On Deviations of the AES S-box when Represented as Vector Valued Boolean Function

Danilo Gligoroski[†] **and Marie Elisabeth Gaup Moe**[†],

[†]Centre for Quantifiable Quality of Service in Communication Systems, Norwegian University of Science and Technology, O.S.Bragstads plass 2E, N-7491 Trondheim, NORWAY

Summary

In this paper we give an explicit representation of the AES S-box as a vector valued Boolean function in $GF(2)^8$ and show several significant deviations in the number of terms that follows from that representation when it is compared with the algebraic representation of randomly generated permutations of 256 elements. We see this as a potential research direction in cryptanalysis of AES.

Key words:

AES, S-box, cryptanalysis, algebraic presentation.

1. Introduction

Algebraic attacks on AES have become an attractive research field in cryptology in recent years. It all started when the Rijndael designers (Daemen and Rijmen) in their proposal [1] for the NIST standard, used a permutation of 256 elements in the field $GF(2^8)$ which allowed the S-box to be algebraically described. They decided to use an affine transformation of the permutation presented by Nyberg in [2]. In their book about the block cipher Rijndael [3], the authors explain in more detail how the S-box looks like in $GF(2^8)$. They first take the function

$$g(x) = \begin{cases} x^{-1}, & x \neq 0 \\ 0 \ , & x = 0 \end{cases} \qquad (1)$$

where every byte represents a polynomial. Then, they apply the affine transformation

$$f(x) = 05\cdot x + 09\cdot x^2 + F9\cdot x^{2^2} + 25\cdot x^{2^3} + F4\cdot x^{2^4} + 01\cdot x^{2^5} + B5\cdot x^{2^6} + 8F\cdot x^{2^7} + 63$$

and obtain the S-box permutation in $GF(2^8)$ that can be expressed as

$$Sbox(x) = f(g(x)) = 05\cdot x^{254} + 09\cdot x^{253} + F9\cdot x^{251} + 25\cdot x^{247} + F4\cdot x^{239} + 01\cdot x^{223} + B5\cdot x^{191} + 8F\cdot x^{127} + 63.$$

In the beginning of the second round of the NIST selection of the AES, Murphy and Robshaw [4] gave comments and remarks about the algebraic properties of Rijndael's S-box and the whole structure of the block cipher, as a possible weak point of the cipher. In 2001, Ferguson, Schroeppel and Whiting [5], using again algebraic properties of Rijndael derived a closed formula for the cipher as a continued fraction. Full development of that formula would generate multivariate polynomials of higher order with 226 unknown variables and 250 terms. This approach was considered a promising direction for the algebraic attacks, however, no successful reports since then have appeared on solving such a complex systems of equations. Then in CRYPTO'02, using again the algebraic structures of Rijndael-AES, Murphy and Robshaw published a paper [6] in which they embedded the cipher into a bigger block cipher called the Big Encryption System or BES, and discussed how to make a cryptanalysis of that cipher. At the same conference, Courtois and Pieprzyk [7] proposed an attack to the AES by solving a system of multivariate quadratic equations by the extended sparse linearisation or XSL algorithm. That method is a version of the extended linearisation or XL algorithm that was presented by Shamir et al. [8] in Eurocrypt 2000. In the attack of Courtois and Pieprzyk, authors claimed that for 128-bit block and 128-bit key Rijndael, recovering the secret key from one single plaintext can be accomplished if a system of 8000 quadratic equations with 1600 binary unknowns is solved in GF(2), and that this has an equivalent workload of about 2^{100} AES encryptions. The claims in that paper were debated and criticized by several authors (see for example [9] and references there). Several other authors have given good surveys about algebraic attacks on block ciphers, see for example [10, 11]. In [12] more precise upper bounds on the dimensions of the spaces of equations in the XL-algorithm were obtained, giving pessimistic evidence for the running time of the algorithm. In [13] the XL method was compared with the Buchberger method for calculation of reduced Groebner bases.

In spite of the large number of papers about algebraic attacks on the AES, we did not find any reference in which its S-box is explicitly represented as a nonlinear Boolean vector function in $GF(2)^8$. The authors of [14]

Manuscript received April 5, 2007
Manuscript revised April 25, 2007

have discussed some linear redundancy properties of these functions but did not provide the functions explicitly in the paper.

Additional motivation for obtaining an explicit representation for the AES S-box we got from the paper [15], where the author in order to obtain a minimal hardware realization of DES, optimized Boolean expressions that represents DES S-boxes. When concerning a hardware realization, having explicit relations of the S-boxes can lead to finding minimal hardware implementations of AES as well.

Finally, an explicit algebraic representation can give additional valuable information about how the S-box correlates the bits when it is used in the block cipher. The "algebraic distinguisher" was mentioned in the paper by Cid, Murphy and Robshaw [16]. The authors refer to the rich algebraic structure of AES as a possible source for finding a polynomial-time distinguisher that can perform successful distinguishing between the cipher and a truly random source. In this paper we will show how the AES S-box manifests some significant deviations from random permutations of 256 elements when it is represented in explicit algebraic form as a vector valued Boolean function in $GF(2)^8$. We point out to these deviations as a possible research direction when attempting to build an algebraic distinguisher. Our initial findings on this topic will be presented in the following three sections.

The structure of the paper is the following. In Section 2 we give a brief description of the linear procedures by which it is possible to obtain the representation of the AES S-box as an 8 dimensional vector valued Boolean function. In Section 3 we give a comparative analysis between the AES S-box and randomly generated permutations of 256 elements based on the statistics of the number of used terms in $GF(2)^8$ and show several significant deviations from randomly generated permutations, and in Section 4 we give some conclusions. In Appendix 1, we give an explicit representation of all 8 Boolean functions that represents the AES S-box.

2. Obtaining AES S-box representation in $GF(2)^8$

Obtaining a representation of the AES S-box as an 8 dimensional vector valued Boolean function is a linear problem. It is relatively easy solvable by any modern mathematical tool that can do symbolic calculations in finite fields and the problem is the following:

Table 1: Comparing obtained statistics from randomly generated permutations of 256 elements and actual values for the AES S-box

Nr. of terms	*Random* (μ,σ^2)	*AES S-box*	*Nr. of terms*	*Random* (μ,σ^2)	*AES S-box*
$\|B_0\|$	(127.4, 8.0)	110	$\|B_1 \cap B_5\|$	(63.7, 7.0)	63
$\|B_1\|$	(127.5, 8.0)	112	$\|B_1 \cap B_6\|$	(63.7, 7.0)	59
$\|B_2\|$	(127.5, 8.0)	114	$\|B_1 \cap B_7\|$	(63.8, 6.9)	56
$\|B_3\|$	(127.4, 8.0)	131	$\|B_2 \cap B_3\|$	(63.8, 6.9)	54
$\|B_4\|$	(127.6, 7.9)	136	$\|B_2 \cap B_4\|$	(63.8, 7.0)	60
$\|B_5\|$	(127.5, 8.0)	145	$\|B_2 \cap B_5\|$	(63.8, 6.9)	66
$\|B_6\|$	(127.5, 8.0)	133	$\|B_2 \cap B_6\|$	(63.7, 6.8)	63
$\|B_7\|$	(127.6, 8.0)	132	$\|B_2 \cap B_7\|$	(63.8, 7.0)	56
$\|B_0 \cap B_1\|$	(63.6, 6.9)	52	$\|B_3 \cap B_4\|$	(63.7, 6.9)	59
$\|B_0 \cap B_2\|$	(63.7, 6.9)	45	$\|B_3 \cap B_5\|$	(63.7, 6.8)	74
$\|B_0 \cap B_3\|$	(63.7, 6.9)	50	$\|B_3 \cap B_6\|$	(63.8, 6.9)	75
$\|B_0 \cap B_4\|$	(63.7, 6.9)	60	$\|B_3 \cap B_7\|$	(63.7, 6.9)	68
$\|B_0 \cap B_5\|$	(63.7, 6.9)	66	$\|B_4 \cap B_5\|$	(63.8, 6.9)	80
$\|B_0 \cap B_6\|$	(63.7, 6.9)	52	$\|B_4 \cap B_6\|$	(63.7, 6.9)	77
$\|B_0 \cap B_7\|$	(63.7, 6.9)	61	$\|B_4 \cap B_7\|$	(63.8, 6.9)	74
$\|B_1 \cap B_2\|$	(63.7, 6.9)	55	$\|B_5 \cap B_6\|$	(63.7, 6.9)	77
$\|B_1 \cap B_3\|$	(63.6, 7.0)	54	$\|B_5 \cap B_7\|$	(63.8, 7.0)	68
$\|B_1 \cap B_4\|$	(63.8, 6.8)	59	$\|B_6 \cap B_7\|$	(63.8, 6.9)	76

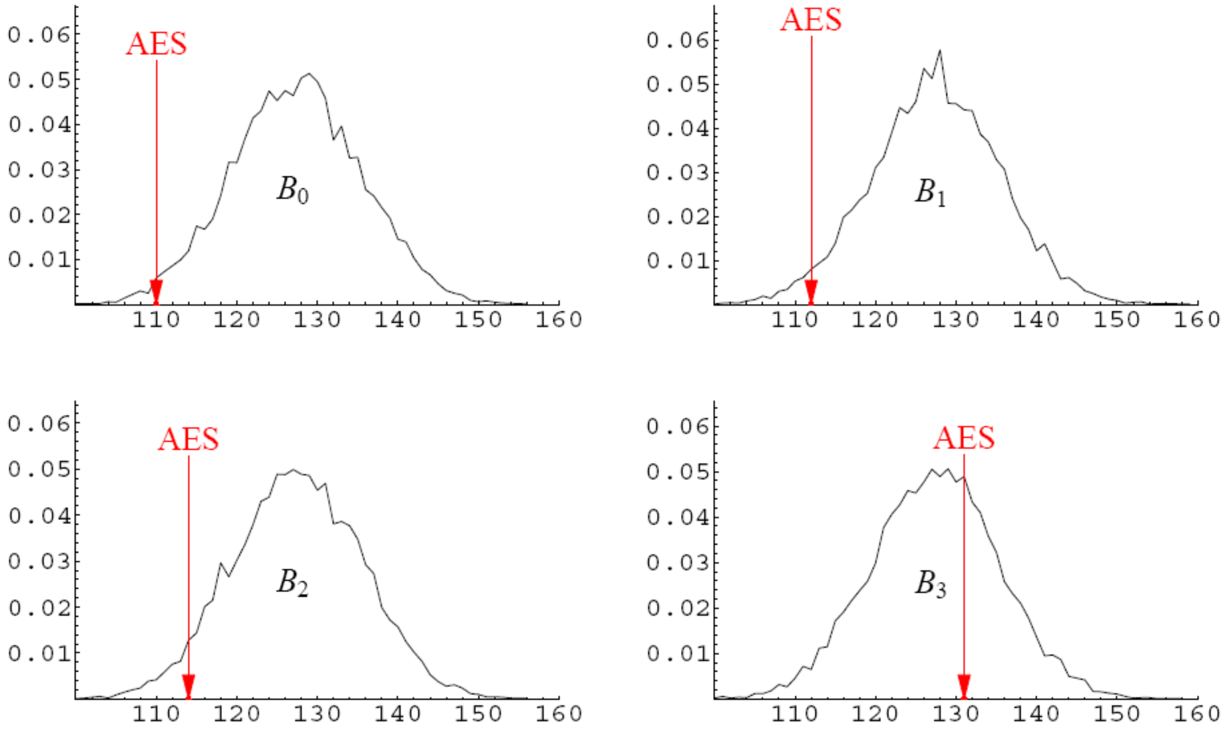

Fig. 1 Distribution of the number of terms in the sets B_i, i = 0, 1, 2, 3. On x axes we represented the number of elements in B_i. The arrow shows the actual number of terms in AES S-box representation.

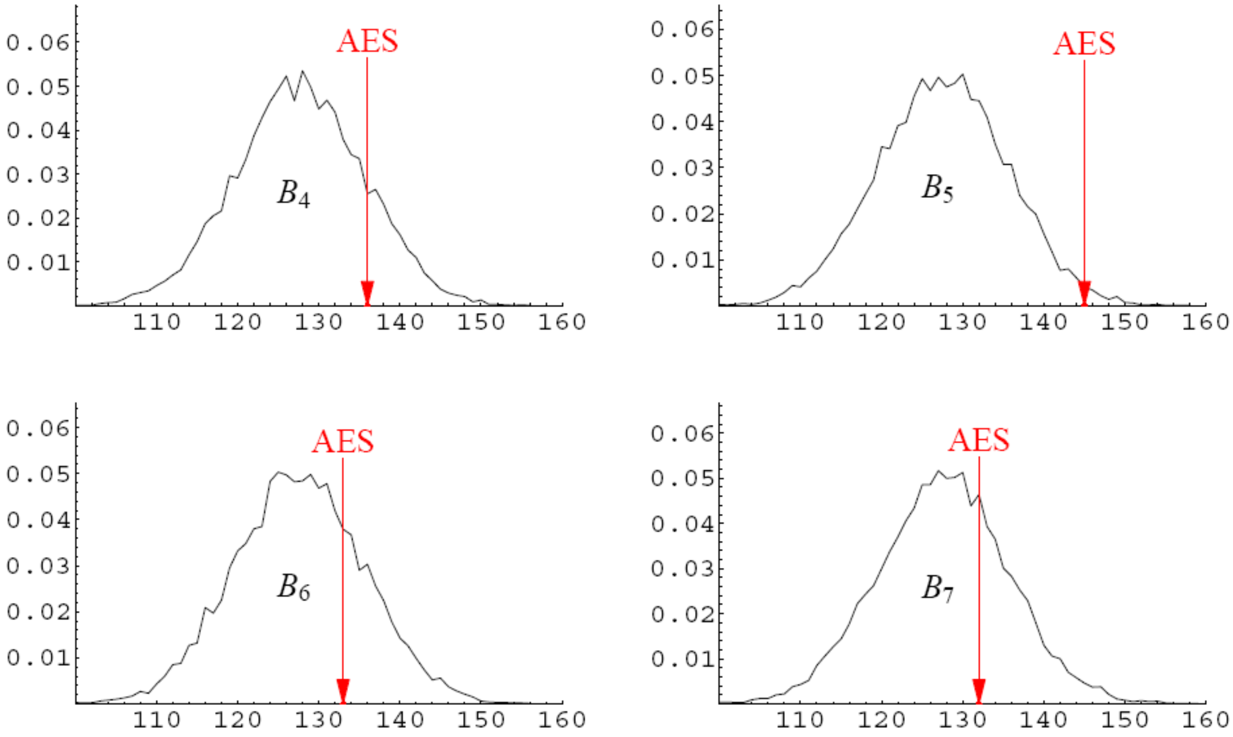

Fig. 2 Distribution of the number of terms in the sets B_i, i = 4, 5, 6, 7. On x axes we represented the number of elements in B_i. The arrow shows

Having a permutation $Sbox(x) : \{0,1\}^8 \rightarrow \{0,1\}^8$, find 8 Boolean functions $f_0, f_1, \dots , f_7, : \{0,1\}^8 \rightarrow \{0,1\}$ in GF(2), such that

$$Sbox(x_0, \dots, x_7) = (f_0(x_0, \dots, x_7), \dots, f_7(x_0, \dots, x_7)).$$

Having in mind that in GF(2) multiplication $x \cdot y \equiv xy$ is equivalent to logical AND, and addition $x+y$ is equivalent to logical XOR, the functions $f_0, f_1, \dots , f_7$ can be represented as

$$\begin{aligned} f_i(x_0,\dots,x_7) = a_0^{(i)} &+ \\ &+ a_1^{(i)} x_0 + \cdots + a_8^{(i)} x_7 + \\ &+ a_9^{(i)} x_0 x_1 + \cdots + a_{36}^{(i)} x_6 x_7 + \\ &\vdots \\ &+ a_{255}^{(i)} x_0 x_1 x_2 x_3 x_4 x_5 x_6 x_7 \end{aligned} \tag{2}$$

where $a_j^{(i)} \in \{0,1\}$ are unknown coefficients. So, for every value $x \in \{0,1\}^8$ we can find its image $Sbox(x) \in \{0,1\}^8$, and replacing these values in Eq. 2 we can establish a linear system with 256 x 8 = 2048 unknown variables $a_j^{(i)}$ $i = 0, 1, \dots, 7, j = 0, 1, \dots, 255$ in GF(2). For example, for $x = 0$ in GF(2^8), i.e. $x = (0, 0, 0, 0, 0, 0, 0, 0)$ in GF$(2)^8$ we have that $Sbox(x) = 99$ in GF(2^8), i.e. $Sbox(x) = (0, 1, 1, 0, 0, 0, 1, 1)$ in GF$(2)^8$. From this we can immediately obtain the values for the free constants $a_0^{(i)}$ $i = 0, 1, \dots, 7$. By exchanging all other 255 values for x we will obtain a linear system with 2048 unknown variables that is easy solvable. For example the function f_0 can be represented as follows:

$f_0(x_0, x_1, \dots, x_7) = x_0 + x_2 + x_3 + x_5 + x_0x_2 + x_0x_6 + x_0x_7 + x_1x_3 + x_1x_5 + x_1x_7 + x_2x_4 + x_3x_5 + x_5x_6 + x_5x_7 + x_0x_1x_3 + x_0x_1x_5 + x_0x_1x_7 + x_0x_2x_3 + x_0x_2x_4 + x_0x_3x_6 + x_0x_4x_7 + x_0x_5x_7 + x_1x_2x_3 + x_1x_2x_5 + x_1x_2x_7 + x_1x_3x_5 + x_1x_4x_6 + x_1x_4x_7 + x_1x_5x_6 + x_1x_6x_7 + x_2x_3x_4 + x_2x_3x_7 + x_2x_4x_5 + x_2x_4x_6 + x_2x_4x_7 + x_2x_5x_7 + x_2x_6x_7 + x_3x_6x_7 + x_4x_5x_6 + x_4x_5x_7 + x_0x_1x_2x_3 + x_0x_1x_2x_5 + x_0x_1x_2x_6 + x_0x_1x_3x_5 + x_0x_1x_3x_6 + x_0x_1x_3x_7 + x_0x_1x_4x_6 + x_0x_1x_4x_7 + x_0x_2x_3x_6 + x_0x_2x_4x_7 + x_0x_3x_4x_5 + x_0x_3x_4x_6 + x_0x_3x_4x_7 + x_0x_3x_5x_7 + x_0x_3x_6x_7 + x_0x_4x_5x_6 + x_0x_4x_5x_7 + x_0x_5x_6x_7 + x_1x_2x_3x_4 + x_1x_2x_4x_7 + x_1x_3x_4x_7 + x_1x_3x_5x_6 + x_1x_3x_5x_7 + x_1x_4x_5x_7 + x_1x_4x_6x_7 + x_2x_3x_5x_6 + x_2x_4x_5x_6 + x_2x_4x_5x_7 + x_2x_5x_6x_7 + x_3x_4x_5x_6 + x_3x_4x_6x_7 + x_3x_5x_6x_7 + x_4x_5x_6x_7 + x_0x_1x_2x_3x_4 + x_0x_1x_2x_3x_6 + x_0x_1x_2x_4x_6 + x_0x_1x_2x_4x_7 + x_0x_1x_2x_5x_6 + x_0x_1x_2x_6x_7 + x_0x_1x_3x_4x_7 + x_0x_1x_3x_5x_6 + x_0x_1x_5x_6x_7 + x_0x_2x_3x_6x_7 + x_0x_2x_5x_6x_7 + x_0x_3x_4x_5x_7 + x_0x_3x_4x_6x_7 + x_1x_2x_3x_4x_5 + x_1x_2x_3x_4x_7 + x_1x_2x_3x_5x_6 + x_1x_2x_3x_6x_7 + x_1x_2x_4x_5x_6 + x_1x_3x_4x_5x_7 + x_1x_3x_5x_6x_7 + x_1x_4x_5x_6x_7 + x_2x_3x_4x_5x_7 + x_2x_3x_4x_6x_7 + x_3x_4x_5x_6x_7 + x_0x_1x_2x_3x_4x_6 + x_0x_1x_2x_3x_5x_7 + x_0x_1x_2x_4x_5x_7 + x_0x_1x_2x_4x_6x_7 + x_0x_1x_3x_4x_5x_7 + x_0x_1x_3x_4x_6x_7 + x_0x_1x_4x_5x_6x_7 + x_0x_2x_3x_4x_5x_7 + x_0x_2x_3x_4x_6x_7 + x_0x_2x_3x_5x_6x_7 + x_1x_2x_3x_4x_5x_7 + x_0x_1x_2x_3x_4x_5x_7 + x_0x_1x_2x_3x_4x_6x_7$

The actual representation of f_1, f_2, f_3, f_4, f_5, f_6 and f_7 can be found in Appendix 1.

3. Significant deviations of the AES S-box from random per mutations of order 256

It is easy to notice that the algebraic order of f_i, i=0, 1, …, 7 is 7. That is of course in compliance with the theoretical result that Nyberg obtained in her paper [2]. Namely, for even n, the inversion mapping described by Eq. 1 in GF(2^n) has algebraic order n - 1. For the AES case, we have $n = 8$, and indeed the order of the polynomials f_i, i=0, 1, …, 7 is 7.

We have performed several measurements, in order to compare how much the AES S-box looks like or differs from a randomly generated permutation of 256 elements. Let us denote by B_0, B_1, …, B_7 the sets of monomials in the 8 coordinate Boolean functions f_i, i=0, 1, …, 7. We have generated 10,000 random permutations of 256 elements. To generate these random permutations we have used the random generator that is built into *Mathematica* 5.1. Then we measured the distribution of the number of used terms in all 8 coordinates, as well as distributions of their intersections and compared them by the actual values obtained from the AES S-box. Some of the statistical measurements that we have made are given in Table 1 and Figures 1 and 2. In Table 1 next to each column that gives the average and standard deviation for each set B_i, i=0, 1, …, 7, we have placed the actual values obtained from the AES S-box. The graphical representations where the actual values for AES S-box are positioned are given in Figures 1 and 2 and are denoter by red arrows.

A standard procedure in this kind of measurement is to compare an actual finding, in this case the values obtained from the AES S-box, with the values we would expect to find assuming that the number of terms is normally distributed. We have performed an empirical evaluation to confirm that the number of terms in the randomly generated sample actually is normally distributed. As an illustration, in Figure 2 we have plotted the probability density function for the standard normal distribution. The values for a random variable that deviate from the mean with more than two times the standard deviation have probability to occur less then 4%, i.e. 96% of the observations will deviate less than 2.05 standard deviations from the mean (the central shaded gray area in Figure 3).

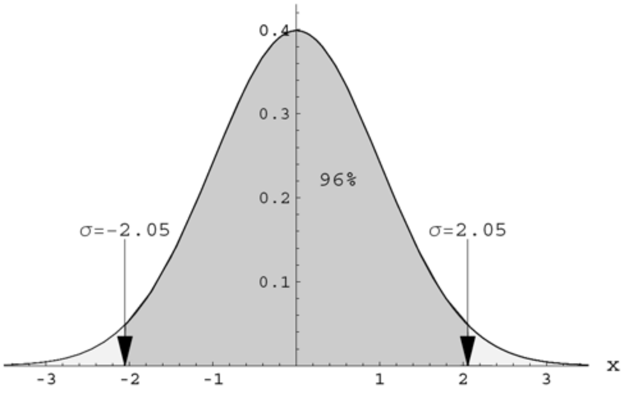

Fig. 3 Shaded area is 96% of the total area beneath the probability density function.

The first impressions comparing the obtained averages from the randomly generated permutations to the actual values for the AES S-box, are that they do not differ much. However, there are several situations where we noticed significant deviations. For example, for the situation $|B_0 \cap B_2|$ (see Table 1) we have the values $(\mu,\sigma^2) = (63.7, 6.9)$ in the randomly generated sample, while for the AES S-box we get $|B_0 \cap B_2|=45$. This value should only be observed with a probability of 0.4% in the randomly generated sample.

In Table 1 notice also high values obtained for the AES S-box for the intersections $|B_4 \cap B_5|= 80$, $|B_4 \cap B_6|= 77$, $|B_5 \cap B_6|=77$ and $|B_6 \cap B_7|=76$. All of them lie outside or at the edge of the regions where 99%, 97%, 97% and 96% of the observations would be expected to be found correspondingly. That reflects further to the intersections $|B_4 \cap B_5 \cap B_6|=46$, $|B_4 \cap B_5 \cap B_7|=49$ and $|B_4 \cap B_5 \cap B_6 \cap B_7|=27$, and for them we can say that they lie outside the regions where 99.6%, 99.9% and 99.7% of the observations are expected to be found. These situations are illustrated in Figure 4.

It would be interesting to see, how this observed biases will be reflected on the output of the whole cipher, as S-box is the only nonlinear part in AES.

4. Conclusion

In this paper we have given explicit representation of the AES S-box as a vector of 8 Boolean functions in GF(2). We have also computed several statistical properties of obtained Boolean functions and found several situations where significant deviations from a randomly generated permutation of 256 elements can be seen. It is natural to ask how these findings could be used to build a distinguisher for the AES, and this will be the subject of our further research.

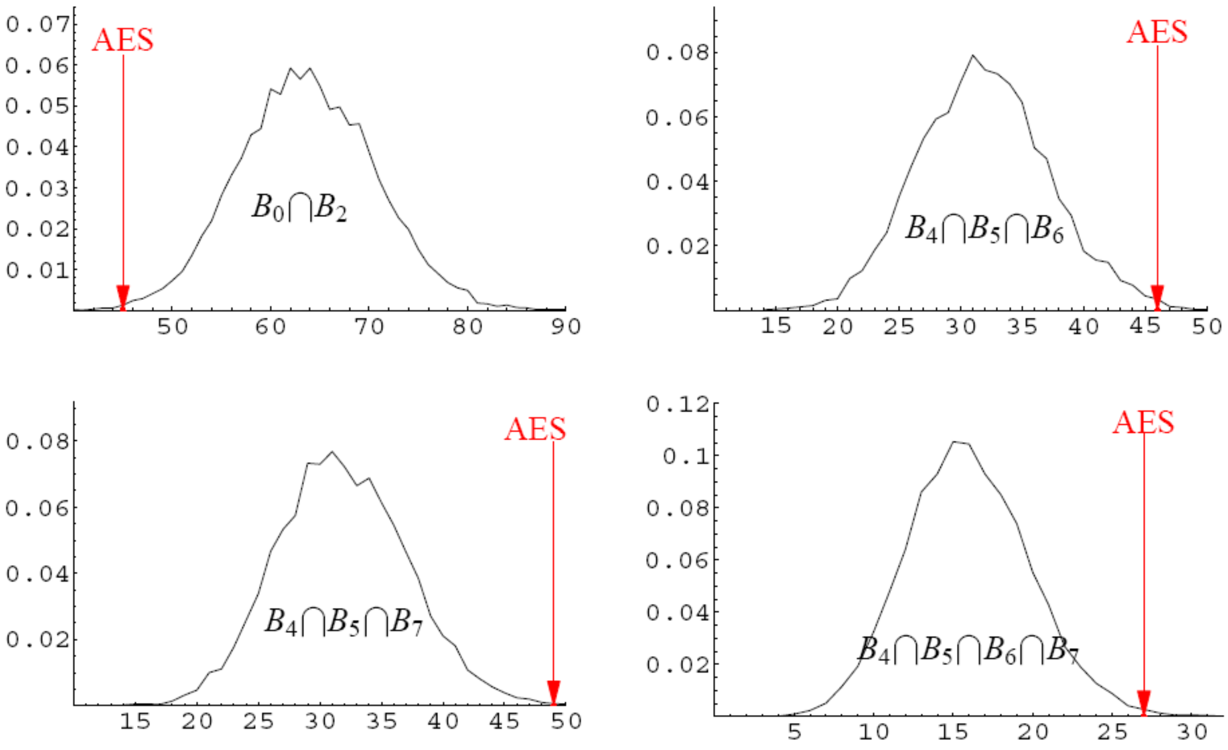

Fig. 4 The situations where the AES S-box representation in $GF(2)^8$ gives significant deviation from "randomly generated" permutations. On *x* axes we represented the number of elements in B_i. The arrow shows the actual number of terms in the AES S-box representation.

References

[1] J. Daemen, V. Rijmen, “The block cipher Rijndael,” available from NIST's AES homepage, URL: http://www.nist.gov/aes

[2] K. Nyberg, “Differentially uniform mappings for cryptography,” Advances in Cryptology, Proc. Eurocrypt'93, LNCS 765, Springer-Verlag, 1994, pp. 55-64.

[3] J. Daemen, V. Rijmen, “The Design of Rijndael: AES - The Advanced Encryption Standard,” ISBN 3-540-42580-2, Springer-Verlag Berlin Heidelberg, 2002.

[4] S. Murphy and M. Robshaw, “New Observations on Rijndael”, initial draft note submitted to NIST as an AES Round Two comment, http://www.isg.rhul.ac.uk/~mrobshaw/rijndael/rijndael.pdf

[5] N. Ferguson, R. Schroeppel and Doug Whiting “A simple algebraic representation of Rijndael”, in S. Vaudenay and A.M. Youssef, ed., Proc. of Selected Areas in Cryptography SAC01, nr. 2259 in LNCS, pp. 103-111. Springer-Verlag, 2001.

[6] S. Murphy and M. Robshaw, “Essential Algebraic Structure Within the AES”, Advances in Cryptology, Crypto 2002, LNCS 2442, pp. 1-16, Springer-Verlag, 2002.

[7] N. Courtois and J. Pieprzyk, “Cryptanalysis of Block Ciphers with Overdefined Systems of Equations”, Advances in Cryptology, Asiacrypt 2002, LNCS 2501, pp. 267-287, Springer-Verlag, 2002.

[8] A. Shamir, J. Patarin, N. Courtois and A. Klimov, “Efficient Algorithms for solving Overdefined Systems of Multivariate Polynomial Equations”, Advances in Cryptology, Eurocrypt2000, LNCS 1807, Springer-Verlag Berlin Heilderberg 2000, pp. 392-407.

[9] S. Murphy and M. Robshaw, “Comments on the Security of the AES and the XSL Technique”, http://www.isg.rhul.ac.uk/~mrobshaw/rijndael/xslnote.pdf

[10] A. Biryukov and C. De Canniere, “Block Ciphers and Systems of Quadratic Equations”, Fast Software Encryption, FSE 2003, LNCS 2887, pp. 274-289, Springer-Verlag, 2003.

[11] A. J. M. Sedgers, A Master's Thesis: “Algebraic Attacks from a Groebner Basis Perspective”, TECHNISCHE UNIVERSITEIT EINDHOVEN, 2004.

[12] C. Diem, “The XL-Algorithm and a Conjecture from Commutative Algebra”, Advances in Cryptology, ASIACRYPT 2004, LNCS 3329, pp. 323-337, Springer-Verlag Berlin Heidelberg, 2004.

[13] G. Ars, J.C. Fauguere, H. Imai, M. Kawazoe, and M. Sugita, “Comparison Between XL and Groebner Basis Algorithms”, Advances in Cryptology, ASIACRYPT 2004, LNCS 3329, pp. 338-353, Springer-Verlag Berlin Heidelberg, 2004.

[14] J. Fuller and W. Millan, “On Linear Redundancy in the AES S-Box”, http://eprint.iacr.org/2002/111

[15] M. Kwan, “Reducing the Gate Count of Bitslice DES”, http://eprint.iacr.org/2000/051

[16] C. Cid, S. Murphy and Matthew Robshaw, “Computational and Algebraic Aspects of the Advanced Encryption Standard”, Proceedings of the Seventh International Workshop on Computer Algebra in Scientific Computing - CASC 2004, pp.93-103, St. Petersburg, 2004.

Danilo Gligoroski received the PhD degree in Computer Science from Institute of Informatics, Faculty of Natural Sciences and Mathematics, at University of Skopje – Macedonia in 1997. His research interests are Cryptography, Computer Security, Discrete algorithms and Information Theory and Coding. Currently he is PostDoc at Q2S – Centre for Quantifiable Quality of Service in Communication Systems at Norwegian University of Science and Technology - Trondheim, Norway.

Marie Elisabeth Gaup Moe received MS in Industrial Mathematics in 2004 from NTNU - Norwegian University of Science and Technology in Trondheim. Her research interests are Cryptography and Computer Security. Currently she is a doctoral student at Q2S – Centre for Quantifiable Quality of Service in Communication Systems.

Appendix 1. Representations of all 8 Boolean functions f_i, i=0, 1, …, 7 of the AES S-box

$f_0(x_0, x_1, \ldots, x_7) = x_0 + x_2 + x_3 + x_5 + x_0x_2 + x_0x_6 + x_0x_7 + x_1x_3 + x_1x_5 + x_1x_7 + x_2x_4 + x_3x_5 + x_5x_6 + x_5x_7 + x_0x_1x_3 + x_0x_1x_5 + x_0x_1x_7 + x_0x_2x_3 + x_0x_2x_4 + x_0x_3x_6 + x_0x_4x_7 + x_0x_5x_7 + x_1x_2x_3 + x_1x_2x_5 + x_1x_2x_7 + x_1x_3x_5 + x_1x_4x_6 + x_1x_4x_7 + x_1x_5x_6 + x_1x_6x_7 + x_2x_3x_4 + x_2x_3x_7 + x_2x_4x_5 + x_2x_4x_6 + x_2x_4x_7 + x_2x_5x_7 + x_2x_6x_7 + x_3x_6x_7 + x_4x_5x_6 + x_4x_5x_7 + x_0x_1x_2x_3 + x_0x_1x_2x_5 + x_0x_1x_2x_6 + x_0x_1x_3x_5 + x_0x_1x_3x_6 + x_0x_1x_3x_7 + x_0x_1x_4x_6 + x_0x_1x_4x_7 + x_0x_2x_3x_6 + x_0x_2x_4x_7 + x_0x_3x_4x_5 + x_0x_3x_4x_6 + x_0x_3x_4x_7 + x_0x_3x_5x_7 + x_0x_3x_6x_7 + x_0x_4x_5x_6 + x_0x_4x_5x_7 + x_0x_5x_6x_7 + x_1x_2x_3x_4 + x_1x_2x_4x_7 + x_1x_3x_4x_7 + x_1x_3x_5x_6 + x_1x_3x_5x_7 + x_1x_4x_5x_7 + x_1x_4x_6x_7 + x_2x_3x_5x_6 + x_2x_4x_5x_6 + x_2x_4x_5x_7 + x_2x_5x_6x_7 + x_3x_4x_5x_6 + x_3x_4x_6x_7 + x_3x_5x_6x_7 + x_4x_5x_6x_7 + x_0x_1x_2x_3x_4 + x_0x_1x_2x_3x_6 + x_0x_1x_2x_4x_6 + x_0x_1x_2x_4x_7 + x_0x_1x_2x_5x_6 + x_0x_1x_2x_6x_7 + x_0x_1x_3x_4x_7 + x_0x_1x_3x_5x_6 + x_0x_1x_5x_6x_7 + x_0x_2x_3x_6x_7 + x_0x_2x_5x_6x_7 + x_0x_3x_4x_5x_7 + x_0x_3x_4x_6x_7 + x_1x_2x_3x_4x_5 + x_1x_2x_3x_4x_7 + x_1x_2x_3x_5x_6 + x_1x_2x_3x_6x_7 + x_1x_2x_4x_5x_6 + x_1x_3x_4x_5x_7 + x_1x_3x_5x_6x_7 + x_1x_4x_5x_6x_7 + x_2x_3x_4x_5x_7 + x_2x_3x_4x_6x_7 + x_3x_4x_5x_6x_7 + x_0x_1x_2x_3x_4x_6 + x_0x_1x_2x_3x_5x_7 + x_0x_1x_2x_4x_5x_7 + x_0x_1x_2x_4x_6x_7 + x_0x_1x_3x_4x_5x_7 + x_0x_1x_3x_4x_6x_7 + x_0x_1x_4x_5x_6x_7 + x_0x_2x_3x_4x_5x_7 + x_0x_2x_3x_4x_6x_7 + x_0x_2x_3x_5x_6x_7 + x_1x_2x_3x_4x_5x_7 + x_0x_1x_2x_3x_4x_5x_7 + x_0x_1x_2x_3x_4x_6x_7$

$f_1(x_0, x_1, \ldots, x_7) = 1 + x_1 + x_2 + x_4 + x_0x_2 + x_0x_4 + x_0x_6 + x_0x_7 + x_1x_3 + x_2x_4 + x_2x_7 + x_3x_7 + x_4x_5 + x_4x_6 + x_0x_1x_2 + x_0x_1x_4 + x_0x_1x_6 + x_0x_2x_4 + x_0x_2x_7 + x_0x_3x_5 + x_0x_3x_6 + x_0x_3x_7 + x_0x_4x_5 + x_0x_5x_6 + x_1x_2x_3 + x_1x_2x_6 + x_1x_2x_7 + x_1x_3x_4 + x_1x_3x_5 + x_1x_3x_6 + x_1x_3x_7 + x_1x_4x_6 + x_1x_4x_7 + x_1x_5x_6 + x_1x_5x_7 + x_2x_3x_7 + x_2x_4x_7 + x_2x_5x_6 + x_2x_5x_7 + x_2x_6x_7 + x_3x_4x_5 + x_3x_4x_6 + x_3x_5x_7 + x_3x_6x_7 + x_4x_6x_7 + x_0x_1x_2x_3 + x_0x_1x_3x_6 + x_0x_1x_3x_7 + x_0x_1x_6x_7 + x_0x_2x_3x_6 + x_0x_2x_4x_5 + x_0x_2x_4x_6 + x_0x_2x_5x_7 + x_0x_3x_4x_6 + x_0x_3x_4x_7 + x_0x_3x_5x_6 + x_0x_4x_5x_7 + x_0x_4x_6x_7 + x_1x_2x_4x_5 + x_1x_3x_4x_5 + x_1x_3x_4x_6 + x_1x_3x_4x_7 + x_1x_3x_5x_7 + x_1x_4x_5x_6 + x_1x_4x_5x_7 + x_1x_4x_6x_7 + x_2x_3x_4x_5 + x_2x_3x_5x_6 + x_2x_3x_5x_7 + x_2x_4x_5x_6 + x_2x_4x_5x_7 + x_2x_5x_6x_7 + x_3x_4x_5x_6 + x_3x_4x_6x_7 + x_3x_5x_6x_7 + x_0x_1x_2x_3x_4 + x_0x_1x_2x_3x_6 + x_0x_1x_2x_3x_7 + x_0x_1x_2x_4x_5 + x_0x_1x_2x_5x_6 + x_0x_1x_2x_5x_7 + x_0x_1x_3x_4x_5 + x_0x_1x_3x_4x_7 + x_0x_1x_3x_5x_7 + x_0x_1x_4x_6x_7 + x_0x_2x_3x_4x_6 + x_0x_2x_3x_5x_7 + x_0x_2x_3x_6x_7 + x_0x_2x_4x_5x_6 + x_0x_3x_4x_5x_6 + x_0x_4x_5x_6x_7 + x_1x_2x_3x_4x_6 + x_1x_2x_3x_5x_6 + x_1x_2x_4x_5x_7 + x_1x_4x_5x_6x_7 + x_2x_3x_4x_5x_6 + x_2x_3x_4x_5x_7 + x_2x_3x_4x_6x_7 + x_2x_4x_5x_6x_7 + x_0x_1x_2x_3x_4x_6 + x_0x_1x_2x_4x_6x_7 + x_0x_1x_2x_5x_6x_7 + x_0x_1x_3x_4x_6x_7 + x_0x_1x_3x_5x_6x_7 + x_0x_2x_3x_4x_6x_7 + x_0x_2x_3x_5x_6x_7 + x_0x_3x_4x_5x_6x_7 + x_1x_2x_3x_4x_6x_7 + x_1x_2x_3x_5x_6x_7 + x_1x_2x_4x_5x_6x_7 + x_0x_1x_2x_3x_4x_6x_7 + x_0x_1x_2x_3x_5x_6x_7$

$f_2(x_0, x_1, \ldots, x_7) = 1 + x_0 + x_1 + x_3 + x_0x_2 + x_1x_3 + x_1x_6 + x_2x_6 + x_3x_4 + x_3x_5 + x_4x_7 + x_0x_1x_2 + x_0x_1x_5 + x_0x_1x_6 + x_0x_2x_3 + x_0x_2x_4 + x_0x_2x_5 + x_0x_2x_6 + x_0x_3x_5 + x_0x_3x_6 + x_0x_3x_7 + x_0x_4x_5 + x_0x_4x_6 + x_0x_6x_7 + x_1x_2x_6 + x_1x_2x_7 + x_1x_3x_6 + x_1x_4x_5 + x_1x_4x_6 + x_1x_5x_6 + x_1x_5x_7 + x_1x_6x_7 + x_2x_3x_4 + x_2x_3x_5 + x_2x_4x_6 + x_2x_4x_7 + x_2x_5x_6 + x_2x_6x_7 + x_3x_5x_6 + x_3x_5x_7 + x_3x_6x_7 + x_4x_6x_7 + x_5x_6x_7 + x_0x_1x_2x_7 + x_0x_1x_3x_4 + x_0x_1x_3x_7 + x_0x_2x_3x_4 + x_0x_2x_3x_5 + x_0x_2x_3x_6 + x_0x_2x_4x_6 + x_0x_2x_6x_7 + x_0x_3x_4x_5 + x_0x_3x_4x_6 + x_0x_3x_4x_7 + x_0x_3x_5x_6 + x_0x_3x_6x_7 + x_0x_4x_6x_7 + x_1x_2x_3x_4 + x_1x_2x_4x_5 + x_1x_2x_4x_6 + x_1x_2x_6x_7 + x_1x_3x_4x_5 + x_1x_3x_4x_6 + x_1x_3x_6x_7 + x_1x_4x_5x_6 + x_2x_3x_4x_5 + x_2x_3x_5x_6 + x_2x_3x_5x_7 + x_2x_3x_6x_7 + x_2x_4x_5x_6 + x_2x_4x_5x_7 + x_2x_5x_6x_7 + x_3x_4x_5x_7 + x_3x_4x_6x_7 + x_4x_5x_6x_7 + x_0x_1x_2x_3x_5 + x_0x_1x_2x_3x_7 + x_0x_1x_2x_4x_5 + x_0x_1x_2x_6x_7 + x_0x_1x_3x_4x_6 + x_0x_1x_3x_6x_7 + x_0x_1x_4x_5x_7 + x_0x_1x_5x_6x_7 + x_0x_2x_3x_4x_7 + x_0x_2x_3x_5x_7 + x_0x_2x_3x_6x_7 + x_0x_2x_4x_5x_7 + x_0x_2x_4x_6x_7 + x_0x_3x_4x_5x_6 + x_0x_3x_4x_5x_7 + x_0x_4x_5x_6x_7 + x_1x_2x_3x_4x_5 + x_1x_2x_3x_4x_6 + x_1x_2x_3x_4x_7 + x_1x_2x_3x_5x_6 + x_1x_2x_3x_6x_7 + x_1x_3x_4x_5x_6 + x_1x_3x_5x_6x_7 + x_1x_4x_5x_6x_7 + x_2x_3x_5x_6x_7 + x_2x_4x_5x_6x_7 + x_3x_4x_5x_6x_7 + x_0x_1x_2x_3x_5x_6 + x_0x_1x_2x_3x_5x_7 + x_0x_1x_2x_4x_5x_6 + x_0x_1x_3x_4x_5x_6 + x_0x_1x_3x_4x_5x_7 + x_0x_2x_3x_5x_6x_7 + x_0x_2x_4x_5x_6x_7 + x_1x_2x_3x_5x_6x_7 + x_1x_2x_4x_5x_6x_7 + x_2x_3x_4x_5x_6x_7 + x_0x_1x_2x_3x_5x_6x_7 + x_0x_1x_2x_4x_5x_6x_7$

$f_3(x_0, x_1, \ldots, x_7) = x_2 + x_4 + x_5 + x_6 + x_7 + x_0x_1 + x_0x_2 + x_0x_4 + x_0x_7 + x_1x_3 + x_1x_6 + x_1x_7 + x_2x_3 + x_2x_4 + x_2x_5 + x_2x_6 + x_2x_7 + x_3x_4 + x_3x_6 + x_3x_7 + x_4x_5 + x_6x_7 + x_0x_1x_2 + x_0x_1x_3 + x_0x_1x_4 + x_0x_1x_5 + x_0x_1x_6 + x_0x_1x_7 + x_0x_2x_3 + x_0x_2x_4 + x_0x_2x_6 + x_0x_3x_4 + x_0x_3x_5 + x_0x_3x_7 + x_1x_2x_4 + x_1x_2x_5 + x_1x_3x_4 + x_1x_3x_7 + x_1x_4x_5 + x_1x_5x_7 + x_2x_3x_4 + x_2x_3x_5 + x_2x_3x_6 + x_2x_3x_7 + x_2x_4x_5 + x_2x_4x_6 + x_2x_4x_7 + x_3x_4x_5 + x_3x_4x_7 + x_3x_5x_6 + x_4x_5x_7 + x_0x_1x_2x_4 + x_0x_1x_3x_4 + x_0x_1x_3x_7 + x_0x_1x_4x_7 + x_0x_1x_5x_6 + x_0x_1x_6x_7 + x_0x_2x_3x_4 + x_0x_2x_4x_5 + x_0x_2x_4x_7 + x_0x_2x_5x_6 + x_0x_2x_5x_7 + x_0x_2x_6x_7 + x_0x_3x_4x_5 + x_0x_3x_4x_6 + x_0x_3x_4x_7 + x_0x_4x_6x_7 + x_1x_2x_3x_7 + x_1x_2x_4x_5 + x_1x_2x_4x_7 + x_1x_2x_6x_7 + x_1x_3x_4x_7 + x_1x_3x_5x_6 + x_1x_3x_6x_7 + x_1x_4x_6x_7 + x_1x_5x_6x_7 + x_2x_3x_4x_6 + x_2x_3x_4x_7 + x_2x_3x_6x_7 + x_2x_4x_5x_6 + x_2x_4x_6x_7 + x_3x_4x_5x_6 + x_3x_4x_5x_7 + x_0x_1x_2x_3x_4 + x_0x_1x_2x_3x_5 + x_0x_1x_2x_3x_6 + x_0x_1x_2x_3x_7 + x_0x_1x_2x_4x_6 + x_0x_1x_2x_4x_7 + x_0x_1x_2x_5x_6 + x_0x_1x_2x_6x_7 + x_0x_1x_3x_4x_5 + x_0x_1x_3x_4x_7 + x_0x_1x_3x_5x_7 + x_0x_1x_4x_6x_7 + x_0x_1x_5x_6x_7 + x_0x_2x_3x_4x_5 + x_0x_2x_3x_4x_6 + x_0x_2x_3x_5x_6 + x_0x_2x_4x_5x_6 + x_0x_2x_4x_5x_7 + x_0x_2x_4x_6x_7 + x_0x_2x_5x_6x_7 + x_0x_4x_5x_6x_7 + x_1x_2x_3x_4x_5 + x_1x_2x_3x_4x_6 + x_1x_2x_3x_5x_6 + x_1x_2x_3x_6x_7 + x_1x_2x_4x_5x_6 + x_1x_2x_4x_5x_7 + x_1x_3x_4x_5x_6 + x_1x_3x_4x_5x_7 + x_1x_3x_4x_6x_7 + x_1x_4x_5x_6x_7 + x_2x_3x_4x_5x_7 + x_2x_3x_5x_6x_7 + x_3x_4x_5x_6x_7 + x_0x_1x_2x_3x_4x_7 + x_0x_1x_2x_4x_5x_6 + x_0x_1x_3x_4x_5x_6 + x_0x_2x_3x_4x_5x_6 + x_0x_2x_3x_4x_5x_7 + x_0x_3x_4x_5x_6x_7 + x_1x_2x_3x_4x_5x_6 + x_1x_2x_3x_4x_6x_7 + x_1x_2x_3x_5x_6x_7 + x_1x_2x_4x_5x_6x_7 + x_2x_3x_4x_5x_6x_7 + x_0x_1x_2x_3x_4x_5x_7 + x_0x_1x_3x_4x_5x_6x_7 + x_0x_2x_3x_4x_5x_6x_7$

$f_4(x_0, x_1, \ldots, x_7) = x_0 + x_1 + x_3 + x_7 + x_0x_1 + x_0x_2 + x_0x_4 + x_0x_6 + x_0x_7 + x_1x_2 + x_1x_4 + x_2x_3 + x_4x_5 + x_4x_7 + x_5x_6 + x_0x_1x_2 + x_0x_2x_4 + x_0x_2x_5 + x_0x_3x_4 + x_0x_4x_5 + x_0x_4x_7 + x_0x_5x_6 + x_0x_5x_7 + x_0x_6x_7 + x_1x_2x_4 + x_1x_2x_6 + x_1x_3x_7 + x_1x_4x_7 + x_1x_5x_6 + x_1x_5x_7 + x_1x_6x_7 + x_2x_3x_5 + x_2x_3x_7 + x_2x_4x_5 + x_2x_5x_6 + x_2x_6x_7 + x_3x_4x_5 + x_3x_4x_6 + x_3x_4x_7 + x_3x_5x_7 + x_3x_6x_7 + x_4x_5x_6 + x_4x_5x_7 + x_4x_6x_7 + x_0x_1x_2x_3 + x_0x_1x_2x_4 + x_0x_1x_2x_6 + x_0x_1x_2x_7 + x_0x_1x_3x_4 + x_0x_1x_3x_5 + x_0x_1x_3x_6 + x_0x_1x_4x_7 + x_0x_1x_5x_6 + x_0x_1x_5x_7 + x_0x_1x_6x_7 + x_0x_2x_3x_5 + x_0x_2x_3x_6 + x_0x_2x_5x_6 + x_0x_3x_4x_6 + x_0x_3x_5x_6 + x_0x_3x_5x_7 + x_0x_4x_5x_6 + x_0x_4x_5x_7 + x_1x_2x_3x_4 + x_1x_2x_3x_5 + x_1x_2x_3x_6 + x_1x_2x_3x_7 + x_1x_2x_4x_6 + x_1x_2x_5x_7 + x_1x_2x_6x_7 + x_1x_3x_4x_6 + x_1x_3x_4x_7 + x_1x_3x_5x_7 + x_1x_3x_6x_7 + x_1x_4x_5x_6 + x_1x_4x_6x_7 + x_2x_3x_4x_5 + x_2x_3x_4x_6 + x_2x_3x_5x_6 + x_2x_4x_5x_6 + x_2x_5x_6x_7 + x_3x_5x_6x_7 + x_4x_5x_6x_7 + x_0x_1x_2x_3x_4 + x_0x_1x_2x_3x_5 + x_0x_1x_2x_3x_7 + x_0x_1x_2x_4x_6 + x_0x_1x_2x_4x_7 + x_0x_1x_2x_5x_6 + x_0x_1x_2x_5x_7 + x_0x_1x_2x_6x_7 + x_0x_1x_3x_4x_6 +$

Manuscript received April 5, 2007
Manuscript revised April 25, 2007

$x_0x_1x_3x_4x_7 + x_0x_1x_4x_5x_7 + x_0x_2x_3x_4x_5 + x_0x_2x_3x_4x_7 + x_0x_2x_3x_5x_7 + x_0x_2x_3x_6x_7 + x_0x_2x_4x_5x_7 + x_0x_2x_5x_6x_7 + x_0x_3x_4x_6x_7 + x_0x_3x_5x_6x_7 +$
$x_0x_4x_5x_6x_7 + x_1x_2x_3x_4x_5 + x_1x_2x_3x_4x_7 + x_1x_2x_3x_5x_6 + x_1x_2x_3x_5x_7 + x_1x_2x_4x_5x_6 + x_1x_2x_4x_5x_7 + x_1x_2x_4x_6x_7 + x_1x_2x_5x_6x_7 + x_1x_3x_4x_6x_7 +$
$x_2x_3x_4x_5x_7 + x_2x_3x_4x_6x_7 + x_2x_4x_5x_6x_7 + x_3x_4x_5x_6x_7 + x_0x_1x_2x_3x_4x_5 + x_0x_1x_2x_3x_5x_6 + x_0x_1x_2x_4x_5x_6 + x_0x_1x_2x_4x_5x_7 + x_0x_1x_3x_4x_5x_6 +$
$x_0x_1x_3x_4x_5x_7 + x_0x_1x_3x_5x_6x_7 + x_0x_1x_4x_5x_6x_7 + x_0x_2x_3x_4x_5x_6 + x_0x_2x_3x_4x_6x_7 + x_0x_2x_3x_5x_6x_7 + x_0x_2x_4x_5x_6x_7 + x_0x_3x_4x_5x_6x_7 +$
$x_1x_2x_3x_4x_5x_6 + x_1x_2x_4x_5x_6x_7 + x_1x_3x_4x_5x_6x_7 + x_0x_1x_2x_3x_4x_5x_7 + x_0x_1x_2x_3x_4x_6x_7 + x_0x_1x_2x_4x_5x_6x_7 + x_1x_2x_3x_4x_5x_6x_7$

$f_5(x_0, x_1, \ldots, x_7) = x_0 + x_2 + x_6 + x_7 + x_0x_1 + x_0x_3 + x_0x_7 + x_1x_2 + x_1x_7 + x_2x_7 + x_3x_4 + x_3x_6 + x_3x_7 + x_4x_5 + x_4x_7 + x_5x_7 + x_0x_1x_3 +$
$x_0x_1x_5 + x_0x_1x_7 + x_0x_2x_6 + x_0x_2x_7 + x_0x_3x_6 + x_0x_3x_7 + x_0x_4x_5 + x_0x_4x_6 + x_0x_4x_7 + x_0x_5x_6 + x_0x_5x_7 + x_0x_6x_7 + x_1x_2x_4 + x_1x_2x_6 + x_1x_3x_4$
$+ x_1x_3x_7 + x_1x_4x_5 + x_1x_4x_7 + x_1x_5x_6 + x_1x_6x_7 + x_2x_3x_4 + x_2x_3x_5 + x_2x_3x_6 + x_2x_3x_7 + x_2x_4x_6 + x_2x_4x_7 + x_2x_5x_6 + x_2x_5x_7 + x_2x_6x_7 +$
$x_3x_4x_5 + x_3x_4x_6 + x_3x_4x_7 + x_3x_5x_6 + x_3x_5x_7 + x_3x_6x_7 + x_4x_5x_7 + x_4x_6x_7 + x_0x_1x_2x_3 + x_0x_1x_2x_4 + x_0x_1x_2x_5 + x_0x_1x_2x_6 + x_0x_1x_3x_5 +$
$x_0x_1x_4x_6 + x_0x_1x_5x_6 + x_0x_2x_3x_5 + x_0x_2x_3x_6 + x_0x_2x_4x_6 + x_0x_2x_4x_7 + x_0x_2x_5x_6 + x_0x_3x_4x_5 + x_0x_3x_4x_7 + x_0x_3x_5x_6 + x_0x_3x_5x_7 + x_0x_4x_5x_7 +$
$x_0x_5x_6x_7 + x_1x_2x_3x_4 + x_1x_2x_3x_5 + x_1x_2x_4x_5 + x_1x_2x_4x_7 + x_1x_2x_6x_7 + x_1x_3x_4x_5 + x_1x_3x_4x_7 + x_1x_3x_5x_7 + x_1x_3x_6x_7 + x_1x_4x_5x_6 + x_1x_4x_6x_7 +$
$x_2x_4x_5x_6 + x_2x_4x_5x_7 + x_2x_4x_6x_7 + x_2x_5x_6x_7 + x_3x_4x_5x_6 + x_3x_4x_5x_7 + x_3x_4x_6x_7 + x_0x_1x_2x_3x_4 + x_0x_1x_2x_3x_6 + x_0x_1x_2x_3x_7 + x_0x_1x_2x_4x_5 +$
$x_0x_1x_2x_4x_6 + x_0x_1x_2x_4x_7 + x_0x_1x_2x_5x_7 + x_0x_1x_3x_4x_5 + x_0x_1x_3x_4x_6 + x_0x_1x_3x_5x_6 + x_0x_1x_3x_6x_7 + x_0x_1x_4x_5x_6 + x_0x_1x_4x_6x_7 + x_0x_1x_5x_6x_7 +$
$x_0x_2x_3x_4x_7 + x_0x_2x_3x_5x_6 + x_0x_2x_4x_5x_7 + x_0x_2x_4x_6x_7 + x_0x_2x_5x_6x_7 + x_0x_3x_4x_5x_7 + x_0x_3x_4x_6x_7 + x_0x_3x_5x_6x_7 + x_0x_4x_5x_6x_7 + x_1x_2x_3x_4x_6 +$
$x_1x_2x_3x_5x_6 + x_1x_2x_3x_5x_7 + x_1x_3x_4x_5x_6 + x_1x_3x_4x_6x_7 + x_1x_4x_5x_6x_7 + x_2x_3x_4x_5x_6 + x_2x_3x_5x_6x_7 + x_2x_4x_5x_6x_7 + x_3x_4x_5x_6x_7 + x_0x_1x_2x_3x_4x_5 +$
$x_0x_1x_2x_3x_5x_7 + x_0x_1x_2x_3x_6x_7 + x_0x_1x_2x_4x_5x_7 + x_0x_1x_2x_4x_6x_7 + x_0x_1x_3x_4x_5x_6 + x_0x_1x_3x_4x_5x_7 + x_0x_1x_3x_4x_6x_7 + x_0x_2x_3x_4x_5x_6 +$
$x_0x_2x_3x_4x_6x_7 + x_0x_2x_3x_5x_6x_7 + x_0x_3x_4x_5x_6x_7 + x_1x_2x_3x_4x_5x_7 + x_1x_2x_3x_5x_6x_7 + x_1x_2x_4x_5x_6x_7 + x_1x_3x_4x_5x_6x_7 + x_2x_3x_4x_5x_6x_7 +$
$x_0x_1x_2x_3x_4x_5x_6 + x_0x_1x_2x_3x_4x_5x_7 + x_0x_1x_2x_3x_4x_6x_7 + x_0x_1x_2x_4x_5x_6x_7 + x_0x_1x_3x_4x_5x_6x_7$

$f_6(x_0, x_1, \ldots, x_7) = 1 + x_0 + x_1 + x_4 + x_7 + x_0x_4 + x_0x_5 + x_0x_6 + x_0x_7 + x_1x_3 + x_1x_5 + x_2x_3 + x_3x_6 + x_3x_7 + x_4x_5 + x_4x_6 + x_4x_7 + x_5x_7 +$
$x_6x_7 + x_0x_1x_2 + x_0x_1x_4 + x_0x_1x_5 + x_0x_1x_7 + x_0x_2x_4 + x_0x_2x_5 + x_0x_2x_7 + x_0x_3x_4 + x_0x_3x_6 + x_0x_3x_7 + x_0x_4x_6 + x_0x_5x_7 + x_1x_2x_4 + x_1x_2x_6 +$
$x_1x_3x_4 + x_1x_3x_7 + x_1x_4x_5 + x_1x_4x_6 + x_1x_6x_7 + x_2x_3x_7 + x_2x_4x_5 + x_2x_4x_6 + x_3x_4x_5 + x_3x_4x_6 + x_3x_4x_7 + x_3x_5x_6 + x_3x_6x_7 + x_4x_5x_6 + x_4x_5x_7$
$+ x_4x_6x_7 + x_0x_1x_3x_4 + x_0x_1x_3x_5 + x_0x_1x_3x_6 + x_0x_1x_3x_7 + x_0x_1x_5x_6 + x_0x_2x_3x_4 + x_0x_2x_3x_5 + x_0x_2x_3x_6 + x_0x_2x_3x_7 + x_0x_2x_4x_5 + x_0x_2x_4x_6 +$
$x_0x_2x_5x_6 + x_0x_3x_4x_5 + x_0x_3x_4x_6 + x_0x_3x_5x_6 + x_0x_3x_6x_7 + x_1x_2x_3x_4 + x_1x_2x_3x_7 + x_1x_2x_4x_6 + x_1x_2x_5x_6 + x_1x_2x_5x_7 + x_1x_2x_6x_7 + x_1x_3x_4x_7 +$
$x_1x_4x_5x_7 + x_1x_4x_6x_7 + x_1x_5x_6x_7 + x_2x_3x_4x_5 + x_2x_3x_4x_6 + x_2x_3x_4x_7 + x_2x_3x_5x_7 + x_2x_3x_6x_7 + x_2x_4x_5x_6 + x_3x_4x_6x_7 + x_0x_1x_2x_3x_5 +$
$x_0x_1x_2x_3x_7 + x_0x_1x_2x_4x_7 + x_0x_1x_2x_5x_7 + x_0x_1x_2x_6x_7 + x_0x_1x_3x_4x_5 + x_0x_1x_4x_5x_7 + x_0x_1x_4x_6x_7 + x_0x_1x_5x_6x_7 + x_0x_2x_3x_4x_5 + x_0x_2x_3x_4x_6 +$
$x_0x_2x_3x_4x_7 + x_0x_2x_4x_5x_7 + x_0x_2x_4x_6x_7 + x_0x_3x_4x_5x_7 + x_0x_3x_4x_6x_7 + x_0x_3x_5x_6x_7 + x_0x_4x_5x_6x_7 + x_1x_2x_3x_4x_6 + x_1x_2x_3x_5x_6 + x_1x_2x_3x_5x_7 +$
$x_1x_2x_3x_6x_7 + x_1x_2x_4x_5x_6 + x_1x_2x_4x_6x_7 + x_1x_3x_4x_5x_6 + x_1x_3x_4x_6x_7 + x_1x_3x_5x_6x_7 + x_1x_4x_5x_6x_7 + x_2x_3x_4x_5x_7 + x_2x_3x_5x_6x_7 + x_3x_4x_5x_6x_7 +$
$x_0x_1x_2x_3x_4x_6 + x_0x_1x_2x_4x_5x_6 + x_0x_1x_2x_4x_5x_7 + x_0x_1x_2x_5x_6x_7 + x_0x_1x_3x_4x_5x_6 + x_0x_1x_3x_4x_5x_7 + x_0x_1x_3x_4x_6x_7 + x_0x_1x_3x_5x_6x_7 +$
$x_0x_2x_3x_4x_5x_7 + x_0x_2x_3x_5x_6x_7 + x_0x_3x_4x_5x_6x_7 + x_1x_2x_3x_4x_5x_6 + x_1x_2x_3x_4x_5x_7 + x_1x_2x_3x_4x_6x_7 + x_1x_2x_4x_5x_6x_7 + x_1x_3x_4x_5x_6x_7 +$
$x_0x_1x_2x_3x_4x_6x_7 + x_0x_1x_2x_3x_5x_6x_7 + x_0x_1x_2x_4x_5x_6x_7 + x_0x_1x_3x_4x_5x_6x_7$

$f_7(x_0, x_1, \ldots, x_7) = 1 + x_3 + x_4 + x_5 + x_7 + x_0x_1 + x_0x_2 + x_0x_5 + x_1x_2 + x_1x_3 + x_1x_5 + x_1x_6 + x_1x_7 + x_2x_7 + x_3x_5 + x_3x_6 + x_3x_7 + x_4x_5 +$
$x_4x_6 + x_5x_6 + x_6x_7 + x_0x_1x_2 + x_0x_1x_4 + x_0x_1x_5 + x_0x_2x_4 + x_0x_2x_5 + x_0x_3x_4 + x_0x_3x_5 + x_0x_3x_7 + x_0x_4x_5 + x_0x_4x_6 + x_0x_5x_7 + x_0x_6x_7 +$
$x_1x_2x_3 + x_1x_2x_5 + x_1x_2x_6 + x_1x_3x_6 + x_1x_3x_7 + x_1x_4x_7 + x_1x_5x_6 + x_1x_5x_7 + x_1x_6x_7 + x_2x_4x_5 + x_2x_4x_7 + x_2x_5x_7 + x_3x_4x_6 + x_3x_4x_7 + x_3x_5x_6$
$+ x_3x_5x_7 + x_3x_6x_7 + x_4x_5x_6 + x_0x_1x_2x_4 + x_0x_1x_2x_6 + x_0x_1x_3x_7 + x_0x_1x_4x_5 + x_0x_1x_4x_6 + x_0x_1x_5x_6 + x_0x_2x_3x_5 + x_0x_2x_3x_6 + x_0x_2x_3x_7 +$
$x_0x_2x_4x_5 + x_0x_2x_5x_7 + x_0x_3x_5x_6 + x_0x_3x_5x_7 + x_0x_3x_6x_7 + x_0x_4x_5x_6 + x_0x_4x_5x_7 + x_0x_5x_6x_7 + x_1x_2x_3x_5 + x_1x_2x_3x_6 + x_1x_2x_3x_7 + x_1x_2x_4x_7 +$
$x_1x_2x_5x_7 + x_1x_3x_4x_6 + x_1x_3x_4x_7 + x_1x_3x_5x_6 + x_1x_4x_5x_6 + x_1x_5x_6x_7 + x_2x_3x_5x_7 + x_2x_3x_6x_7 + x_2x_4x_5x_6 + x_2x_4x_5x_7 + x_2x_5x_6x_7 + x_4x_5x_6x_7 +$
$x_0x_1x_2x_3x_5 + x_0x_1x_2x_4x_7 + x_0x_1x_2x_5x_6 + x_0x_1x_2x_6x_7 + x_0x_1x_3x_4x_7 + x_0x_1x_3x_5x_6 + x_0x_1x_3x_5x_7 + x_0x_1x_4x_6x_7 + x_0x_1x_5x_6x_7 + x_0x_2x_3x_4x_5 +$
$x_0x_2x_3x_4x_6 + x_0x_2x_3x_4x_7 + x_0x_2x_3x_5x_7 + x_0x_2x_4x_5x_6 + x_0x_2x_4x_6x_7 + x_0x_2x_5x_6x_7 + x_0x_3x_5x_6x_7 + x_0x_4x_5x_6x_7 + x_1x_2x_3x_4x_5 + x_1x_2x_3x_4x_7 +$
$x_1x_2x_3x_5x_6 + x_1x_2x_3x_6x_7 + x_1x_2x_4x_5x_6 + x_1x_2x_4x_6x_7 + x_1x_3x_4x_5x_7 + x_1x_3x_4x_6x_7 + x_1x_4x_5x_6x_7 + x_2x_3x_4x_5x_7 + x_2x_3x_5x_6x_7 + x_3x_4x_5x_6x_7 +$
$x_0x_1x_2x_3x_4x_5 + x_0x_1x_2x_3x_4x_6 + x_0x_1x_2x_3x_5x_6 + x_0x_1x_2x_3x_6x_7 + x_0x_1x_2x_4x_5x_7 + x_0x_1x_2x_5x_6x_7 + x_0x_1x_3x_4x_5x_6 + x_0x_1x_3x_4x_6x_7 +$
$x_0x_1x_4x_5x_6x_7 + x_0x_2x_3x_4x_5x_7 + x_0x_2x_3x_5x_6x_7 + x_0x_2x_4x_5x_6x_7 + x_0x_3x_4x_5x_6x_7 + x_1x_2x_3x_4x_5x_7 + x_1x_3x_4x_5x_6x_7 + x_0x_1x_2x_3x_4x_5x_7 +$
$x_0x_1x_2x_3x_5x_6x_7 + x_0x_1x_3x_4x_5x_6x_7$