Parameters of Algebraic Representation vs. Efficiency of Algebraic Cryptanalysis

Hossein Arabnezhad-Khanoki*

Babak Sadeghiyan

Abstract

The aim of an algebraic attack is to find the secret key by solving a collection of relations that describe the internal structure of a cipher for observations of plaintext/cipher-text pairs. Although algebraic attacks are addressed for cryptanalysis of block and stream ciphers, there is a limited understanding of the impact of algebraic representation of the cipher on the efficiency of solving the resulting collection of equations. In this paper, we investigate on how different S-box representations affect the complexity of algebraic attacks, in an empirical manner. In the literature some algebraic properties are intuitively proposed to evaluate optimality of an algebraic description of S-boxes for algebraic cryptanalysis. In this paper, we compare different S-box representation for algebraic cryptanalysis with doing experiments with SR family of block ciphers. We also show that the so-called *Forward-Backward* representation which is in contrast with all mentioned criteria for optimal representations criteria, practically gives better results than the compliant representations. We also compare the representations for both $GF(2)$ and $GF(2^n)$ fields.

1 Introduction

The Rijndael block cipher was chosen as the winner of the AES competition and was later approved by NIST as the Advanced Encryption Standard (AES) [10]. The simple and elegant algebraic structure of AES provides a strong motivation for the development of new cryptanalysis techniques.

These techniques involve describing a block cipher as a system of relations and then solving the system of equations. Courtois and Pieprzyk proposed one such approach using an extended sparse linearization (XSL) algorithm to solve the system of relations [9]. The AES block cipher is represented by a system of polynomial equations over the field $GF(2)$, and the XSL algorithm is used to recover the secret key. It was estimated that this attack would be slightly faster than an exhaustive search.

The XSL attack turns out to be ineffective against AES block cipher [4, 5]. The system of relations that describe the AES block cipher with 128-bit keys

*Department of Computer Engineering & Information Technology, Amirkabir University of Technology, Tehran, Iran
Email: {arabnezhad,basadegh}@aut.ac.ir

consists of 8000 nonlinear relations in 1600 variables, making it impossible to experimentally verify the claimed complexity of the XSL attack. Cid et al. [3] proposed a family of scalable versions of AES to study and experiment with cryptanalysis of AES.

In general, algebraic attacks progress in two following steps:

1. finding a system of relations that describe the block cipher, where an adversary can observe plaintext and ciphertext pairs. The unknowns are bits/bytes of secret key,

2. solving the obtained system of relations using appropriate algorithm (XL, F4, SAT solvers, ...).

There are virtually an infinite number of ways to algebraically describe a block cipher. As a result, the adversary (cryptanalyzer) would like to form these algebraic relations in such a way that they can be solved as fast as possible.

The effectiveness of cryptographic attacks against cryptographic primitives is measured by the attack complexity, which takes into account both the time and space complexity of running the attack. In algebraic attack, since the cipher is described by a collection of polynomial equations, the complexity of attack is measured by complexity of solving such a system of equations. In general solving a system of polynomial equations of degree 2 or more with coefficients in a finite field is an NP-hard computational problem.

In general, the computational complexity of Gröbner basis would be of order

$$O(\binom{n}{D_{reg}}^{\omega})$$

where ω is the matrix multiplication exponent ($\omega = 2.373$) and D_{reg} is the degree of regularity of the system which is the degree in which the system of equations in Gröbner basis computation will be solved . Determining D_{reg} is not straightforward and asymptotic estimations only reported for the so-called *semi-regular* systems. In [12] it has been shown that the system of equations arising from block ciphers are not semi-regular. Therefore currently it is not possible to compare efficiency of different representation of S-boxes theoretically.

In [1] different approaches for S-box representation were evaluated for their effectiveness in algebraic cryptanalysis. Surprisingly, the results showed that utilizing a higher degree representation of S-boxes, referred to as the FWBW representation, might result in a more efficient algebraic attack against block ciphers, compared to using a degree 2 representation.

There are many S-box representation that have been proposed in the literature for algebraic cryptanalysis. In this paper, we compare different S-box representation that have been proposed in the literature in properties that intuitively might lead to efficient algebraic cryptanalysis. We also consider S-box representation in $GF(2^n)$ field.

Additionally, our study extends the research done in [1] by considering the impact of different S-box representations on the complexity of algebraic attacks on larger S-boxes. We consider S-box representations for 5 and 6 bit S-boxes which might provides valuable insight into how representations can affect the security of block ciphers that use larger S-boxes.

The paper is organized as follows. In Section 2, we discuss different methods for achieving algebraic representation of S-boxes . Then, in Section 3, we report

different properties that discussed in the literature about algebraic representation of S-boxes for the aim of algebraic cryptanalysis. The empirical results with the SR block cipher are also reported in this section. In Section 4, we study representations in $GF(2^n)$. Section 5 provides an insight into effect of S-box representation for larger S-boxes. We give conclusions and future research directions in Section 6.

2 S-box description

S-boxes play a crucial role in many block ciphers, as they provide the non-linearity to transform the input (plaintext) into the output (ciphertext). Typically, S-boxes are implemented as lookup tables that efficiently implement a vector of Boolean functions, and are defined as mapping from $GF(2)^n$ to $GF(2)^m$. If we define x_i for $1 \leq i \leq n$ as the variables that denote input bits and y_j for $1 \leq j \leq m$, as the variables that denote output bits , an S-box can be represented as a vectorial Boolean function, using the following collection of polynomials:

$$\begin{aligned} f_1(x_1, \ldots, x_n) + y_1 \\ \vdots \\ f_m(x_1, \ldots, x_n) + y_m \end{aligned} \tag{1}$$

In the polynomial ring $GF(2)[x_1, \ldots, x_n, y_1, \ldots, y_n]$, there exists a polynomial ideal, denoted as $I_{\mathcal{S}}$, which contains all possible polynomials that define a relation between the input and output of the S-box [1]. This ideal can be expressed as:

$$I_{\mathcal{S}} = \langle f_1, \ldots, f_n \rangle + Q \tag{2}$$

where Q is the ideal of field polynomials. Any generating set for the ideal $I_{\mathcal{S}}$ would be a set that define the S-box, uniquely. As a result, there exist virtually infinite ways to describe an S-box.

The literature presents several methods for generating a set of polynomials that describe the S-box [8, 7, 1, 2, 9]. These methods can be grouped into three main categories:

1. Methods based on the hardware implementation of the S-box
2. Methods based on linear algebra
3. Methods based on algebraic properties

We will discuss each of these categories in the following.

2.1 Hardware Implementation

As the S-box might finally implemented in hardware, one way to describe it algebraically is through its Boolean circuit realization. For example the Boolean polynomials in (3) for the S-box of $SR(n, 2, 1, 4)$, represent the Boolean realiza-

tion of the S-box which are used during the encryption process.

$$\begin{cases} f_0 : y_0 + & x_0x_1x_2 + x_0x_1x_3 + x_0x_2x_3 + x_0x_2 + x_0x_3 + x_0 + \\ & x_1x_2x_3 + x_1x_3 + x_1 + x_3 \\ f_1 : y_1 + & x_0x_1x_3 + x_0x_2x_3 + x_1x_2x_3 + x_1x_2 + x_1 + x_2x_3 + x_3 + 1 \\ f_2 : y_2 + & x_0x_1x_2 + x_0x_1 + x_0x_2x_3 + x_0 + x_1x_2 + x_1x_3 + x_2x_3 + x_2 + x_3 + 1 \\ f_3 : y_3 + & x_0x_1x_2 + x_0x_1x_3 + x_0x_1 + x_0x_3 + x_0 + x_2x_3 + x_3 \end{cases} \tag{3}$$

The backward equations for the S-box are showed in (4). In these equations inputs bits to the S-box are described by the output bits of the S-box. These polynomials are used during the decryption process.

$$\begin{cases} w_0 : x_0 + y_0y_1 + y_0y_2y_3 + y_0 + y_1y_2y_3 + y_1y_2 + y_2y_3 + y_2 + y_3 \\ w_1 : x_1 + y_0y_1y_3 + y_0y_1 + y_0y_2y_3 + y_0y_2 + y_0 + y_1y_2y_3 + y_1y_3 + y_1 + 1 \\ w_2 : x_2 + y_0y_1y_2 + y_0y_2y_3 + y_0y_3 + y_1y_2y_3 + y_2 + 1 \\ w_3 : x_3 + y_0y_1y_2 + y_0y_1y_3 + y_0y_1 + y_1y_2y_3 + y_1y_2 + y_1 + y_2y_3 + y_2 + 1 \end{cases} \tag{4}$$

In [1], the first set of equations are referred to as forward (FW), and the second as backward (BW). The FWBW representation in [1], is created by combining the FW and BW representations.

The number of gates required for implementing an S-box is a crucial factor in hardware implementation. In [8], an optimal hardware implementation of the DES block cipher S-box was used to perform an algebraic attack on the cipher with the help of a SAT-solver.

One of important aspects in hardware implementation of an S-box is the number of required gates for S-box realization. In [8], an optimal hardware implementation of the DES block cipher S-box is used to perform an algebraic attack on the cipher with a SAT-solver. In [7], a SAT-solver based approach is proposed to achieve optimal implementation of S-boxes. To generate a set of polynomials for S-box circuit, the approach consider different criteria, including:

1. Multiplicative Complexity (MC): This is the minimum number of AND gates required to implement the S-box circuit, without considering the number of required XOR gates.

2. Gate Complexity: This is the minimum number of two input gates of any type: XOR, AND, OR, NAND, NOR, XNOR.

3. Bit-Sliced Gate Complexity: The minimum number of two input gates such as XOR, AND, OR, NAND, NOR, XNOR for bit-sliced implementation with SIMD instructions of the CPU.

4. NAND Complexity: The minimum number of two inputs NAND gate.

Taking into account that in algebraic cryptanalysis with computation of Gröbner basis, the cipher is described as system of polynomial equations in *Algebraic Normal Form* (ANF) in a polynomial ring, the only relevant criteria would be multiplicative complexity. We refer to this kind of description as Minimal Multiplicative Complexity (MMC) representation. Considering the $SR(n, 2, 1, 4)$, we derived following system of equations for S-box $SR(n, 2, 1, 4)$ using the method outlined in [7]. The multiplicative complexity, i.e. the minimum of number of

Boolean multiplications (AND gates) of the resulting system for this S-box is 5.

$$\begin{cases} w_0 = x_0 + x_1 + x_2 + x_3 \\ w_1 = x_0 + x_1 \\ q_0 = w_0 w_1 \\ w_2 = x_1 + q_0 \\ w_3 = x_0 + x_3 \\ q_1 = w_2 w_3 \\ w_4 = x_0 + x_2 + x_3 + q_0 + q_1 \\ w_5 = x_1 + x_2 + x_3 + q_0 \\ q_2 = w_4 w_5 \\ w_6 = x_2 + q_0 + q_1 + q_2 \\ w_7 = x_0 + x_3 + q_0 + q_2 \\ q_3 = w_6 w_7 \\ w_8 = x_0 + x_2 + q_0 + q_1 + q_2 \\ w_9 = x_0 + x_1 + x_3 \\ q_4 = w_8 w_9 \\ y_0 = x_1 + x_3 + q_1 \\ y_1 = x_0 + x_1 + q_3 + q_4 + 1 \\ y_2 = x_0 + x_1 + x_3 + q_0 + q_1 + q_2 + q_3 + q_4 + 1 \\ y_3 = x_3 + q_0 + q_1 + q_3 \end{cases} \tag{5}$$

2.2 Linear Algebra

Another approach to find algebraic relations between outputs and inputs of an S-box is to utilize linear algebra. In [2] the algorithm is described with an example. Consider following 3-bit S-box:

$$[7, 6, 0, 4, 2, 5, 1, 3]$$

A matrix with a row for each selected term is created to find linear-independent polynomials for the given S-box. In this example, terms x_i, y_i, and $x_i y_j$ with $i = 0, 1, 2$ are selected. Each row in the matrix contains 2^n elements representing the corresponding input values for an n-bit S-box. Gaussian Elimination is then applied to the matrix, and corresponding row operations are performed on the terms. At the end of the process, some zero rows may appear in the matrix, and the corresponding relations are the desired equations.

For the mentioned example, the left matrix shows the value of each term for all inputs. The right matrix yields from Gaussian Elimination and the

corresponding operations on the corresponding terms.

$$\left[\begin{array}{c|cccccccc}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
x_0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
x_1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
x_2 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
y_0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
y_1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\
y_2 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\
x_0y_0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
x_0y_1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
x_0y_2 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\
x_1y_0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
x_1y_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
x_1y_2 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
x_2y_0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
x_2y_1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\
x_2y_2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0
\end{array}\right]
\rightarrow
\left[\begin{array}{l|cccccccc}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
x_0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
x_1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
1+x_0+x_1+y_0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\
x_2 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
1+x_1+y_1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
1+x_0+x_1+y_0+y_1+y_2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
x_0+x_0y_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\hline
x_2+y_0+y_1+x_0y_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1+x_1+y_1+x_0y_0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1+x_0+x_1+y_0+y_1+y_2+x_1y_0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
x_0+x_0y_2+x_1y_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1+x_1+x_2+y_0+x_0y_2+x_1y_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
y_0+y_2+x_0y_2+x_2y_0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
x_0+x_2+y_0+y_2+x_2y_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1+x_0+x_1+y_1+x_0y_2+x_2y_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}\right] \tag{6}$$

The expressions in the right matrix of (6) are valid relations for the S-box, as they are satisfied for all inputs. However, to fully describe the S-box, there needs to be enough equations. A notable example is DES S-boxes which there is not enough quadratic relations to define the S-boxes uniquely and an exact description must also consider monomials of degree three [8]. In [2], various system of equations for S-boxes of some block ciphers is reported with this method. For the S-box of $SR(n, 2, 1, 4)$ we derive equations in 7 with monomials of degree one and monomials of degree two which are of the form x_ix_j and y_iy_j . This system of equations is similar to the one proposed for S-box of SERPENT block cipher in [2].

$$\begin{cases}
h_0 : y_1y_2 + y_1y_3 + x_1x_3 + x_1 + x_2x_3 + x_2 + x_3 + 1 \\
h_1 : y_0y_3 + y_2 + x_0x_2 + x_0x_3 + x_1x_2 + x_1x_3 + x_2 + 1 \\
h_2 : y_0y_3 + y_0 + y_1y_3 + y_3 + x_0x_1 + x_1 + x_3 \\
h_3 : y_0y_1 + y_2y_3 + y_3 + x_0x_3 + x_1x_2 + x_3 \\
h_4 : y_0y_1 + y_0 + y_1y_3 + y_1 + x_0x_3 + x_0 + x_1x_3 + 1 \\
h_5 : y_0y_1 + y_0y_2 + y_1 + y_2y_3 + y_3 + x_0x_1 + x_2x_3 + 1
\end{cases} \tag{7}$$

2.3 Algebraic Properties

For the so-called inverse S-boxes such as AES S-box, it is possible to derive the set of equations based on their algebraic structure, as described in [9]. These kind of S-boxes are composed of G(2)-linear transformations and patched version of inverse function in $G(2^n)$:

$$y \leftarrow \begin{cases} 0 & x = 0 \\ x^{-1} & x \neq 0 \end{cases} \tag{8}$$

In case of $x \neq 0$, the relation $xy = 1$ gives 8 bi-linear quadratic equations, considering valid relations $x^2y = x$ and $xy^2 = y$ we could derive 16 more equations. This method allows for the description of the AES S-box with 24 relations, with one of them being valid for a probability of 255/256. The same method can

also be applied to the S-box of $SR(n, 2, 1, 4)$ to derive 21 quadratic equations. The main property of these equations is that they are overdefined, which will be explained later. For instance, the S-box of ciphers from $SR(n, r, c, 4)$ can be described using an overdefined system of 21 polynomials of degree 2 with 36 monomials.

$$\begin{cases} x_2x_3 + y_1y_2 + y_0x_1 + y_0x_0 + y_2 + y_1 + y_0 + 1 \\ x_1x_3 + y_0x_2 + y_0x_1 + y_0y_3 + y_0y_2 + x_0 + y_3 + y_1 + y_0 + 1 \\ x_1x_2 + y_1y_2 + y_0x_2 + y_0x_0 + y_0y_1 + x_1 + y_2 \\ x_0x_3 + y_0x_1 + y_0x_0 + y_0y_1 + x_3 + x_2 + x_1 + x_0 + y_2 + 1 \\ x_0x_2 + y_1y_2 + y_0y_2 + x_3 + y_3 + y_2 + y_1 + y_0 + 1 \\ x_0x_1 + y_0x_2 + y_0x_0 + y_0y_2 + x_2 + x_0 + y_2 + y_0 + 1 \\ y_3x_3 + y_0x_1 + y_0y_3 + y_0y_1 + x_3 + x_2 + x_1 + x_0 + y_2 + 1 \\ y_3x_2 + y_1y_2 + y_0x_2 + y_0x_1 + y_0x_0 + y_0y_3 + y_0y_2 + x_0 + y_2 + y_0 \\ y_3x_1 + y_0x_1 + y_0x_0 + y_0y_3 + y_0y_1 + x_2 + y_3 + y_2 + y_0 + 1 \\ y_3x_0 + y_0x_2 + y_0x_1 + x_3 + x_0 + y_1 + 1 \\ y_2x_3 + y_1y_2 + y_0x_1 + y_0y_2 + y_0y_1 + x_0 + y_3 + y_2 + y_0 \\ y_2x_2 + y_0x_2 + y_0x_0 + y_0y_3 + y_0y_1 + x_2 + y_2 + 1 \\ y_2x_1 + y_1y_2 + y_0x_2 + y_0y_3 + y_0y_2 + y_2 + y_0 \\ y_2x_0 + y_1y_2 + y_0y_3 + y_0y_2 + x_2 + 1 \\ y_2y_3 + y_1y_2 + y_0x_2 + y_0x_1 + y_0y_1 + x_2 + x_0 + y_3 + 1 \\ y_1x_3 + y_0x_2 + y_0x_0 + x_3 + x_0 + y_3 + y_1 + y_0 + 1 \\ y_1x_2 + y_1y_2 + y_0x_0 + y_0y_2 + y_0y_1 + x_3 + x_2 + x_0 + y_3 + y_2 \\ y_1x_1 + y_0x_2 + y_0y_3 + y_0y_1 + x_1 + y_1 + 1 \\ y_1x_0 + y_0x_1 + y_0y_3 + y_0y_1 + x_3 + x_2 + x_1 + x_0 + y_3 + y_2 + 1 \\ y_1y_3 + y_0x_2 + y_0x_0 + y_0y_3 + y_0y_2 + x_3 + x_2 + x_1 + x_0 + y_3 + y_2 + 1 \\ y_0x_3 + y_0x_0 + y_0y_3 + y_0y_1 + x_3 + x_0 + y_3 + y_1 + y_0 + 1 \end{cases} \tag{9}$$

It should be noted that the system of polynomials used to describe the S-box of ciphers from $SR(n, r, c, 4)$ is derived using the technique discussed in the previous section.

3 Properties of S-box Description

In the field of algebraic cryptanalysis of block ciphers, various methods for describing S-boxes with different properties have been proposed. For instance, the MQ description of S-boxes,which was introduced in [9] for XSL attack, is an overdefined set of equations. But, [8] used an S-box description with low gate or multiplicative complexity property to attack the DES block cipher.

We enumerated following properties for various S-box descriptions that discussed in the literature.

- degree of polynomials.
- overdefinedness [6, 9].
- number of free terms [2].
- the ratio number of terms to number equations [2].
- multiplicative complexity [8].

Generally one of main property that is being discussed for algebraic description of S-boxes is to have a system of equations with minimum degree. If the degree of the system is low, then the number of possible monomials will also be reduced, potentially we might expect a simpler complexity in algorithms such as XL.

In [6] it is shown that if the system is overdefined the system might be solved with less complexity. Suppose that an s-bit S-box could be described with r linearly independent polynomials in t different monomials with maximum degree d. Then system is considered to be:

- sparse if $t \ll \binom{s}{d}$.
- overdefined if $r \gg s$.

An example of an overdefined system of equations for the S-box of $SR(n, r, c, 4)$ ciphers is given by the system defined in Equation (9). We refer to this type of system of equations as *Overdefined Multivariate Quadratic* (OD-MQ) system.

Another property that discussed for description of S-box which might be effective in algebraic cryptanalysis, is the number of free terms [2]. This refers to the number of linearly independent terms if the system is treated as a linear system in monomials. In order to achieve a minimal number of free terms in an S-box description, the difference between the number of terms and the number of equations, should be minimized. The description in (7) gives such a system of polynomials for S-box of $SR(n, r, c, 4)$ ciphers. In this paper, we call such polynomial systems *Sparse Multivariate Quadratic* (S-MQ) system.

An important property of a hardware implementation of an S-box is the number of gates that required to implement the logic. In [8] an efficient hardware description for DES is used to attack the cipher. In [7], a method is proposed to achieve an optimal hardware description for S-boxes using Boolean Satisfiability to prove its optimality. The system in (5) represents such a system for S-box of $SR(n, r, c, 4)$ ciphers, which we refer to as the *Minimal Multiplicative Complexity* (MMC) system of equations

To study the effect of aforementioned properties of S-box description in algebraic cryptanalysis of block ciphers based on Gröbner basis, we consider following six different descriptions: OD-MQ, S-MQ, MMC, FW, FWBW and OD-FWBW. The last system, combines FWBW and S-MQ systems to result in an overdefined system.

In Tables 1, 3 and 2, a comparison between the arising system of equations with different descriptions for $SR(n, 2, 1, 4)$, is reported. The properties considered in the comparison include the degree of the polynomials, the number of polynomials, the number of free terms, the ratio of the number of monomials to the number of polynomials, and the number of variables. The parameter $\Gamma = ((t-r)/s)^{\lceil (t-r)/s \rceil}$ is also calculated, which is used to determine the complexity of the XSL attack in [9]. In Table 1 OD-MQ, MMC and S-MQ are of degree 2, while others have a degree of 3. OD-MQ have the highest number of equations and monomials, with 21 equations and 36 monomials, while the FW have the least number of equations and monomials, with 4 equations and 14 monomials respectively. Without considering the system of equations for the whole block cipher, MMC description have the minimum number free terms i.e. 9, while FWBW have the most number of free terms, i.e. 20. From XSL attack point of view (Γ parameter), MMC should have minimum complexity ($\Gamma = 2^{3.97}$)

Table 1: Comparison of different representation of S-box

desc	deg	#eqs	#mon	#free	Γ
OD-MQ	2	21	36	15	$2^{8.0}$
MMC	2	19	28	9	$2^{3.97}$
FW	3	4	14	10	$2^{4.38}$
FWBW	3	8	28	20	$2^{14.35}$
OD-FWBW	3	14	28	14	$2^{7.63}$
S-MQ	2	6	20	14	$2^{7.63}$

while FWBW gives ($\Gamma = 2^{14.35}$) which cause the maximum complexity for such attack among the descriptions. In Table 2, we provide the number of variables,

Table 2: Comparison of the number of variables and equations $SR(n, 2, 1, 4)$

N_r	#var	OD-MQ		FW		FWBW		OD-FWBW		S-MQ		MMC		
		#eqs	#mon	#eqs	#mon	#eqs	#mon	#eqs	#mon	#eqs	#mon	#var	#eqs	#mon
5	88	428	1093	88	1654	168	1694	288	1709	128	517	388	388	489
6	104	512	1335	104	2057	200	2097	344	2115	152	631	464	464	585
7	120	596	1577	120	2459	232	2499	400	2521	176	745	540	540	681
8	136	680	1819	136	2861	264	2901	456	2927	200	859	616	616	777
9	152	764	2061	152	3263	296	3303	512	3333	224	973	692	692	873
10	168	844	2303	168	3666	328	3706	568	3739	248	1087	768	768	969

monomials, and equations for rounds 5 to 10 of $SR(n, 2, 1, 4)$. Additionally, we have added another column to indicate the number of variables in the MMC description. As seen from the table, MMC involves more variables compared to the other descriptions, but it also has the least number of free terms, which is 9. Furthermore, MMC has the minimum ratio of the number of monomials to the number of polynomials, which is 1.26. On the contrast, FWBW description has the most number of free terms and a large ratio of the number of monomials to equations.

3.1 Empirical Study

We conducted an empirical study to investigate how the aforementioned properties affect algebraic cryptanalysis using MAGMA version 2.21 and the SageMath computer algebra system version 6.7-x86_64 [13]. The experiments were performed on a desktop computer with a Core i7 4770 processor, 32 GB of RAM on a single core.

Experiments are done with $SR(n, 2, 1, 4)$ cipher. For each experiment run, we generated a collection of polynomial equations for 50 instances of the cipher with randomly chosen plaintexts and keys. The computation of Gröbner basis was done using the **degrevlex** monomial ordering, in which key variables were assigned the lowest order. Table 3 presents the average running time in seconds for computing the Gröbner basis for N_r rounds of the cipher . The numbers in parentheses indicate the number of solved instances out of 50, and the entries marked with $\perp$ indicate cases where the computation of Gröbner basis failed.

Table 3: Comparison of running time for different representations

N_r	OD-MQ	MMC	FW	FWBW	OD-FWBW	S-MQ
1	0.18	0.3	0.17	0.17	0.17	0.18
2	0.32	0.72	0.25	0.32	0.26	0.41
3	1.0	3.24	0.8	0.53	0.69	5.89
4	3.26	14.76	2.42	1.17	2.63	35.73
5	11.17	56.18	6.67	1.82	4.82	105.38
6	23.86	197.11	12.45	3.97	10.45	212.79
7	88.24	472.04	46.26	6.39	24.42	423.05
8	128.72	951.67	91.17	12.58	40.1	615.57
9	267.92	2042.18	155.12	16.07	62.42	897.29
10	327.29	$\perp$	217.08	22.81	87.73	1406.35

On average, computing the Gr"obner basis for a system of equations based on the MMC description of S-boxes for $SR(9, 2, 14)$ would take 2042.18 seconds for the 9-round version of the cipher. However, our tool would fail to solve the system of equations for the 10-round cipher. In contrast, for the OD-MQ description, the average running time would be 267.92 and 327.29 seconds for the 9 and 10-round versions, respectively. For the FWBW description, such a system of equations for $SR(9, 2, 14)$ and $SR(10, 2, 1, 4)$ would be solved on average in just about 16.07 and 22.81 seconds, respectively. On the other hand, for the S-MQ description, the average running time is 897.29 and 1406.35 seconds for the 9 and 10-round versions, respectively. For FW, the average running time for computing the Gr"obner basis for $SR(9, 2, 14)$ and $SR(10, 2, 1, 4)$ is 155.12 and 217.08 seconds, respectively. Finally, with OD-FWBW, the average running time is 62.42 and 87.72 seconds for the 9 and 10-round versions, respectively.

The empirical results presented in Table 3 show a significant difference between the expected and actual running times for Gr"obner basis computation using different algebraic descriptions. Surprisingly, the MMC description of S-boxes, which has a minimal number of free terms, a minimum Γ parameter, and polynomials of maximum degree 2, performs the worst in terms of running time. In contrast, the FWBW description, which has a maximum number of free terms, maximum Γ parameter, and all polynomials of degree 3, performs the best. These results contradict what one might expect based on the properties of these descriptions presented in Tables 1 and 2. Additionally, the OD-FWBW description, which is a superset of FWBW, has a longer running time than FWBW, which is unexpected.

To give more insight of Table 3, Figures 1, 2 and 3 might show other aspects of the experiments. Figure 1 provides a visualization of the average running time in logarithmic scale as a function of the number of rounds. This figure demonstrates how the running time for solving the system of equations arising from different representations evolves with respect to the number of rounds. The figure reveals that the running time for the FWBW representation approximately doubles with each additional round, whereas other representations exhibit a steeper increase in running time. Figures 2 and 3 give the average running time in logarithmic scale against the number of monomials and equations, respectively. From figure 2, we can categorize these representations into three

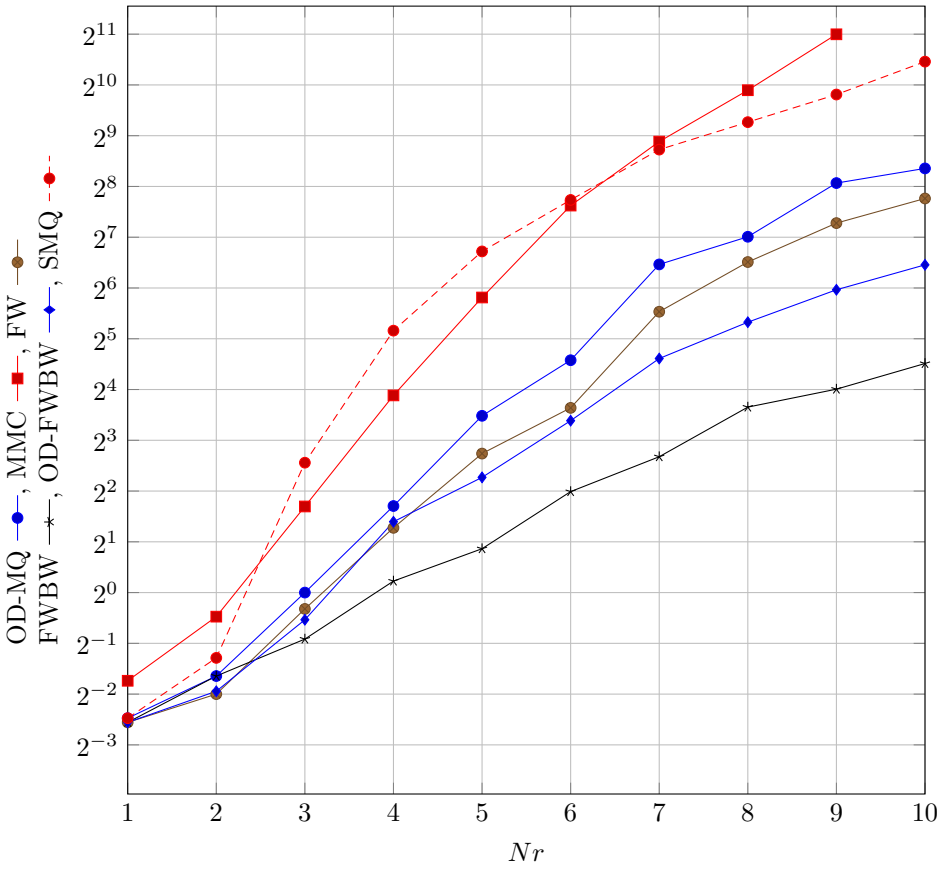

Figure 1: Comparison of running time for different representations

groups. MMC and SMQ representations have the fewest number of monomials and show similar average running times in logarithmic scale. FWBW, FW, and OD-FWBW can be seen as another group with very similar numbers of monomials, while the OD-MQ representation falls between the two previous groups. This observation suggests that good algebraic representations might include a reasonable number of monomials, but further investigation is needed in this direction.

4 Description in $GF(2^4)$

In this section, we investigate the impact of different S-box representations in $GF(2^4)$. Representing the cipher in $GF(2^4)$ results in a system of equations with fewer equations and variables. However, a rough description can lead to polynomials of high degree. To address this, the BES method for AES description was proposed by Murphy [11]. This technique embeds the main cipher in the space of a special big Cipher, resulting in the non-linear component of the AES S-Boxes being described by only one quadratic equation, which efficiently reduces the number of equations. We experimentally examine this approach for

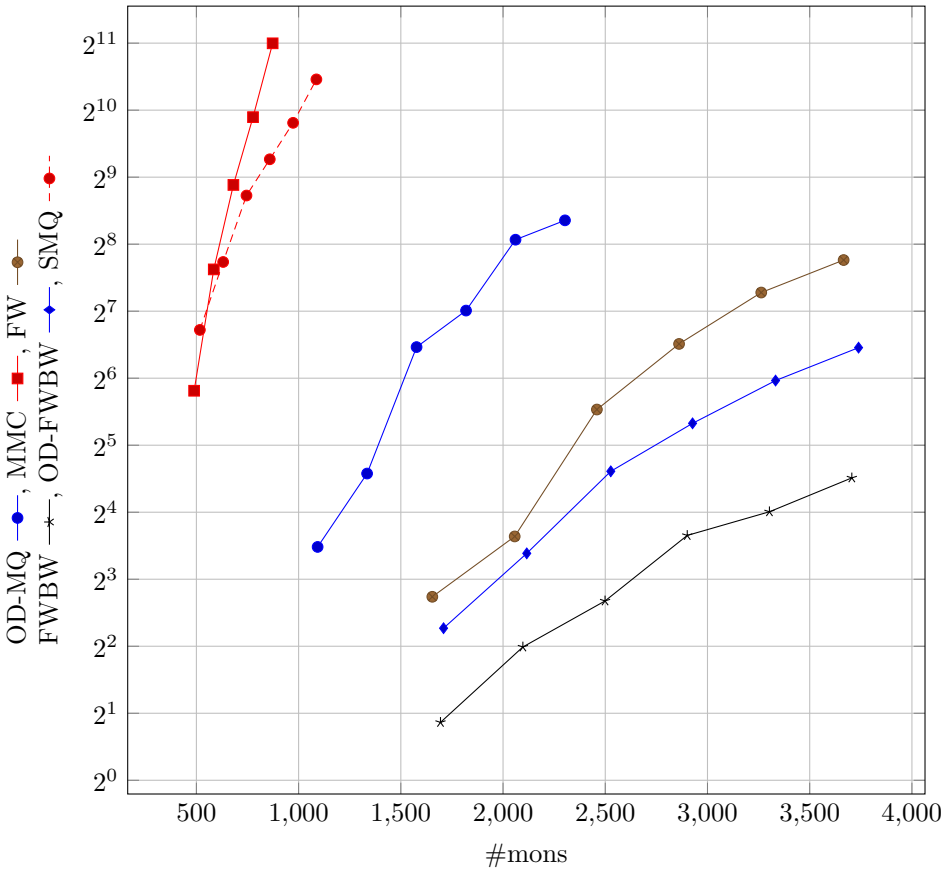

Figure 2: Comparison of running time for different representations against number of monomials

$SR(n, 2, 1, 4)$ and compare the results with previously published ones.

4.1 FWBW

One way to derive an algebraic description of S-boxes in $GF(2^n)$ is to use interpolation. In this technique, the Lagrange formula is used to interpolate a polynomial description for the S-box.

$$f(x) = \sum_{i=0}^{2^n} y_i \prod_{j=0, j\neq i}^{d} \left(\frac{x - x_j}{x_i - x_j} \right) \tag{10}$$

where x_i/ y_i are input/output of S-box as elements in $GF(2^n)$. For the S-box of $SR(n, 2, 1, 4)$ we can derive following polynomial in $GF(2^4)$, where the output of S-box is derived based on the input.

$$f : Y + 5X^{14} + X^{13} + CX^{11} + 5X^7 + 6 \tag{11}$$

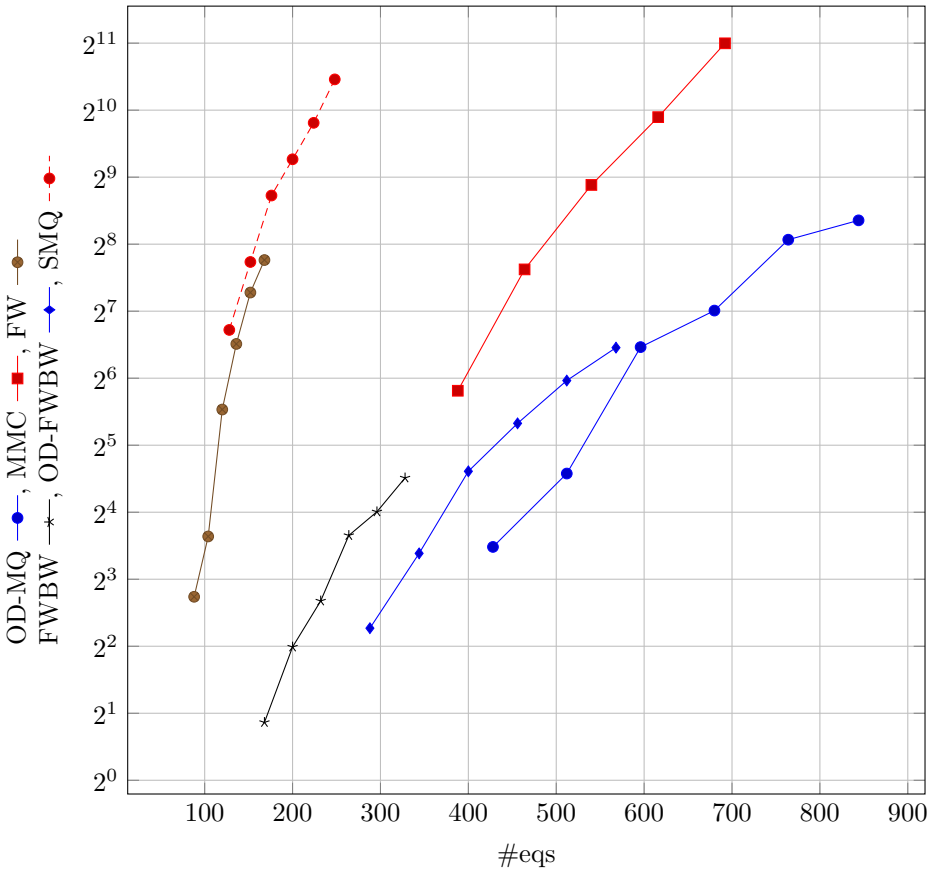

Figure 3: Comparison of running time for different representations against number of equations

Another polynomial that describes the input of S-box based on its output is as follows:

$$g : X+5Y^{14}+2Y^{13}+2Y^{12}+AY^{11}+Y^{10}+7Y^{8}+Y^{7}+Y^{6}+BY^{5}+BY^{4}+6Y^{3}+FY^{2}+3Y+E \quad (12)$$

In equations (11) and (12), the coefficients of the polynomial are expressed in hexadecimal notation and are elements in $GF(2^4)$ with irreducible polynomial $\alpha^4 + \alpha + 1$. The polynomial in equation (11) is called the Forward polynomial, while the polynomial in equation (12) is called the Backward polynomial. Another approach for the description of the block cipher in $GF(2^4)$ is to use the BES method, as proposed by [11].

4.2 BES

In [11] BES approach is introduced for AES to achieve a simpler algebraic description for the cipher in $GF(2^8)$. In this section, we will explain the BES representation for $SR(n, 2, 1, 4)$ ciphers with adoptions from [3]. In this approach, a big cipher is defined such that any intermediate state of the original

cipher could be mapped to a valid intermediate state in the big cipher. Considering that the space of the big cipher is much larger than the original one, if intermediate states have a special property, they would be mapped to the intermediate state of the original cipher. The big cipher is defined based on *vector conjugate*. To describe the cipher in $GF(2^n)$, the cipher is described as a set operation over vectors with elements in $\mathbf{F} = GF(2^n)$. For $SR(n, 2, 1, 4)$, intermediate state in each step of encryption is represented as a vector in $\mathbf{F}^2$.

Definition 1 ([11]) *for each element $a \in \mathbf{F}$, vector conjugate of a is defined as follows:*

$$\tilde{\mathbf{a}} = (a^{2^0}, a^{2^1}, a^{2^2}, a^{2^3})$$

where elements of the vector $\tilde{\mathbf{a}}$ are $GF(2)$-conjugate of a.

Therefore the vector conjugate mapping ϕ is defined as $\tilde{\mathbf{a}} = \phi(a)$. This mapping is easily extended from $\mathbf{F}^n$ to $\mathbf{F}^{4n}$. Therefore the n-dimensional vector $\mathbf{a} = (a_0, a_1, \ldots, a_{n-1}) \in \mathbf{F}^n$ is mapped to $\tilde{\mathbf{a}} = \phi(\mathbf{a}) = (\phi(a_0), \phi(a_1), \ldots, \phi(a_{n-1}))$. This mapping has two important algebraic properties: $\phi(\tilde{\mathbf{a}}) + \phi(\tilde{\mathbf{b}}) = \phi(\tilde{\mathbf{a}} + \tilde{\mathbf{b}})$ and $\phi(\mathbf{a}^{-1}) = \phi(\mathbf{a})^{-1}$. In the following intermediate state of $SR(n, 2, 1, 4)$ is denoted by $\mathbf{A} = \mathbf{F}^2$. For the equivalent BES-SR the intermediate state is denoted by $\mathbf{B} = \mathbf{F}^{4\times 2}$. Each round of $SR(n, 2, 1, 4)$ consists following operations [3]:

1. **Addition of Subkey**: In this step, the intermediate vector state is added with subkey vector of the round.
2. **S-box application**: In this operation, the S-box is applied on each element of intermediate state vector. S-box mapping consists of three steps:
   (a) inversion mapping over $GF(2^4)$.
   (b) $GF(2)$-linear map.
   (c) Constant addition.
3. **MixColumn**: In this operation, an $M_{2\times 2}$ is multiplied to the intermediate state vector.

In the following we describe the corresponding operations for BES-SR, where the definitions are adopted from [3].

Subkey Addition. Both $SR(n, 2, 1, 4)$ and BES-SR behave similarly. In $SR(n, 2, 1, 4)$ the vector $\mathbf{a} \in \mathbf{A}$ is combined with the subkey of round i with addition operation, $\mathbf{a} \leftarrow \mathbf{a} + \mathbf{k}^i_{\mathbf{A}}$. For BES-SR we have $\mathbf{a} \leftarrow \mathbf{a} + \mathbf{k}^i_{\mathbf{A}}$.

Inversion mapping of S-box. The inversion operation for S-box of $SR(n, 2, 1, 4)$ is defined as inversion of each element of intermediate state vector, $\mathbf{a} \leftarrow \mathbf{a}^{-1}$. Similarly, for BES-SR for each element $\mathbf{b} \in \mathbf{B}$ we apply $\mathbf{b} \leftarrow \mathbf{b}^{-1}$.

$GF(2)$-linear mapping. In this operation matrix L is multiplied to the bit representation of input.

$$L = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$

This mapping can be represented by a polynomial in $\mathbf{F}$,

$$f(X) = \lambda_0 X^{2^0} + \lambda_1 X^{2^1} + \lambda_2 X^{2^2} + \lambda_3 X^{2^3}$$

where $(\lambda_0, \lambda_1, \lambda_2, \lambda_3) = (5, 1, C, 5)$. The effect of this polynomial could be represented by a multiplication of a matrix in F to a vector $b \in B$. Therefore for BES-SR the equivalent operation would be

$$L' = \begin{bmatrix} \lambda_0 & \lambda_1 & \lambda_2 & \lambda_3 \\ \lambda_3^2 & \lambda_0^2 & \lambda_1^2 & \lambda_2^2 \\ \lambda_2^4 & \lambda_3^4 & \lambda_0^4 & \lambda_1^4 \\ \lambda_1^8 & \lambda_2^8 & \lambda_3^8 & \lambda_0^8 \end{bmatrix}$$

MixColumn operation for the collection of $SR(n, 2, 1, 4)$ consist of a multiplication of a matrix with elements in $\mathbf{F}$ to the intermediate state vector $\mathbf{a}$. To preserve conjugacy, the corresponding operation would be a multiplication of a matrix to a vector $b \in B$. Multiplication of an element z in F could be represented by following matrix multiplication:

$$D_z = \begin{bmatrix} z & 0 & 0 & 0 \\ 0 & z^2 & 0 & 0 \\ 0 & 0 & z^4 & 0 \\ 0 & 0 & 0 & z^8 \end{bmatrix}$$

Therefore the MixColumn operation in BES-SR cipher can be described with following matrix:

$$\left[\begin{array}{c|c} D_3 & D_2 \\ \hline D_2 & D_3 \end{array}\right]$$

The above definitions could be also extended for the key schedule part of the cipher. For each step the following relation is satisfied for intermediate state vector $\mathbf{a}$ in $SR(n, 2, 1, 4)$ and $\mathbf{b}$ in BES-SR:

$$\mathbf{b} = \phi(\mathbf{a})$$

Therefore the state space of ciphers $SR(n, 2, 1, 4)$ could be embedded in the state space of BES-SR, with ϕ mapping. If plaintext and key vectors for BES-SR satisfy conjugacy property, then we have:

$$\mathbf{a} = \phi^{-1}(\mathbf{b})$$

Taking into account that there exists algebraic description of degree 2 for BES-SR, we could derive a degree 2 description in $GF(2^4)$ for $SR(n, 2, 1, 4)$. The approach for description of BES-SR is similar to [1], but with the difference that operation here are done over vectors in $GF(2^4)$. We generate the system

of equations for BES-SR by relations in (13).

$$\begin{cases} p_{0,i}^2 + p_{0,i+1} & \text{for } i = 0, \ldots, 2 & \text{conjugacy relation} \\ p_{1,i}^2 + p_{1,i+1} & \text{for } i = 0, \ldots, 2 & \text{conjugacy relation} \\ sbox(p_0 + k_{0,0}, y_{0,0}, x_{0,0}) & & \\ sbox(p_1 + k_{0,1}, y_{0,1}, x_{0,1}) & & \\ sbox(D_3 x_{i-1,0} + D_2 x_{i-1,1} + k_{i,0}, y_{i,0}, x_{i,0}) & \text{for } i = 1, \ldots, n & \\ sbox(D_2 x_{i-1,0} + D_3 x_{i-1,1} + k_{i,1}, y_{i,1}, x_{i,1}) & \text{for } i = 1, \ldots, n & \\ c_0 + D_3(x_{n,0} + D_2 x_{n,1} + k_{n,0}) & & \\ c_1 + D_2(x_{n,0} + D_3 x_{n,1} + k_{n,1}) & & \\ c_{0,i}^2 + c_{0,i+1} & \text{for } i = 0, \ldots, 2 & \text{conjugacy relation} \\ c_{1,i}^2 + c_{1,i+1} & \text{for } i = 0, \ldots, 2 & \text{conjugacy relation} \\ sbox(k_{i,0}, k'_{i+1,0}, k_{i+1,0} + rc_i) & \text{for } i = 0, \ldots, n-1 & \\ sbox(k_{i,1}, k'_{i+1,0}, k_{i+1,1} + rc_i) & \text{for } i = 0, \ldots, n-1 & \\ k_{0,0,i}^2 + k_{0,0,i+1} & \text{for } i = 0, \ldots, 2 & \text{conjugacy relation} \\ k_{0,1,i}^2 + k_{0,1,i+1} & \text{for } i = 0, \ldots, 2 & \text{conjugacy relation} \end{cases} \tag{13}$$

In (13) all operation are done in $GF(2^4)$. In the above $sbox()$ notation is used to indicate generation of equations related to the S-box. The equations are generated as follows:

$$sbox(X, Y, X') \rightarrow \begin{cases} X_i^2 + X_{i+1} & \text{for } i = 0, \ldots, 2 & \text{conjugacy relation} \\ Y_i^2 + Y_{i+1} & \text{for } i = 0, \ldots, 2 & \text{conjugacy relation} \\ X_i Y_i + 1 & \text{for } i = 0, \ldots, 3 & \text{inversion relation} \\ X' + L'Y + C & \text{for } i = 0, \ldots, 3 & \text{linear mapping and constant addition} \end{cases} \tag{14}$$

where X is input vector of S-box and Y is the intermediate vector and X' denotes input variables to the next round. In this paper we call this kind of system of polynomials for algebraic S-box representation BES-MQ. In the following we report another possible description for S-boxes in $GF(2^n)$.

4.3 Gröbner Basis

As mentioned, the polynomials derived using interpolation normally are of high degree. Another possible approach may be using the Gröbner basis of the S-box that is derived based on a degree-based order of variables to get polynomials with lower degrees. This can be achieved by computation of Gröbner basis for following ideal:

$$I = \langle f, g, X^{16} + X, Y^{16} + Y \rangle$$

where the f and g polynomials are forward and backward polynomials that mentioned in relations (11) and (12). For S-box of $SR(n, 2, 1, 4)$ we computed

following Gröbner basis:

$$
\begin{aligned}
&Y^6 + 3Y^5 + FX^2Y^2 + 2Y^4 + 4X^2Y + 2XY^2 + BY^3 + 6XY + FY^2 + 3X + 8Y + 1\\
&X^5 + CY^5 + FX^4 + 2X^3Y + BX^2Y^2 + 9XY^3 + 9Y^4 + 5X^3 + 5X^2Y + 4XY^2 + 3Y^3 + 3X^2 +\\
&\qquad CXY + 8Y^2 + DX + 2Y + 3\\
&X^4Y + 3Y^5 + 3X^4 + FX^3Y + 5X^2Y^2 + 7XY^3 + 9Y^4 + 6X^3 + 5X^2Y + XY^2 + 2Y^3 + 2X^2 +\\
&\qquad 3XY + BY^2 + DX + CY + 2 \qquad (15)\\
&X^3Y^2 + 5Y^5 + FX^4 + X^3Y + 8X^2Y^2 + 9XY^3 + AY^4 + 8X^3 + 6X^2Y + 3XY^2 + 6Y^3 + DX^2 +\\
&\qquad 5XY + 2Y^2 + 5Y + 4\\
&X^2Y^3 + FY^5 + 5X^2Y^2 + 4Y^4 + 7X^2Y + CY^3 + 8X^2 + DXY + AY^2 + 8X + 7Y + E\\
&XY^4 + EY^5 + X^4 + 3X^3Y + CX^2Y^2 + 4XY^3 + FY^4 + EX^3 + 5X^2Y + XY^2 + 5Y^3 + 8XY +\\
&\qquad 3Y^2 + AX + EY + 6
\end{aligned}
$$

The polynomials in (15) have degree less than 7, while FW and BW polynomials are of degree 14.

4.4 Empirical Study

In this section, we compare studied representation based $SR(n, 2, 1, 4)$. Table 4 reports the number of terms and equations for describing $SR(n, 2, 1, 4)$ for different descriptions.

Table 4: comparison of number of terms and equations for $SR(n, 2, 1, 4)$ in $GF(2^4)$

N_r	FWBW			GRB			BES-MQ		
	var	#eqs	#mon	var	#eqs	#mon	#var	#eqs	#mon
1	6	10	79	6	26	75	24	48	57
2	10	18	333	10	50	223	40	88	217
3	14	26	576	14	74	371	56	128	377
4	18	34	850	18	98	519	72	168	537
5	22	42	1100	22	122	667	88	208	697
6	26	50	1360	26	146	815	104	248	857
7	30	58	1619	30	170	963	120	288	1017
8	34	66	1878	34	192	1111	136	328	1177
9	38	74	2131	38	218	1259	152	368	1337
10	42	82	2383	42	242	1407	168	408	1497

Considering the Table 4 FWBW representation leads to fewer equations and fewer monomials in comparison with other descriptions. Interestingly both BES-MQ and GRB lead to a similar number of monomials, but GRB generates less equations. Another point that may be mentioned is that in GRB representation have monomials of degree 7 while in BES-MQ the maximum degree is two.

The experiments are done with the same system in the previous section. Table 5 presents the average running time in seconds for computing the Gröbner bases. N_r stands for the number of cipher rounds. The numbers in parentheses

Table 5: Average running time for computation of Gröbner basis in $GF(2^4)$ (seconds)

N_r	FWBW	GRB	BES-MQ
1	0.19	0.17	0.21
2	0.82	0.23	0.41
3	1.89	0.73	0.99
4	15.42	1.77	3.64
5	30.72	4.69	12.36
6	104.54(48)	11.83	22.69
7	101.61(41)	18.85	38.46
8	$\perp$	30.39	56.07
9	$\perp$	52.87	74.44
10	$\perp$	78.94	103.24

show the number of solved instances (out of 50). The entries with $\perp$ indicate cases where computation of Gröbner basis has failed.

Table 5 reports the average running time for solving the system of equations that derived with FWBW, GRB and BES-MQ representations. According to Table 5 FWBW representation leads to solving 7 round of $SR(n, 2, 1, 4)$ and average running time is much higher in comparison with GRB and BES-MQ. The GRB representation leads to less running time in comparison with BES-MQ. The interesting point is that GRB have more degree and less equations with respect to BES-MQ. Another point is that BES-MQ description is valid only if the inversion relation of S-box is valid, i.e. $xy = 1$ and the inputs of all of the S-boxes must be non-zero. But for GRB description there is not such limitation and the resulting system of equation is valid for all inputs and outputs of S-boxes. Therefore we conclude that GRB is more efficient description than BES-MQ for $GF(2^4)$.

5 Experimenting with larger S-boxes

Considering the interesting behavior of FWBW representation of 4-bit S-boxes it would be interesting to analyze it for S-boxes with larger dimensions. In [3] block cipher $SR(n, 2, 1, 4)$ and $SR(n, 2, 1, 8)$ are only defined, which have 4 and 8 bit S-boxes respectively. Our experiments failed for even small rounds of $SR(n, 2, 1, 8)$. To achieve a better understanding of how FWBW and MQ representations might affect algebraic cryptanalysis of a cipher with larger S-boxes, we defined similar ciphers to $SR(n, 2, 1, 4)$ and $SR(n, 2, 1, 8)$ but with 5, 6 and 7 bit S-boxes. The designed S-boxes follow same structure as $SR(n, 2, 1, 4)$ and $SR(n, 2, 1, 8)$ ones, i.e these S-boxes are based on inversion mapping and have following algebraic structure:

$$S(x) = \psi^{-1}(A_{n\times n} \times \psi(x^{-1})) + C \tag{16}$$

In relation (16) the inversion is applied over $GF(2^n)$ and ψ is a mapping from $GF(2^n)$ to $GF(2)^n$. A is a $n \times n$ binary matrix and C is constant in $GF(2^n)$. ψ^{-1} is the inverse of ψ and a mapping from $GF(2)^n$ to $GF(2^n)$. The definition

Table 6: Comparison of number of terms, equations and avergae solving time for FWBW and MQ representaion of $SR(n, 2, 1, 5)$

N_r	#var	$SR(r, 2, 1, 5)$					
		OD-MQ			FWBW		
		#eqs	#mon	T	#eqs	#mon	T
1	30	106	191	0.22	50	181	0.22
2	50	202	568	0.83	90	2542	0.76
3	70	298	945	19.87	130	4903	2.26
4	90	394	1322	115.08	170	7264	451.44
5	110	490	1699	660.66	210	9625	404.41
6	130	586	2076	977.73	250	11986	1284.01
7	150	682	2453	2286.1	290	14347	1535.99
8	170	778	2830	4197.01	330	16708	2480.7
9	190	874	3207	$\perp$	370	19069	2523.43
10	210	970	3584	9211.54	410	21430	3991.09

of new S-boxes is reported in Appendix A. The system of equations for the cipher is generated as section 2.

5.1 Empirical Study

In this section, we report some experiments to compare the effect of S-box description in solving the resulting system of equations. The setup for experiments is as before. The experiments are applied for $SR(n, 2, 1, 5)$ and $SR(n, 2, 1, 6)$. Table 6 reports the number of terms, equations and average running time for computation of Gröbner basis for different description of S-box of $SR(n, 2, 1, 5)$. Considering the Table 6, it is obvious that generally FWBW yields a more efficient analysis in comparison with OD-MQ representation. But if we compare the results with the effect of FWBW for $SR(n, 2, 1, 5)$, there is not much difference until round 7. With an increase in the number of rounds the difference is more. Another interesting thing is that for OD-MQ representation after each round the running time is doubled, but for FWBW it happens for each 2 rounds. But because FWBW has more degree than OD-MQ, it is counter-intuitive.

Table 7 reports the number of terms, equations, and average running time for $SR(n, 2, 1, 6)$. Considering Table 7, OD-MQ and FWBW did not lead to solving a system of equation for $SR(n, 2, 1, 6)$ with number of round greater than 3. But taking into account the Table 6, we can not conclude that necessarily that MQ representation would lead to more efficient analysis. The interesting point is that with the increase of dimension of S-boxes the running time for computation of Gröbner basis increases exponentially.

6 Conclusion

In this paper, we have studied the different of S-box representations that discussed in the literature, and we have compared their efficiency in algebraic cryptanalysis. For the mentioned representations many properties have been

Table 7: Comparison of number of terms, equations and avergae solving time for FWBW and MQ representaion of $SR(n, 2, 1, 6)$

N_r	#var	$SR(r, 2, 1, 6)$					
		OD-MQ			FWBW		
		#eqs	#mon	T	#eqs	#mon	T
1	36	128	271	0.24	60	373	0.32
2	60	244	813	1.66	108	12138	10.72
3	84	360	1355	19.87	156	23911	530.51
4	108	476	1897	$\perp$	204	35684	$\perp$
5	132	592	2439	$\perp$	252	47449	$\perp$
6	156	708	2951	$\perp$	300	59214	$\perp$
7	180	824	3523	$\perp$	348	70987	$\perp$
8	204	940	4065	$\perp$	396	82760	$\perp$
9	228	1056	4607	$\perp$	444	94533	$\perp$
10	252	1172	5149	$\perp$	492	106306	$\perp$

discussed in the literature. Our study shows that these properties do not reflect the actual efficiency of these representations in algebraic attacks. In particular, the FWBW representation, which is against all mentioned criteria such as degree of polynomials, number of free terms, etc., gives the best running time for solving the system of equations for ciphers $SR(n, 2, 1, 4)$ with Gröbner basis methods.

Another aspect that we have studied in this paper, was how these representations might behave for S-boxes with dimensions greater than 4. In [3] only 4-bit and 8-bit S-boxes have been defined for SR family of ciphers. But our experiments even failed to run for ciphers with 8-bit S-boxes, therefore we defined new ciphers with the same structure as $SR(n, 2, 14)$ but with 5, 6 and 7 bit S-boxes. Our investigations showed with increase of one bit in dimension of S-box, the running time for solving the system of equations increases significantly. For example, the system of equations for $SR(10, 2, 1, 5)$ with FWBW representation, solves in 3991 seconds in average, but for $SR(10, 2, 1, 4)$ solves in just about 22.81 seconds on average. This shows that for a more secure design against algebraic cryptanalysis it might be better to use S-boxes with dimensions greater than 4. Another interesting observation was that the FWBW representation gives a more efficient algebraic cryptanalysis in comparison with MQ representation even with 5-bit S-boxes.

We also tried to extend our results for algebraic representation in larger fields such as $GF(2^n)$. We compared three different approaches, i.e FWBW, GRB and BES-MQ to algebraically represent S-boxes in $GF(2^n)$. BES-MQ gives a system of equations with polynomials of degree 2 and only could be derived if S-box have a special algebraic structure, even though the equations are held with probability. The GRB representation is derived from computation of Gröbner basis of FWBW representation with field polynomials in $GF(2^n)$. Although GRB has high degree polynomials, it is still leads to faster solving time for equations that arises from the cipher. In contrast to $GF(2)$-FWBW, the $GF(2^4)$ counterpart gives the worst running time among the representations.

References

[1] Arabnezhad-Khanoki, H., Sadeghiyan, B., Pieprzyk, J.: S-boxes representation and efficiency of algebraic attack. IET Information Security 13, 448–458(10) (September 2019), https://digital-library.theiet.org/content/journals/10.1049/iet-ifs.2018.5201

[2] Biryukov, A., Cannière, C.D.: Block Ciphers and Systems of Quadratic Equations. In: Johansson, T. (ed.) Fast Software Encryption. No. 2887 in Lecture Notes in Computer Science, Springer Berlin Heidelberg (Feb 2003), http://link.springer.com/chapter/10.1007/978-3-540-39887-5_21

[3] Cid, C., Murphy, S., Robshaw, M.J.B.: Small Scale Variants of the AES. In: Gilbert, H., Handschuh, H. (eds.) Fast Software Encryption, pp. 145–162. No. 3557 in Lecture Notes in Computer Science, Springer Berlin Heidelberg (2005)

[4] Cid, C.: Some Algebraic Aspects of the Advanced Encryption Standard. In: Dobbertin, H., Rijmen, V., Sowa, A. (eds.) Advanced Encryption Standard – AES, pp. 58–66. No. 3373 in Lecture Notes in Computer Science, Springer Berlin Heidelberg (2004)

[5] Cid, C., Leurent, G.: An Analysis of the XSL Algorithm. In: Roy, B. (ed.) Advances in Cryptology - ASIACRYPT 2005, pp. 333–352. No. 3788 in Lecture Notes in Computer Science, Springer Berlin Heidelberg (2005)

[6] Courtois, N., Klimov, A., Patarin, J., Shamir, A.: Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In: Preneel, B. (ed.) Advances in Cryptology — EUROCRYPT 2000, pp. 392–407. No. 1807 in Lecture Notes in Computer Science, Springer Berlin Heidelberg (2000), http://link.springer.com/chapter/10.1007/3-540-45539-6_27

[7] Courtois, N., Mourouzis, T., Hulme, D.: Exact logic minimization and multiplicative complexity of concrete algebraic and cryptographic circuits. International Journal On Advances in Intelligent Systems 6(3), 165–176 (2013)

[8] Courtois, N.T., Bard, G.V.: Algebraic Cryptanalysis of the Data Encryption Standard. In: Galbraith, S.D. (ed.) Cryptography and Coding 2007, pp. 152–169. No. 4887 in Lecture Notes in Computer Science, Springer Berlin Heidelberg (2007)

[9] Courtois, N.T., Pieprzyk, J.: Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. In: Zheng, Y. (ed.) Advances in Cryptology — ASIACRYPT 2002, pp. 267–287. No. 2501 in Lecture Notes in Computer Science, Springer Berlin Heidelberg (2002)

[10] Daemen, J., Rijmen, V.: AES proposal: Rijndael, in NIST AES Proposal (1998)

[11] Murphy, S., Robshaw, M.J.B.: Essential Algebraic Structure within the AES. In: Yung, M. (ed.) Advances in Cryptology — CRYPTO 2002, pp.

1–16. No. 2442 in Lecture Notes in Computer Science, Springer Berlin Heidelberg (Aug 2002), http://link.springer.com/chapter/10.1007/3-540-45708-9_1

[12] Pyshkin, A.: Algebraic cryptanalysis of block ciphers using Gröbner Bases. PhD, Darmstadt, Technische Universität (2008)

[13] The Sage Developers: SageMath, the Sage Mathematics Software System (Version 6.7) (2015), http://www.sagemath.org

A Definition of larger S-boxes for SR family of Block ciphers

S-box of $SR(r, 2, 1, 4)$ **[3]** The inverse operation for S-box is defined as $\alpha^4 + \alpha + 1$.

$$S(x) = \psi^{-1}(\begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix} \times \psi(x^{-1})) + 6$$

S-box of $SR(r, 2, 1, 5)$ The inverse operation for S-box is defined as $\alpha^5+\alpha^2+1$.

$$S(x) = \psi^{-1}(\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 \end{bmatrix} \times \psi(x^{-1})) + 3$$

S-box of $SR(r, 2, 1, 6)$ The inverse operation for S-box is defined as $\alpha^6+\alpha+1$.

$$S(x) = \psi^{-1}(\begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \times \psi(x^{-1})) + 43$$

S-box of $SR(r, 2, 1, 7)$ The inverse operation for S-box is defined as $\alpha^7+\alpha+1$.

$$S(x) = \psi^{-1}(\begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \times \psi(x^{-1})) + 63$$

S-box of $SR(r, 2, 1, 8)$ **[3]** The inverse operation for S-box is defined as $\alpha^8 + \alpha^4 + \alpha^3 + \alpha + 1$.

$$S(x) = \psi^{-1}(\begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \times \psi(x^{-1})) + 63$$