An Evolutionary Multiobjective Approach to Design Highly Non-linear Boolean Functions

Hernán Aguirre Hiroyuki Okazaki Yasushi Fuwa
ahernan@shinshu-u.ac.jp okazaki@cs.shinshu-u.ac.jp fuwa@cs.shinshu-u.ac.jp

Graduate School of Science and Technology, Shinshu University
4-17-1 Wakasato, Nagano, 380-8553 JAPAN

ABSTRACT

The proliferation of all kinds of devices with different security requirements and constraints, and the arms-race nature of the security problem are increasingly demanding the development of tools to help on the automatic design of Boolean functions with security application. Nowadays, the design of strong cryptographic Boolean functions is a multiobjective problem. However, so far evolutionary multiobjective algorithms have been largely overlooked and not much is known about this problem from a multiobjective optimization perspective. In this work we focus on non-linearity related criteria and explore a multiobjective evolutionary approach aiming to find several balanced functions of similar characteristics satisfying multiple criteria. We show that the multiobjective approach is an efficient alternative to single objective optimization approaches presented so far. We also argue that it is a better framework for automatic design of cryptographic Boolean functions.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search; G.1.6 [**Numerical Analysis**]: Optimization

General Terms

Algorithms, Design, Performance, Security

Keywords

Evolutionary Multiobjective Optimization, Non-linear Boolean Functions, Information Security, Cryptography

1. INTRODUCTION

Boolean functions are used as fundamental components in several different types of cryptographic applications, including block ciphers, stream ciphers, and hash functions. Multiple desirable criteria for cryptographic Boolean functions have been identified in the past, among them balance, maximum distance to linear functions, maximum distance to linear structures, high algebraic degree, correlation immunity of high order, and so on. All these criteria have proved important to reduce the effects of advanced modern cryptanalytic attacks, such as correlation attacks [1], algebraic approximations, linear cryptanalysis [2, 4], and differential cryptanalysis [3, 4]. Recently, the tradeoffs between some of these criteria have received a lot of attention in the Boolean function literature, see for example [5] and references therein. A clear tradeoff has been shown for correlation immunity and algebraic degree. That is, a cryptosystem high resistant to correlation attacks will be less resistant to linear complexity attacks. Also, it is well known that a function cannot at the same time be balanced and have maximum algebraic degree, maximum distance to linear functions, and maximum distance to linear structures. On the other hand, less is known about the tradeoffs among non-linearity criteria, such as distance to linear functions, distance to linear structures, and algebraic degree.

The proliferation of all kinds of devices with different security requirements and constraints, and the arms-race nature of the security problem (weaknesses in the functions we currently use are frequently being reported) are increasingly demanding the development of tools to help on the automatic design of Boolean functions with security application. Traditionally, algebraic approaches have been used to design suitable functions with specific properties. However, generating cryptographic functions purely by constructive algebraic methods becomes increasingly difficult as the number of criteria to be satisfied augment. Recently, some works have sought to combine algebraic construction with deterministic computer search methods [5, 6, 7] and some authors have attempted using search heuristics such as genetic algorithms, hill climbers, and simulated annealing to generate Boolean functions [8, 9, 10, 11]. These are important steps towards automatic generation of cryptographic functions. However, these works have mostly focused on generating strong functions under a specific criterion rather than satisfying simultaneously several criteria.

As mentioned above, the design of strong cryptographic Boolean functions is a multiobjective problem constrained by devices and applications. Using multiobjective evolutionary approaches [12, 13] would seem a better option to tackle this problem aiming to develop more flexible, scalable, efficient, and probably more effective methods to generate ap-

GECCO'07, July 7–11, 2007, London, England, United Kingdom.

propriate functions. However, evolutionary multiobjective algorithms have been largely overlooked for the design of cryptographic functions, and not much is known about this problem from a multiobjective optimization perspective.

In this work we focus on non-linearity related criteria and use an evolutionary multiobjective algorithm aiming to find various balanced functions of similar characteristics satisfying multiple criteria. That is, we are interested in functions with similar high fitness but diverse in variable space. This is important because in most applications the security system is composed of several different primitive Boolean functions, where the security of the whole system is given by the weaker primitive function. Since not much is known about the tradeoffs among non-linearity criteria, we enumerate small spaces of balanced functions to gain some understanding about the multiobjective problem at hand. In larger spaces, we incrementally add non-linearity criteria to observe the effectiveness of the multiobjective approach. We show that the multiobjective approach is an efficient alternative to single objective optimization approaches presented so far. We also argue that it is a better framework for automatic design of cryptographic Boolean functions.

2. DEFINITIONS

Boolean Function. A Boolean function $f : Z_2^n \rightarrow Z_2$ is a mapping from n binary inputs variables to one binary output. Basic representations of a Boolean function are the binary truth table and the polarity truth table. Other important and useful representations are the Walsh-Hadamard Transform and the Algebraic Normal Form.

Balance. A function is said to be *balanced* if the number of combinations mapping to 0 is the same as the number of combinations mapping to 1.

Binary Truth Table. The binary truth table lists the output bits for each of the 2^n possible inputs to the Boolean function. The binary truth table uses the set $\{0, 1\}$ for the output. Note that for n variables there could be totally 2^{2^n} different Boolean functions.

Polarity Truth Table. The polarity truth table represents a Boolean function with output symbols in the set $\{1, -1\}$. The polarity truth table is useful in calculations and can be easily obtained from the binary truth table by

$$\hat{f}(x) = (-1)^{f(x)} = 1 - 2f(x). \tag{1}$$

It should be noted that the group $\{0, 1, \oplus\}$ is isomorphic to $\{1, -1, *\}$, where XOR operation $\oplus$ is bitwise modulo 2 addition, and $*$ denotes multiplication on integers.

Hamming Distance. The Hamming distance between two functions f and g is defined as the number of truth table positions in which the functions disagree, i.e.

$$hd(f, g) = \Big|\{x \mid f(x) \neq g(x)\}\Big|. \tag{2}$$

Functions Similarity. The similarity between two functions f and g can be defined as the number of truth table positions in which the functions agree, i.e. $\Big|\{x \mid f(x) = g(x)\}\Big|$. It can be expressed in terms of Hamming distance by

$$s(f, g) = 2^n - hd(f, g). \tag{3}$$

Cross Correlation. The cross correlation between two functions f and g is defined as

$$c_a(f, g) = s(f, g) - hd(f, g), \tag{4}$$

which can be calculated from the polarity truth table by

$$c_a(f, g) = \sum_{x \in Z_2^n} \hat{f}(x)\hat{g}(x). \tag{5}$$

The normalized cross correlation is expressed by

$$c_n(f, g) = \frac{c_a(f, g)}{2^n}. \tag{6}$$

Linear Boolean Function. A linear Boolean function is defined as a linear combination of a subset of the input variables and can be expressed by

$$L_w(x) = w_1x_1 \oplus w_2x_2 \oplus \cdots \oplus w_nx_n, \tag{7}$$

where $x \in Z_2^n$ is the set of variables, $w \in Z_2^n$ is a mask that determines which input variables are considered by setting $w_i = 1$ or $w_i = 0$, w_ix_i denotes the bitwise AND of the w_i bit mask and x_i variable, and $\oplus$ denotes bitwise XOR.

Affine Boolean Functions. The set of Affine functions is the set of linear functions and their complements $A_{w,c}(x) = L_w(x) \oplus c$, where $c \in Z_2$.

Walsh-Hadamard Transform. The Walsh-Hadamard Transform (WHT) provides another means of representing Boolean functions. The WHT expresses a Boolean function in terms of its cross correlation to all linear functions. Thus, for a given linear function L_w specified by $w \in Z_2^n$, the WHT of function f denoted as F is equivalent to

$$F(w) = c_a(f, L_w) \tag{8}$$

The relationship between the WHT and the polarity truth table of a Boolean function can be expressed as

$$F = H_n \cdot \hat{f} \tag{9}$$

where H_n is a ± 1 matrix of size $2^n \times 2^n$ which is defined recursively using the Kronecker product of matrices as follows

$$H_n = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \otimes H_{n-1}, \; H_o = [1]. \tag{10}$$

Linear Structure. For a linear (or Affine) Boolean function f the value $f(x + s)$ and $f(x)$, for every fixed s, are either always equal or always different. However many functions have this property without being linear or Affine. This property is called the *linear structure*.

Autocorrelation. The autocorrelation function gives an indication of the imbalance of all first order derivatives of a Boolean function and provides a measure of self-similarity for Boolean functions. The derivative of a Boolean function $f(x)$, taken with respect to a vector s, where x and $s \in Z_2^n$, is defined as $f(x) \oplus f(x \oplus s)$. Similarly, the derivative of the polarity form of a Boolean function $\hat{f}(x)$ is defined as $\hat{f}(x)\hat{f}(x \oplus s)$. Thus, the *autocorrelation function* $r_{\hat{f}}(s)$ of a Boolean function f is defined by the polarity truth table to be

$$r_{\hat{f}}(s) = \sum_x \hat{f}(x)\hat{f}(x \oplus s) \tag{11}$$

Low values are considered good, whereas maximal values are a serious weakness related to linear structures.

Algebraic Normal Form. The Algebraic Normal Form (ANF) [14] is a representation that describes a Boolean function as the sum of all products of the variables, i.e. an XOR sum of logical AND products of sub-sets of inputs variables.

The ANF can be expressed by

$$f(x_1, x_2, \cdots, x_n) = a_o \oplus \sum_{1 \le i \le n} a_i x_i \oplus \sum_{1 \le i < j \le n} a_{ij} x_i x_j \oplus \cdots \oplus a_{12\cdots n} x_1 x_2 \cdots x_n. \quad (12)$$

3. NONLINEARITY CRITERIA

The method of confusion and diffusion is used as a fundamental technique to achieve security in cryptographic systems. Confusion is reflected in the non-linearity of certain Boolean functions describing the cryptographic transformations. Non-linearity is crucial since most linear systems are easily breakable. In this context, there are several criteria which can serve as measures for non-linearity. Here, we explain non-linearity criteria related to the multiple objectives we want to optimize.

Two important non-linearity criteria are expressed in terms of a distance to appropriate sets of cryptologically weak Boolean functions. One is expressed in terms of the distance to the set of Affine functions. The other one is expressed in terms of the distance to the set of functions with linear structures. Perfect non-linear functions have been shown to be optimum with respect to both the set of Affine functions and the set of functions with linear structures. However, a perfect linear function is never balanced.

A third non-linearity criterion is the algebraic order of the function.

3.1 Nonlinearity Respect to Affine Functions

The nonlinearity Nl of a Boolean function f with respect to Affine functions is defined as the minimum Hamming distance of f to all members of the set of Affine functions. Since the Walsh-Hadamard Transform expresses a Boolean function in terms of its correlation to all linear functions, Nl can be calculated directly from this transform by

$$Nl = \frac{1}{2}(2^n - F_{max}). \quad (13)$$

where F_{max} is the largest absolute cross correlation value occurring in $F(w)$ for all $w \in Z_2^n$, i.e.

$$F_{max} = \max_{w \in Z_2^n} |F(w)| \quad (14)$$

The maximization of Nl, or equivalently the minimization of the largest absolute correlation to linear functions $|F_{max}|$, assures a suitable resistance to linear cryptanalysis. Perfect non-linear (not balanced) functions are known to have minimum correlation $\pm 2^{n/2}$ to all Affine functions.

3.2 Nonlinearity Respect to Linear Structures

The nonlinearity Sl of a Boolean function f with respect to the set of functions with linear structures is defined as the minimum Hamming distance of f to all members of the set of functions with linear structures. This non-linearity Sl can be calculated by observing the largest absolute value in the autocorrelation spectra of a function f denoted by

$$Ac = \max_{s \in Z_2^n, s \neq 0} |\hat{r}_f(s)| \quad (15)$$

The maximization of Sl, or equivalently the minimization of Ac, is the basis for protecting against differential cryptanalysis [15, 16]. It should be mentioned that a function f has optimum Sl if for every nonzero vector $s \in Z_2^n$ the values $f(x+s)$ and $f(x)$ are equal for exactly half of the arguments $x \in Z_2^n$. In previous works, Ac has been minimized instead of trying to maximize Sl. For comparison purposes in this work we also seek to minimize Ac.

3.3 Algebraic Order

The algebraic order Or of a Boolean function f is given by the number of variables in the largest product term (monomial) that exists in the ANF. Affine Boolean functions are thus the functions with algebraic order < 2. The algebraic order of the functions should be maximized to resist simple algebraic approximations.

4. RELATED WORK

Previous works aiming to generate highly non-linear balanced functions by search heuristics have used single objective optimization approaches, targeting directly either the non-linearity Nl respect to the set of affine functions (equivalently F_{max}) or the non-linearity respect to the set of Boolean functions with linear structures (equivalently Ac) [8, 9, 10]. We call these methods direct-single objective approaches.

Recently, Clark et al. have proposed a two-stage optimization approach [11]. In the first stage, an annealing-based method is used to evolve functions restricting the spread of Walsh values using a function of the type

$$C_{RX} = \sum_w \Big| |F(w)| - X \Big|^R, \quad (16)$$

where X and R are parameters. In the second stage, hill climbing respect to Nl orAc is applied to the best function obtained in the first stage. This method also focuses on single objective optimization, although at the end of the run other properties of the function are also measured. Best results in terms of the individual objectives have been achieved by the two-stage optimization approach than by the direct-single objective methods.

The two-stage approach is an important improvement towards the automatic generation of highly non-linear Boolean functions with cryptographic application. However, it suffers from some drawbacks and limitations. First, the objective function used during the first stage introduce two parameters and needs to be fine tuned in order to produce good results. Second, the method offers very limited function diversity (functions with same non-linearity characteristics but different at the genotype), especially for large n ($n = 8$). Third, the method is not scalable in the sense that it does not support the addition of criteria that could be related to non-linearity or to other cryptographic characteristics we might want an evolved function to have.

5. MULTIOBJECTIVE APPROACH

5.1 Evaluation criteria

Here, we describe the evaluation criteria used in this work to evolve functions. The first evaluation criterion used is the non-linearity Nl, i.e.

$$f_1 = N_f = \frac{1}{2}(2^n - F_{max}). \quad (17)$$

The second criterion is Ac, the largest absolute value in the autocorrelation spectra, i.e.

$$f_2 = Ac = \max_{s \in Z_2^n, s \neq 0} |\hat{r}_f(s)| \quad (18)$$

Step 1 Initialize the single parent p with a random balanced string of length N, assuring that the number of 1s equal the number of 0s (balanced function), evaluate p, add p to *Archive*, add p to *Population*, set the total number of evaluations T, and set the number of evaluations counter to $t = 1$

Step 2 While ($t < T$)

Step 2.1 Create a random permutation $\boldsymbol{\pi}$=$(\pi_1, \pi_2, \ldots, \pi_N)$ of the N string positions, set the permutation index to $i = 1$, and set *stagnation* to *yes*

Step 2.2 While ($i < N$ and $t < T$)

- Determine a random position j among the bits with complementary value to the bit at position π_i.
- Clone the parent p and swap bits between positions π_i and j to create child c. The child c is a two bits neighbor of p, and remains balanced similar to p.
- Evaluate c
- If c DOMINATES p, replace p with c and set *stagnation* to *no*
- If c is NOT DOMINATED by p, update *Archive* and *Population* with c
- Increment t and i, $t = t + 1$ and $i = i + 1$

Step 2.3 If *stagnation* is *yes*, RESTART the search by replacing p with an individual of the *Population*. If *Population* is empty, p is initialized anew with a random balanced string and added to *Population*. Increment t, $t = t + 1$

Step 3 Return *Archive*.

Figure 1: The flow of multiobjective random bit climber moRBC(ρ : 1 + 1) to search for balanced functions

These two criteria have been used in direct-single objective approaches to try to find appropriate functions satisfying a particular criterion.

In addition, we use a third criterion that measures the correlation deviation of a Boolean function f from a perfect non-linear function, taking into account the correlation spectra to all Affine functions. This correlation deviation is given by

$$f_3 = \sigma_c = \sqrt{\sum_w (|F_w(x)| - 2^{n/2})^2} \qquad (19)$$

Recall that perfect non-linear (not balanced) functions have minimum correlation $\pm 2^{n/2}$ to all Affine functions. The above criterion gives a measure of how far f is being from being a perfect non-linear function. Thus by minimizing σ_c we can expect to focus the search around functions with good Nl and Ac.

In our study we use the above described evaluation criteria to search for Boolean functions using one, two, or three objectives. In all cases we report the characteristics of the functions on terms of (Nl,Ac,Or), whether Nl and Ac are used or not as evaluation criteria. Note that in this work Or is only measured but never used as one of the evaluation criteria. In addition, we search on the space of balanced functions, so the initialization procedure and the genetic operators are implemented to guarantee that all generated trial solutions are balanced. That is, always the number of 1s and 0s remains equal.

5.2 Multiobjective Optimizer

In this work we use a multiobjective random bit climber (moRBC) that uses elitism and bias selection by Pareto dominance to search on the space of balanced Boolean functions. The algorithm we use is a version of moRBC($\rho : 1+1$) [17] that preserve bits balance.

moRBC(ρ : 1 + 1) uses only one parent individual from which it creates one child by swapping two complementary bit values. The position π_i of the first bit to be swapped is given by a random permutation. The position of the second bit to be swapped is randomly determined among the bits with complementary value to the bit at position π_i. Since the initial parent is balanced (same number of 0s and 1s) the child is also balanced and a two-bit neighbor of the parent. Due to the random permutation, children of the same parent created in subsequent iterations are different.

The parent for the next iteration is decided between parent and child based on their Pareto dominance relation. If the child dominates the parent then it replaces the parent. Otherwise, the parent remains. moRBC(ρ : 1 + 1) keeps the individuals *non-dominated by the parent* in a *Population* of maximum size ρ in order to sustain its search. Climbing continues until no improvement has been detected, i.e. there has not been parent replacement after all bits indicated by the random permutation have been tested in subsequent iterations. In that case, the algorithm opts for a restart. If the *Population* is not empty, the algorithm restarts the search from one of the collected individuals. Otherwise, it restarts the search from a balanced randomly created individual. moRBC(ρ : 1 + 1) also uses an *Archive* of limited capacity where it keeps the set of non-dominated solutions found by the algorithm. **Fig. 1** illustrates the flow of moRBC(ρ : 1 + 1).

It should be noted that the procedure that updates the *Population* first tries to delete solutions that have been already climbed in order to make room for a new member when the *Population* has reached its maximum size ρ. If no local optimum solutions are in the *Population* then it deletes the most crowded non-dominated solution using NSGA-II's diversity preserving mechanism in objective space [12](p.236). On the other hand, the procedure that updates the *Archive* only checks for crowding distance in order to make room for

a new member when the *Archive* has reached its maximum size. In addition, in both procedures we do not allow clones, i.e. same genotype individuals are forbidden.

moRBC($\rho : 1 + 1$) using a one-bit climbing strategy has shown robust and better performance than other state of the art multiobjective algorithms on sub-classes of non-linear epistatic binary problems [17, 18]. For additional details about moRBC($\rho : 1 + 1$) and the effectiveness of population climbing the reader is referred to [17].

(In this work, moRBC(ρ : 1 + 1) is also used for single objective climbing. In this case the *Population* of non-dominated individuals consists only of individuals with fitness equal to the parent individual. Since in our algorithm we do not allow clones, the same fitness individuals are all different at the genotype level.)

6. RESULTS BY ENUMERATION ON SMALL SPACES OF BALANCED-FUNCTIONS

The characteristics of the multiobjective landscapes defined by the criteria used to evaluate cryptographic Boolean functions are not known. In this section, in order to have some insight into the multiobjective landscape induced by Nl, Ac and Or, we present results by enumerating all balanced functions with $n = 4$ and $n = 5$ input variables. Enumeration for $n \geq 6$ is not possible within a reasonable time. The size of the search space of balanced functions is given by the combination $\binom{2^n}{2^{n-1}}$, where n is the number of the input variables in the function. Thus, there are totally $B_n = 12,870$ balanced functions for $n = 4$ and $B_n = 601,080,390$ for $n = 5$.

Table 1 (a) and **(b)** show for $n = 4$ and $n = 5$, respectively, the number of balanced functions $|f|$ with characteristics Nl, Ac and Or, and the percentage $|f|$ represents respect to the total number of balanced functions, $\% = 100 \times |f|/B_n$. Also, it separates fronts with a horizontal line and includes the non-dominated front the functions belong to. Recall that we seek functions with high Nl, low Ac, and high Or.

From **Table 1** the following observations are important. First, it can be seen that for $n = 4$ there are only 4 different points in objective space (4 different characteristics of functions) grouped in 3 fronts, whereas for $n = 5$ there are only 20 different points in objective space grouped in 10 fronts.

Second, for a given combination of characteristics (for each point in the objective space) there are many different functions in variable space.

Third, the fraction of solutions in the first front for $n = 4$ represents about 78.32% of the search space, whereas for $n = 5$ the first front represents only about 6.49% of the whole search space of balanced functions. Conform n increases we expect the fraction of solutions in the first front to reduce drastically respect to the whole search space. Also, although the number of fronts would increase, as well as the number of points in objective space within each front, we would still have many points in variable space mapping to the same point in objective space.

Fourth, a clear tradeoff between characteristics Nl, Ac, and Or is observed for $n = 5$ in the first front as well as in other fronts. In $n = 4$ there is no tradeoff in the first front (there is only one point in objective space) but a tradeoff can be seen in the second front. This is important because it suggests tradeoffs among non-linearity criteria on functions with larger n. Tradeoffs between algebraic order and correlation immunity have been shown, but not much discussion exists about the tradeoffs among non-linearity criteria.

Table 1: Number of balanced Boolean functions $|f|$ with characteristics (Nl, Ac, Or) found by enumeration, $n = 4$ and $n = 5$.

Nl	Ac	Or	$\|f\|$	%	$Front$
4	8	3	10,080	78.32	1
4	16	2	840	6.53	2
2	16	3	1,920	14.92	
0	16	1	30	0.23	3

(a) $n = 4$

Nl	Ac	Or	$\|f\|$	%	$Front$
12	8	3	5,332,992	0.89	1
12	16	4	1,666,560	0.28	
10	8	4	31,997,952	5.32	
12	16	3	1,666,560	0.28	2
10	16	4	306,647,040	51.02	
12	32	2	27,776	4.6E-3	3
10	24	4	36,664,320	6.10	
10	32	4	444,416	0.07	4
8	16	4	103,326,720	17.19	
8	16	3	10,554,880	1.76	5
8	24	4	81,244,800	13.52	
8	32	3	1,145,760	0.19	6
6	16	4	8,888,320	1.48	
8	32	2	8,680	1.4E-3	7
6	24	4	9,999,360	1.66	
6	32	4	555,520	0.09	8
4	24	4	833,280	0.14	
4	32	3	59,520	9.9E-3	9
2	32	4	15,872	2.6E-3	
0	32	1	62	1.0E-5	10

(b) $n = 5$

Fifth, it has been suggested that Or is maximized by maximizing Nl or alternatively by minimizing Ac. **Table 1** shows that indeed high values of Or are achieved by maximizing Nl. However, by looking the characteristics of solutions in the first front, it should be noticed that not always maximum Or corresponds to maximum Nl, nor maximum Or always corresponds to minimum Ac.

7. RESULTS BY THE MULTIOBJECTIVE APPROACH AND DISCUSSION

An important objective of this work is to learn whether a multiobjective optimization approach could be effective to find functions with high non-linearity. In this section we discuss the characteristics of the evolved functions using single and multiple criteria to evaluate solutions by focusing on functions with $n = 8$ variables. The characteristics of the functions are expressed in terms of their non-linearity respect to Affine functions Nl, the largest value in the autocorrelation spectra Ac, and algebraic order Or, i.e. (Nl, Ac, Or). We also analyze the number of different alternative functions (genotype diversity) evolved for the same combination of (Nl, Ac, Or) because we are interesting in finding not one but many functions with similar character-

istics. This is important because in most applications the security system is composed of several different primitive Boolean functions, where the security of the whole system is given by the weaker primitive function. Results presented below show all evolved functions kept in the *Archive* by the multiobjective random bit climber, collected from 50 different runs of the algorithm. In each run the multiobjective random bit climber expended $T = 50.000$ evaluations, setting the maximum size of its *Population* and *Archive* to 250.

7.1 Results by single and two objective optimization using Nl and Ac

First, we show results by using only one evaluation criterion at the time. **Table 2** shows the number of different functions for each obtained combination of (Nl, Ac, Or) by using Nl as the sole evaluation criterion. From **Table 2**, we can see that mostly functions with $Nl = 112$ and some with $Nl = 114$ can be found by focusing exclusively on Nl. In this case, where Ac is not targeted directly, autocorrelation of the obtained functions is in the range [80, 40] (best achieved autocorrelation $Ac = 40$). Functions with better characteristics can be found by the two-stage approach. From **Table 9** note that the two-stage approach could find 8 functions with $Nl = 116$ and $Ac = 24$. Regarding algebraic degree, we can see that functions with very high order $Or = 6$ and $Or = 7$ can be obtained. Note that since we are evolving balanced functions, the maximum achievable order is 7 ($n - 1$).

Similarly, **Table 3** shows results by using Ac as the sole evaluation criterion. In this case, we can see that functions with better autocorrelation can be found, i.e. $Ac = 32$, but no function with $Ac = 24$ is found. However, since Nl is not targeted directly, the number of functions achieving $Nl = 114$ reduce substantially and many functions with values of $Nl < 112$ are found. Overall, these results are in accordance with previous works reporting that it is not effective to find functions with simultaneously very high Nl and low Ac by using direct-single objective optimization on these objectives.

Second, **Table 4** shows results by using simultaneously Nl and Ac as the two evaluation criteria in a multiobjective optimization fashion. From the table, we can see that by including both criteria the obtained functions cluster in objective space around $Nl = \{112, 114\}$ and $Ac = 32$. This suggests that the multiobjective algorithm is more reliable than the single objective approach to find simultaneously better solutions in terms of Nl and Ac. However, by targeting directly and simultaneously Nl and Ac still no function with $Nl = 116$ or $Ac = 24$ could be found. This is because by using directly Nl and Ac there are still too many functions mapping to the same point in objective space, in which case selection becomes ineffective since it cannot discriminate among solutions with same fitness.

7.2 Results by two and three objective optimization using σ_c with Nl and Ac

Next, we include σ_c as one of the evaluation criteria and observe its effects when is used in conjunction with Nl and Ac. Remember that σ_c gives a measure of the distance of the Boolean function f to a perfect non-linear function. Thus by minimizing σ_c we can expect to focus the search around functions with good Nl and Ac. In addition, σ_c helps to discriminate among solutions having the same value of Nl and Ac.

Table 2: Number of Boolean functions with characteristics (Nl, Ac, Or) found by using Nl as the sole evaluation criterion, $n = 8$.

Ac	Nl	Nl	
	112	*112*	*114*
40	11	978	14
48	119	5560	76
56	86	3563	20
64	17	797	1
72	3	103	
80	1	12	
	$Or = 6$	$Or = 7$	

Table 3: Number of Boolean functions with characteristics (Nl, Ac, Or) found by using Ac as the sole evaluation criterion, $n = 8$.

Ac	Nl		Nl					
	108	*112*	*104*	*106*	*108*	*110*	*112*	*114*
32	6	9		22	698	3882	1460	2
40	14	5	51	467	2034	1621	58	
	$Or = 6$		$Or = 7$					

Table 4: Number of Boolean functions with characteristics (Nl, Ac, Or) found by using simultaneously Nl and Ac as the two evaluation criteria, $n = 8$.

Ac	Nl	Nl	
	112	*112*	*114*
32	18	2628	1007
40			22
	$Or = 6$	$Or = 7$	

First, we use simultaneously Nl and σ_c as the two evaluation criteria. Results are shown in **Table 5**. From this table we can see that by including σ_c, functions with better Nl and Ac characteristics could be found. Note that in this case the algorithm was able to find 3 functions with $Nl = 116$ and $Ac = 24$, which was not possible by targeting only Nl or Ac as shown in **Table 2** and **Table 3**, respectively, or by targeting both Nl and Ac in a multiobjective fashion as shown in **Table 4**.

Second, we try simultaneously Ac and σ_c as the two evaluation criteria. Results are shown in **Table 6**. From this table, we can see that the inclusion of σ_c helps finding functions with better $Ac = 24$, which was not possible by targeting Ac alone or Ac in conjunction with Nl as shown in **Table 3** and **Table 4**, respectively. Also, note that many more functions with $Ac = 24$ are found by focusing on Ac and σ_c than by focusing on Nl and σ_c. However, quality respect to Nl reduces substantially. Note that no function with $Nl = 116$ was found; actually, only one function with $Nl = 114$ was found.

Third, **Table 7** shows results by using simultaneously Nl, Ac, and σ_c as the three evaluation criteria. From this table

Table 5: Number of Boolean functions with characteristics (Nl, Ac, Or) found by using simultaneously Nl and σ_c as the two evaluation criteria, $n = 8$.

Ac	Nl		Nl				
	112	*116*	*108*	*110*	*112*	*114*	*116*
24	4	1		16	297	77	3
32	15	20	1	641	2707	1274	5
40	1	13	2	254	691	467	
48				15	28	34	
56						1	
	$Or = 6$		$Or = 7$				

Table 6: Number of Boolean functions with characteristics (Nl, Ac, Or) found by using simultaneously Ac and σ_c as the two evaluation criteria, $n = 8$.

Ac	Nl		Nl				
	112	*116*	*108*	*110*	*112*	*114*	*116*
24	8		1	29	1027	1	
32				33	15		
40			3				
	$Or = 6$		$Or = 7$				

Table 7: Number of Boolean functions with characteristics (Nl, Ac, Or) found by using simultaneously Nl, Ac, and σ_c as the three evaluation criteria, $n = 8$.

Ac	Nl		Nl				
	112	*116*	*108*	*110*	*112*	*114*	*116*
24	20	1	1	18	1332	171	249
32		16	1	36	185	184	1
40		6			1	1	
48						1	
	$Or = 6$		$Or = 7$				

we can see that evaluating solutions with these three objectives increases the chances of finding functions with high Nl and low Ac. Note that in this case the algorithm was able to find 249 functions with characteristics $Nl = 116$, $Ac = 24$, and $Or = 7$. Recall that the two-stage optimization approach was able to find only 8 functions with similar characteristics. This larger number of found functions with high Nl and low Ac is significant because in most applications where Boolean functions are used we need several different functions (different genotype) with high non-linearity. For example, an S-Box, with n inputs and m outputs, will require m Boolean functions for its implementation (typically, $n = 8$ and $m = 8$). Remember that a block cipher uses several S-Boxes. Also, many approaches to Hash functions are based on S-Boxes.

Summarizing, Nl and Ac used within a multiobjective approach seem to complement each other to focus the search around promising regions of the search space. In addition σ_c is required to discriminate better among solutions with same values of Nl and Ac in order to find functions with high non-linearity.

Table 8: Summary of results by the multiobjective approach using simultaneously Nl, Ac, and σ_c as the three evaluation criteria, $n = \{5, 6, 7, 8\}$.

	f			
n	Nl	Ac	Or	$\lvert f\rvert$
5	12	8	3	100
5	12	16	4	12350
5	10	8	4	50
6	26	16	5	12500
7	56	16	5	7
7	54	16	6	100
8	116	24	7	249

Table 9: Summary of results by the two-stage optimization approach. NLT and ACT show results when the second stage focuses on Nl and Ac, respectively. NA indicates data non available

	f			NLT		ACT	
n	Nl	Ac	Or	X-Best	$\lvert f\rvert$	X-Best	$\lvert f\rvert$
5	12	8	3	-4	80	-4	74
5	12	16	4	$[-2, 10]$	100	$[-2, 4]$	100
6	26	16	5	4	59	6	52
7	56	16	6	$\{-4, 0, 6, 8\}$	2	$\{6, 8\}$	2
8	112	16	5	8	22	NA	NA
8	116	24	7	10	8	NA	NA

8. COMPARISON WITH TWO-STAGE APPROACH

In this section we briefly compare results by the multiobjective approach used in this work with the two-stage optimization approach proposed in [11] generating functions with $n = \{5, 6, 7, 8\}$ variables.

Table 8 summarizes results obtained on 50 runs by the multiobjective approach, where n indicates the number of variables, and $|f|$ indicates the number of different functions (at the genotype level) found with characteristics Nl, Ac, and Or. As indicated before, the multiobjective approach expended $T = 50,000$ evaluations in each run setting the size of its *Population* and *Archive* to 250.

Table 9 summarizes best results by the two-stage optimization approach reported in [11]. NLT/ACT shows results when Nl/Ac was targeted on the second stage (hill climbing). X-best indicates the values of parameter X for best performance ($R = 3.0$). The above results were collected after performing 100 different runs for each combination of parameter setting X and R, varying X in the range $[-10, 16]$ on intervals of 2 ($n = 8$) and $R = \{2.0, 2.5, 2.75, 3.0\}$. The range of X for other n changes slightly. The algorithm was set to expend a maximum of 160.000 iterations (evaluations) in each run during the first stage (annealing). However, no indication is given about the number of evaluations spent during the second stage (hill climbing).

From these tables, we can see that the multiobjective approach can find many more functions with similar characteristics (Nl,Ac,Or) than the two-stage optimization approach, especially for $n = \{5, 6, 8\}$. For example, for $n = 6$ the

multiobjective approach finds 12,500 different functions with characteristics $Nl = 26$, $Ac = 16$, and $Or = 5$, whereas the two stage optimization approach could find only 59 functions with similar characteristics.

For $n = 5$, where we know all solutions by enumeration, it should be mentioned that the multiobjective optimization approach is able to find functions with all three combination of characteristics (Nl,Ac,Or) present in the true Pareto front as shown in **Table 1**. However, looking closely at **Table 8**, is worth noting that the multiobjective optimization approach finds 12,350 functions with ($Nl = 12$, $Ac = 16$, $Or = 4$) and only 50 functions with ($Nl = 10$, $Ac = 8$, $Or = 4$), despite the fact that the percentage of functions with those characteristics respect to the whole search space are 0.28% and 5.32%, respectively (see **Table 1**). These results suggest that the evaluation functions we use bias the search towards optimum regions of Nl better than to optimal regions of Ac. For $n = 6$ the multiobjective approach is by far more efficient than the two-stage approach. However, in $n = 7$ is should be noted that the two-stage approach found 2 functions with ($Nl = 56$, $Ac = 16$, $Or = 6$) and the multiobjective approach could find none. In $n = 8$, the multiobjective approach found many more functions with ($Nl = 116$, $Ac = 24$, $Or = 7$) than the two stage approach, but it could not find non-dominated solutions that favor Ac at the expense of Nl and Or as the two-stage approach did.

Overall, the performance by the multiobjective approach seems promising, especially considering that many more functions with similar characteristics can be found expending fewer runs and much fewer evaluations than the two-stage optimization approach. These results encourage us to develop further the multiobjective approach for automatic generation of Boolean functions with cryptographic characteristics.

9. CONCLUSIONS

In this work we have focused on non-linearity related criteria and used an elitist, Pareto dominance-based, multiobjective algorithm to generate several balanced functions of similar characteristics. The multiobjective approach is more efficient than a two-stage optimization approach, requiring much fewer runs and evaluations to generate functions with high non-linearity. These results encourage us to develop further the multiobjective approach for automatic generation of Boolean functions with cryptographic characteristics. An important aspect of the problem analyzed in this work is that there are many functions mapping to the same combination of (Nl, Ac, Or) characteristics. Thus, ranking by non-dominance and diversity in objective space loses its effectiveness when the population is composed by different functions of similar characteristics. As future works, it would be worth exploring genotype diversity as a way to bias selection in order to guide better the search. Also, we would like to include other objectives in addition to non-linearity criteria, for which tradeoffs are well known. Furthermore, the multiobjective approach should be tried on functions with more input variables.

10. REFERENCES

[1] T. Siegenthaler, "Correlation Immunity of Non-linear Combining Functions for Cryptographic Applications", *IEEE Transactions on Information Theory*, vol.30, pp.776-780,1984.

[2] M. Matsui, "Linear Cryptanalysis Method for DES Cipher", *Proc. EUROCRYPT'93*, Springer-Verlag, Lecture Notes in Computer Science, vol.765, pp.386-397, 1994.

[3] E. Biham and A. Shamir "Differential Cryptanalysis of DES-like Cryptosystems", *Journal of Cryptology*, vol.4, no.1, pp.3-72, 1991.

[4] H. M. Heys, "A Tutorial on Linear and Differential Cryptanalysis", Technical Report CORR 2001-17, Centre for Applied Cryptographic Research, Department of Combinatorics and Optimization, University of Waterloo, Mar. 2001.

[5] S. Maitra and E. Pasalic, "Further Constructions of Resilient Boolean Functions with Very High Nonlinearity", *IEEE Transactions on Information Theory*, 48(7):1825-1834, July 2002.

[6] E. Pasalic, S. Maitra, T. Johansson, and P. Sarkar, "New Constructions of Resilient and Correlation Immune Boolean Functions Achieving Upper Bound on Nonlinearity", *Proc. Workshop on Coding and Cryptography - WCC 2001*, Electronic Notes in Discrete Mathematics, vol.6, Elsevier Science, 2001.

[7] P. Sarkar and S. Maitra, "Nonlinearity Bounds and Construction of Resilient Boolean Functions", *Advances in Cryptology - Crypto 2000*, Springer-Verlag, Lecture Notes in Computer Science, vol.1880, pp.515-532, 2000.

[8] W. Millan, A. Clark and E. Dawson, "An Effective Genetic Algorithm for Finding Highly Non-linear Boolean Functions", *Proc. First International Conference on Information and Communication Security*, Springer-Verlag, Lecture Notes in Computer Science, vol.1334, pp.149-158, 1997.

[9] W. Millan, A. Clark and E. Dawson, "Heuristic Desing of Cryptographically Strong Balanced Boolean Functions", *Proc. Advances in Cryptology - EUROCRYPT'98*, Springer-Verlag, Lecture Notes in Computer Science, vol.1403, pp.489-499, 1998.

[10] W. Millan, A. Clark and E. Dawson, "Boolean Functions Desing Using Hill Climbing Methods", *Proc. 4th Australasian Conference on Information, Security and Privacy*, Springer-Verlag, Lecture Notes in Computer Science, vol.1587, pp.1-11, 1999.

[11] J.A. Clark and J. L. Jacob, "Two-Stage Optimization in the Design of Boolean Functions", *Proc. 5th Australasian Conference on Information, Security and Privacy- ACISP 2000*, Springer-Verlag, Lecture Notes in Computer Science, vol.1841, pp.242-254, 2000.

[12] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*, John Wiley & Sons,2001.

[13] C. Coello, D. Van Veldhuizen, and G. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Kluwer Academic Publishers, Boston, 2002.

[14] C. J. A. Jansen and D.E. Boekee, "The Algebraic Normal Form of Arbitrary Functions over Finite Fields", *Proc. 8th Symposium of Information Theory in the Benelux*, pp.69-76, 1987.

[15] X. Lai, "Additive and Linear Structures of Cryptographic Functions", *Proc. Fast Software Encription - FSE Leuven Workshop 1994*, Springer-Verlag, Lecture Notes in Computer Science, pp.75-85, 1994.

[16] W. Meier and O. Staffelbach, "Nonlinearity Criteria for Cryptographic Application", *Proc. EUROCRYPT'89*, Springer-Verlag, Lecture Notes in Computer Science, vol.434, pp.549-562, 1990.

[17] H. Aguirre and K. Tanaka, "Effects of Elitism and Population Climbing on Multiobjective MNK-Landscapes", *Proc. 2004 IEEE Congress on Evolutionary Computation*, IEEE Center, pp.449-456, 2004.

[18] H. Aguirre and K. Tanaka, "Selection, Drift, Recombination, and Mutation in Multiobjective Evolutionary Algorithms on Scalable MNK-Landscapes", *Proc. Third Intl. Conf. on Evolutionary Multi-Criterion Optimization*, Springer, LNCS, vol.3410, pp. 355-369, 2005.