DE GRUYTER
OPEN

TATRA
MOUNTAINS
Mathematical Publications

DOI: 10.1515/tmmp-2016-0040
Tatra Mt. Math. Publ. **67** (2016), 205–219

MRHS EQUATION SYSTEMS THAT CAN BE SOLVED IN POLYNOMIAL TIME

Pavol Zajac

ABSTRACT. In this article we study the difficulty of solving Multiple Right--Hand Side (MRHS) equation systems. In the first part we show that, in general, solving MRHS systems is NP-hard. In the next part we focus on special (large) families of MRHS systems that can be solved in polynomial time with two algorithms: one based on linearisation of MRHS equations, and the second one based on decoding problems that can be solved in polynomial time.

1. Introduction

Non-linear Multiple Right-Hand Side (MRHS) equation systems used in algebraic cryptanalysis were introduced by S e m a e v in [6]. Non-linear MRHS equation can be written as an inclusion $x\mathbf{M} \in S$, where x is a single vector, $\mathbf{M}$ is left-hand side matrix, and S is a set of vectors (right-hand sides). Individual right hand side sets are typically small and explicitly enumerated. A compressed MRHS representation [5] uses binary-decision diagrams to store the right-hand side sets in a more compact way. We summarize basic definitions more formally in Section 2.

MRHS equation systems can be used to efficiently encode ciphers for algebraic cryptanalysis. Typically, for a particular cipher, the right hand side sets are constructed directly from S-boxes. Then left-hand side matrix represents linear combinations of internal bits and key bits that are used as S-box inputs and outputs. For ciphers with complex linear layers (such as AES) this leads to a more efficient representation than k-CNF encoding used for algebraic cryptanalysis based on SAT-solvers. Moreover, the MRHS representation is more compact than the corresponding multivariate equations [3].

2010 Mathematics Subject Classification: 94A60, 68P25.
Keywords: algebraic cryptanalysis, MRHS equations, NP-hard, post-quantum cryptography.
Support by NATO's Public Diplomacy Division in the framework of "Science for Peace", Project MD.SFPP 984520, is acknowledged.

Each MRHS equation corresponds to a system of multivariate polynomial equations. There are various methods proposed for solving general MRHS systems based on Gluing [3], [6], [7] or transformation to another type of problem [11], [13]. All of them have a worst-case exponential complexity. In general, finding a solution of a MRHS equation system is a difficult problem. In Section 3, we formally show that it is NP-hard.

In this article, we want to study easy to solve instances of the MRHS problem. Our primary motivation lies in post-quantum cryptography area. Some of the NP-hard problems have been successfully used to construct quantum-resistant cryptosystems. McEliece cryptosystem is based on the hardness of syndrome decoding problem [4]. Hardness of MQ problem is used as a basis of security for multivariate cryptosystems [2]. All these systems exploit the fact that although the selected problem is (under $P \neq NP$) difficult in general, there are instances that can be solved efficiently with the help of some trapdoor information.

We do not identify such concrete trapdoors, but study the systems in a more general way. This research is similar to a research of restricted parametrised versions of 3-SAT problem. It is known that instances of 3-SAT with restricted number of occurrences of literals below 3 can be solved in polynomial time [8]. Similarly, random instances of SAT that combine 2-SAT and 3-SAT formulas in a specified ratio can be solved in polynomial time [14]. We are interested, whether similar observation can be made about MRHS equation systems, and what kinds of systems are easy to solve.

In Section 4, we study two polynomial time algorithms that can solve MRHS equation systems from large infinite families of systems. In the first part, we study in more detail the linearisation algorithm introduced in [3]. We identify a family of MRHS systems that can be solved solely by linearisation in polynomial time. In the second part, we extend a new algorithm proposed in [12]. This algorithm is based on restricted syndrome decoding. We again identify a family of MRHS systems that can be solved by this technique in polynomial time. In Section 5, the results are summarised and compared.

2. Notation and preliminaries

In this section we summarize basic definitions and notations. We will mostly follow the definitions and notations from [13]. In this article:

- we will always use the underlying finite field $\mathbb{F} = GF(2)$ (although many of the results can be generalized to any finite field);
- all vectors are row vectors, denoted by u, v, w;
- matrices are typeset in bold, e.g., $\mathbf{M}$;

- number of columns of matrix $\mathbf{M}$ is denoted by $cols(\mathbf{M})$;
- if S is a set of vectors from $\mathbb{F}^l$, then we say l is the dimension of S, denoted by $l = dim(S)$;
- if S is a set of vectors, then $\mathbf{S}$ is a matrix with rows from S (in arbitrary order);
- if S is a set of vectors, and $\mathbf{T}$ a matrix, then $S \cdot \mathbf{T} = \{v\mathbf{T}; v \in S\}$;
- m, n, l, k are all natural numbers (with a special meaning related to MRHS system's dimensions).

Definition 1. Multiple Right-Hand Sides (MRHS) equation is defined by a formal predicate on $x \in \mathbb{F}^n$

$$x\mathbf{M} \in S, \tag{1}$$

where $\mathbf{M}$ is an $n \times l$ matrix over $\mathbb{F}$, and S is a set of k vectors from $\mathbb{F}^l$. We say that $v \in \mathbb{F}^n$ is a solution of MRHS equation, if the predicate (1) holds for this particular $x = v$.

Multiple Right-Hand Sides (MRHS) equation system (MRHS system) is a conjunction of m MRHS equations:

$$x\mathbf{M}_1 \in S_1 \wedge x\mathbf{M}_2 \in S_2 \wedge \cdots \wedge x\mathbf{M}_m \in S_m. \tag{2}$$

Vector v is a solution of MRHS system, if and only if it is a solution of each of the equations in the system. Each MRHS system can be written as a single MRHS equation

$$x\mathbf{M} = x\left(\mathbf{M}_1\mathbf{M}_2 \ldots \mathbf{M}_m\right) \in S_1 \times S_2 \times \cdots \times S_m. \tag{3}$$

Set S is now the Cartesian product $S_1 \times S_2 \times \cdots \times S_m$. In the text, we will denote a particular MRHS system of this form by $\mathbb{M}$. A set of all solutions of MRHS system $\mathbb{M}$ will be denoted by $Sol(\mathbb{M})$.

We will always suppose that $\mathbf{M}$ has maximal rank. Otherwise we can reduce the number of variables by a change of basis. If $rank(\mathbf{M}) = cols(\mathbf{M})$, it is trivial to compute $Sol(\mathbb{M})$: for every vector $w \in S$, we can find a corresponding set of solutions by solving linear equation system $x\mathbf{M} = w$. On the other hand, if $cols(\mathbf{M}) > rank(\mathbf{M})$, only a fraction of vectors from S will have a corresponding solution x. In this case, left-hand side matrix $\mathbf{M}$ is a generator matrix of some linear code $\mathcal{C}$ (we will call it the left-hand side code). The solution space of the MRHS equation/system can then be expressed as $Sol(\mathbb{M}) = \mathcal{C} \cap S$. If the set S is a set of randomly chosen vectors, and left-hand side code $\mathcal{C}$ is a random linear code over $\mathbb{F} = GF(2)$, the expected number of solutions is $|Sol(\mathbb{M})| = 2^{n-cols(\mathbf{M})}|S|$.

There are various methods proposed for solving general MRHS systems, all of them with worst-case exponential complexity. It is possible to check the MRHS predicate for all 2^n potential vectors x (exhaustive search). This can be done

coordinate-wise, by guessing values of some coordinate x_i, replacing it in the system by constant, and checking whether it is still consistent on individual MRHS equation level. The Gluing algorithm and its improved versions [3], [6], [7] are based on replacing two MRHS equations in a system by computing partial Cartesian products $S_{i,j} = S_i \times S_j$, merging left-hand side matrices, and removing conflicting vectors from $S_{i,j}$, which can be identified when $rank(M_i|M_j) < rank(M_i) + rank(M_j)$. Final system will have $rank(\mathbf{M}) = cols(\mathbf{M})$, and a single joined right-hand side set. In [11], a new type of algorithm for solving MRHS systems was presented, that changes the problem to a group factoring instance. It is further possible to transform this problem to a specific syndrome decoding problem [13].

3. MRHS systems and NP-completeness

In this section we will show formally that solving MRHS systems is in general an NP-hard problem. To study MRHS systems from a computational complexity view, we start by defining a decision problem related to MRHS equations.

Definition 2. Let $x\mathbf{M} \in S_1 \times S_2 \times \cdots \times S_m$ define an MRHS equation system $\mathbb{M}$. MRHS solution problem is the following decision problem: Given $\mathbb{M}$, return "yes" if and only if $Sol(\mathbb{M}) \neq \emptyset$.

In the decision version, we only ask, whether some MRHS equation system has a solution or not. If we have an oracle for a decision version, it is easy to find all solutions of $\mathbb{M}$ by a search algorithm with backtracking. In each step of the algorithm we append solvable $\mathbb{M}$ with the equation of the type $x_i = 0$ (or replace variable with a constant, respectively). If the resulting system is still solvable, we know that $x_i = 0$ is a part of the solution (otherwise, it is $x_i = 1$).

We show that MRHS solution problem is in NP, if we restrict instances to a family of polynomially sized systems (with a single complexity parameter n).

Definition 3. Let $\mathcal{M}_n$ denote a set of MRHS equation systems over $GF(2)$ in n variables of the form $x\mathbf{M} \in S_1 \times \cdots \times S_m$. We will call a family of sets $\mathcal{M}_n$ polynomially bounded in n if all $m, k_i, l_i < poly(n)$, where $k_i = |S_i|$, and $l_i = dim(S_i)$.

Lemma 1. *MRHS solution problem is in NP for instances $\mathbb{M} \in \mathcal{M}_n$, where $\mathcal{M}_n$ is from a family of polynomially bounded MRHS systems.*

Proof. Let $\mathbb{M}$ be a solvable MRHS system. Let $x \in GF(2)^n \in Sol(\mathbb{M})$. We can verify that x is indeed a solution of the system by computing $s = x\mathbf{M}$, and verifying that each part of s belongs to a corresponding set S_i. Matrix $\mathbf{M}$ has n rows and $M = \sum_{i=1}^{m} l_i$ columns. Vector-matrix multiplication then takes at most

nM multiplications (ANDs), and $(n-1)M$ additions (XORs). As $m, l_i < poly(n)$, we also get that $nM < poly(n)$. Condition that $x\mathbf{M} \in S_1 \times \cdots \times S_m$ can be checked with at most $\sum_{i=1}^{m} k_i$ comparisons. Again, due to $k_i < poly(n)$, the number of comparisons is also polynomially bounded w.r.t. n. Thus, MRHS solution problem restricted to polynomially bounded MRHS systems is in NP. □

We note that in general, some family of MRHS systems might not be polynomially bounded, as the dimension and size of sets S_i can grow exponentially (e.g., a sequence of systems during the Gluing algorithm). However, representing such systems in computer is impractical. On the other hand, systems which arise in algebraic cryptanalysis of ciphers can be typically considered as polynomially bounded systems, as the size of sets S_i is restricted to some constant (size of S-boxes/non-linear elements), and the number of equations m (related to the number of operations in the cipher instance) can be considered as polynomially bounded in n (otherwise, it would be impractical to implement such a family of ciphers).

Having shown MRHS solution problem to be in NP, we can furthermore show that this problem is NP-hard, and thus NP-complete. To show this, we show a polynomial reduction from the 3-SAT problem to the MRHS solution problem.

Lemma 2. *MRHS solution problem is NP-hard.*

Proof. We show that 3-SAT polynomially reduces to MRHS solution problem. Let $\varphi = C_1 \wedge C_2 \wedge \ldots \wedge C_m$ be an instance of 3-SAT. Each C_i can be written as

$$\left(x_{i,j_1}^{c_{i,j_1}} \vee x_{i,j_2}^{c_{i,j_2}} \vee x_{i,j_3}^{c_{i,j_3}}\right),$$

where c denotes a presence of a complement: $x_i^0 = x_i$, and $x_i^1 = \neg x_i$. We can now define an MRHS system $\mathbb{M}$

$$x\left(\mathbf{M}_1\mathbf{M}_2\cdots\mathbf{M}_m\right) \in S_1 \times S_2 \times \cdots \times S_m,$$

where each $\mathbf{M}_i \in GF(2)^{(n\times 3)}$. The tth column of $\mathbf{M}_i$ has a single non-zero element at row j_t. Each set S_i consists of 7 vectors, which are all of the possible satisfying assignments to C_i. We use natural mapping between $GF(2)$ and Boolean values: 0 is mapped to false and 1 to true (and vice-versa). Namely, $S_i = GF(2)^3 \setminus \{(c_{i,j_1}, c_{i,j_2}, c_{i,j_3})\}$. Note that complement notation was chosen, so that under this mapping we can write $x_i^{c_i} = x_i + c_i$.

It is now easy to show that φ is satisfiable if and only if $\mathbb{M}$ is solvable. Suppose that x is a solution of $\mathbb{M}$. Then it must hold that $x\mathbf{M}_i \in S_i$ for each i. Sets S_i hold all satisfying assignments for which C_i is true. Thus each C_i is true, which means that φ is true.

Similarly, if we have some truth assignment for x_i's for which φ is satisfied, we can map them to some vector $x \in GF(2)^n$. Now $x\mathbf{M}_i$ have values $(x_{i,j_1} + c_{i,j_1}, x_{i,j_2} + c_{i,j_2}, x_{i,j_3} + c_{i,j_3})$. For φ to be true, each of C_i's must be true.

This means that at least one of $x_{i,j}+c_{i,j}=1$ for each i. Thus, $(x_{i,j_1}+c_{i,j_1}, x_{i,j_2}+c_{i,j_2}, x_{i,j_3}+c_{i,j_3}) \neq (c_{i,j_1}, c_{i,j_2}, c_{i,j_3})$, and $x\mathbf{M}_i \in S_i$. From this we can conclude that x is a solution of $\mathbb{M}$. □

Note that $\mathbb{M}$ from the proof of Lemma 2 can be understood as polynomially bounded in n, as sizes of S_i are constant, and $m \leq n(n-1)(n-2)/6$. Thus MRHS solution problem restricted to polynomially bounded systems is an NP-complete problem.

4. Easy and hard instances of MRHS problem

As far as we know, all of the known algorithms to solve NP-complete problems in general require superpolynomial time. However, there are some restricted classes of problems that can be solved in polynomial time. Although general SAT problem is NP, we know that 2-SAT, Horn-SAT or XOR-SAT are in P. Also some restricted versions of 3-SAT problem are also known to be in P: instances with restrictions on the number of occurrences of literals [1], [8], or instances that combine 2-SAT and 3-SAT formulas in a specified ratio [14]. In this section we identify two (infinite) families of MRHS systems, which can be solved with known (probabilistic) polynomial time algorithms.

In the following, we will study "random MRHS systems". Formally, by a random MRHS system, we will understand an MRHS system chosen uniformly random from a set of all systems $x\mathbf{M} \in S_1 \times S_2 \times \ldots S_m$, where $\mathbf{M}$ is a $n \times ml$ matrix over $GF(2)$ with $n < ml$, and $rank(\mathbf{M}) = n$. Each S_i is a set of k distinct vectors from $GF(2)^l$. We will denote this set by $\mathcal{M}_{n,m,k,l}$, or just $\mathcal{M}$ if we do not need to specify concrete parameters.

4.1. Linearisation

We start with the study of the method that was introduced in [3]. The goal of the method is to produce ordinary linear equations from MRHS equations. We call this method "linearisation" of MRHS system. In this paper, we study this method in more details, and analyse, when it is possible to use this method to solve whole MRHS systems in polynomial time.

There exists a well-defined family of MRHS systems that can be linearised. This family has right-hand side sets with small number of solutions. We can always linearise MRHS equation with $k \leq l$ by using the following lemma:

Lemma 3. *Let $x\mathbf{M} \in S$ be an MRHS equation with $|S| \leq dim(S)$. We can find a linear equation $x \cdot a^T = b$ and a smaller MRHS equation $x\mathbf{M}' \in S'$ such that $v\mathbf{M} \in S$ holds if and only if both $v \cdot a^T = b$ and $v\mathbf{M}' \in S'$ hold for some v.*

P r o o f. Let $\mathbf{S}$ be a matrix with rows constructed from vectors in S. Dimensions of $\mathbf{S}$ are $k \times l$. We can find a regular $l \times l$ matrix $\mathbf{T}$, such that $\mathbf{R} = (\mathbf{ST})^T$ is in a reduced row echelon form. As $k \leq l$, either $\mathbf{R}$ is an identity matrix, or has some all-zero rows at the end.

Suppose that the last row of $\mathbf{R}$ is all-zero. Then the last column of $\mathbf{ST}$ contains only zeros. Now, for any x for which $x\mathbf{M} \in S$, it must hold that $x\mathbf{MT} \in S \cdot \mathbf{T}$. Let a^T be the last column of $\mathbf{MT}$. As any vector in $\mathbf{ST}$ can only contain zero at the last position, then clearly $x \cdot a^T = 0$. Furthermore, it still holds that $x\mathbf{M}' \in S'$, where $\mathbf{M}'$ contains the first $l-1$ columns of $\mathbf{MT}$, and similarly, S' is a set of vectors from $S \cdot \mathbf{T}$ restricted to $l-1$ coordinates.

If $\mathbf{R}$ is an $l \times l$ identity matrix, we can apply further transform $\mathbf{T}_2$, which adds the first $l-1$ columns to the last column of $\mathbf{ST}$. Then we will get column of all ones, and we can extract linear equation $x \cdot a^T = 1$ similarly to the previous case.

The proof of opposite direction can be done by gluing the linear equation and $x\mathbf{M}' \in S'$ (append a^T to the left-hand side, and the all-zero or all one column to the right-hand side), and applying the inverse transform $\mathbf{T}^{-1}$ to obtain the original $x\mathbf{M} \in S$. □

If the number of right-hand sides is small, we can apply Lemma 3 multiple times on the same MRHS equation. Furthermore, we can combine linear equations $xa_i^T = b_i$ obtained from multiple MRHS equations in the MRHS system, as they must all be satisfied for each solution of the MRHS system. Putting all these extracted linear equations together, we obtain a linear system

$$x\mathbf{A} = x\left(a_1^T a_2^T \cdots a_N^T\right) = (b_1 b_2 \ldots b_N) = b. \tag{4}$$

Vector x is a solution of the original MRHS system $\mathbb{M}$, if it is a solution of the linear system (4), and the remaining (part of the original) MRHS system $\mathbb{M}'$. Let $x \in GF(2)^n$, and let $rank(\mathbf{A}^T) = n - c$. Solution space of (4) then contains 2^c vectors. Thus, we can construct $Sol(\mathbb{M})$ by verifying whether each of the 2^c solutions of (4) is also a solution of $\mathbb{M}'$. Suppose that for some family of polynomially bounded MRHS systems we can bound $c = O(\log n)$. Then solution space of (4) is polynomially bounded in n. Furthermore, each verification against $\mathbb{M}'$ takes only polynomial time. Thus, in this case, we can solve instances of MRHS solution problem in polynomial time.

From the MRHS system with m MRHS equations, all of them having $k \leq l$, we can extract at least $m(k-l+1)$ linear equations. For the purpose of asymptotic analysis, we can suppose that these linear equations are independent. The reduced system will have right-hand sides with dimensions $2l-k-1$. If $m(k-l+1) \geq n - c\log n$ for some constant c, we can expect at most n^c solutions to the extracted linear system, which can then be tested against the reduced system in polynomial time. Thus, random MRHS systems with $k \leq l$, and

$m \geq \frac{n}{k-l+1} - c' \log n$, with $c' = \frac{c}{k-l+1}$, can be solved in polynomial time by linearisation. Note that the linearisation algorithm can also decide in polynomial time that the MRHS system has no solution. This happens if either the linear system has no solution, or if no solution of the linear system is also a solution of the reduced system $\mathbb{M}'$.

If m is smaller then the bound $\frac{n}{k-l+1} - c' \log n$, we can still at least reduce the system size by linearisation (using Lemma 3). From the extracted system of $m(k-l+1)$ linear equations, we can express up to $m(k-l+1)$ variables as affine functions of remaining $n - m(k-l+1)$ variables. We can then substitute these variables in the system $x\mathbf{M}' \in S'_1 \times \cdots \times S'_m$. The reduced MRHS system will have matrix $\mathbf{M}'$ with dimensions $\big(n - m(k-l+1)\big) \times \big(m(2l-k-1)\big)$ and right-hand sides with dimensions $2l-k-1$ (k remains unchanged).

Suppose, that $m > n$, and the MRHS system has exactly one solution. This means in each set S_i there is exactly one vector (valid vector), that is a part of the correct solution. We can run a probabilistic algorithm, that will remove $k-l$ vectors at random from each S_i. Then it will try to find the solution by linearisation. The algorithm will succeed, if no valid vector was removed from any S_i. This happens with probability approximately $(l/k)^m$. We can expect the solution after approximately $(k/l)^m$ re-tries. Thus, the expected complexity of such algorithm is exponential for any system with $k > l$. However, depending on the parameters, this can be more efficient than exhaustive search over all 2^n possible vectors $x \in GF(2)^n$. For example, we can model a cipher with an AND-XOR circuit. We can then transform this circuit into a MRHS system on AND gate level, giving us right-hand side sets with $k = 4$, $l = 3$. For a random instance of this kind of MRHS system, we can expect 1 solution when $m = n$. As $k - l = 1$, we can linearise system by removing one vector from each S_i. We can solve the system by linearisation with probability $(3/4)^n$. This means, that the solution will be found in average with $2^{0.415n}$ retries, instead of 2^{n-1} (expected average complexity of the exhaustive search).

The reduction algorithm can also be useful if we have some extra information or structure in the system, and we can remove $(k-l)$ invalid vectors from S_i with probability 1. For some types of systems, this can be accomplished by local reduction techniques, such as the method of syllogisms [9], [10]. Other example of exploiting linear subsystem is a combination of guessing and linearisation from [3].

4.2. Syndrome decoding

In [12] a new probabilistic polynomial time algorithm for solving MRHS systems with $l = 2$ was presented. The algorithm works in a dual code to the code generated by MRHS system left-hand side matrix $\mathbf{M}$. Here, we generalise and expand this algorithm, and study its complexity in more details.

The main idea of the algorithm is to find solutions of an MRHS system by checking whether solutions from the set

$$S = S_1 \times \cdots \times S_m$$

are code words of the left-hand side linear code. This is done by using parity check matrix of the code. After transformation, the problem will change to a problem of finding a zero sum of vectors, each from a different set. The polynomial time algorithm does this in two phases. Firstly, it uses an algorithm from [11], [13] to transform the problem of solving MRHS system into a special decoding problem (Algorithm 1). Then it tries to solve the decoding problem by looking for special vectors in the echelonized matrix (Algorithm 2).

Algorithm 1 Preparation phase of solving MRHS system in a dual code

Require: MRHS system $x\mathbf{M} \in S_1 \times S_2 \times \cdots \times S_m$.

Ensure: Vector u and block matrix $\mathbf{U}$, such that any solution c of $\mathbf{U}c^T = u^T$ with limited weight in each block can be one-to-one mapped to a solution x.

1: Compute parity check matrix: find $\mathbf{H}$ such that $\mathbf{MH}^T = \mathbf{0}$.

2: Construct block diagonal matrix

$$\mathbf{S} = \begin{pmatrix} \mathbf{S}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{S}_m \end{pmatrix}$$

3: Compute block matrix $\mathbf{V} = \mathbf{SH}^T$.

4: Compute block matrix $\binom{u}{\mathbf{U}} = \mathbf{TV}$, where $\mathbf{T}$ is a transformation matrix in a block form:

$$\mathbf{T} = \begin{pmatrix} \mathbf{T}_1 & \mathbf{T}_1 & \cdots & \mathbf{T}_1 \\ \mathbf{T}_2 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{T}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{T}_2 \end{pmatrix}.$$

Matrix $\mathbf{T}_1$ is a $1 \times k$ matrix $(1, 0, 0, \ldots, 0)$, and matrix $\mathbf{T}_2$ is a $(k-1) \times k$ matrix:

$$\mathbf{T}_2 = \begin{pmatrix} 1 & 1 & 0 & \cdots & 0 \\ 1 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \\ 1 & 0 & 0 & \cdots & 1 \end{pmatrix}.$$

Matrix $\mathbf{V}$ in Algorithm 1 is a block matrix. In the following, $v_{i,j}$ (and similarly $u_{i,j}$) denotes jth row (vector) in the ith block of the corresponding block matrix. Each block has k rows of dimension $r = ml - n$ (the number of parity bits). Thus $v_{i,j}$ is $(ik + j)$th row of matrix $\mathbf{V}$.

Suppose that we can find exactly one row in each block of $\mathbf{V}$ in such a way, that their sum is 0. Then we can identify a correct codeword c in S: if $v_{i,j}$ is a part of the zero-sum combination, than jth vector in S_i is the corresponding projection of c we were looking for.

We can change the problem slightly by computing block matrix $\mathbf{U}$ and vector u by Algorithm 1. Vector u is a sum of all first vectors from each block of $\mathbf{V}$, i.e., $u = \sum_{i=1}^{m} v_{i,1}$. The rows of the matrix $\mathbf{U}$ are given as $u_{i,j} = v_{i,1} + v_{i,j+1}$. By adding $u_{i,j}$ to u we replace $v_{i,1}$ in the sum by $v_{i,j+1}$. Thus, our goal is to remove all non-zero elements of u by adding at most one vector from each of the blocks of $\mathbf{U}$. If we can do this, we can identify the correct zero-sum of the rows of $\mathbf{V}$, compute the correct right-hand side vector s, and finally, solve the linear system $x\mathbf{M} = s$ to obtain a solution of the original MRHS system.

All the steps so far have polynomial complexity in n (for polynomially sized MRHS system), so the only difficult part is finding a proper sum of rows of $\mathbf{U}$ that cancel out the non-zero elements of u. However, as pointed in [13], this is a special version of the syndrome decoding problem, which is NP-hard in general. In Algorithm 2 we try to obtain a solution by linear algebra. We can do equivalent column operations (column swaps, column additions) without changing the problem. We will try to bring some rows to a reduced form with a single non-zero element. We will call the row reduced, and the non-zero element a pivot. After we prepare a reduced form of the whole matrix $\mathbf{U}$, a restricted search can identify solutions in some cases.

Note that we exclude the case when u is zero. In such a case, solution is trivial. However, the chance of getting $u = 0$ for a random system is negligible. Furthermore, we suppose that column rank of the matrix $\binom{u}{\mathbf{U}}$ is maximal. If this is not true, we can remove some unnecessary columns, and work with a smaller version of the problem.

Algorithm 2 changes syndrome u to $(1, 0, \ldots, 0)$, and with linear algebra reduces some rows of $\mathbf{U}$. Note that under condition of maximal rank of $\binom{u}{\mathbf{U}}$, there is a one-to-one mapping between reduced rows and columns of $\mathbf{U}$ after the first one, given by positions of reduced rows' pivots.

The algorithm tries to combine up to t remaining non-reduced rows from different blocks in such a way that the sum contains a leading one (to cancel up the one in u). Finally, it tries to cancel the remaining non-zero elements by adding reduced rows. The algorithm can succeed only if each reduced row is taken from a different block (we can only add one vector to u from each block), and if no additional reduced row must be taken from the originally chosen t blocks.

Algorithm 2 Restricted syndrome decoding phase of solving MRHS system in a dual code

Require: Vector $u \neq 0$ and matrix $\mathbf{U}$ of dimension $m(k-1) \times r$.
Ensure: Set of indexes $R = \{(i,j)\}$, with no i duplicated, and $u = \sum_R u_{i,j}$.
1: With column operations on $\left(\begin{smallmatrix} u \\ \mathbf{U} \end{smallmatrix}\right)$ change u to reduced form $(1,0,0,\ldots,0)$.
2: **for** $a = 2,\ldots,r$ **do**
3: If all rows of $\mathbf{U}$ are reduced, return "NO SOLUTION".
4: Find vector $u_{i,j}$ with a non-zero element in position $b \geq a$. If there are more options, choose from the block with minimum number of reduced rows randomly.
5: Use the following column operations: Add bth column to each column of $\mathbf{U}$ which contains 1 in row $u_{i,j}$. Note that all previous reduced rows are unchanged, because bth column has value 0 at these rows (because $b \geq a$). Finally, swap bth column with ath column. This will change row $u_{i,j}$ to $(0,\ldots,0,1,0,\ldots,0)$, where the only 1 is in position a.
6: **end for**
7: Construct set C of candidate indexes (i,j) of rows, which were not reduced in previous steps.
8: **for all** t-tuples D of vectors from C with different i's **do**
9: Verify that $v = \sum_D u_{i,j}$ has non-zero element at position 1.
10: Let P be a set of positions of non-zero element of v after position 1.
11: Compute set E of indexes (i,j), such that $u_{i,j}$'s are reduced rows with pivots in positions given by P.
12: Let $R = D \cup E$.
13: **if** no i is duplicated in R **then**
14: **return** R
15: **end if**
16: **end for**
17: **return** "CANNOT FIND SOLUTION"

Thus, the algorithm can only find some of the solutions of the system, and can fail. Moreover, it can only verify that no solution exists in a special case: when the number of rows of $\mathbf{U}$ is less than number of parities r. This however means that

$$(k-1)m < ml - n, \quad \text{or} \quad n < m\big(l-(k-1)\big).$$

Expression $m\big(l-(k-1)\big)$ is positive only if $l \geq k$. However, we have already shown in Section 4.1 that systems with $l \geq k$ can be solved by linearisation.

Let us study the case, when the number of rows of $\mathbf{U}$ is greater than number of parities r, so that C is not empty. We will denote this size by

$$N = |C| = m(k-1) - (r-1).$$

To test all t-tuples of vectors from C, the complexity can be bounded by $O(N^t)$, which stays polynomial in n for fixed t (and polynomially sized MRHS systems). For each t-tuple, in a random system, we expect that first bit is 1 with probability $1/2$. Each vector v defines a function $f : \{2, 3, \ldots, r\} \to \{0, 1, \ldots, r\}$ that maps positions j to 0 if the jth bit of v is zero, or to a, where a is the block number of the corresponding reduced row. Furthermore each v is associated with a set $T \subset \{1, \ldots, m\}$ with values corresponding to selected blocks for vectors in the t-tuple. We have two conditions: $range(f) \cap T = \emptyset$ (no reduced row in blocks selected by the choice of the t-tuple), and $|range(f) \setminus \{0\}| = w_H(v) - 1$ (at most one reduced row in each block).

Let us compute the (approximate) probability that the Algorithm 2 succeeds. For simplicity, we will understand f as a result of a series of independent random processes: for each input, randomly choose one of $r - 1$ positions, and then flip a coin to select bit 0/1. The probability of randomly hitting one of the t selected positions out of $r - 1$ possible is $\frac{1}{2}\frac{t}{r-1}$. We repeat this $r - 1$ times, so the probability of missing the selected t positions is

$$p_1 = \left(1 - \frac{1}{2}\frac{t}{r-1}\right)^{r-1} \approx e^{-t/2}.$$

For each of $r - 1$ positions, we can count, how often it is selected by f. We can simplify the model to consider repeating at each place independent experiments with probability of hit $\frac{1}{2}\frac{1}{r-1}$. Probability that in $r - 1$ tries we get at most one hit at some position is

$$p_2 = \left(1 - \frac{1}{2(r-1)}\right)^{r-2},$$

and probability that this happens independently on $(r - 1 - t)$ positions is $p_3 = p_2^{(r-1-t)}$. Finally, we can combine probability of not hitting t positions and hitting remaining positions at most once as

$$p_4 = p_1 p_3 = \left(1 - \frac{t}{2(r-1)}\right)^{r-1} \left(1 - \frac{1}{2(r-1)}\right)^{(r-2)(r-1-t)}.$$

Approximately we get

$$p_4 \approx e^{-\frac{t}{2}} e^{-\frac{r-2}{2} + \frac{t(r-2)}{2(r-1)}} = e^{-\frac{r-2}{2}} e^{\frac{t}{2(r-1)}}.$$

Note, that the analysis is intentionally simplified, because we are only interested in the general asymptotic behaviour, and not precise probabilities. Having obtained estimate p_4 for success per one vector v, we can now try to find N such that expected number of successes is greater than 1. That is

$$\frac{N^t}{2} e^{-\frac{r-2}{2}} e^{\frac{t}{2(r-1)}} \geq 1,$$

or after taking logarithms

$$t \log N - \frac{r-2}{2} + \frac{t}{2(r-1)} - \log 2 \geq 0.$$

This gives

$$\log N \geq \frac{r-2}{2t} - \frac{1}{2(r-1)} + \log 2.$$

Part $\frac{1}{2(r-1)} - \log 2$ is asymptotically small, so it can be removed. After this simplification, and a change back to original parameters of MRHS system ($r = ml - n$ and $N = m(k-1) - (r-1) = m(k-l-1) + n + 1$) we get

$$\log\big(n + m(k-l-1) + 1\big) > \frac{ml - n - 2}{2t}.$$

For polynomially bounded systems, and large enough n, we can bound $n + m(k - l - 1) + 1$ by some n^c. Thus, we can replace the restriction to:

$$m < \frac{n}{l} + \frac{2}{l}\big(1 + ct \log(n)\big).$$

This means that the number of MRHS equations in the system can be greater than n/l by only a logarithmic factor, if we want to have a non-negligible chance to solve the system by combinations of Algorithm 1 and 2. Recall that if $m < n/l$, we can solve the MRHS system trivially, as for every vector in S we can find x by linear algebra. Parameter t can increase the range where the algorithm is applicable, but also increases its complexity, which is $O(n^t)$.

We can thus define a family of algorithms depending on the choice of t. If t is constant for some family of MRHS systems, we get a polynomial time algorithm, but cover only a negligible fraction of the polynomially sized MRHS systems. If we allow t to grow polynomially with n (essentially, we test all options), we can cover all such systems, but the algorithm requires exponential time. If we choose t to grow with $(\log n)^\varepsilon$ for some ε, we get a quasi-polynomial time algorithm (for $\varepsilon > 1$), which can succeed for systems with $m - n/l = O\left((\log n)^{1+\varepsilon}\right)$.

In this place, one important remark is required on random systems that can be solved in polynomial time. These systems can be restricted to instances where $m = n/l + c/l \log_2 n$ for some constant c. The expected number of solutions of such a system is $2^{n-ml}|S| = 2^{-c \log_2 n}|S| = |S|/n^c$. Thus, we can simply solve the system in probabilistic polynomial time by taking vectors from S and test, whether they are codewords of the code generated on the left-hand side. This however, is exactly what the combination of Algorithm 1 and 2 does, but in a more systematic way. On the other hand, for some non-random systems (e.g., systems with sparse dual code), the systematic approach may be more efficient in practice (especially for sparse systems).

5. Conclusions

In this article we have studied selected easy instances of MRHS problem. We have shown formally that MRHS solution problem is NP-hard, and NP-complete if we restrict the problem to a family of polynomially bounded MRHS systems. This means that so far we do not know any polynomial time algorithms that can solve the MRHS problem in general. However, we have shown there are large (infinite) families of MRHS systems that can be solved by linearisation, or by restricted syndrome decoding.

In the linearisation case, algorithm can find solutions and decide that the system is not solvable, for systems with enough MRHS equations with small number of vectors in right-hand side sets ($\sum(dim(S_i) - |S_i| + 1)$ must be at least $n - c\log(n)$). In this case, a random system has typically a small number of solutions or none at all. On the other hand, the restricted syndrome decoding can be used to find a solution for systems with small number of MRHS equations $m = n/l + O(\log(n))$. However, random systems of this type have typically a large number of solutions. The algorithms can be parametrised in such a way, that we can make a trade-off between complexity and success probability. In general, polynomial time versions of the algorithms have a negligible success probability, while exponential time version of the algorithms can solve all types of systems.

It remains an open question, whether we can efficiently use some trap-door information to transform a seemingly hard instance of the MRHS problem to an easy one. E.g., in a linearisation case, we can efficiently mark the "correct" vectors in each (large enough) S_i, and then solve the system by linear algebra. If the marking stays hidden, the system cannot be solved, as we do not know which vectors to remove. The "marking" however should be more efficient than just storing solution vector x, and the "marked" system should be indistinguishable from a random system.

REFERENCES

[1] BERMAN, P.—KARPINSKI, M.—SCOTT, A. D.: *Computational complexity of some restricted instances of 3-SAT*, Discrete Appl. Math. **155** (2007), 649–653.

[2] DING, J.—YANG, B.-Y.: *Multivariate public key cryptography*, in: Post-Quantum Cryptography (D. J. Bernstein et al., eds.), Springer-Verlag, Berlin, 2009, pp. 193–241.

[3] RADDUM, H.—SEMAEV, I.: *Solving multiple right hand sides linear equations*, Des. Codes Cryptogr. **49** (2008), 147–160.

[4] REPKA, M.—ZAJAC, P.: *Overview of the Mceliece cryptosystem and its security*, Tatra Mt. Math. Publ. **60** (2014), 57–83.

[5] SCHILLING, T. E.—RADDUM, H.: *Analysis of Trivium using compressed right hand side equations*, in: Information Security and Cryptology—ICISC '11 (H. Kim, ed.), 14th Internat. Conf., Seoul, Korea, 2011, Lecture Notes in Comput. Sci., Vol. 7259, Springer-Verlag, Berlin, 2012, pp. 18–32.

[6] SEMAEV, I.: *On solving sparse algebraic equations over finite fields*, Technical report, Department of Informatics, University of Bergen, 2005.
[7] SEMAEV, I.: *Improved agreeing-gluing algorithm*, Math. Comput. Sci. **7** (2013), 321–339.
[8] TOVEY, C. A.: *A simplified NP-complete satisfiability problem*, Discrete Appl. Math. **8** (1984), 85–89.
[9] ZAJAC, P.: *On the use of the method of syllogisms in algebraic cryptanalysis*, in: Proc. of the 1st Plenary Conf. of the NIL-I-004, University of Bergen, Norway, 2009, pp. 21–30.
[10] ZAJAC, P.: *Solving trivium-based Boolean equations using the method of syllogisms*, Fund. Inform. **114** (2012), 359–373.
[11] ZAJAC, P.: *A new method to solve MRHS equation systems and its connection to group factorization*, J. Math. Cryptol. **7** (2013), 367–381.
[12] ZAJAC, P.: *Some notes on MRHS equations over $GF(2)$ with two left-hand sides*, in: Norwegain-Slovakian Workshop in Crypto, Bergen, Norway, 2016, Slovak University of Technology, Bratislava, 2016, pp. 65–70.
[13] ZAJAC, P.: *Upper bounds on the complexity of algebraic cryptanalysis of ciphers with a low multiplicative complexity*, Des. Codes Cryptogr. 2016, 1–14.
[14] ZHAO, Y.—DENG, X.—LEE, C. H.—ZHU, H.: *$(2 + f(n))$-SAT and its properties*, Discrete Appl. Math. **136** (2004), 3–11.

Received August 17, 2016

Institute of Computer Science and Mathematics
Faculty of Electrical Engineering and Information Technology
Slovak University of Technology in Bratislava
Ilkovičova 3
SK–812-19 Bratislava
SLOVAKIA
E-mail: pavol.zajac@stuba.sk