing equational unification algorithms for the resulting theories for use in protocol analysis was initiated in [13]. Building on the work reported in [8], we have so far explored four related unification theories involving modular multiplication. These are described below.

The motivation for studying these equational theories comes from building a tool for analyzing authentication protocols against possible attack. The proposed unification algorithms are planned to be integrated into a software tool called the *NRL Protocol Analyzer* (*NPA*) developed by Catherine Meadows at the Naval Research Laboratory [10]. This tool employs a state-based approach for analyzing attacks on an authentication protocol, and has been used effectively to analyze, e.g., the Internet Engineering Task Force's (IETF) Internet Key Exchange [11], Group Domain of Interpretation [12] protocols, demonstrate known integrity flaws in the Encapsulating Security Protocol [19] (an Internet Standard), and discover a new attack on the Simmons Selective Broadcast Protocol [18]. Currently, the tool uses simple unification (over the empty theory) and a *narrowing* procedure for state exploration. Narrowing is used to find solutions with respect to (henceforth, abbreviated as *wrt*) a set of terminating rewrite rules capturing the semantics of primitive operations used in a protocol, thus simulating unification wrt the associated equational theory. However, if these primitive operations have the associativity and commutativity properties, which is often the case, then narrowing either does not work or is extremely inefficient.

To review the various equational theories discussed in this paper, consider the set of axioms in Figure 1 relating modular multiplication with modular exponentiation. The first four equations (A, C, U, Inv) characterize an Abelian group with $\cdot$ representing multiplication modulo a prime number p; for brevity, this axiomatization is referred by AG. The last three equations characterize an Abelian monoid with $\circ$ standing for multiplication modulo $p-1$. Operators $\cdot$ and $\circ$ share the unit 1. Modular exponentiation is denoted by $\hat{}$; $x\hat{}y$ is also written as x^y. Note also that $Exp5$ implies a 'kind of' associativity for $\circ$, since

$$x^{(u\circ(v\circ w))} =_{Exp5} ((x^u)^v)^w =_{Exp5} x^{((u\circ v)\circ w)}.$$

We have investigated various theories that are obtained by keeping AG fixed and varying the theories that $\circ$ satisfies. Our main results so far are:

$$
\begin{array}{rl}
x \cdot (y \cdot z) = (x \cdot y) \cdot z & A \\
x \cdot y = y \cdot x & C \\
x \cdot 1 = x & U \\
x \cdot x^{-1} = 1 & Inv \\
x^1 = x & Exp1 \\
1^x = 1 & Exp2 \\
(x \cdot y)^z = (x^z) \cdot (y^z) & Exp3 \\
(x^y)^z = (x^z)^y & Exp4 \\
(x^y)^z = x^{y \circ z} & Exp5 \\
x \circ (y \circ z) = (x \circ y) \circ z & A' \\
x \circ y = y \circ x & C' \\
x \circ 1 = x & U'
\end{array}
$$

Fig. 1. Axioms for Modular Exponentiation

(i) The unification problem for the equational theory consisting of AG and axioms $(Exp1, Exp2, Exp3)$ is decidable; this theory is denoted as $\mathcal{E}_3$. This theory is a subset of the equational theories in (ii) and (iii) below. A unification algorithm for this problem is given by reducing it to that of finding constrained solutions of linear equations over $\mathsf{Z}\!<\!X_1, \ldots, X_n\!>$, an algebraic structure similar to a polynomial ring over the integers, except that the indeterminates do not commute (i.e., the multiplication operation on the terms is not commutative — this structure is often called a *monoid semiring*). This latter problem is solved by computing *strong* Gröbner bases[2] of right ideals over $\mathsf{Z}\!<\!X_1, \ldots, X_n\!>$. A new algorithm for computing strong Gröbner bases for right ideals over $\mathsf{Z}\!<\!X_1, \ldots, X_n\!>$ is given; in contrast to the earlier algorithms proposed by Baader [1] for weak Gröbner bases and by Madlener and Reinert [9] for strong Gröbner bases assuming right admissible orderings with the *boundedness property*[3], the proposed algorithm computes strong Gröbner bases using any right admissible ordering.

Techniques employed here for solving linear equations over $\mathsf{Z}\!<\!X_1, \ldots, X_n\!>$ are novel since the variables of the equations being solved have additional constraints of the form that certain indeterminates cannot appear in their solutions (these constraints are related to the occur check condition in simple unification, i.e., over the empty theory[4]). Such constraints are handled by defining somewhat *unusual* kinds of admissible term orderings reflecting such constraints. These term order-

[2] A *weak* Gröbner basis of a (right) ideal simplifies polynomials in the (right) ideal to 0; however, polynomials equivalent with respect to the (right) ideal need not be simplified by the weak Gröbner basis to the *same* canonical form. In contrast, a *strong* Gröbner basis of a (right) ideal simplifies every polynomial to a unique canonical form such that equivalent polynomials with respect to the (right) ideal have the same canonical form; the canonical form of every polynomial in the (right) ideal is 0; see [6, 1, 9] for more details.

[3] An ordering has the boundedness property if and only if for every element, there are only finitely many elements smaller than the element in the ordering.

[4] and also the *linear constant restrictions* introduced by Baader and Schulz [3]

ings are very different from usual term orderings typically used in the literature for computing Gröbner bases of ideals, such as total degree and lexicographic, pure lexicographic, reverse lexicographic as well as block orderings.

(ii) When axiom $Exp4$ is added to $\mathcal{E}_3$, the unification problem for the extended equational theory remains decidable, as shown in [8]; we will denote this theory by $\mathcal{E}_{3,4}$. The unification algorithm uses an algorithm for computing strong Gröbner bases of polynomial ideals in $\mathsf{Z}[x_1, \cdots, x_n]$ developed by one of the authors in 1984 [6]. Linear equations are solved under constraints in a way similar to (i) above. For details, the reader can refer to [8].

(iii) If the equational theory in (i) is extended by adding axioms $Exp5$, A', C' and U' instead of $Exp4$, then the unification problem for this equational theory, denoted by $\mathcal{E}_{3,5}$, is undecidable [20]. (It is easy to see that $Exp4$ follows from C' and $Exp5$, thus implying that $Exp5$ is strictly stronger than $Exp4$ in this context.) The undecidability is mainly due to the distributivity property since if the distributivity axiom $Exp3$ is dropped from this equational theory—thus just keeping $Exp5$—then the unifiability problem is NP-complete [13].

(iv) If the equational theory in (i) is extended by adding axioms $Exp5$, A' and U' (in other words, $\cdot$ forms an Abelian group, whereas $\circ$ forms a *monoid*), denoted by $\mathcal{E}'_{3,5}$, the unification problem is still undecidable. This is shown by reducing Hilbert's tenth problem to the unification problem by adapting a construction developed by one of the authors [14] for showing undecidability of solving linear equations over polynomial semirings. The proof is given in Section 2.

The result mentioned in (ii) above was obtained earlier and is reported in [8] with detailed proofs. The undecidability result mentioned in (iii) is similar to the undecidability result reported in [8].

The paper thus brings together techniques from several different fields, particularly symbolic computation (ideal theory and Gröbner basis algorithms) and unification theory, to address problems arising in state-based cryptographic protocol analysis.

We show in the next section that unifiability for $\mathcal{E}'_{3,5}$ is undecidable. The rest of the paper focuses on the decidability of the unifiability check for $\mathcal{E}_3$. Section 3 relates unification over $\mathcal{E}_3$ to unification over $AGnH$ with a *linear constraint,* an occur-check-like condition. Section 4 discusses an algorithm for solving the unification problem over $AGnH$ with a linear constraint. The algorithm uses a strong Gröbner basis algorithm to solve linear equations over a monoid semiring $\mathsf{Z}<X_1, \ldots, X_n>$. To ensure that the solutions to the unification problem satisfy the occur-check like condition, unusual term orderings capturing this constraint are defined. The Gröbner basis algorithm is discussed in Section 5. It is shown that the algorithm computes a Gröbner basis for a right ideal using arbitrary right admissible term orderings including those which do

not have the boundedness property. This algorithm is new since earlier algorithms worked only for admissible term orderings satisfying the boundedness property.

2 Undecidability of Unification over $\mathcal{E}'_{3,5}$

Recall again that $\mathcal{E}'_{3,5}$ consists of axioms

$$\begin{aligned}
x \cdot (y \cdot z) &= (x \cdot y) \cdot z && \text{(A)}\\
x \cdot y &= y \cdot x && \text{(C)}\\
x \cdot 1 &= x && \text{(U)}\\
x \cdot x^{-1} &= 1 && \text{(Inv)}\\
x^1 &= x && \text{(Exp1)}\\
1^x &= 1 && \text{(Exp2)}\\
(x \cdot y)^z &= (x^z) \cdot (y^z) && \text{(Exp3)}\\
(x^y)^z &= x^{y \circ z} && \text{(Exp5)}\\
x \circ (y \circ z) &= (x \circ y) \circ z && \text{(A')}\\
x \circ 1 &= x && \text{(U')}
\end{aligned}$$

This theory has the following AC-convergent system $\mathcal{R}'_{3,5}$ (the precedence used is $\hat{} \succ {}^{-1} \succ \cdot \succ \circ$):

$$\begin{aligned}
x \cdot 1 &\to x\\
x \cdot x^{-1} &\to 1\\
x \cdot (x^{-1} \cdot y) &\to y\\
(x^{-1})^{-1} &\to x\\
1^{-1} &\to 1\\
(x \cdot y)^{-1} &\to x^{-1} \cdot y^{-1}\\
x^1 &\to x\\
1^x &\to 1\\
(x \cdot y)^z &\to (x^z) \cdot (y^z)\\
(x^y)^z &\to x^{y \circ z}\\
(x^{-1})^y &\to (x^y)^{-1}\\
(x \circ y) \circ z &\to x \circ (y \circ z)\\
x \circ 1 &\to x\\
1 \circ x &\to x
\end{aligned}$$

Through the rest of this section we use the symbol $\approx$ to abbreviate $=_{\mathcal{E}'_{3,5}}$.

To prove the undecidability of the unification problem over this theory, we need to introduce the following claims and lemmas. In the following, we use $\bigcirc_k(b)$ to denote $\underbrace{b \circ b \ldots \circ b}_{k}$ and a^i to denote $\underbrace{a \cdot a \ldots \cdot a}_{i}$ if i is positive, and $\underbrace{a^{-1} \cdot a^{-1} \ldots \cdot a^{-1}}_{i}$ if i is negative. (Note: k cannot be a negative number.) We also use $+$ and the integers

merely for abbreviation; e.g., a^{x+y} is the abbreviation of $a^x \cdot a^y$, a^{2x} is the abbreviation of $a^x \cdot a^x$, and a^{-x} stands for $(a^{-1})^x \approx (a^x)^{-1}$. It is not hard to see that the equation $x \circ b \approx b \circ x$ forces $x \approx \bigcirc_k(b)$ for some $k \geq 0$.

Claim 2.1. Equation $y^b \approx y$ has only one solution $y \approx 1$.

Claim 2.2. Equation $y^b \cdot a^i \approx y$ forces $y \approx 1$ and $i = 0$.

Claim 2.3. Equation $x^b \approx y^c$ forces $x \approx y \approx 1$.

Claim 2.4. Equations $x^b \cdot w \approx x$ and $y^c \cdot w \approx y$ force $x \approx y \approx w \approx 1$.

Proof. If $x^b \cdot w \approx x$ and $y^c \cdot w \approx y$, then $x^b \cdot (y^{-1})^c \approx x \cdot y^{-1}$. This is clearly impossible, since no cancellation can occur between x^b and $(y^{-1})^c \approx (y^c)^{-1}$.

Claim 2.5. Let i be a nonnegative integer. The equations $x^b \cdot w \cdot a^i \approx x$ and $y^c \cdot w \approx y$ force $x \approx y \approx w \approx 1$ and $i = 0$.

The proof is roughly the same as that of Claim 2.4.

Lemma 2.1. $x^b \cdot a \approx x \cdot a^{\bigcirc_k(b)}$ *iff* $x \approx a^{\bigcirc_{k-1}(b)+\bigcirc_{k-2}(b)+\ldots+1}$.

Proof. The "if" part is simple, since

$$\begin{aligned}\left(a^{\bigcirc_{k-1}(b)+\bigcirc_{k-2}(b)+\ldots+1}\right)^b \cdot a &\approx a^{\bigcirc_{k-1}(b)\bigcirc b+\bigcirc_{k-2}(b)\bigcirc b+\ldots+b} \cdot a \\ &\approx a^{\bigcirc_k(b)+\bigcirc_{k-1}(b)+\ldots b+1} \\ &\approx x \cdot a^{\bigcirc_k(b)}.\end{aligned}$$

To prove the "only if" part, assume equation $x^b \cdot a \approx x \cdot a^{\bigcirc_{k_1}(b)}$ has more than one solution. Let Z_1, Z_2 be two different solutions. We then have:

$$\begin{aligned}{Z_1}^b \cdot a &\approx Z_1 \cdot a^{\bigcirc_{k_1}(b)} \quad \ldots\ldots(1) \\ {Z_2}^b \cdot a &\approx Z_2 \cdot a^{\bigcirc_{k_1}(b)} \quad \ldots\ldots(2)\end{aligned}$$

From (2) we should have:

$$\left({Z_2}^{-1}\right)^b \cdot a^{-1} \approx {Z_2}^{-1} \cdot (a^{-1})^{\bigcirc_{k_1}(b)} \quad \ldots\ldots(3)$$

From (1) and (3), we get

$$\left(Z_1 \cdot {Z_2}^{-1}\right)^b \approx Z_1 \cdot {Z_2}^{-1}$$

According to **Claim 2.1**, $Z_1 \cdot {Z_2}^{-1} \approx 1$. Therefore $Z_1 \approx Z_2$, contradicting the assumption that Z_1, Z_2 are two different solutions.

Lemma 2.2. *Let $v \approx a^i$ for some i. Then the equation $y^b \cdot v \approx a^{\bigcirc_k(b)+\ldots+1} \cdot y$ has only one solution, namely*

$$\{v \approx a^{k+1},\ y \approx a^{(\bigcirc_{k-1}(b)+\ldots+i\bigcirc_{k-i}(b)+\ldots+k)}\}.$$

Proof. The above is clearly a solution. Assume there are two different solutions $\{y \approx Y_1, v \approx a^i\}$ and $\{y \approx Y_2, v \approx a^j\}$ for this equation. That is,

$$\begin{aligned} {Y_1}^b \cdot a^i &\approx a^{\bigcirc_k(b)+\ldots+1} \cdot Y_1 \\ {Y_2}^b \cdot a^j &\approx a^{\bigcirc_k(b)+\ldots+1} \cdot Y_2 \end{aligned}$$

So $(Y_1 \cdot {Y_2}^{-1})^b \cdot a^{(i-j)} \approx Y_1 \cdot {Y_2}^{-1}$. Based on **Claim 2.2**, we get $Y_1 \approx Y_2$ and $i - j = 0$, causing a contradiction.

Lemma 2.3. *The following two equations have a solution iff $k_1 = k_2$ and $v_1 \approx a^{k_1+1}$:*

$$\begin{aligned} {y_1}^b \cdot v_1 &\approx a^{\bigcirc_{k_1}(b)+\ldots+1} \cdot y_1 \\ {y_2}^c \cdot v_1 &\approx a^{\bigcirc_{k_2}(c)+\ldots+1} \cdot y_2 \end{aligned}$$

Proof. The proof is by contradiction. Suppose the claim is not true. Now

$$\{v_1 \approx a^{k_1+1},\ y_1 \approx a^{(\bigcirc_{k_1-1}(b)+\ldots+i\bigcirc_{k_1-i}(b)+\ldots+k_1)}\}$$

is clearly a solution for the first equation. Similarly,

$$\{v_1 \approx a^{k_2+1},\ y_2 \approx a^{(\bigcirc_{k_2-1}(b)+\ldots+i\bigcirc_{k_2-i}(b)+\ldots+k_2)}\}$$

is a solution for the second equation. Without loss of generality assume $k_1 > k_2$. Let $i = k_1 - k_2$. Let

$$v_1 \approx \hat{v_1},\ y_1 \approx \alpha_1,\ y_2 \approx \alpha_2$$

be a solution for the equations where $\hat{v_1}$ is not of the required form. Let

$$\beta_1 = a^{(\bigcirc_{k_1-1}(b)+\ldots+i\bigcirc_{k_1-i}(b)+\ldots+k_1)}$$

and $\beta_2 = a^{(\bigcirc_{k_2-1}(b)+\ldots+i\bigcirc_{k_2-i}(b)+\ldots+k_2)}$. Let $v_1' = \hat{v_1} \cdot (a^{k_1+1})^{-1}$. But then the equations

$$\begin{aligned} {w_1}^b \cdot v_1' &\approx w_1 \\ {w_2}^c \cdot v_1' \cdot a^i &\approx w_2 \end{aligned}$$

are solvable too, with

$$\{w_1 \approx \alpha_1 \cdot \beta_1^{-1},\ w_2 \approx \alpha_2 \cdot \beta_2^{-1}\},$$

but this is impossible by Claim 2.5. □

Lemma 2.4. *The following two equations force $v_2 \approx a^{k(k-1)/2}$.*

$$z_1{}^b \cdot v_2 \approx a^{(\bigcirc_{k-1}(b)+\ldots+i\bigcirc_{k-i}(b)+\ldots+k)} \cdot z_1$$
$$z_2{}^c \cdot v_2 \approx a^{(\bigcirc_{k-1}(c)+\ldots+i\bigcirc_{k-i}(c)+\ldots+k)} \cdot z_2$$

The proof of this lemma is very similar to that of **Lemma 2.3**.

Lemma 2.5. *The following two equations force* $w_2 \approx a^{\bigcirc_k(b)}$.

$$x_3{}^b \cdot a \approx x_3 \cdot a^{w_2}$$
$$y_3{}^b \cdot a^k \approx x_3 \cdot y_3$$

Proof. It is obvious that w_2 should be of the form $a^{\bigcirc_i(b)}$. By **Lemma 2.1**, the first equation forces x_3 to be $a^{(\bigcirc_{i-1}(b)+\ldots+1)}$. According to **Lemma 2.2**, k should be equal to i, thus forcing w_2 to be $a^{\bigcirc_k(b)}$. □

Theorem 2.1. *Unification over* $\mathcal{E}'_{3,5}$ *is undecidable.*

Proof. The reduction is from Hilbert's tenth problem. Thus, it is necessary to simulate numbers as well as multiplication over numbers. Using the algebraic relation $i * j = ((i+j)^2 - (i-j)^2)/4$, if we can simulate i^2, multiplication is obtained. Given $u \approx \bigcirc_k(b)$, the following set of equations can be used to force w to be $\bigcirc_{k^2}(b)$, thus simulating the squaring of numbers.

1. $x_1{}^b \cdot a \approx x_1 \cdot a^u$	6. $z_2{}^c \cdot v_2 \approx y_2 \cdot z_2$
2. $x_2{}^c \cdot a \approx x_2 \cdot a^{u'}$	7. $s_1 \approx v_2 \cdot v_2$
3. $y_1{}^b \cdot v_1 \approx x_1 \cdot y_1$	8. $s_2 \approx s_1 \cdot v_1$
4. $y_2{}^c \cdot v_1 \approx x_2 \cdot y_2$	9. $x_3{}^b \cdot a \approx x_3 \cdot a^w$
5. $z_1{}^b \cdot v_2 \approx y_1 \cdot z_1$	10. $y_3{}^b \cdot s_2 \approx x_3 \cdot y_3$

According to **Lemma 2.1**, the first and second equations force $x_1 \approx a^{\bigcirc_{k-1}(b)} \cdot a^{\bigcirc_{k-2}(b)} \cdot \ldots \cdot a$ and $x_2 \approx a^{\bigcirc_{k_2-1}(c)} \cdot a^{\bigcirc_{k_2-2}(c)} \cdot \ldots \cdot a$ for some k_2. According to **Lemma 2.3** the third and fourth equations together force v_1 to be a^k as well as $k_2 = k$. In a similar fashion, the fifth and sixth equations force v_2 to be $a^{k\cdot(k-1)/2}$. It is easy to see that the seventh and eighth equations make s_1 and s_2 to be $a^{k\cdot(k-1)}$ and a^{k^2}, respectively. Finally, by **Lemma 2.5**, the ninth and tenth equations force w to be $\bigcirc_{k^2}(b)$.

3 Relating Unification over $\mathcal{E}_3$ to Unification over *AGnH*

We consider the theory $\mathcal{E}_3$ consisting of AG and axioms $Exp1$, $Exp2$ and $Exp3$. We show that the equational unification problem for $\mathcal{E}_3$ is equivalent to the unification problem wrt the theory of Abelian groups with *noncommuting* homomorphisms, denoted by $AGnH$, but with an additional constraint. (The n stands for the number of homomorphisms.)

It is shown in [2] that the theory $AGnH$ is *unitary* with respect to unification without constants and it is also unitary with respect to unification with constants[5]. In Section 5 of [1], Baader showed that the unification problem of $AGnH$ reduces to solving linear equations over the polynomial semiring $\mathsf{Z}<h_1, \ldots, h_n>$ where $h_1, \ldots, h_n$ are the noncommuting homomorphisms of $AGnH$, treated as indeterminates in the polynomial semiring. He gave a unification algorithm for $AGnH$, which uses a weak Gröbner basis of a right ideal over $\mathsf{Z}<h_1, \ldots, h_n>$ for solving linear equations; as a result, Baader also proposed an algorithm for computing a weak Gröbner basis. We generalize Baader's algorithm by adding an additional key step to ensure that a given *linear constraint* (capturing an occur-check like condition) is satisfied by the unifier, so as to apply it to the equational unification problem for $\mathcal{E}_3$. This generalization needs a strong Gröbner basis of a right ideal over $\mathsf{Z}<h_1, \ldots, h_n>$. In a later section, we develop a new algorithm for computing a strong Gröbner basis of a right ideal over $\mathsf{Z}<h_1, \ldots, h_n>$.

3.1 Unification over $\mathcal{E}_3$ as a Combination of Theories

Let $\Sigma = \{\cdot, ^{-1}, 1, \hat{}\}$, the signature of $\mathcal{E}_3$.

Definition 3.1. *An $\mathcal{E}_3$-unification problem S over Σ is called an* AG*-unification problem if each equation in S is of the form $x =^? t$, where x is a variable and t is a term over the signature of AG.*

Definition 3.2. *An $\mathcal{E}_3$-unification problem S on Σ is called an* exponent *$\mathcal{E}_3$-unification problem, abbreviated as EXP, if every equation in S is of the form $x =^? y^z$, where x and y are variables and z is a variable or a free constant. Also if z is a variable, z is called an* exponent variable*; otherwise, it is called an* exponent constant.

Definition 3.3. *An $\mathcal{E}_3$-unification problem S on Σ is called a* simple *$\mathcal{E}_3$-unification problem if $S = S_1 \bigcup S_2$, where S_1 is an AG-unification problem and S_2 is an exponent $\mathcal{E}_3$-unification problem.*

It is easy to see that using abstraction, any $\mathcal{E}_3$-unification problem can be transformed into a simple $\mathcal{E}_3$-unification problem. For example, consider $S = \{w =^? (x^{(y^{u\cdot v})^{-1}\cdot z^{u'\cdot v'}})^{-1}\}$. Using abstractions, S is transformed to S':

$$\{ \quad 1.\ w =^? z_1^{-1}, \quad 2.\ z_1 =^? x^{z_2}, \quad 3.\ z_2 =^? z_3^{-1}\cdot z_4, \quad 4.\ z_3 =^? y^{z_5},$$
$$5.\ z_5 =^? u\cdot v,\ 6.\ z_4 =^? z^{z_6}, \quad 7.\ z_6 =^? u'\cdot v'\ \},$$

where $z_1, z_2, z_3, z_4, z_5, z_6$ are new variables introduced for alien subterms in S.

[5] A theory is unitary if a minimal complete set of unifiers always exists and its cardinality is at most one.

3.2 Relating $AG + EXP$ to $AGnH$

Given a simple $\mathcal{E}_3$-unification problem S, for each equation of the form $x =^? y^w$ in S, we transform it into $x =^? h_w(y)$ where h_w is a homomorphism corresponding to the symbol w. Let $\mathcal{H}(S)$ denote the set of all homomorphisms introduced in this way, and, let $\Sigma_h = \{\cdot, ^{-1}, 1\} \cup \mathcal{H}(S)$. We call the transformed $AGnH$ problem a *h-image* of S.

For the above example, its *h-image* T is:

$$\{ \quad 1.\ w =^? z_1^{-1}, \qquad 2.\ z_1 =^? h_{z_2}(x), \quad 3.\ z_2 =^? z_3^{-1} \cdot z_4, \quad 4.\ z_3 =^? h_{z_5}(y),$$
$$5.\ z_5 =^? u \cdot v, \quad 6.\ z_4 =^? h_{z_6}(z), \quad 7.\ z_6 =^? u' \cdot v' \ \},$$

It is not hard to see that the unifiability of T wrt $AGnH$ does not necessarily imply that the original problem S has a unifier wrt $\mathcal{E}_3$. For instance, the problem $\{x =^? a^x\}$ has no unifier wrt $\mathcal{E}_3$, whereas its h-image $\{x =^? h_x(a)\}$, when simply taken as a unification problem wrt $AGnH$ (with $n = 1$) is solvable. Thus something like an extended occur check must be enforced; this is done by solving the unifiability problem of T wrt $AGnH$ subject to *linear* constraints (including) $x \succ h_x$ for every homomorphism $h_x \in \mathcal{H}(S)$, i.e., a unifier θ of T should satisfy the condition that for every $x \in Var(T)$, $\theta(x)$ *does not contain* any occurrence of h_x, where $Var(T)$ denotes the set of all variables in T.

Definition 3.4. *Given a simple $\mathcal{E}_3$-unification problem S and its h-image T wrt $AGnH$, a* linear constraint *C is a total ordering $\succ_C$ over $Var(T) \cup \mathcal{H}(S)$ such that $x \succ_C h_x$ for all exponent variables x in S.*

Definition 3.5. *A substitution β whose domain is $Var(T)$, is said to satisfy a linear constraint C if and only if the following holds: for every $x \in Var(T)$, $\beta(x)$* does not contain *any of the function symbols below x in C. In other words, if $x \succ_C h_y$, then $\beta(x)$ does not contain any occurrence of h_y.*

Definition 3.6. *A unifier θ for a $\mathcal{E}_3$-unification problem S is said to be a* discriminating *unifier if and only if the following hold for all variables in $Var(S)$:*

1. *$\theta(u) \neq_{\mathcal{E}_3} 1$ for all u.*
2. *$\theta(v) =_{\mathcal{E}_3} \theta(w)$ iff $v = w$.*

Theorems 2 and 3 below relate the unification problem S wrt $\mathcal{E}_3$ to its h-image T wrt $AGnH$.

Theorem 3.1. *Given a simple $\mathcal{E}_3$-unification problem S and its h-image T wrt $AGnH$, if S has a discriminating unifier, then T is unifiable. Furthermore, there is a linear constraint C that the unifier of T satisfies.*

Proof. Let θ be such a unifier of a simple $\mathcal{E}_3$-unification problem S. Consider all the exponent equations in S :

$$\begin{aligned}\{\ x_{u_1} &=^? {x_{v_1}}^{x_{w_1}}\\ &\vdots \quad\ \vdots\\ x_{u_i} &=^? {x_{v_i}}^{x_{w_i}}\\ &\vdots \quad\ \vdots\\ x_{u_k} &=^? {x_{v_k}}^{x_{w_k}}\}\end{aligned}$$

The unifier θ includes $\{x_{u_i} \leftarrow t_{u_i}, x_{v_i} \leftarrow t_{v_i}, x_{w_i} \leftarrow t_{w_i} (1 \leq i \leq k)\}$. Thus, $t_{u_i} =_{\mathcal{E}_3} {t_{v_i}}^{t_{w_i}} (1 \leq i \leq k)$.

For each t_{w_i}, we introduce a homomorphism $h_{t_{w_i}}$. Let $\mathcal{H}''(S)$ denote the set of homomorphisms introduced for all t_{w_i}. Let $\Sigma'' = \{\cdot, ^{-1}\} \bigcup \mathcal{H}''(S)$. Let $\mathcal{E}_3' = AG \bigcup \{h_{t_{w_i}}(u_1 \cdot u_2) = h_{t_{w_i}}(u_1) \cdot h_{t_{w_i}}(u_2)\}$ for all i $(1 \leq i \leq k)$. We also define a recursive function ***rep*** as follows:

$\boldsymbol{rep}(a) = a$, where a is a constant in Σ.

$\boldsymbol{rep}(A \cdot B) = \boldsymbol{rep}(A) \cdot \boldsymbol{rep}(B)$, where A, B are terms on Σ.

$\boldsymbol{rep}(A^x) = h_x(\boldsymbol{rep}(A))$, where x, A are terms on Σ.

It is easy to see that the function ***rep*** removes all occurrences of the exponent operator from terms over Σ. Since θ is a discriminating unifier for S, for each $t_{w_i}, t_{w_j} (i \neq j)$, we have $t_{w_i} \neq_{\mathcal{E}_3} t_{w_j}$. Also it is easy to show that $s =_{\mathcal{E}_3} t$ if and only if $\boldsymbol{rep}(s) =_{\mathcal{E}_3'} \boldsymbol{rep}(t)$ for all terms s, t over Σ. Therefore, for $1 \leq i \leq k$,

$$\boldsymbol{rep}(t_{u_i}) =_{\mathcal{E}_3'} \boldsymbol{rep}({t_{v_i}}^{t_{w_i}}) =_{\mathcal{E}_3'} h_{t_{w_i}}(\boldsymbol{rep}(t_{v_i})) \text{ ——(1).}$$

Now for each $h_{t_{w_i}} (1 \leq i \leq k)$, we introduce the homomorphism $h_{x_{w_i}}$ which is the same homomorphism we introduced for x_{w_i} in the *h-image* T of S. We also define function ***rep'*** as:

$\boldsymbol{rep'}(a) = a$, where a is a constant on Σ'.

$\boldsymbol{rep'}(A \cdot B) = \boldsymbol{rep'}(A) \cdot \boldsymbol{rep'}(B)$, where A, B are terms on Σ'.

$\boldsymbol{rep'}(h_{t_{w_i}}(A)) = h_{x_{w_i}}(\boldsymbol{rep'}(A))$, where A is a term on Σ.

Obviously ***rep'*** maps all $h_{t_{w_i}} (1 \leq i \leq k)$ to the corresponding $h_{x_{w_i}}$. So from (1) and the definition of $\mathcal{E}'$,

$$\boldsymbol{rep'}(\boldsymbol{rep}(t_{u_i})) =_{\mathcal{E}'} h_{x_{w_i}}(\boldsymbol{rep'}(\boldsymbol{rep}(t_{v_i}))).$$

That means T is solvable and a unifier β for T can be constructed as follows: $\beta(x) \leftarrow \boldsymbol{rep'}(\boldsymbol{rep}(\theta(x)))$ for every $x \in Var(S)$.

A linear constraint that β satisfies is constructed by comparing $\theta(x_i)$ for variables in $Var(T)$ using a term ordering that is total on ground terms. The reader should note that this requirement is one of the key differences between the proposed algorithm and Baader's algorithm for $AGnH$. □

Theorem 3.2. *Given a simple $\mathcal{E}_3$-unification problem S and its h-image T wrt $AGnH$, if T has a solution which satisfies a linear constraint, then S is solvable.*

Proof. Consider all the exponent equations $\{x_{u_1} =^? {x_{v_1}}^{x_{w_1}}, \ldots, x_{u_k} =^? {x_{v_k}}^{x_{w_k}}\}$ in S; similarly, also consider their corresponding equations

$$\{x_{u_1} =^? h_{x_{w_1}}(x_{v_1}), \ldots, x_{u_k} =^? h_{x_{w_k}}(x_{v_k})\}$$

in the h-image T for S.

Let β be a ground unifier of T that satisfies a linear constraint C. From C, we can get a subconstraint C' on variables in $Var(T)$. Let $Var(T) = \{x_1, \ldots, x_n\}$. Assume without loss of generality that $C' = x_n \succ_C \cdots \succ_C x_i \succ_C \cdots \succ_C x_1$. Now we will use induction on C' to form θ. As the basis step, consider the first variable x_n in C'. Since x_n is the first variable, $\beta(x_n)$ should not contain any item below x_n in C, implying that $\beta(x_n)$ is composed of constants. Define $\theta(x_n) = \beta(x_n)$.

Assume that $\theta(x_{j'})$ $(j \leq j' \leq n)$ is already defined. For variable x_{j-1}, the following cases arise:

1. $\beta(x_{j-1})$ is composed of constants. In this case, define $\theta(x_{j-1}) = \beta(x_{j-1})$.
2. $\beta(x_{j-1})$ is composed of constants and some $h_{x_{w_i}}$ $(1 \leq i \leq n)$ where each $h_{x_{w_i}} \succ_C x_{j-1}$. Since $x_{w_i} \succ_C h_{x_{w_i}}$, $x_{w_i} \succ_C h_{x_{w_i}} \succ_C x_{j-1}$. By the induction hypothesis, all these $\theta(x_{w_i})$ are already defined. Therefore, we can define $\theta(x_{j-1}) = repp\,(\beta(x_{j-1}))$ where the function $repp$ is defined as:

   $repp(a) = a$, where a is a constant on Σ_h.

   $repp(A \cdot B) = repp(A) \cdot repp(B)$, where A, B are terms on Σ_h.

   $repp(h_{x_{w_i}}(A)) = repp(A)^{\theta(x_{w_i})}$, where A is a term on Σ_h.

It can be shown that $s =_{AGnH} t$ iff $repp(s) =_{\mathcal{E}_3} repp(t)$. Therefore, $repp(\beta(x_{u_i})) =_{\mathcal{E}_3} repp(h_{x_{w_i}}(\beta(x_{v_i})))$ for all $1 \leq i \leq n$. So $\theta(x_{u_i}) =_{\mathcal{E}_3} (\theta(x_{v_i}))^{\theta(x_{w_i})}$ for all $1 \leq i \leq n$. Thus θ is a solution for S. □

In the next section, we generalize Baader's algorithm for unifiability check for $AGnH$ to work with a linear constraint.

4 Unification over *AGnH* with a Linear Constraint

The unifiers of a unification problem wrt $AGnH$, where $h_1, \ldots, h_n$ are the noncommuting homomorphisms, correspond to the solutions of (nonhomogeneous) linear equations over the polynomial semiring $\mathsf{Z}<h_1, \ldots, h_n>$ with $h_1, \ldots, h_n$ as indeterminates in the polynomial semiring. The set of solutions of a homogeneous equation $p_1U_1 + \ldots + p_kU_k = 0$, where $p_1, \ldots, p_k \in \mathsf{Z}<h_1, \ldots, h_n>$ and $U_1, \ldots, U_k$ are variables, is a finitely generated right $\mathsf{Z}<h_1, \ldots, h_n>$–semimodule. The nonhomogeneous equation $p_1U_1 + \ldots + p_kU_k = p_0$ has a solution iff p_0 is a member of the right ideal generated by $p_1, \ldots, p_k$. And, the membership problem for finitely generated right ideals is decidable.

Let NHE be a set of linear equations $\{p_{11}U_1 + \cdots + p_{1k}U_k = p_1, \; \vdots \; \vdots \; p_{m1}U_1 + \cdots + p_{mk}U_k = p_m\}$ where $p_{11}, \cdots, p_{1k}, \cdots, p_{m1}, \cdots, p_{mk}, p_1, \cdots, p_m$ are in $\mathsf{Z}<h_1, \ldots, h_n>$.

Baader [1] gave an algorithm for finding a solution for such nonhomogeneous linear equations. In his algorithm, he first computes a *weak Gröbner basis* G for a polynomial right ideal, then computes a right syzygy basis for homogeneous linear equations, and finally, a particular solution for the nonhomogeneous equations by using G. The algorithm is nontrivial; we will not discuss the details here but suggest the reader to refer to [1] for details. Let $\pi = (q_1, \ldots, q_k)$, where $\pi(U_i) = q_i$, be a particular solution for the above set NHE of nonhomogeneous equations obtained, for instance, using Baader's algorithm.

Let SB denote a right syzygy basis $\{(q_{11}, \cdots, q_{1k}), \ldots, (q_{w1}, \cdots, q_{wk})\}$ for the set HE of the homogeneous equations

$$\begin{array}{c} \{p_{11}U_1 + \cdots + p_{1k}U_k = 0, \\ \vdots \qquad\qquad \vdots \\ p_{m1}U_1 + \cdots + p_{mk}U_k = 0\}. \end{array}$$

Proposition 4.1. *Let π be a particular solution for NHE. $\pi' = (q'_1, \ldots, q'_k)$ is equivalent to π with respect to SB and hence is also a particular solution for NHE iff there exist multipliers $b_1, \cdots, b_w$ such that $q_i - q'_i = q_{1i}b_1 + \cdots + q_{wi}b_w$ for each $1 \leq i \leq k$.*

4.1 Solutions Satisfying a Linear Constraint

We discuss how a solution satisfying a linear constraint C, if any, is searched for among all solutions of NHE.

Let $z_1, \ldots, z_k$ be new indeterminates. Consider the following set MSB of polynomials in $\mathsf{Z}<h_1, \ldots, h_n, z_1, \ldots, z_k>$ constructed from SB above:

$$\{z_1\, q_{11} + \ldots + z_k\, q_{1k},\ \ldots\ ,\ z_1\, q_{w1} + \ldots + z_k\, q_{wk}\}.$$

Note that we only have to consider terms of the form $z_j\, \omega$, where $\omega \in (\mathcal{H}(S))^*$, where $\mathcal{H}(S) = \{h_1, \cdots, h_n\}$. Below, we define a right compatible ordering $\succ_t$ on such linear terms to capture linear constraints C in the following sense: if in a term $z_j\, \omega_j$, $h_i \succ_C U_j$ for each h_i appearing in ω_j, whereas in another term $z_{j'}\, \omega_{j'}$ we have some $h_{i'}$ in $\omega_{j'}$ such that $U_{j'} \succ_C h_{i'}$ then $z_j\, \omega_j \prec_t z_{j'}\, \omega_{j'}$. In other words, a term that "violates" a linear constraint will be above a term that does not.

The ordering $\succ_t$ is used to construct a strong Gröbner basis $GMSB$ for the set MSB of polynomials. The polynomial $\pi_p = z_1 q_1 + \ldots z_k q_k$ corresponding to the particular solution π above is then normalized using the Gröbner basis $GMSB$. Since the equivalence relation induced by MSB preserves solutions of NHE, the canonical (normal) form of π_p wrt $GMSB$ also corresponds to a particular solution, say π'. If this particular solution π' satisfies C (i.e., all terms are *good* in the sense defined in the next subsection), then we get a unifier for the unification problem wrt $AGnH$ satisfying C. If π' does not satisfy C, then the unification problem wrt $AGnH$ does not have a solution satisfying C, since a polynomial corresponding to a solution satisfying C must be smaller than the canonical form of π_p wrt $\succ_t$.

In the next subsection, we introduce how to define $\succ_t$ given a linear constraint C.

4.2 Term Orderings Capturing Linear Constraints

A linear constraint C on an extended alphabet

$$\Delta = \{Y_0, \ldots, Y_l, a_1, \ldots, a_l\},$$

where $\{U_1, \ldots, U_k\} \subseteq \{Y_0, \ldots, Y_l\}$ and $\{h_1, \ldots, h_n\} \subseteq \{a_1, \ldots, a_l\}$, is written as:

$$Y_l \succ_C a_l \succ_C \ldots \succ_C a_2 \succ_C Y_1 \succ_C a_1 \succ_C Y_0.$$

In the above, upper case symbols are used for variables, and lower case symbols are used for constants. Extra symbols are introduced to ensure that between every two variables, there is a constant in the ordering. For ease of exposition, we denote the ordering this induces on $\{U_1, \ldots, U_k\}$ as

$$U_{i_k} \succ \ldots \succ U_{i_1}$$

where $\{U_1, \ldots, U_k\} = \{U_{i_1}, \ldots, U_{i_k}\}$.

Consider any two terms s, t in $\mathsf{Z}<h_1, \ldots, h_n, z_1, \ldots, z_k>$ which are linear in $\{z_1, \ldots, z_k\}$. Define $s \succ_t t$ iff $nf(s\#) >' nf(t\#)$, where nf stands for the normal form wrt the reduction rules defined below to capture the linear constraint C. After defining nf and $>'$, we show that $\succ_t$ is right admissible.

The term $nf(s\#)$ is over the extended alphabet

$$\Delta_1 = \{a_1, \ldots, a_l, v_1, \ldots, v_l, a'_1, \ldots, a'_l, z_1, \ldots, z_k, z'_1, \ldots, z'_k\},$$

where the a_i's in Δ_1 are copies of the corresponding a_i's in Δ; further, corresponding to every U_i in Δ, we have z_i, z'_i in Δ_1.

Below, *legal* term, *good* term, and *bad* term are defined on $\{a_1, \ldots, a_l, z_1, \ldots, z_k\}$ based on whether the term satisfies the linear constraint C.

Definition 4.1. *A term $s = z_i\ s_a$, where $s_a \in \{a_1, \cdots, a_l\}^*$ is called a* legal *term (only such terms appear in the polynomials in the basis MSB and in the computation of a Gröbner basis from MSB).*
A legal term $s = z_i\ s_a$ is called a good *term if for each a_j appearing in s_a, $a_j \succ_C U_i$ in C, i.e., s_a satisfies the linear constraint C with respect to U_i.*
A legal term $s = z_i\ s_a$ is called a bad *term if there exists a_j appearing in s_a such that it is not the case that $a_j \succ_C U_i$ in C, i.e., s_a does not satisfy the linear constraint C with respect to U_i. (A legal term that is not good, is bad.)*

To capture the restrictions imposed by the linear constraint C on terms, we define the reduction rules on legal terms as:

$$\begin{array}{ll}
z_i\ a_j \to a'_i\ z_j & \text{if } a_j \succ_C U_i. \\
z_i\ a_j \to v_j\ a'_j\ z_i & \text{if } U_i \succ_C a_j. \\
z_i\ \# \to z'_i. & \\
a'_i\ z'_j \to z'_j\ a'_i. & \\
a'_i\ v_j \to v_j\ a'_i. & \\
v_i\ v_j \to v_j\ v_i & \text{if } U_i \succ_C U_j.
\end{array}$$

The normal form of a legal term $z_i\ s_a$ wrt the above rules is either

(i) $z_i'\ s_{a'}$, where $s_{a'}$ is obtained by systematically replacing every a_j in s_a by a'_j. or
(ii) $s_v\ z_i'\ s_{a'}$, where $s_{a'}$ is obtained from s_a as in (i) and s_v is a string of v_j's.

To compare $nf(s\#)$ with $nf(t\#)$ using the above reduction rules, we define the following ordering $>'$ on symbols in Δ_1:

$$v_1 >' \ldots >' v_l >' z_{i_k}' >' \ldots >' z_{i_1}' >' a'_1 >' \ldots >' a'_l$$

(The ordering on the other symbols in Δ_1 does not matter since they do not appear in the normal forms.) This ordering is *extended* to terms over Δ_1 using pure lexicographic comparison.

Definition 4.2. *Given two terms $s = x_1 \ldots x_m$, $t = y_1 \ldots y_r$ on Δ_1 and the alphabet ordering on Δ_1 as defined above, we define $s >' t$ iff t is a prefix of s or there exists a* k *such that $x_k > y_k$ and $x_{k'} = y_{k'}$ for all $1 \leq k' < k$. Given two legal terms s, t on Δ_1, $s \succ_t t$ iff $nf(s\#) >' nf(t\#)$.*

By the above reduction rules, the normal form of a bad term $\succ_t$ the normal form of a good term because only the normal form of a bad term has some v_j's which are $>'$ all a_i''s and z_i''s. The ordering clearly does not have the boundedness property.

We now show that the ordering $\succ_t$ on legal terms in $\mathsf{Z}<h_1, \cdots, h_n, z_1, \ldots, z_k>$ is right admissible.

For any $s \neq 1$, it is easy to see that $s \succ_t 1$. The following lemma ensures that if $s \succ_t t$, then for any u, $s\,u \succ_t t\,u$, by proving that $nf(s\#) >' nf(t\#)$ iff $nf(su\#) >' nf(tu\#)$.

Lemma 4.1. *Let Δ_1, $>'$, and nf be as defined above. Then $nf(s\#) >' nf(t\#)$ iff $nf(su\#) >' nf(tu\#)$ for all legal terms s, t, su and tu.*

Proof. (i) If $nf(s\#) >' nf(t\#)$ then $nf(su\,\#) >' nf(tu\,\#)$: To prove this, it is enough to prove that for any symbol a_p in Σ_1, $nf(sa_p\,\#) >' nf(ta_p\,\#)$ if $nf(s\#) >' nf(t\#)$. Assume $s = z_i w_s$, $t = z_j w_t$ where $w_s, w_t \in \{a_1, \cdots, a_l\}^*$. We have the following cases:

1. $U_i \succeq_C U_j$. So $z_i \geq' z_j$. After right multiplying s and t by a_p, we have the following sub-cases:
   (a) $a_p \succ_C U_i \succ_C U_j$. Right multiplying s and t by a_p will contribute a_p' to both $nf(sa_p\,\#)$ and $nf(ta_p\,\#)$. Therefore $nf(sa_p\,\#) >' nf(ta_p\,\#)$.
   (b) $U_i \succ_C U_j \succ_C a_p$. Right multiplying s and t by a_p will contribute $v_p a_p'$ to both $nf(sa_p\,\#)$ and $nf(ta_p\,\#)$. Therefore $nf(sa_p\,\#) >' nf(ta_p\,\#)$.
   (c) $U_i \succ_C a_p \succ_C U_j$. Right multiplying s and t by a_p will contribute $v_p a_p'$ to $nf(sa_p\,\#)$ and a_p' to $nf(ta_p\,\#)$. Therefore $nf(sa_p\,\#) >' nf(ta_p\,\#)$.

2. $U_i \prec_C U_j$. So $z_i <' z_j$. Since $nf(s\#) >' nf(t\#)$, there must exist some v_q in $nf(s\#)$ such that $U_i \succ_C a_q$ and the power of v_q in $nf(s\#)$ is larger than the power of v_q in $nf(t\#)$. We have the following sub-cases:

   (a) $a_p \succ_C U_j$. Right multiplying s and t by a_p will contribute a_p' to both $nf(sa_p\,\#)$ and $nf(ta_p\,\#)$. Therefore $nf(sa_p\,\#) >' nf(ta_p\,\#)$.
   (b) $a_p \prec_C U_i$. Right multiplying s and t by a_p will contribute $v_p a_p'$ to both $nf(sa_p\,\#)$ and $nf(ta_p\,\#)$. Therefore $nf(sa_p\,\#) >' nf(ta_p\,\#)$.
   (c) $U_i \prec_C a_p \prec_C U_j$. Right multiplying s and t by a_p will contribute a_p' to $nf(sa_p\,\#)$ and $v_p a_p'$ to $nf(ta_p\,\#)$. But since $a_q \prec_C U_i \prec_C a_p$, it must be that $v_q >' v_p$, and we still have $nf(sa_p\,\#) >' nf(ta_p\,\#)$.

(ii) If $nf(su\,\#) >' nf(tu\,\#)$, then $nf(s\#) >' nf(t\#)$: This part is easier and similar. (Observe that $>'$ is a total ordering on terms.)

5 A Gröbner Basis Algorithm for Right Ideals

Gröbner bases for polynomials over Z have been considered in for example [6], and more generally in [7] for polynomials over Euclidean rings. Here, we are interested in constructing Gröbner bases for finitely generated right ideals in $\mathsf{Z}<X_1, \ldots, X_n>$, where $X_1, \ldots, X_n$ are the noncommuting indeterminates. In [1], Baader gave an algorithm for constructing weak Gröbner bases for right ideals in $\mathsf{Z}<X_1, \ldots, X_n>$. But in his algorithm only right compatible orderings with the boundedness property can be used to construct weak Gröbner bases.[6] In [9], an algorithm for computing strong Gröbner bases for right ideals in $\mathsf{Z}<X_1, \ldots, X_n>$ is given but that too is for right admissible term orderings with the boundedness property. In this section, we give an algorithm which can be used to construct Gröbner bases for polynomial right ideals in $\mathsf{Z}<X_1, \ldots, X_n>$ based on any right admissible term ordering.

For $m, n \in \mathsf{Z}$, we define the ordering $\succ_{\mathsf{Z}}$ as follows: $m \succ_{\mathsf{Z}} n$ if either $|m| > |n|$, or $|m| = |n|$ and $m < n$. Thus $-5 \succ_{\mathsf{Z}} 5$.

The set of terms we consider is the monoid $W_n = \{X_1, \ldots, X_n\}^*$.

Definition 5.1. *A total ordering $\succ$ on W_n is called right admissible iff it satisfies:*

1. *for any term $t \neq 1$, $t \succ 1$, and*
2. *for any terms s, t, u, if $s \succ t$, then $s\ u \succ t\ u$.*

A right admissible ordering $\succ$ on terms can be extended to monomials as follows: Let $a, b \in \mathsf{Z}$ and $s, t \in W_n$. Then $as \succ bt$ iff

1. $s \succ t$, or
2. $s = t$ and $a \succ_{\mathsf{Z}} b$.

Given an integer c, c induces an equivalence relation on Z as follows: $a =_c b$ iff there exists a k_1 such that $a = k_1 * c + b$ where $a, b, k_1 \in \mathsf{Z}$. Using this equivalence relation $=_c$, we define the *remainders* with respect to c as follows: Consider all equivalence classes induced by $=_c$. From each equivalence class, the smallest element wrt $\succ_{\mathsf{Z}}$ is a *remainder* of c. For example, the remainders of 5 are $0, 1, -1, 2, -2$; 3 is not a remainder of 5; similarly, the remainders of 4 are $0, 1, -1, 2$.

Let $\succ$ be a right admissible ordering and let $f = at + g$ be a polynomial in $\mathsf{Z}<X_1, \ldots, X_n>$ such that $t \in W_n$ is the greatest term in f with respect to $\succ$ and $a \in Z \backslash \{0\}$ is the coefficient of t in f. Then t is called the *head term* of f, denoted as $HT(f)$; a is called the *head coefficient* of f, denoted as $HC(f)$; $a\ t$ is called the *head monomial* of f, denoted as $HM(f)$, and $g = f - a\ t$ is called the *rest* of f, denoted as $R(f)$.

[6] Recall that an ordering has the boundedness property if and only if for every element, there are only finitely many elements less than it in the ordering.

A right-admissible ordering $\succ$ on monomials can be extended in the natural way to an ordering $\succ$ on polynomials by comparing head monomials first and if these are equal, recursively comparing the rest of the polynomials. Sets of polynomials can be compared using its multiset extension $\succ\succ$. Since $\succ$ is well-founded, $\succ\succ$ are well-founded too.

Definition 5.2. *Given a set F of polynomials, for any $f, g \in \mathsf{Z}<X_1, \ldots, X_n>$, f can be reduced to g wrt F, denoted by $f \rightarrow_F g$, iff*

1. *f contains a term t with coefficient a, (i.e. f contains a monomial at).*
2. *F contains a polynomial h such that $t = HT(h) \cdot s$ for some s and a is not a* remainder *of $HC(h)$.*
3. *$g = f - b\ h\ s$ where $a = b\ HC(h) + c$ and c is a* remainder *of $HC(h)$.*

We use $nf_F(f)$ to denote a normal form of f with respect to $\rightarrow_F$.

Lemma 5.1. *Let $F = \{f_1, \ldots, f_k\}$ be a set of polynomials. For any $f_i\ (1 \leq i \leq k)$, if $f_i \rightarrow^*_{F \setminus \{f_i\}} f_i'$, then $F \setminus \{f_i\} \cup \{f_i'\}$ generates the same right ideal as F.*

Proof. Based on the definition of reduction rules $\rightarrow_{F \setminus \{f_i\}}$, $f_i = f_i' + \sum a_j\ f_{h_j}\ t_j)$ where $f_{h_j} \in F \setminus \{f_i\}$. So $F \setminus \{f_i\} \cup \{f_i'\}$ generates the same right ideal as F. □

Algorithm 5.1: Gröbner basis Algorithm for Right Polynomial Ideals.

In the beginning, $F_0 = \{p_1, \ldots, p_m\}$ and all pairs of indices are unmarked. Without loss of generality, we assume all $HC(p_i) > 0$ $(1 \leq i \leq m)$. Assume that F_k $(k \leq 0)$ is already defined. If there is the zero polynomial 0 in F_k, we erase it. As long as there are $f = p_i$ and $g = p_j$ in F_k such that:

(a) (i, j) is not marked and
(b) $f = at + R(f)$ and $g = btr + R(g)$ for $a, b \in Z^+$, and $t, r \in W_n$, we do the following:
   1. $a \leq b$. Let $b = c\ a + d$ for some $c,\ d$, where d is a remainder of a. Define $g_1 = g - c\ f\ r = d\ t\ r + R(g) - c\ R(f)\ r$. Reduce g_1 to its normal form wrt F_k. If $HC(nf_{F_k}(g_1)) \geq 0$, define $g_1' = nf_{F_k}(g_1)$. Otherwise define $g_1' = -nf_{F_k}(g_1)$. $F_{k+1} = (F_k \cup \{g_1'\} \setminus \{g, 0\})$. Since $g = g_1 + c\ f\ r$, F_{k+1} generates the same right ideal as F_k by **Lemma 1**. Note that $HM(g_1') \prec HM(g)$.
   2. $r \neq 1, a > b$ and b does not divide a. Let $a = c\ b + d$ for some $c,\ d$, where d is a remainder of b. We define $g_1 = f\ r - c\ g = d\ t\ r + R(f)\ r - c\ R(g)$. Since $d < b$, g is reducible by g_1. Let $b = c_1\ d + \ d_1$ where d_1 is a remainder of d, and $g_2 = g - c_1\ g_1$. Reduce g_1, g_2 to their normal forms $g_1' = nf_{F_k}(g_1)$ and $g_2' = nf_{F_k}(g_2)$ wrt F_k. If $HC(g_1') \geq 0$, define $g_1'' = g_1'$, otherwise define $g_1'' = -g_1'$. Similarly if $HC(g_2') \geq 0$, define $g_2'' = g_2'$ and otherwise define $g_2'' = -g_2'$. Now

$F_{k+1} = (F_k \cup \{g_1'', g_2''\} \backslash \{g, 0\})$. In other words, g is deleted and non-zero normal forms are added to F_k. Since $g = g_2 + c_1\ g_1$, F_{k+1} generates the same ideal as F_k by **Lemma 1**. Again, $HM(g_1'), HM(g_2') \prec HM(g)$

3. $r \neq 1, a > b$ and b divides a. That is, there exists c such that $a = c\ b$. Define $g_1 = f\ r - c\ g = R(f)\ r - c\ R(g)$. Reduce g_1 to its normal form wrt F_k. If $nf_{F_k}(g_1) \geq 0$, define $g_1' = g_1$, otherwise define $g_1' = -g_1$. Define $F_{k+1} = F_k \cup \{g_1'\}$. Since $g_1 = f\ r - c\ g$, F_{k+1} generates the same ideal as F_k by **Lemma 1**. Mark (i, j). Here $HT(g_1) \prec HT(g)$.

Note that there is no need to mark the pair in cases 1 and 2, since g is deleted in both cases. For cases $1, 2$, clearly $F_k \succcurlyeq F_{k+1}$. For case 3, however, it is possible that $F_{k+1} \succcurlyeq F_k$. It is easy to see that if $p, q \in F_k$ such that $HT(p) = HT(q)$, then due to case 1 above, eventually one of these polynomials will be deleted from the basis leaving at most one polynomial with any head term.

Lemma 5.2. *Algorithm 5.1 terminates on any finite input basis of polynomials.*

Proof. Our proof is by contradiction. Define S to be the set of all finite sets of polynomials on which Algorithm 5.1 does not terminate, i.e.,

$$S = \{F \mid F \text{ is a finite set, Algorithm 5.1 does not terminate on } F\}.$$

Let $G_0 = \{f_1, \ldots, f_n\} \in S$ be the lowest element in S with respect to $\succcurlyeq$.

Claim 1: No two distinct polynomials in G_0 has the same head term.

Let $G_0, G_1, \ldots, G_k, \ldots$ be an infinite sequence that the algorithm produces.

Claim 2: For all $0 \leq i < j$, if polynomials $p \in G_i$ and $q \in G_j$ have the same head term, then either $p = q$ or $HC(p) > HC(q)$.

Without loss of generality, we may assume that f_1 is the highest polynomial in G_0 with respect to the ordering $\succ^p$. f_1 has to be a *persistent* polynomial, i.e., it must be that f_1 exists in all G_k $(k \geq 0)$; otherwise, if there exist some k, $f_1 \notin G_k$, then $G_k \in S$ and $G_0 \succcurlyeq^p G_k$, which contradicts the assumption that $G_0 = \{f_1, \ldots, f_n\} \in S$ is the smallest item in S.

Among the infinite sequence $G_0, \ldots, G_k, \ldots$, there must exist an infinite subsequence $G_{u_1}, \ldots, G_{u_i}, \ldots$ such that G_{u_i} is created because case 3 occurs between f_1 and f_j $(j > 1)$ in $G_{u_i - 1}$. If that is not the case, then there must be a t such that for the infinite sequence $G_t, G_{t+1}, \ldots$, case 3 does not occur between f_1 and any other polynomial in any $G_v\,(v \geq t)$. Cases $1, 2$ also will not occur between f_1 and any

f_j $(j > 1)$, because otherwise f_1 will be reduced to a smaller polynomial, but we already proved f_1 should exist in any G_k in this infinite sequence. Note also that f_1 cannot be involved in any reduction step since all other polynomials are below it in the ordering $\succ$. Therefore we can "get rid of" f_1, i.e., $G_t \setminus \{f_1\} \prec\!\!\prec^p G_0$ will produce an infinite sequence, which contradicts the the assumption that $G_0 = \{f_1, \ldots, f_n\} \in S$ is the smallest item in S.

But since for $HT(f_1)$, there are only finitely many prefixes, it is impossible that case 3 occurs infinitely often between f_1 and some other polynomials. □

The correctness of the above Gröbner basis algorithm is established by a series of lemmas, patterned after the proof in [1] using ideas from [6] to establish that the proposed algorithm indeed computes a strong Gröbner basis for a right ideal over $\mathsf{Z}\!<\!X_1, \ldots, X_n\!>$.

Lemma 5.3. *Starting with a finite input basis G_0, let Algorithm 5.1 terminate in G_n by generating a sequence of bases $G_0, G_1, \ldots, G_n$. For any term t, if $f = \sum_{g_i \in G_0} g_i a_i$ with $HT(g_i a_i) \prec t$, then $f = \sum_{g'_i \in G_n} g'_i a'_i$ with $HT(g'_i a'_i) \prec t$.*

To prove this lemma, it is enough to prove the following claim.

Claim 3: If $f = \sum_{g_i \in G_k} g_i a_i$ with $HT(g_i a_i) \prec t$, then $f = \sum_{g'_i \in G_{k+1}} g'_i a'_i$ with $HT(g'_i a'_i) \prec t$.

Proof. Assume the algorithm is currently running on $G_k = \{g_1, \ldots, g_s\}$ $(0 \leq k \leq n-1)$. If case 3 of the algorithm happens, we will have $G_k \subseteq G_{k+1}$, so the claim is true. Otherwise, if case 1 of the algorithm happens, then we will have either $G_{k+1} = G_k \setminus \{g_u\} \bigcup \{nf_{G_k}(h)\}$ or $G_{k+1} = G_k \setminus \{g_u\} \bigcup \{nf_{G_k}(h) \cdot (-1)\}$ where $g_u = c g_v r + h$ with $HT(h) \preceq HT(g_u)$. By keeping track of the reduction steps of how h is reduced to $nf_{G_k}(h)$, we can get $h = nf_{G_k}(h) + \sum_{g_{h_i} \in G_k \setminus \{g_u\}} (g_{h_i} b_i)$ with $HT(g_{h_i} b_i) \preceq HT(h)$. Therefore, if $f = \sum_{g_i \in G_k} (g_i a_i)$, then $f = \sum_{g_i \in G_k, i \neq u} (g_i a_i) + g_u a_u = \sum_{g_i \in G_k, i \neq u} (g_i a_i) + (c g_v r + nf_{G_k}(h) + \sum_{g_{h_i} \in G_k \setminus \{g_u\}} (g_{h_i} b_i)) a_u$. Since $HT(g_{h_i} b_i) \preceq HT(h) \preceq HT(g_u)$, we will have $HT(g_{h_i} b_i a_u) \leq HT(h a_u) \preceq HT(g_u a_u) \prec t$, and the claim is true. In the same way, we can prove that if case 2 happens, the claim still holds. □

Lemma 5.4. *Starting with a finite input basis G_0, let Algorithm 5.1 terminate in G_n by generating a sequence of bases $G_0, G_1, \ldots, G_n$. Let $G_n = \{g_1, \ldots, g_k\}$. If $HT(g_i)$ is a prefix of $HT(g_j)$ $(1 \leq i, j \leq k)$, in other words, $HT(g_j) = HT(g_i) r$ for some r, then $HC(g_i) = c HC(g_j)$ for some $c \in Z \setminus \{0\}$, and $g_i r - g_j c = \sum_{g_{u_v} \in G_n} g_{u_v} a_v$ with $HT(g_{u_v} a_v) \prec HT(g_j)$ for some v $(1 \leq v \leq k)$.*

Proof. Consider two polynomials g_i, g_j in G_n such that $HT(g_i)$ is a prefix of $HT(g_j)$. Let $g_i = at + R(g_i)$, $g_j = btr + R(g_j)$. We have the following cases:

1. $r = 1$. When $r = 1$, the condition of case 1 in the algorithm will be satisfied. If this case happens, the algorithm will replace f_i or f_j with a smaller polynomial. Thus one of them would have been deleted. But since the algorithm terminates on G_k, this case is not possible.

2. $r \neq 1$ and $a \leq b$. For the same reason as above, this case is also not possible.

3. $r \neq 1,\ a > b$ and b does not divide a. This case is also not possible, since case 2 of the algorithm will apply here.

4. $r \neq 1,\ a > b$, b divides a. Thus $a = bc$ for some c and the condition of case 3 of the algorithm is satisfied. Since the algorithm terminates on G_n, (i, j) must be marked. According to the algorithm there exists $l\,(1 \leq l \leq n)$ such that $G_{l+1} = G_l \bigcup nf_{G_l}(R(g_i)r - R(g_j)c)$. By keeping track of the reduction steps, we can get $R(g_i)r - R(g_j)c = nf_{G_l}(R(g_i)r - R(g_j)c) + \sum_{g_{h_u} \in G_l} g_{h_u} a_u$ with $HT(g_{h_u}a_u) \preceq HT(R(f_i)r - R(f_j)c) \prec HT(g_j)$. So by Lemma 5.4, we can get $R(g_i)r - R(g_j) = \sum_{g_{h'_u} \in G_n} f_{h'_u} a'_u$ with $HT(g_{h'_u}) \prec HT(g_j)$. □

Lemma 5.5. *Let Algorithm 5.1 generate $G_n = \{g_1, \cdots, g_m\}$ from a finite input basis G. For any $f \in <G>$, there exist polynomials $a_1, \ldots, a_n$ such that $f = \sum_{g_i \in G} g_i a_i$ and $|\{\ i \mid HT(g_i a_i) = HT(f)\}| = 1$.*

Proof. We prove this lemma by contradiction. For brevity, let $I = \{\ i \mid HT(g_i a_i) = HT(f)\}$.

Assume there exist $f \in <G>$ such that for any set of polynomials $\{a_1, \ldots, a_n\}$ which satisfies $f = \sum_{g_i \in G} g_i a_i$, we have $|I| > 1$. Choose the set of polynomials $\{a_1, \ldots, a_n\}$ which have the smallest $|I|$. Assume $|I| = k$. Since $k > 1$, there exist $i,\ j$ $(1 \leq i, j \leq n)$ such that $HT(g_i a_i) = HT(g_j a_j) = HT(f)$. Let $a_i = c_i r_i, a_j = c_j r_j$. Since $HT(g_i a_i) = HT(g_j a_j)$, either $HT(g_i)$ is a prefix of $HT(g_j)$ or vice versa. Without loss of generality, we assume $HT(g_i) = HT(g_j)r$. So $r_j = r r_i$. By Lemma 5.4, there must be some c such that $HC(g_j) = HC(g_i)c$, and $g_j r - g_i c = \sum_{g_{h_k} \in G} g_{h_k} b_k$ with $HT(g_{h_k} b_k) \prec HT(g_i)$. Therefore $g_j r_j - g_i r_i c = (g_j r - g_i c) r_i = \sum_{g_{h_k} \in G} g_{h_k} b_k r_i$ with $HT(g_{h_k} b_k r_i) \prec HT(g_i r_i)$. So we have

$$\begin{aligned} f &= (g_j r_j - g_i r_i c)c_j + g_i r_i c c_j + g_i r_i c_i + \sum_{g_v \in G, v \neq i,j} g_v a_v \\ &= \sum_{g_{h_k} \in G} g_{h_k} b_k r_i c_j + g_i r_i (c c_j + c_i) + \sum_{g_v \in G, v \neq i,j} g_v a_v. \end{aligned}$$

For this representation of f, since $HT(h_k b_k r_i c_j) \prec HT(g_i r_i) = HT(f)$, $|I| = k-1 < k$, which yields a contradiction. □

Lemma 5.6. *Let Algorithm 5.1 generate $G_n = \{g_1, \cdots, g_m\}$ from a finite input basis G_0. Any polynomial $f \in <G_0>$ can be reduced to 0 with respect to $\rightarrow_{G_n}$.*

Proof. Since $f \in < G_0 >$, $f = \sum_{g_i \in G_0} g_i a_i$ for some polynomials a_i. We prove this lemma by induction on f.

Base case: When $f = 0$, there is nothing to prove.

Assumption: Assume the lemma is true for all $g \prec f$. For f, based on Lemma 5.5, f can be expressed by $f = \sum_{g_i \in G} g_i a_i$ with $|I| = |\{\ i \mid HT(g_i a_i) = HT(f)\}| = 1$. Assume $I = \{j\}$. Since $HT(f) = HT(g_j a_j), HC(f) = HC(g_j)HC(a_j)$, f can be reduced by g_j to a smaller polynomial $f_1 = f - g_j a_j$. By assumption, $f_1 \rightarrow_G^* 0$, so $f \rightarrow_G^* 0$. □

Theorem 5.1. *Starting with a finite input basis G_0, let Algorithm 5.1 terminate in G_n by generating a sequence of bases $G_0, G_1, \ldots, G_n$. Then, G_n is a* strong *Gröbner basis.*

The key idea in the proof is that for any two polynomials p_1 and p_2, if $p_1 - p_2$ reduces to 0, then one of p_1 and p_2 are reducible by $\rightarrow_{G_n}$. Further, the ordering on Z satisfies the unique remainder property discussed in [6].

Proof. G_n is weak Gröbner basis by Lemma 5.6. Let p_1 and p_2 be the smallest (with respect to the ordering $\succ^p$) distinct polynomials such that

1. $p_1 - p_2 \in <G_0>$ and
2. p_1, p_2 are in normal form with respect to G_n.

Let t be the highest term whose coefficients in p_1 and p_2 differ. Let c_1 be the coefficient of t in p_1 and c_2 be the coefficient of t in p_2 where $c_1 \neq c_2$. Thus the head monomial of $p_1 - p_2$ is $(c_1 - c_2)t$. By Lemma 5.5, there is a polynomial g_j in G_n such that $HT(g_j)$ is a prefix of t and $HC(g_j)$ divides $(c_1 - c_2)$. But then it cannot be that both c_1 and c_2 are remainders of $HC(g_j)$; hence one of p_1 and p_2 must be *reducible* by G_n, which is a contradiction. □

6 Conclusion

Unification problems for theories admitting modular multiplication and exponentiation are discussed. As shown above, for theories that admit the distributivity property

of exponentiation over multiplication, unifiability checks are undecidable. However, if this property is excluded, then unification algorithms for various restricted theories, depending upon properties of exponentiation used, can be obtained. In most cases, these algorithms are of high complexity.

In order to integrate these unification algorithms into the NRL Protocol Analyzer (NPA) and effectively use them for cryptographic protocol analysis, it will be useful to further specialize these algorithms so as to keep their complexity manageable in practice. That is likely to be the main challenge in investigating the effectiveness of the approach proposed in [13]. This issue needs further investigation.

References

1. F. Baader. Unification in Commutative Theories, Hilbert's Basis Theorem, and Gröbner Bases. *J. ACM,* 40 (3), 1993, 477–503.
2. F. Baader and W. Nutt. Adding Homomorphisms to Commutative/Monoidal Theories, or: How Algebra Can Help in Equational Unification. Proc. *4th Intl. Conf. on Rewriting Techniques and Applications (RTA 91),* LNCS 488, 1991, 124–135.
3. F. Baader and K.U. Schultz. Unification in the Union of Disjoint Equational Theories: Combining Decision Procedures. Proc. *11th Conference on Automated Deduction (CADE-11),* Saratoga Springs, NY, Springer LNAI 607, 1992, 50–65.
4. J. Clark and J. Jacob. *A Survey of Authentication Protocol Literature: Version 1.0.* Unpublished Technical Report, Department of Computer Science, University of York, UK, Nov 1997. Available at the URL: `www-users.cs.york.ac.uk/~jac/papers/drareviewps.ps`.
5. M. Davis. *Computability and Unsolvability.* Dover Publications, 1982.
6. A. Kandri-Rody and D. Kapur. Computing the Gröbner Basis of a Polynomial Ideal over Integers. Proc. *Third MACSYMA Users' Conference,* Schenectady, NY, July 1984, 436–451.
7. A. Kandri-Rody and D. Kapur. An Algorithm for Computing the Gröbner Basis of a Polynomial Ideal over an Euclidean Ring. *Journal of Symbolic Computation* 6 (1), August 1988, 37-57.
8. D. Kapur, P. Narendran, and L. Wang. *A Unification Algorithm for Analysis of Protocols with Blinded Signatures.* TR 02-5, Department of Computer Science, SUNY, Albany, NY. To appear in the Festschrift for Jörg Siekmann (Dieter Hutter, Werner Stephan, eds.), Lecture Notes in Artificial Intelligence 2605, Springer. An expanded version is available at the URL: `www.cs.albany.edu/~dran/my_research/papers/techrep_fest.ps`.
9. K. Madlener and B. Reinert. On Gröbner bases in Monoid and Group Rings. SEKI Report SR-93-08, Universität Kaiserslautern, Germany.
10. C. Meadows. The NRL Protocol Analyzer: An Overview. *J. Logic Programming,* 26(2), 1996, 113–131.
11. C. Meadows. Analysis of the Internet Key Exchange protocol using the NRL Protocol Analyzer. In: Proc. the 1999 Symp. on Security and Privacy, IEEE.
12. C. Meadows, P. Syverson and I. Cervesato. Formal Specification and Analysis of the Group Domain of Interpretation Protocol using NPATROL and the NRL Protocol Analyzer. To appear in the *Journal of Computer Security.*
13. C. Meadows and P. Narendran. A Unification Algorithm for the Group Diffie-Hellman Protocol. *Workshop on Issues in the Theory of Security (WITS 2002),* Portland, OR, Jan 2002.
14. P. Narendran. On solving linear equations over polynomial semirings. In: Proc. 11th Annual Symp. on Logic in Computer Science (LICS), NJ, July 96, 466–472.
15. P. Narendran, F. Pfenning, and R. Statman. On the Unification Problem for Cartesian Closed Categories. *Journal of Symbolic Logic* 62 (2), June 97, 636–647.
16. O. Pereira and J.-J. Quisquater. A Security Analysis of the Cliques Protocols Suites. Proc. *14th IEEE Computer Security Foundations Workshop,* June 2001.

17. R.L. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public Key Cryptosystems. *Communications of the ACM* 21 (2), 1978, 120–126.
18. G. Simmons and C. Meadows. The Role of Trust in Information Integrity Protocols. *Journal of Computer Security* 3 (2), 1994.
19. S.G. Stubblebine and C. Meadows. On Searching for Known and Chosen Cipher Pairs using the NRL Protocol Analyzer. Presented at the DIMACS Workshop on Design and Formal Verification of Security Protocols, September 1997.
20. L. Wang. *Equational Unification techniques for Cryptographic Protocol Analysis.* Ph.D. dissertation, Dept. of Computer Science, University at Albany–SUNY, Albany, NY, 2003.