The Design of S-Boxes by Simulated Annealing

Article *in* New Generation Computing · July 2004

DOI: 10.1007/BF03037656 · Source: DBLP

3 authors, including:

John A. Clark
The University of York
201 PUBLICATIONS **5,772** CITATIONS

SEE PROFILE

Susan Stepney
The University of York
354 PUBLICATIONS **3,667** CITATIONS

SEE PROFILE

The Design of S-Boxes by Simulated Annealing

John A. Clark, Jeremy L. Jacob, Susan Stepney
Dept. of Computer Science, University of York, Heslington, York, YO10 5DD, UK
Email: [jac,jeremy,susan]@cs.york.ac.uk

***Abstract*— Substitution boxes are important components in many modern day block and stream ciphers. Their study has attracted a great deal of attention over many years. The development of a variety of cryptosystem attacks over the years has lead to the development of criteria for resilience to such attacks. Some general criteria such as high non-linearity and low autocorrelation have been proposed as useful criteria (providing some protection against attacks such as linear cryptanalysis and differential cryptanalysis). There has been little application of evolutionary search to the development of S-boxes. In this paper we show how a cost function that has found excellent single output Boolean functions can be generalised to provide improved results for small S-boxes.**

I. Introduction

Substitution provides a significant role in modern cryptography. For some applications the substitutions are formed by simple Boolean functions (which take several Boolean inputs and give a single output as a result). The design of suitable functions has received significant attention from cryptographers for decades. Recently, meta-heuristic search has emerged as a potentially very powerful tool for the design of such functions [9], [13], [12], [2]. Most recently, metaheuristic search in combination with theory has found functions with properties unattained by any other means [4], [6].

Substitution is typically implemented by *substitution boxes* (S-boxes for short). These are multiple-input, multiple-output functions. Perhaps the most famous (notorious) S-boxes are those of the Data Encryption Standard [14]. Like many modern ciphers DES is an iterated block cipher; the algorithm is implemented by repeating a smaller and simpler cipher a number of times or 'rounds'. Within each round the most significant contribution to security is made by eight 6-input, 4-output functions, shown as S1–S8 in Figure 1. These are specified via lookup tables. The DES algorithm has been subject to a great deal of controversy. Much of this has revolved around the particular substitutions implemented by the eight S-boxes. (Another controversial aspect was the reduction of the initially suggested key-length to 56 bits). The S-box idea has a firm hold in modern day cryptography. The new international symmetric key cryptography standard, the Advanced Encryption Standard (AES), also uses S-boxes to perform substitutions.

Unlike single-output Boolean functions, there has been little application of evolutionary or other metaheuristic search to the design of S-boxes. In this paper we show how generalising an unusual cost function developed for single output Boolean functions can be generalised for the case of S-boxes to produce significant improvements on previous work. (Results of attempts to find highly nonlinear small S-boxes have been reported in [10], [11].)

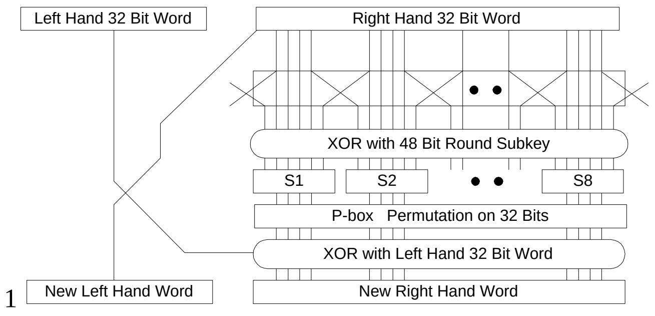

Fig. 1. Round of Data Encryption Standard

II. Preliminaries

A. Background

This section provides some definitions of relevance to Boolean functions with cryptographic application. We denote the substitution table of an n-input k-output Boolean function by $f : \mathbf{B}^n \rightarrow \mathbf{B}^k$, mapping each combination of n Boolean input values to some combination of k Boolean output values. For single-output functions if the number of combinations mapping to 0 is the same as the number mapping to 1 then the function is said to be *balanced*. For the multiple-output case, if each k-bit output value appears the same number of times, we say that the function is *regular*.

For the single-output case the substitution table is generally referred to as a 'truth table'. The *polarity truth table* is a particularly useful representation for our purposes. It is defined by $\hat{f}(x) = (-1)^{f(x)}$. Two functions f and g are said to be uncorrelated when $\sum_{x \in \mathbf{B}^n} \hat{f}(x)\hat{g}(x) = 0$. If so, if you try to approximate f by using g, you will be right half the time and wrong half the time.

An area of particular importance for cryptanalysts is the ability to approximate a function f by a simple linear function. One of the cryptosystem designer's tasks is to make such approximation as difficult as possible (by making the function f suitably *nonlinear*). Linearity is a form of structure crypto-designers clearly strive to avoid. One form of attack that exploits linearity is known as linear cryptanalysis, introduced by Matsui [8]. It has attracted a great deal of attention. Another form of structure that is to be avoided is *differential* structure. Essentially, particular differences in input words (difference defined by simple bitwise XOR) may be associated with particular differences of output words (again defined by bitwise XOR) with some strong bias (i.e. the output difference is not uniform for a particular input difference). This can often be exploited by a form of attack known as differential cryptanalysis, introduced by Biham and Shamir [1]. It has also attracted a great deal of attention. An excellent introduction to linear and differential cryptanalysis can be found in Heys' tutorial [5].

Substitution boxes are essentially n-input k-output functions. These can be viewed as a combination k individual single-output Boolean functions. Several important security criteria are actually defined in terms of single-output function criteria and so it is essential to understand first the basic Boolean function definitions and concepts. We then extend these to cater for the multiple-output case.

B. Cryptographic Criteria for Single-output Functions

Two formal criteria have been defined for the single-output case to capture some aspects of resilience to the sorts of attacks indicated above. These are high nonlinearity and low autocorrelation and are defined below together with other terminology used in this paper.

Linear Boolean Function. A linear Boolean function, selected by $\omega \in \mathbf{B}^n$, is denoted by
$L_\omega(x) = \omega_1 x_1 \oplus \omega_2 x_2 \cdots \oplus \omega_n x_n$
where $w_i x_i$ denotes the bitwise AND of the ith bits of ω and x, and $\oplus$ denotes bitwise XOR.

Affine Boolean Function. The set of affine functions is the set of linear functions and their complements
$A_{\omega,c}(x) = L_\omega(x) \oplus c$, where $c \in \mathbf{B}$.

Walsh Hadamard Transform. For a Boolean function f the Walsh Hadamard Transform $\hat{F}_f$ is defined by $\hat{F}_f(\omega) = \sum_{x\in\mathbf{B}^n} \hat{f}(x)\hat{L}_\omega(x)$. We denote the maximum absolute value taken by the transform by $WH_{max}(f) = \max_{\omega\in\mathbf{B}^n} \left|\hat{F}_f(\omega)\right|$. It is related to the nonlinearity of f.

Nonlinearity. The nonlinearity N_f of a Boolean function f is its minimum distance to any affine function. It is given by $N_f = \frac{1}{2}(2^n - WH_{max}(f))$.

Parseval's Theorem. This states that $\sum_{\omega\in\mathbf{B}^n}(\hat{F}_f(\omega))^2 = 2^{2n}$. A consequence of this result is that $WH_{max}(f) \geq 2^{n/2}$. This fact forms the starting point for the principal cost functions in this paper.

Autocorrelation Transform. The autocorrelation transform of a Boolean function f is given by $\hat{r}_f(s) = \sum_x \hat{f}(x)\hat{f}(x \oplus s)$. We denote the maximum absolute value in the autocorrelation spectra of a function f by AC_f, i.e., $AC_f = \max_s \left|\sum_x \hat{f}(x)\hat{f}(x\oplus s)\right|$. Here x and s range over $\mathbf{B}^n$ and so produces a result in $\mathbf{B}^n$.

C. Extensions to S-boxes

For each k-output S-box, we can extract a single-output Boolean function by simply XOR-ing some subset of the output bits together. If $f(x) : \mathbf{B}^n \to \mathbf{B}^k$ is an n-input k-output S-box then each $\beta \in \mathbf{B}^k$ defines a function that is a linear combination $f_\beta(x)$ of the m outputs of f. This is given by

$$f_\beta(x) = \beta_1 f_1(x) \oplus \cdots \oplus \beta_k f_k(x) \tag{1}$$

For each such function f_β the Walsh-Hadamard values $\hat{F}_\beta(\omega)$ and autocorrelation values $\hat{r}_\beta(s)$ are defined in the usual way. (Each such function is now a single-output function defined over the n inputs.)

There are $2^k - 1$ non-trivial functions obtainable in this way. The notions of non-linearity and autocorrelation are readily extended to the multiple output case. For the k-output case the non-linearity is the worst (lowest) non-linearity of all the $2^k - 1$ non-trivial single output functions obtained as indicated above. Similarly, the autocorrelation is the worst (highest) over all such derived single-output functions.

III. Cost Functions

A. Traditional Cost Functions

In virtually all work done so far existing optimisation-based work aimed at producing highly nonlinear functions has generally used nonlinearity itself as the fitness function, i.e. the fitness of a function f on n input variables is given by

$$fitness(f) = N_f = \frac{1}{2}\left(2^n - \max_\omega \left|\hat{F}(\omega)\right|\right) \tag{2}$$

or, when viewed as a minimisation problem, the cost function is given by

$$cost(f) = WH_{max}(f) = \max_\omega \left|\hat{F}(\omega)\right| \tag{3}$$

Similarly, with low autocorrelation as the target, the autocorrelation itself has been used as the cost function, i.e. the cost function is given by

$$cost(f) = AC(f) = \max_{s\neq 0}\left|\sum_x \hat{f}(x)\hat{f}(x\oplus s)\right| = \max_{s\neq 0}|\hat{r}(s)|. \tag{4}$$

Previous optimisation approaches to evolving Boolean functions with desirable cryptographic properties have been generalised to the multiple output case. Millan has compared random generation and hill-climbing as means of evolving highly nonlinear bijective S-boxes [10]. Burnett *et al.* have investigated the use of genetic algorithms and hill-climbing to evolve regular S-boxes [11]. Both high nonlinearity and low autocorrelation were targets. The fitness and cost measures for an S-box were the nonlinearity and autocorrelation values of that S-box. For the S-box case, the researchers above have used extensions of the basic definitions as cost functions. For non-linearity the cost function was:

$$cost(f) = \min_{\beta\in\mathbf{B}^k} \sum_{\omega\in\mathbf{B}^n} \left||\hat{F}_\beta(\omega)| - X\right|^R \tag{5}$$

For autocorrelation the cost function was:

$$cost(f) = \max_{\beta\in\mathbf{B}^k} \sum_{s\in\mathbf{B}^n} \left||\hat{r}_\beta(s)| - X\right|^R \tag{6}$$

B. Spectrum Based Cost Functions

In 2000 a new cost function family was proposed that offered significant improvements for the single-output case. This cost function, after significant experimentation, was shown capable of producing functions with exceptional profiles of security criteria. Rather than base the cost on extreme values (as per the definition of non-linearity and aurocorrelation), it defined a cost over the whole Walsh Hadamard spectrum and autocorrelation spectrum. The details of the experiments for the single output case and the detailed motivation for the cost function adopted are defined in an accompanying paper [3]. For present purposes we simply provide the cost function family below:

$$cost(f) = \sum_{\omega} \left| |\hat{F}(\omega)| - X \right|^R \tag{7}$$

where X and R are real-valued parameters. It is difficult to predict what the best such parameter values should be and considerable experimentation is needed. However, as indicated above, they have produced some exceptional results (effectively equalling the best results of theoreticians for functions of 8-inputs or less).

Since spectrum based approaches generated interesting results for the single-output case an obvious question to pose is 'Can the spectrum-based approaches be generalised to allow S-boxes to be evolved with desirable properties?' Two cost functions can now be defined for use in S-box evolution. A cost function based on Walsh-Hadamard spectra is given by

$$cost(f) = \sum_{\beta \in \mathbf{B}^k} \sum_{\omega \in \mathbf{B}^n} \left| |\hat{F}_\beta(\omega)| - X \right|^R \tag{8}$$

and a similar cost function based on autocorrelation spectra is given by

$$cost(f) = \sum_{\beta \in \mathbf{B}^k} \sum_{s \in \mathbf{B}^n} \left| |\hat{r}_\beta(s)| - X \right|^R \tag{9}$$

The single output cost functions have been applied to each function defined as a linear combination of the outputs and the results summed over all such combinations.

IV. The General Approach

All the S-boxes we are concerned with are regular: all outputs appear an equal number of times. In the case of bijective S-boxes, each output appears only once. (They are effectively permutations on the inputs). In the case of n-input k-output regular S-boxes, each output appears 2^{n-k} times. A variety of move strategies are possible, but we adopt the simplest: simply swap the output values associated with two input values. The result is guaranteed to maintain regularity and bijectivity. In addition, we swap only dissimilar output values. A search starts with a regular (but otherwise random) function and moves by a sequence of such swaps around the search space.

The approach is as follows:

1) Use an annealing-based search to minimise the value of the new cost function (suitably parametrised) given in Equation 7. Let the best solution produced during the search be f_{sa}.
2) Hill-climb from f_{sa} with respect to nonlinearity (or autocorrelation) to produce the final solution f_{sahc}
3) Measure the nonlinearity, autocorrelation and algebraic degree of f_{sahc}.

Although nonlinearity, autocorrelation and algebraic degree are all of interest, our approach is somewhat unusual in that Stage 1 targets none of the criteria directly, Stage 2 considers only one of the first two, and algebraic degree is never considered at all (it is simply measured at the end). The motivation for the application of this function to the single output case is explained in detail in an accompanying paper. In this paper, however, we wish to investigate its extension to the derivation of S-boxes.

We omit here a description of the annealing algorithm used for Stage 1. It is very much a 'vanilla' annealing. A full description is given in the Appendix.

V. Experiments Performed

Two approaches have been used in experiments. In the first, the second-stage hill-climbing is with respect to nonlinearity. We refer to this approach as the NLT (Non-Linearity Targeted) approach. In the second, the second-stage hill-climbing is with respect to autocorrelation. We refer to this as the ACT (Auto-Correlation Targeted) approach.

A. Experiments and Results

Table I records the best nonlinearity values achieved in Millan's experiments comparing the ability of random search and hill-climbing to evolve 5×5, 6×6, 7×7 and 8×8 bijective S-boxes. The cost functions defined by equations 8 and 9 have been used to evolve S-boxes of similar dimensions. At the end of each run hill-climbing was carried out with respect to nonlinearity and autocorrelation respectively. 50 runs were carried out for each value of X in the set $-4, -3, -2, -1, 0, 1, 2, 3, 4$. $R = 3.0$ was used throughout. Table I records the best *joint* values of nonlinearity and autocorrelation achieved by either technique (i.e. functions were generated which possessed both the indicated nonlinearity value and the indicated autocorrelation value).

The results for the bijective S-boxes are not optimal. 6×6 boxes with nonlinearity of 24 have been provided by construction but they seem quite rare (Millan [10] attempted one million random generation and hill-climbing attempts and found only a nonlinearity of 20). Deriving bijective S-boxes is not an easy task for annealing. As k increases by 1 the number of derived linear combinations to check doubles. An 8×8 bijective S-box with the parameter values shown takes about 20 minutes on 1.4 GHz Pentium PC. However, again this is not easy. Only one $(104, 80)$ function was generated from 200 runs. Similarly, for $n = 7$ only one $(48, 48)$ function was generated. Does this matter? We address this issue below.

	Millan [10]		Annealing	
n	rnd	HC	SA	AC-SA
5	8	10	10	16
6	20	20	22	32
7	44	46	48	48
8	98	100	102	80

TABLE I

SUMMARY RESULTS FOR BIJECTIVE N (AUTOCORRELATION RESULTS ALSO SHOWN FOR ANNEALING

Original	Final Non-linearity					
Non-linearity	90	92	94	96	98	100
80				2	2	
82			1	4		
84			2	29	7	
86			5	80	34	
88	1	3	20	258	127	
90		1	65	886	447	
92		2	91	1919	1112	
94			26	1946	1826	
96				246	827	3
98					17	

TABLE II

MILLAN'S IMPROVMENT OF BIJECTIVE 8×8 S-BOXES FROM A SAMPLE OF 10000

Burnett *et al.* applied genetic algorithms followed by hill-climbing to evolve $8 \times k$ regular S-boxes (for $k = 2, \ldots, 8$). Table III records the best nonlinearity and autocorrelation values achieved (individually). The new cost functions were again used to evolve regular S-boxes of similar dimensions (with $R = 3.0$ and the same range of X as before). Table I records the best *joint* values of nonlinearity and autocorrelation achieved by each technique. Burnett *et al.* presented their results as their 'current conjectures for the achievable bounds'. The results of applying the annealing-based approaches with the new cost functions is fairly dramatic (the hill-climbing second stage with respect to nonlinearity or autocorrelation rarely improves matters). As m increases the same general patterns of declining nonlinearity and increasing autocorelation are witnessed as by Burnett et al. However, the new cost functions and annealing-based searches have found functions that simultaneously improve nonlinearity and autocorrelation. Most typically, for the best functions, nonlinearity is 4 higher and autocorrelation is 16 lower.

		Burnett et al. [11]				Spectrum Based	
		NL		AC		Joint (NL,AC)	
n	m	rnd	GAs	rnd	GAs	SNLT	SACT
8	2	108	110	56	48	(114,32)	(114,32)
8	3	106	108	64	56	(112,40)	(112,40)
8	4	104	106	72	64	(110,56)	(110,48)
8	5	102	104	72	72	(108,64)	(108,56)
8	6	100	104	80	80	(106,64)	(106,64)
8	7	98	102	80	80	(104,80)	(104,72)

TABLE III

NONLINEARITY AND AUTOCORRELATION VALUES ACHIEVED FOR 8 BY M S-BOXES

VI. CONCLUSIONS

Comparison with theoretical approaches is difficult. On specific criteria it is clear that the derived S-boxes are not optimal. Nyberg, for example, has demonstrated 8×8 S-boxes with nonlinearity 112. For present purposes we note that spectrum-based cost functions have promise and have provided improvements on previous optimisation-based work.

Single-output Boolean functions have been a highly important application area for evolutionary and other metaheuristic search. Our techniques have already equalled and exceeded the combined achievements of theoreticians for the case of functions on small numbers of inputs (for $n \leq 8$). For some higher number of inputs the techniques have also produced functions unattained by other means. For larger numbers of inputs theoreticians clearly have the upper hand.

S-boxes, the multiple-output variant, are relatively unexplored and present significant challenges for the evolutionary commuting community. The search spaces involved are clearly vast. For example, although an 8×8 S-box would generally be considered 'small', there are actually 256^{256} possible S-boxes of this size (each of 256 inputs can take any of 256 possible outputs). The message is very clear: S-box design gets hard very quickly. There has been considerable interest in S-boxes in the cryptography community for many years; the problem is a real-world design one. The design problems can be clearly stated and the success criteria can be easily measured. They seem ideal tests problems for evaluation of evolutionary and other nature inspired techniques.

We encourage the evolutionary computing community to attack this problem. The results reported here are targets to attack.

REFERENCES

[1] E. Biham and A. Shamir. Differential Cryptanalysis of DES-like Cryptosystems (Extended Abstract). In Alfred J. Menezes and Scott A. Vanstone, editors, *Advances in Cryptology - Crypto '90*, pages 2–21, Berlin, 1990. Springer-Verlag. Lecture Notes in Computer Science Volume 537.

[2] John A Clark and Jeremy L Jacob. Two Stage Optimisation in the Design of Boolean Functions. In Ed Dawson, Andrew Clark, and Colin Boyd, editors, *5th Australasian Conference on Information Security and Privacy, ACISP 2000*, pages 242–254. Springer Verlag LNCS 1841, july 2000.

[3] John A. Clark, Jeremy L. Jacob, and Susan Stepney. Searching for cost functions. In *CEC 2004: International Conference on Evolutionary Computation, Portland, USA, June 2004*, 2004. (these proceedings).

[4] John A Clark, Jeremy L Jacob, Susan Stepney, Subhamoy Maitra, and William Millan. Evolving Boolean Functions Satisfying Multiple Criteria. In *Progress in Cryptology - INDOCRYPT 2002*, pages 246–259. Springer Verlag LNCS 2551, 2002.

[5] Howard Heys. A tutorial on linear and differential cryptanalysis. Technical report, Electrical and Computer Engineering, University of Newfoundland, St. John's, Newfoundland, Canada.

[6] Subhamoy Maitra John A Clark, Jeremy L Jacob and Pantelimon Stanica. Almost Boolean functions: the Design of Boolean Functions by Spectral Inversion. In *Conference on Evolutionary Computation — CEC-03*, December 2003.

[7] S. Kirkpatrick, Jr. C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science*, 220(4598):671–680, May 1983.

[8] M. Matsui. Linear Cryptanalysis Method for DES Cipher. In Tor Helleseth, editor, *Advances in Cryptology - EuroCrypt '93*, pages 386–397, Berlin, 1993. Springer-Verlag. Lecture Notes in Computer Science Volume 765.

[9] W. Millan. Smart Hill Climbing Finds Better Boolean Functions. In *Boolean Functions Workshop on Selected Areas on Cryptography, SAC 97*, pages 50–63, 1997.

[10] W Millan. How to Improve the Non-linearity of Bijective S-boxes. In C. Boyd and E. Dawson, editors, *3rd Australian Conference on Information Security and Privacy*, pages 181–192. Springer-Verlag, April 1998. Lecture Notes in Computer Science Volume 1438.

[11] W. Millan, L. Burnett, G. Carter, A. Clark, and E. Dawson. Evolutionary Heuristics for Finding Cryptographically Strong S-Boxes. In *ICICS 99*, 1999.

[12] W. Millan, A. Clark, and E. Dawson. Boolean Function Design Using Hill Climbing Methods. In Bruce Schneier, editor, *4th Australian Conference on Information Security and Privacy*. Springer-Verlag, april 1999. Lecture Notes in Computer Science Volume 1978.

[13] William Millan, Andrew Clark, and Ed Dawson. Heuristic Design of Cryptographically Strong Balanced Boolean Functions. In *Advances in Cryptology EUROCRYPT'98*, pages 489–499. Springer Verlag LNCS 1403, 1998.

[14] National Bureau of Standards. Data Encryption Standard. *NBS FIPS PUB 46*, 1976.

Appendix A - Description of Simulated Annealing

In 1983 Kirkpatrick et al. [7] proposed *simulated annealing*, a new search technique inspired by the cooling processes of molten metals. It merges hill-climbing with the probabilistic acceptance of non-improving moves. The basic algorithm is shown in Figure 2. The search starts at some initial state $S := S_0$. There is a control parameter T known as the temperature. This starts 'high' at T_0 and is gradually lowered. At each temperature, a number MIL (Moves in Inner Loop) of moves to new states are attempted. A candidate state Y is randomly selected from the neighborhood $N(S)$ of the current state. The change in value, δ, of f is calculated. If it improves the value of $f(S)$ (i.e., if $\delta < 0$ for a minimisation problem) then a move to that state is taken is taken ($S = Y$); if not, then it is taken with some probability. The worse a move is, the less likely it is to be accepted. The lower the temperature T, the less likely is a worsening move to be accepted. Probabilistic acceptance is determined by generating a random value U in the range $(0 \ldots 1)$ and performing the indicated comparison. Initially the temperature is high and virtually any move is accepted.

As the temperature is lowered it becomes ever more difficult to accept worsening moves. Eventually, only improving moves are allowed and the process becomes 'frozen'. The algorithm terminates when the stopping criterion is met. Common stopping criteria, and the ones used for the work in this paper, are to stop the search after a fixed number $MaxIL$ of inner loops have been executed, or else when some maximum number MUL of consecutive unproductive inner loops have been executed (i.e., without a single move having been accepted). Generally the best state achieved so far is also recorded (since the search may actually move out of it and subsequently be unable to find a state of similar quality). At the end of each inner loop the temperature is lowered. The simplest way of lowering the temperature is to multiply by a constant cooling factor α in the range $(0 \ldots 1)$; this is known as *geometric cooling*. The basic simulated annealing algorithm has proven remarkably effective over a range of problems.

```
$S := S_0$
$T := T_0$
**repeat**
{
    for (int $i = 0$; $i < MIL$; $i++$)
    {
        select $Y \in N(S)$
        $\delta := f(Y) - f(S)$
        if ($\delta < 0$) then
            $S := Y$
        else
            generate $U := rnd(0,1)$
            if ($U < \exp(-\delta/T)$) then $S := Y$
    }
    $T = T \times \alpha$
}
**until** stopping criterion is met
```

Fig. 2. Basic Simulated Annealing for Minimization Problems