Correlation Immunity of Boolean Functions: An Evolutionary Algorithms Perspective

Stjepan Picek
Faculty of Electrical Engineering and Computing
University of Zagreb, Croatia
stjepan@computer.org

Claude Carlet
LAGA, UMR 7539, CNRS
University of Paris XIII and University of Paris VIII, France
Claude.Carlet@univ-paris8.fr

Domagoj Jakobovic
Faculty of Electrical Engineering and Computing
University of Zagreb, Croatia
domagoj.jakobovic@fer.hr

Julian F. Miller
Department of Electronics
University of York, UK
julian.miller@york.ac.uk

Lejla Batina
Radboud University
Nijmegen, The Netherlands
lejla@cs.ru.nl *

ABSTRACT

Boolean functions are essential in many stream ciphers. When used in combiner generators, they need to have sufficiently high values of correlation immunity, alongside other properties. In addition, correlation immune functions with small Hamming weight reduce the cost of masking countermeasures against side-channel attacks. Various papers have examined the applicability of evolutionary algorithms for evolving cryptographic Boolean functions. However, even when authors considered correlation immunity, it was not given the highest priority. Here, we examine the effectiveness of three different EAs, namely, Genetic Algorithms, Genetic Programming (GP) and Cartesian GP for evolving correlation immune Boolean functions. Besides the properties of balancedness and correlation immunity, we consider several other relevant cryptographic properties while maintaining the optimal trade-offs among them. We show that evolving correlation immune Boolean functions is an even harder objective than maximizing nonlinearity.

Categories and Subject Descriptors

I.2.8 [**Computing Methodologies**]: Artificial Intelligence—*Problem Solving, Control Methods, and Search*

Keywords

Boolean Functions; Cryptography; Cartesian Genetic Programming; Genetic Programming

*This work was supported in part by the Technology Foundation STW (project 12624 - SIDES), The Netherlands Organization for Scientific Research NWO (project ProFIL 628.001.007) and the ICT COST action IC1204 TRUDEVICE.

GECCO '15, July 11 - 15, 2015, Madrid, Spain

DOI: http://dx.doi.org/10.1145/2739480.2754764

1. INTRODUCTION

Evolutionary algorithms (EAs) found their place in many real-world applications where they offer a good balance between the ease of use and results. In particular, one area where EAs have shown good results is the generation of Boolean functions that are suitable for cryptography. However, since there are many possible design criteria for Boolean functions in cryptography, this domain is still wide open for research. In the rest of this paper, when we discuss Boolean functions suitable for cryptography, we primarily mean Boolean functions suitable to be used in combiner generators. In this paper, we are concentrating on the correlation immunity property of Boolean functions. To date, this property has not been the primary concern when evolving Boolean functions and although there exist papers considering this property, they are mainly concerned with other properties. Furthermore, correlation immunity is given a limited span of desired values (i.e. correlation immunity of first or second order) and only for balanced Boolean functions.

When discussing methods of generating Boolean functions with appropriate cryptographic properties, they can be divided into three groups: algebraic constructions, random generation and heuristics (and of course, different combinations between those methods). Among various heuristic approaches, EAs offer highly competitive results when generating Boolean functions for cryptography. This is discussed in Section 3. In this paper, we use EAs to experiment with two very different usages of Boolean functions, but where in both applications correlation immunity property has high importance.

The first application explores the evolution of Boolean functions appropriate for usage in combiner generators. The second deals with the correlation immune Boolean functions that have a small Hamming weight. Such functions have application in side-channel countermeasures [2]. Although we explore only those two applications, correlation immune functions have more utilizations. For instance, they are also directly related to those designs in mathematics called orthogonal arrays [4].

1.1 Our Contribution

Our first contribution is an extensive analysis of the difficulty of obtaining balanced Boolean functions suitable for cryptography that have various values of correlation immunity property. The second contribution is, as far as the authors know, the first application of EAs to find Boolean functions with minimal Hamming weight and different correlation immunity values. Both contributions have real-world applications.

The rest of this paper is organized as follows. In Section 2, we give necessary information about Boolean functions, their properties and trade-offs between some of those properties. Next, Section 3 enumerates some of the relevant work. In Section 4, we present experimental setup, Section 5 gives results, short discussion and some guidelines for the future work. Finally, Section 6 offers a summary of our work.

2. BACKGROUND AND MOTIVATION

Notation.

The inner product of two vectors $\vec{a}$ and $\vec{b}$ is denoted as $\vec{a}\cdot\vec{b}$ and equals $\vec{a}\cdot\vec{b} = \oplus_{i=1}^{n} a_i b_i$. Here, " $\oplus$ " represents bitwise XOR. The set of all n-tuples of elements in the field $\mathbb{F}_2$ is denoted as $\mathbb{F}_2^n$ where $\mathbb{F}_2$ is the Galois field with 2 elements. An (n, m)-function (also called vectorial Boolean function) is any mapping F from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$. If m equals 1 then the function f is called a Boolean function. The Hamming weight (HW) of a vector $\vec{a}$, where $\vec{a} \in \mathbb{F}_2^n$, is the number of nonzero positions in the vector. The support *supp* of a Boolean function f is given by [5]:

$$supp(f) = x \in \mathbb{F}_2^n | f(x) = 1. \tag{1}$$

The Hamming weight of a Boolean function f on $\mathbb{F}_2^n$ is the size (cardinality) of its support.

Representations.

A Boolean function f on $\mathbb{F}_2^n$ can be uniquely represented by a truth table (TT), which is a vector (e.g. $f(\vec{0}), ..., f(\vec{1})$) that contains the function values of f, ordered lexicographically, i.e. $\vec{a} \leq \vec{b}$ [5].

A second unique representation of a Boolean function f is the Walsh transform W_f. It represents the correlation between $f(\vec{x})$ and the linear function $\vec{a}\cdot\vec{x}$ [5]. The Walsh transform of a Boolean function f equals:

$$W_f(\vec{a}) = \sum_{\vec{x}\in\mathbb{F}_2^n} (-1)^{f(\vec{x})\oplus\vec{a}\cdot\vec{x}}. \tag{2}$$

A third unique representation of a Boolean function f on $\mathbb{F}_2^n$ is a polynomial in $\mathbb{F}_2[x_0, ..., x_{n-1}]/(x_0^2 - x_0, ..., x_{n-1}^2 - x_{n-1})$. This form is called ANF and it is a multivariate polynomial P defined as [5]:

$$P(\vec{x}) = \oplus_{\vec{a}\in\mathbb{F}_2^n} h(\vec{a})\vec{x}^{\vec{a}}, \tag{3}$$

where $h(\vec{a})$ is defined by the Möbius inversion principle [5].

The Properties.

A Boolean function is balanced if its Hamming weight (HW) is equal to 2^{n-1} [5]. Boolean functions need to be balanced; otherwise (for sufficiently large n) it would be possible to distinguish between the keystream output by the function and some random sequence when using filter and combiner generators [5].

The nonlinearity N_f of a Boolean function f can be expressed in terms of the Walsh coefficients as [5]:

$$N_f = 2^{n-1} - \frac{1}{2} max_{\vec{a}\in\mathbb{F}_2^n} |W_f(\vec{a})|. \tag{4}$$

A Boolean function needs to have high nonlinearity value in order to resist fast correlation attacks [5]. By high value we mean a value that is near the covering radius bound $2^{n-1} - 2^{\frac{n}{2}-1}$.

The algebraic degree *deg* of a Boolean function f is defined as the number of variables in the largest product term of the functions' ANF having a non-zero coefficient. A Boolean function needs to have high enough value of algebraic degree to resist Berlekamp-Massey attack [5].

For a Boolean function f, we say it is correlation immune of order t (in brief, $CI(t)$) if the output of the function is statistically independent of the combination of any t of its inputs [5]. A characterization exists using the Walsh spectrum and it equals:

$$W_f(\vec{a}) = 0, \text{ for } 0 \leq HW(\vec{a}) \leq t. \tag{5}$$

A high value of correlation immunity property is necessary for resisting Siegenthaler correlation attack when Boolean functions are used in combiner generators [28].

A Boolean function f is t-resilient if it is balanced and with correlation immunity of degree t [5].

The algebraic immunity AI of a Boolean function f is the lowest degree of a nonzero function g from $\mathbb{F}_2^n$ into $\mathbb{F}_2$ for which $f\cdot g = \vec{0}$ or $(f \oplus \vec{1})\cdot g = \vec{0}$ where f and g are Boolean functions [5]. Boolean functions need to have high values of algebraic immunity in order to resist algebraic attacks [15].

Bounds and Trade-offs.

Sarkar and Maitra showed that if a CI(t) Boolean function f has an even number of inputs n and $k \leq \frac{n}{2} - 1$ then its nonlinearity N_f is upper bounded as follows [27]:

$$N_f \leq 2^{n-1} - 2^{\frac{n}{2}-1} - 2^k, \tag{6}$$

where k equals $t+1$ if f is balanced or has Hamming weight divisible by 2^{t+1} and k equals t otherwise.

In the case when $k > \frac{n}{2} - 1$ then the nonlinearity is upper bounded with:

$$N_f \leq 2^{n-1} - 2^k. \tag{7}$$

The correlation immunity $CI(t)$ and the algebraic degree *deg* are conflicting properties and it is not possible to obtain a function with both properties optimal. For a resilient function where $t > 1$ and $t \neq n - 1$, the following Siegenthaler inequality holds [28]:

$$t \leq n - deg - 1. \tag{8}$$

Otherwise, the inequality is as follows:

$$t \leq n - deg. \tag{9}$$

The algebraic immunity is upper bounded by the number of input variables, more precisely [14]:

$$AI \leq \lceil \frac{n}{2} \rceil. \tag{10}$$

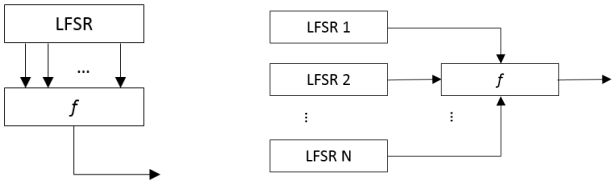
The next inequality shows a connection between the algebraic immunity and the algebraic degree properties:

$$AI \leq deg. \tag{11}$$

2.1 Motivation and Objectives

Symmetric key cryptography can be divided into block and stream ciphers [18]. In both categories, nonlinear elements play an important role by adding confusion to the algorithm. Informally speaking, without some nonlinear element, a cryptographic algorithm would be easy to break. Boolean functions are considered the best known nonlinear elements and they are used in stream ciphers, while vectorial Boolean functions (commonly known as Substitution boxes or S-boxes) are used in block ciphers.

Being more specific, many stream ciphers are realized as the combination by a Boolean function of some bits extracted from one or several Linear Feedback Shift Registers (LFSR), where LFSR is a linear construction used for generating a sequence of binary bits. Two often explored models that use an LFSR together with a nonlinear transformation are the so-called combiner and the filter generator models [5]. A combiner generator uses several LFSRs in parallel and then combines their output in a nonlinear way. In a filter generator, the output is obtained by a nonlinear function of some taps of one longer LFSR [5]. Both models are depicted in Figures 1a and 1b.



(a) Filter generator. (b) Combiner generator.

Figure 1: Principles of filter and combiner generators.

In general, for a Boolean function to be useful in such framework it needs to be balanced, with high nonlinearity, high algebraic immunity and a high algebraic degree. Moreover, to be used in a combiner generator it also needs to have a high value of correlation immunity [5, 23, 28]. However, if the correlation immunity is high, it is difficult to find high algebraic immunity, which has resulted in a decreased use of combiner generators.

For our **first** fitness objective, we are interested in Boolean functions that are appropriate for usage in combiner generators. Furthermore, we concentrate only on Boolean functions with eight inputs that have various values of correlation immunity property. As evident from the Section 3, EAs can be also used to evolve Boolean functions of larger size. However, we decided to experiment with only one input size in order to better answer the question of the difficulty of evolving Boolean functions with different correlation immunity values. Eight inputs is a relatively small size for Boolean functions for stream ciphers (it is commonly considered that 13 is a strict minimum for resisting algebraic attacks), but we still believe it presents an interesting case. Indeed, for instance ciphers RAKAPOSHI [12] and Achterbahn [17] use Boolean functions of that size.

In addition, we are interested in Boolean functions that can help resist side-channel attacks. Side-channel attacks do not rely on the security of the underlying algorithm, but rather on the implementation of the algorithm in a device. Implementations running on small devices often provide an adversary with unintentional output channels, often called side channels. On some occasions, these types of information leakages can be linked either to the types of operations that the cipher is performing, or to the data. This makes the leakages explorable by the adversary trying to extract the secret key as they are always looking for shortcuts in cryptanalysis [22]. One class of countermeasures against side-channel attacks are masking schemes. In masking schemes one randomizes the intermediate values that are processed by the cryptographic device. One obvious drawback of such an approach is the masking overhead which can be substantial in embedded devices or smart cards.

However, correlation immune Boolean functions can reduce the masking overhead in two ways [6, 9]. The first way is by applying leakage squeezing method [8, 9] and the other method uses Rotating S-box masking [6]. For the second case the key is to generate Boolean functions with as high as possible correlation immunity value and minimal Hamming weight of their support. In the first case, there is a constraint that f is the indicator of a permutation of $\mathbb{F}_2^{\frac{n}{2}}$.

In accordance to that, evolution of Boolean functions with minimal Hamming weight is the goal of our **second** objective. We note here that most of algebraic constructions are not a viable choice for this objective since they are designed to generate balanced Boolean functions. Since this is the first time EAs are to be used to evolve Boolean functions with such properties, we again work with eight input Boolean functions.

We conclude the case for our motivation with the citation from [5], page 97: "But designing constructions leading to large numbers of functions achieving good trade-offs between the nonlinearity, the algebraic degree and the resiliency order (if possible, on any numbers of variables) are still necessary for permitting to choose in applications cryptographic functions satisfying specific constraints."

3. RELATED WORK

In this section, we first briefly discuss applications of EAs to the evolution of Boolean functions where correlation immunity is a property of interest.

Millan et al. use GAs to evolve Boolean functions that have high nonlinearity [24]. In conjunction with the GA they use hill-climbing and a resetting step in order to find Boolean functions with even higher nonlinearity. They experiment with Boolean function sizes of up to 12 inputs and they find balanced eight input Boolean functions with nonlinearity 112 and correlation immunity equal to one.

Clark et al. use simulated annealing (SA) to generate Boolean functions with cryptographically relevant properties [13]. In their work they consider balanced function with high nonlinearity and with the correlation immunity less and equal to two.

Burnett et al. present two heuristic methods where the goal of the first method is generating balanced Boolean functions with high nonlinearity and low autocorrelation. The second method aims at generating resilient functions with high nonlinearity and algebraic degree that maximizes Siegenthaler inequality [3].

McLaughlin and Clark on the other hand use SA to generate Boolean functions that have optimal values of algebraic immunity, fast algebraic resistance and algebraic degree [23].

In their work they experiment with Boolean functions of size up to 16 inputs.

Picek et al. experiment with GA and GP to find Boolean functions that possess several optimal properties among which is correlation immunity [25]. As far as the authors know, this is the first application of GP when evolving cryptographically suitable Boolean functions.

When discussing algebraic constructions and bounds there exists a plethora of results on correlation immune Boolean functions. We give here only a subset of relevant works.

Tarannikov suggests a method to construct n-variable and t-resilient with maximal possible nonlinearity for $\frac{2n-7}{3} \leq m \leq n - \log_2 \frac{n-2}{3} - 2$ [29].

Carlet and Sarkar show the necessary condition on the ANF of correlation immune Boolean functions that reach maximal possible nonlinearity [10].

Maitra and Sarkar present constructions able to generate n-variable and t-resilient Boolean functions that maximize Siegenthaler inequality [21, 26].

Le Bars and Viola present a complete characterization of the first order correlation immune Boolean functions of up to seven inputs [20].

Carrasco et al. use the approach from [20] to develop a combinatorial method able to randomly generate first order correlation immune functions [11]. With that method they are able to generate eight input Boolean function with correlation immunity equal to one in slightly more than five seconds.

4. EXPERIMENTAL SETUP

In this section, we briefly discuss the algorithms we use and their parameters. We conduct a short two parameter sweep in order to investigate the best combination of parameters for each of the algorithms. Since there are more than two parameters for each algorithm, we try to consider only the most obvious choices for the tuning.

4.1 Representations and Common Parameters

The GA represents the individuals as strings of bits where values are truth tables of Boolean functions. GP uses a representation where individuals are trees of Boolean primitives which are then evaluated according to the truth table they produce. CGP solutions are directed graphs with Boolean primitives as nodes, that are also evaluated using the truth table they produce.

The following parameters of the experiments are common for every objective. The size of Boolean function is eight (the size of the corresponding truth table is 256) and the number of independent trials M for each experiment is 50 (the mean of best solutions from 50 runs is reported if not stated otherwise). Stopping criterion for all algorithms is 500 000 evaluations. Common parameters are also given in Table 1.

Table 1: Common parameters for EAs.

Parameter	Parameter value
Boolean function input size n	8
Number of experiments	50
Stopping criterion	500 000 evaluations

4.2 Genetic Programming

The function set for GP in all experiments is OR, XOR, AND, XNOR and AND with one input inverted. Terminals correspond to n Boolean variables. Boolean functions may be represented with only XOR and AND operators, but it is quite easy to transform it from one notation to the other. A small number of experiments were also conducted to select the appropriate operators, and based on that we used a simple tree crossover with 90% bias for functional nodes and a subtree mutation. We experiment with tree depth sizes of 5, 7, 8, 9 and 11 and population sizes of 100, 200, 500, 1 000 and 2 000.

4.3 Cartesian Genetic Programming

The function set n_f for the CGP is the same as for the GP. Setting the number of rows to be 1 and levels-back parameter to be equal to the number of columns is regarded as the best and most general choice [1]. We experiment with genotype sizes of 300, 700, 1 100, 1 500 and 1 900 and for mutation rates with values of 1%, 4%, 7%, 10% and 13%.

The number of input connections n_n for each node is two and the number of program output connections n_o is one. The population size for CGP equals five in all our experiments. For CGP individual selection we use a (1 + 4)-ES in which offspring are favored over parents when they have a fitness better than or equal to the fitness of the parent. The mutation operator is one-point mutation where the mutation point is chosen with a fixed probability. The number of genes mutated is defined as fixed percentage of the total number of genes. Note, the single output gene is not mutated and is taken from the last node in the genotype. The genes chosen for mutation might be a node representing an input connection or a function.

4.4 Genetic Algorithm

We use a simple GA with elimination tournament selection with size 3 [16]. A mutation is selected uniformly at random between a simple mutation, where a single bit is inverted, and a mixed mutation, which randomly shuffles the bits in a randomly selected subset. The crossover operators are one-point and uniform crossover, performed uniformly at random for each new offspring. For each of the fitness functions we experiment with population sizes of 50, 100, 200, 500 and 1 000 and mutation probabilities of 0.1, 0.3, 0.5, 0.7 and 0.9.

4.5 Fitness Functions

As discussed previously, we evolve Boolean functions for two different settings. In the first setting, we want to evolve balanced Boolean functions that have good correlation immunity values and that maximize the Siegenthaler inequality. First, we need to establish what are good values for correlation immunity of Boolean functions with eight inputs.

When a Boolean function is used in a filter generator, correlation immunity value of one is usually considered to be sufficient. To obtain this value, often it is possible to replace a Boolean function that has other good properties with a linearly equivalent function that has the correlation immunity equal to one [7].

However, when using Boolean functions in a combiner generator, high values for correlation immunity are necessary [28]. Therefore, in our experiments, we look for Boolean functions that have the correlation immunity *equal or larger*

than two which we believe is sufficient for a Boolean function with eight inputs. We stop with a correlation immunity value of five in order to leave high enough value of the algebraic degree. Furthermore, such Boolean functions need to have the maximal nonlinearity value according to the Sarkar-Maitra divisibility bound. Therefore, for each of the target correlation immunity values t, we know what are the maximal values that the other properties can reach. In accordance to that, we can identify the global optimum for each target correlation immunity value.

We aim to *maximize* the following fitness function:

$$fitness_1 = BAL + t + \delta_{t,x}(N_f + AI + deg), \qquad (12)$$

where $\delta_{t,x}$ is the Kronecker delta function that takes value one when $t = x$ and zero otherwise. Here, x is defined as the target correlation immunity value in the range $[2, 5]$.

The balancedness property (BAL) is calculated in the following way; if a function is balanced then we give it a value of zero, otherwise we give it a negative value that amounts to the absolute difference of the number of zeros and ones in the truth table. A two stage fitness is used here in which a fitness bonus equal to the sum of the nonlinearity, algebraic immunity AI and algebraic degree deg is awarded only to a solution that is t-resilient; otherwise, the fitness is only the sum of the balancedness penalty and the correlation immunity.

In the second setting, we look for *unbalanced* Boolean functions that have correlation immunity values t in the range $[1, 5]$ and minimal Hamming weight of their truth table representation (therefore, minimal cardinality of their support). These conditions are presented in the following equation where the goal is *maximization*:

$$fitness_2 = \begin{cases} t + \delta_{t,x}(2^n - HW(TT)), \text{if } t = x \\ (2^n - HW(TT) - 2^n \times abs(t - x), \text{otherwise.} \end{cases}$$

When the correlation immunity reaches some predefined value, then the fitness equals the sum of the correlation immunity and the difference between the maximal HW of a Boolean function's truth table and a current one. For solutions with incorrect correlation immunity values, we assign them a value that allows those solutions to be differentiated by their Hamming weight.

We do not apply multiobjective optimization since we regard different properties as constraints of a fixed target value, rather than independent objectives. This allows us to focus on the best fitness values we can obtain while the selected properties, such as balancedness, are held as constraints.

5. RESULTS AND DISCUSSION

5.1 The First Objective

When considering fitness function as given in Eq. (12), we know what are the optimal values for each of the properties we investigate. We give those values for each level of correlation immunity in Table 2. As explained in the previous section, it is obvious that the increase in the value of one property results in the decrease of the values of other properties. We can see that the particularly interesting case is when the correlation immunity equals two and three. For those values the nonlinearity and the algebraic immunity properties have the same values. When reporting the obtained property values for a certain solution, we use the following notation: (N_f, t, deg, AI) and we only present balanced solutions. In Table 3, we present mean fitness values for CGP with the fitness function 1 for various target correlation immunity values.

Table 2: Optimal values of properties, $fitness_1$

t	N_f	AI	deg	Global optimum value
2	112	4	5	123
3	112	4	4	123
4	96	3	3	106
5	64	2	2	73

Table 3: Average fitness, $fitness_1$, CGP.

	Gen./mut.	1	4	7	10	13
t=2	300	101.68	99.48	97.04	101.3	100.84
	700	105.08	114.38	106.26	110.36	112.7
	1 100	110.32	116.56	114.68	116.16	115.84
	1 500	109.84	118.6	117.08	110.54	114.34
	1 900	111.72	116.12	117.42	116.92	115.28
t=3	300	78.8	79.76	87.06	88.16	82.52
	700	93.86	95.04	95.66	99.52	97.52
	1 100	90.94	103	103.56	101.9	100.6
	1 500	93.42	105.4	104.06	101.86	99.86
	1 900	95.3	103.38	100.5	101.12	101.76
t=4	300	56.28	58.18	59.64	49.94	60.06
	700	60.28	81.18	71.76	77.82	79.8
	1 100	75.68	79.88	84.62	90.68	75.78
	1 500	75.24	88.64	88.62	82.58	88.46
	1 900	73.04	82.56	85.3	87.26	89.92
t=5	300	20.02	32.88	21.92	25.84	26.0
	700	34.16	32.78	35.56	34.34	39.28
	1 100	36.98	39.58	41.04	38.62	43.64
	1 500	35.56	49.02	42.46	50.5	42.4
	1 900	30.44	50.42	54.12	40.9	38.62

We give the number of times that each CGP variant reached the global optimum in Table 4. When the target correlation immunity equals three, there were no successful runs and therefore we skip that case in the table. For all other correlation immunity values, CGP proves to be an efficient algorithm (when using fitness function as in Eq. (12)). The easiest case seems to be when the correlation immunity equals five, after that is the case when it is equal to four and then two. Finally, the most difficult case is when the target correlation immunity value equals three.

Next, we give average fitness values for GP and fitness function 1 in Table 5. In Table 6, we display the number of times that GP found the global optimum. As is the case with CGP, when the correlation immunity equals three, the algorithm could not find any global optimum solutions. Besides that, it is interesting to notice that the cases when the correlation immunity equals four and five are easier for GP than CGP. This is particularly apparent when correlation immunity equals five, since in that case GP has 100% successfulness in reaching the global optimum. However, when the correlation immunity equals two, then the performance of CGP is substantially better than that of GP.

When experimenting with GA and fitness function 1 we were unable to find any optimal solutions for any of the correlation immunity values and therefore we do not present similar tables with results for GA. We note that the maximal

Table 4: # of optimal solutions, $fitness_1$, CGP.

	Gen./mut.	1	4	7	10	13
t=2	300	0	2	2	4	6
	700	3	15	14	14	12
	1 100	7	17	21	16	14
	1 500	6	**25**	19	11	8
	1 900	12	20	22	20	15
t=4	300	6	11	13	7	16
	700	14	21	21	24	27
	1 100	21	25	30	**33**	27
	1 500	18	32	30	29	30
	1 900	21	29	23	32	28
t=5	300	11	20	12	15	15
	700	21	20	22	21	25
	1 100	23	25	26	24	28
	1 500	22	32	27	33	27
	1 900	18	33	**36**	26	24

Table 5: Average fitness, $fitness_1$, GP.

	Pop./depth	5	7	8	9	11
t=2	100	110.2	113.8	113.24	110.98	110.18
	200	109.94	112.46	113.7	111.16	110.6
	500	110.42	113.86	113.1	112.9	110.4
	1 000	112	112.3	111.54	113.24	110.48
	2 000	115.08	115.8	113.68	113.72	111.12
t=3	100	105.9	106	16.04	106.04	106.02
	200	105.9	106.02	106.02	106.06	106
	500	105.98	106.04	106	106.02	106.04
	1 000	106	106.08	106.02	106.02	106
	2 000	106.04	106	106.02	106.06	106
t=4	100	85.24	102.8	99.6	99.6	99.6
	200	89.8	96.36	98.92	99.58	94.44
	500	92.56	102.16	100.24	100.24	99.56
	1 000	97.04	101.52	97.66	102.16	97.66
	2 000	98.96	102.8	100.88	98.32	96.96
t=5	100	73	73	73	73	73
	200	73	73	73	73	73
	500	73	73	73	73	73
	1 000	73	73	73	73	73
	2 000	73	73	73	73	73

value of correlation immunity we were able to find with GA and bitstring encoding equals one.

When working with two-stage fitness function, there is always the possibility that the algorithm will not perform well since it cannot reach the second stage. This is obviously true for the GA case where most of the population at the end of each run still have negative fitness values which means that those solutions are unbalanced. In that case, solutions can drastically improve their fitness only by becoming balanced and with a correlation immunity value larger than zero. However, the results implicate there is not enough information (diversity) in the search pool to converge.

Therefore, we note that we could write the two-stage fitness function differently. Since we know what is the best possible nonlinearity value we can rewrite it as:

$$fitness_1 = BAL + N_f + \delta_{N_f,x}(t + AI + deg). \quad (13)$$

Next, we experiment with the Eq. (13) for cases when nonlinearity equals 112 (correlation immunity equal to two and three). We disregard the cases when correlation immunity equals four and five since those seems to be much easier problems. We do not present the results here, but

Table 6: # of optimal solutions, $fitness_1$, GP.

	Pop./depth	5	7	8	9	11
t=2	100	0	7	12	6	10
	200	2	11	13	8	7
	500	2	12	11	12	8
	1 000	1	8	7	15	5
	2 000	6	**19**	15	14	9
t=4	100	18	**45**	40	40	40
	200	25	35	39	40	32
	500	29	44	41	41	40
	1 000	36	43	37	44	37
	2 000	39	**45**	42	38	36
t=5	100	50	50	50	50	50
	200	50	50	50	50	50
	500	50	50	50	50	50
	1 000	50	50	50	50	50
	2 000	50	50	50	50	50

mention that again both CGP and GP reach the same value corresponding to the global optimum relatively easily. Furthermore, with this fitness function the GA also succeeds in reaching the same value as GP and CGP.

However, we observe that all algorithms, although reaching the global optimum value of the modified criteria, in this case do not find the Boolean functions we are originally looking for. Instead, in most of the cases solutions are either $(N_f, t, deg, AI) = (112, 0, 7, 4)$ or $(112, 1, 6, 4)$. All our experiments show that it is much easier to obtain a high algebraic degree value than the high correlation immunity value. In accordance to that, we need to further improve our fitness function and not allow the algebraic degree to go over a certain threshold value. Therefore, our new fitness function equals:

$$fitness_1 = BAL + N_f + \delta_{N_f,x}(t + AI + \delta_{deg,x}deg). \quad (14)$$

In this function, we look only for Boolean functions with nonlinearity 112 and algebraic degree less than a threshold value (here, we set it to four and five). When not allowing the algebraic degree to go over a value of five, we can reach the maximal correlation immunity of two. In this case, GP and CGP perform similarly as in fitness function given in Eq. (12). This means that the best solutions are of the form (112, 2, 5, 4).

However, when not allowing algebraic degree to be higher than four, we are able to find solutions that have correlation immunity equal to three. The caveat is that with this fitness function the nonlinearity does not go over the value 96. Therefore, the best solutions are of the form (96, 3, 4, 4).

Again, GA with this fitness function does not succeed in evolving solutions with correlation immunity equal to two or three. Actually, our results show that for the GA there is no statistical difference when using fitness function as in Eqs. (13) or (14).

When comparing the efficiency of all three algorithms we observe that GA is by far the worst. However, this might be the problem of fitness function although we experimented with several variations and never obtained a solution that has a correlation immunity higher than one. However, when trying to compare the efficiency of GP and CGP, the situation is much more fuzzy.

We have only two algorithms and four objectives (one fitness function, but four possible correlation immunity values)

Table 7: Results for $fitness_2$, GP and CGP.

	t=2	t=3	t=4	t=5
CGP, 500	197.73	175.8	132	88.33
	(242)	(227)	(132)	(133)
CGP, 1500	206.27	169.93	134.13	131.67
	(242)	(243)	(196)	(133)
GP, 500	178	171	138.4	133
	(242)	(243)	(196)	(133)
GP, 5000	195.6	176.3	151.2	133
	(242)	(243)	(196)	(133)

where one objective (correlation immunity equal to three) behaves the same for both algorithms. In the case when we use fitness function as in Eq. (12) then both algorithms cannot find any correct solution. In the case of fitness function as in Eq. (14), both algorithms always reach the same set of values (96, 3, 4, 4). Notice that in this case the maximal obtained nonlinearity value is the same as in the case when the correlation immunity equals four. Therefore, with regards to the nonlinearity property, these solutions are worse. Naturally, such solutions have slightly higher algebraic immunity which is a consequence of a trade-off between the algebraic degree and the algebraic immunity properties.

5.2 The Second Objective

In this phase, we do not perform a thorough parameter search; on the basis of the results from the previous phase, we select tree depth to be seven when using GP and mutation probability of 7% when using CGP.

To understand the difficulty of this problem, consider the function that has all ones in its truth table. Therefore, the cardinality of the support equals 256. In that case the correlation immunity equals eight. However, by randomly flipping a single bit, correlation immunity drops to zero (while algebraic degree increases). Therefore, it is to be expected that EAs will have problems to evolve such functions due to the extreme sensitivity of some of the properties.

In Table 7, we give average fitness values for two CGP and two GP variations when the correlation immunity ranges from two to five. Furthermore, in parentheses we give the maximal value that each algorithm reached. Recall that we aim for fitness values that equal the sum of correlation immunity and the number of zeros in truth table. Here, we write zeros since we aim to maximize the fitness function and we are looking for a minimal number of ones in the truth table.

We do not present results for the correlation immunity value equal to one since it appears to be a trivial case where both algorithms find the global optimum easily. We note that GA also succeeds in finding several minimal Hamming weights when the correlation immunity equals to one. However, for all other cases GA is unsuccessful.

When evaluating the quality of the obtained solutions, we refer to data from [2]. We see that when the correlation immunity equals two, the minimal value of the cardinality of support equals 12. However, the best solution we could find is 16. Next, for the correlation immunity value of three, the best cardinality equals 16 which is the same value as found in our experiments. Furthermore, for correlation immunity values of four and five the minimal cardinalities of support are 64 and 128, respectively. We observe that both CGP and GP reach those values.

Therefore, this scenario is in a sense similar to the first one, i.e. for fitness function 1. For one of the correlation immunity values EAs cannot find the optimal solutions, but for the other cases both CGP and GP are able to find the optimal values. This suggests that in general EAs are efficient enough for a problem of this size. However, we note that before giving the final verdict about the difficulty of these problems, one must find the missing solutions as well as conduct the analysis with more algorithms.

5.3 Future Work

As already mentioned, future work must encompass more experiments in an effort to reach all known optimal solutions for both fitness functions. There exists one simple improvement of fitness function 2 we could have added, but decided not to in order to assess the efficiency of our algorithms. Since we look for a Boolean function with a minimal weight, we could add a penalty to all solutions that have the correct correlation immunity value, but are balanced.

Furthermore, one natural research avenue is to investigate Boolean functions with more than eight inputs. It is said that for the bigger sizes (e.g. $n \geq 9$), EAs cannot compete with some combinations of algorithms (e.g. theory supplementing optimization methods) [19]. In [2], we can see that for the correlation immunity equal to four, five and six and Boolean functions of sizes 11, 12 and 13 there are no known minimal values for the cardinality of support. Finding those values would be an interesting goal not only from the EA perspective, but also from the cryptographic and mathematic perspectives, in design theory and even in statistics.

Finally, here we are interested only in Boolean functions that possess certain properties, while future research should include experiments where the goal is a Boolean function (a circuit) that is at the same time easy to implement [17].

6. CONCLUSION

This paper explores the performance of three evolutionary algorithms on finding Boolean functions for cryptographic applications with the focus on the correlation immunity property. In the first application scenario, the goal is to find balanced functions with varying correlation immunity, that allows their usage in combiner generators. The second objective considers Boolean functions with minimal Hamming weight, while attaining the desired correlation immunity. Although we encountered varying success for different algorithms, the results indicate that EAs are capable of obtaining optimal solutions for known function sizes, which supports their application in conditions where the optimum is not known in advance.

7. REFERENCES

[1] In J. F. Miller, editor, *Cartesian Genetic Programming*, Natural Computing Series. Springer Berlin Heidelberg, 2011.

[2] S. Bhasin, C. Carlet, and S. Guilley. Theory of masking with codewords in hardware: low-weight dth-order correlation-immune boolean functions. Cryptology ePrint Archive, Report 2013/303, 2013. http://eprint.iacr.org/.

[3] L. Burnett, W. Millan, E. Dawson, and A. Clark. Simpler methods for generating better Boolean

functions with good cryptographic properties. *Australasian Journal of Combinatorics*, 29:231–247, 2004.

[4] P. Camion and A. Canteaut. Correlation-immune and resilient functions over a finite alphabet and their applications in cryptography. *Des. Codes Cryptography*, 16(2):121–149, 1999.

[5] C. Carlet. Boolean Functions for Cryptography and Error Correcting Codes. In Y. Crama and P. L. Hammer, editors, *Boolean Models and Methods in Mathematics, Computer Science, and Engineering*, pages 257–397. Cambridge University Press, New York, NY, USA, 1st edition, 2010.

[6] C. Carlet, J.-L. Danger, S. Guilley, and H. Maghrebi. Leakage squeezing of order two. In S. Galbraith and M. Nandi, editors, *Progress in Cryptology - INDOCRYPT 2012*, volume 7668 of *Lecture Notes in Computer Science*, pages 120–139. Springer Berlin Heidelberg, 2012.

[7] C. Carlet and K. Feng. An Infinite Class of Balanced Functions with Optimal Algebraic Immunity, Good Immunity to Fast Algebraic Attacks and Good Nonlinearity. In J. Pieprzyk, editor, *Advances in Cryptology - ASIACRYPT 2008*, volume 5350 of *Lecture Notes in Computer Science*, pages 425–440. Springer Berlin Heidelberg, 2008.

[8] C. Carlet and S. Guilley. Side-channel indistinguishability. In *Proceedings of the 2Nd International Workshop on Hardware and Architectural Support for Security and Privacy*, HASP '13, pages 9:1–9:8, New York, NY, USA, 2013. ACM.

[9] C. Carlet and S. Guilley. Correlation-immune boolean functions for easing counter measures to side-channel attacks. In H. E. Niederreiter, O. E. Alina, and e. a. Daniel, Panario (Ed.), editors, *Algebraic Curves and Finite Fields. Cryptography and Other Applications.*, pages 41–70. Berlin, Boston: De Gruyter., 2014.

[10] C. Carlet and P. Sarkar. Spectral Domain Analysis of Correlation Immune and Resilient Boolean Functions. *Finite Fields Appl.*, 8(1):120–130, Jan. 2002.

[11] N. Carrasco, J.-M. L. Bars, and A. Viola. Enumerative encoding of correlation-immune boolean functions. *Theoretical Computer Science*, 487(0):23 – 36, 2013.

[12] C. Cid, S. Kiyomoto, and J. Kurihara. The RAKAPOSHI Stream Cipher. In S. Qing, C. Mitchell, and G. Wang, editors, *Information and Communications Security*, volume 5927 of *Lecture Notes in Computer Science*, pages 32–46. Springer Berlin Heidelberg, 2009.

[13] J. A. Clark, J. L. Jacob, S. Stepney, S. Maitra, and W. Millan. Evolving Boolean Functions Satisfying Multiple Criteria. In *Progress in Cryptology - INDOCRYPT 2002*, pages 246–259, 2002.

[14] N. Courtois. Fast Algebraic Attacks on Stream Ciphers with Linear Feedback. In D. Boneh, editor, *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 176–194. Springer Berlin Heidelberg, 2003.

[15] N. Courtois and W. Meier. Algebraic Attacks on Stream Ciphers with Linear Feedback. In E. Biham, editor, *Advances in Cryptology - EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 345–359. Springer Berlin Heidelberg, 2003.

[16] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Springer-Verlag, Berlin Heidelberg New York, USA, 2003.

[17] B. M. Gammel, R. Gottfert, and O. Kniffler. The Achterbahn Stream Cipher, 2005.

[18] J. Katz and Y. Lindell. *Introduction to Modern Cryptography*. Chapman and Hall/CRC, Boca Raton, 2008.

[19] S. Kavut, S. Maitra, and M. D. Yücel. There exist Boolean functions on n (odd) variables having nonlinearity $> 2^{n-1} - 2^{(n-1)/2}$ if and only if n > 7, 2006.

[20] J.-M. Le Bars and A. Viola. Equivalence classes of boolean functions for first-order correlation. In *Information Theory, 2007. ISIT 2007. IEEE International Symposium on*, pages 181–185, June 2007.

[21] S. Maitra and P. Sarkar. Highly Nonlinear Resilient Functions Optimizing Siegenthaler's Inequality. In M. Wiener, editor, *Advances in Cryptology - CRYPTO' 99*, volume 1666 of *Lecture Notes in Computer Science*, pages 198–215. Springer Berlin Heidelberg, 1999.

[22] S. Mangard, E. Oswald, and T. Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards (Advances in Information Security)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.

[23] J. McLaughlin and J. A. Clark. Evolving balanced Boolean functions with optimal resistance to algebraic and fast algebraic attacks, maximal algebraic degree, and very high nonlinearity. Cryptology ePrint Archive, Report 2013/011, 2013. http://eprint.iacr.org/.

[24] W. Millan, A. Clark, and E. Dawson. Heuristic design of cryptographically strong balanced Boolean functions. In *Advances in Cryptology - EUROCRYPT '98*, pages 489–499, 1998.

[25] S. Picek, D. Jakobovic, and M. Golub. Evolving cryptographically sound boolean functions. In *Proceedings of the 15th Annual Conference Companion on Genetic and Evolutionary Computation*, GECCO '13 Companion, pages 191–192, New York, NY, USA, 2013. ACM.

[26] P. Sarkar and S. Maitra. Construction of nonlinear boolean functions with important cryptographic properties. In B. Preneel, editor, *EUROCRYPT*, volume 1807 of *Lecture Notes in Computer Science*, pages 485–506. Springer, 2000.

[27] P. Sarkar and S. Maitra. Nonlinearity bounds and constructions of resilient boolean functions. In M. Bellare, editor, *CRYPTO*, volume 1880 of *Lecture Notes in Computer Science*, pages 515–532. Springer, 2000.

[28] T. Siegenthaler. Correlation-immunity of nonlinear combining functions for cryptographic applications (corresp.). *IEEE Trans. Inf. Theor.*, 30(5):776–780, Sept. 2006.

[29] Y. Tarannikov. On Resilient Boolean Functions with Maximal Possible Nonlinearity. In *Proceedings of INDOCRYPT 2000, Lecture Notes in Computer Science*. Springer-Verlag, 2000.