Soft Nonlinearity Constraints and their Lower-Arity Decomposition

Venkatesh Ramamoorthy
Array Networks, Inc.
Milpitas, CA 95035, U.S.A
venkatesh@arraynetworks.net

Marius C. Silaghi
Florida Institute of Technology
Melbourne, FL 32901, U.S.A
msilaghi@cs.fit.edu

Toshihiro Matsui
Nagoya Institute of Technology
Nagoya, Aichi, 466-8555, Japan
matsui.t@nitech.ac.jp

Katsutoshi Hirayama
Kobe University
Kobe, 657-8501, Japan
hirayama@maritime.kobe-u.ac.jp

Makoto Yokoo
Kyushu University, Hakozaki 6-10-1
Higashi-ku, Fukuoka, 812-8581, Japan
yokoo@is.kyushu-u.ac.jp

Abstract

In this paper we express nonlinearity requirements in terms of soft global n-ary constraints. We describe a method to project global nonlinearity constraints into redundant lower-arity hard constraints. The nonlinearity constraints apply to the inputs and outputs of discrete functions $f : \mathbb{Z}_{2^n} \rightarrow \mathbb{Z}_{2^m}$ mapping n-bit inputs to m-bit outputs, $n > m$. No output bit (or linear function on a subset of output bits) of the function f should be too close to a linear function of (a subset of) its input bits. For example, if we select any output bit position and any subset of the six input bit positions, the fraction of inputs for which this output bit equals the exclusive-OR of these input bits should not be close to 0 or 1, but rather should be near $\frac{1}{2}$. We analyze this constraint and find that the obtained redundant constraints increase the efficiency of an arc consistency maintenance solver by several orders of magnitude.

Keywords: Constraints and Search, CSP Model, Soft Constraint, S-boxes, DES, 3DES, Nonlinearity, Cryptanalysis, Global Constraint, Projection, Optimization, n-ary Constraint

Introduction

The design of S-box functions is crucial to the security of modern symmetric cryptosystems and we show how to improve by orders of magnitude the efficiency of an approach to the automatic generation of such functions. The nonlinearity constraint is proposed in (Ramamoorthy et al. 2011) to model nonlinearity requirements that are essential for the security of cryptographic algorithms (ciphers). If substitution operations in ciphers could be represented as linear relations, their parameters could be easily obtained by solving a system of such equations connecting pairs of inputs and outputs. Even when the functions are not perfectly linear, any success in approximating them with linear functions can increase the chances of success in guessing their parameters. As such, (Coppersmith 1994) defines one of the main nonlinearity requirements as: *No output bit of the function f should be too close to a linear function of the input bits.* The requirement is extended in (Matsui 1994) to the relation of any linear combination of any subset of output bits to any linear combination of any subset of input bits. In this paper, we discuss our formulation of the nonlinearity constraint as a soft global n-ary constraint of a constraint satisfaction problem (CSP) (Rossi, Beek, and Walsh 2006) and prove a method to obtain its projections on subsets of the variables as a set of hard, redundant constraints. The nonlinearity constraints apply the possible inputs and outputs of discrete functions $f : \mathbb{Z}_{2^n} \rightarrow \mathbb{Z}_{2^m}$ which map n-bit inputs to m-bit outputs, $n > m$. Such functions are commonly referred to as Substitution boxes (S-boxes) and their design is an important problem. We analyze these constraints and find that the obtained redundant constraints increase the efficiency of an arc consistency maintenance solver (Bessière and Régin 2001) by orders of magnitude.

13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Figure 1: The 3DES 6×4 S-box S_8

Background

We now introduce the nonlinearity requirement, originally defined in (Coppersmith 1994).

S-box Criteria and Nonlinearity An $n \times m$ substitution box (S-box) that scrambles (substitutes) an n-bit input data to yield an m-bit output, is a function $S : \mathbb{Z}_{2^n} \rightarrow \mathbb{Z}_{2^m}$ where $\mathbb{Z}_k$ stands for the set of integers $\{0, ...k-1\}$ modulo-k. S is not necessarily invertible. Substitution-permutation networks (Shannon 1949), a common cipher architecture, use S-boxes in the generation of a parametrized substitution of x. These S-boxes have to be nonlinear and invertible. One of the SP-network versions, the Feistel cipher architecture (Feistel 1973), relaxes this constraint by removing the invertibility requirement. Namely, it proceeds through iterations of the function:

$$F : \mathbb{Z}_{2^m} \times \mathbb{Z}_{2^m} \rightarrow \mathbb{Z}_{2^m} \times \mathbb{Z}_{2^m},$$

$$F(x, y) = split(y \cdot 2^m + f(x, E_K(y)), m, m).$$

Here, $E_K : \mathbb{Z}_{2^m} \rightarrow \mathbb{Z}_{2^m}$ is a linear expansion function that mixes the encryption key K. The function f is:

$$f : \mathbb{Z}_{2^m} \times \mathbb{Z}_{2^n} \rightarrow \mathbb{Z}_{2^m}, f(x, y) = x \oplus S(y),$$

S-1 Each S-box has six bits of input and four bits of output.

S-2 No output bit of an S-box should be too close to a linear function of the input bits. (That is, if we select any output bit position and any subset of the six input bit positions, the fraction of inputs for which this output bit equals the exclusive-OR of these input bits should not be close to 0 or 1, but rather should be near $\frac{1}{2}$).

S-3 If we fix the leftmost and rightmost input bits of the S-box and vary the four middle bits, each possible 4-bit output is attained exactly once as the middle four input bits range over their 16 possibilities.

S-4 If two inputs to an S-box differ in exactly one bit, the corresponding outputs must differ in at least two bits.

S-5 If two inputs differ in the two middle bits exactly, the outputs must differ in at least two bits.

S-6 If two inputs differ in the first two bits and are identical in the last two bits, the two outputs must be different.

S-7 For any nonzero 6-bit difference between inputs $\Delta I_{i,j}$, no more than eight of the 32 pairs of inputs exhibiting $\Delta I_{i,j}$ may result in the same output difference $\Delta O_{i,j}$.

Table 1: The nonlinearity criteria used by IBM for designing 3DES S-boxes (Coppersmith 1994)

and the function *split* is:

$$split : \mathbb{Z}_{2^{n+m}} \times \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{Z}_{2^n} \times \mathbb{Z}_{2^m},$$
$$split(z, n, m) = \left(\lfloor \frac{z}{2^m} \rfloor, z - 2^m \cdot \lfloor \frac{z}{2^m} \rfloor \right)$$

which splits the $(n+m)$-bit number z in two parts of n bits and m bits respectively, $\mathbb{Z}$ being the set of (non-negative) integers. The obtained substitution function $f(x, \cdot)$ is a parametrized bijection, and it is therefore often referred to as a permutation function. This function of parameter y should necessarily be nonlinear (Coppersmith 1994). Since it is difficult to design and verify a nonlinear S-box function that works on a large sequence of bits n, cipher designers replace it with a set of smaller S-boxes, where each of them handles a fraction of the sequence of bits.

One of the most commonly-used ciphers in Netscape's Secure Sockets Layer (SSL) protocol and the newer Transport Layer Security (TLS) protocol is Triple-DES (3DES), which is an example of a Feistel architecture. Other known Feistel architectures are Blowfish, Twofish, RC5, Camellia, etc. 3DES works by a triple-application of the old Data Encryption Standard (DES) developed by IBM (DES 1988). It employs eight 6×4 S-boxes numbered $S_1, S_2, \ldots, S_8$, with S_8 shown in Figure 1. Each 6×4 S-box has 4 rows and 16 columns. The row and column numbers begin from 0 rather than 1. An S-box substitution of 4 bits for a 6-bit input i is obtained by indexing into the row number formed by the first and last bits of i, and the column number formed by the remaining middle bits of i. For example, input of 45 ($= 101101_2$) to S-Box S_8 yields $8 = (1000_2)$, obtained by reading the entry in the fourth row (which is row number $3 = 11_2$), seventh column (or column number $6 = 0110_2$) of Figure 1. The S-boxes are so designed to satisfy criteria numbered **S-1**, **S-2**, and so on (Coppersmith 1994), which are listed in Table 1. The S-box nonlinearity constraint **S-2** states that *the output of an S-box should be highly nonlinear*. A proposal by Matsui (Matsui 1994) compiles this criteria into a complex metric but which allows for a quantitative comparison of S-boxes. This is the metric that we employ here under the form of a soft global nonlinearity n-ary constraint.

Concepts and Problem Formulation

Notations A `0x` prefixed to the left of a number, such as `0x2ab3`, specifies it is in hexadecimal notation. $|x|$ denotes the absolute value of a number x. For a set S, $|S|$ represents its cardinality while for a set expressed using braces, its cardinality is denoted by preceding the braces with a $\#$. The symbols $\cdot$ and $\oplus$ represent the bit-wise AND and exclusive-OR (XOR) operation respectively, on two identical-sized bit patterns. A linear Boolean function $L_\omega(x)$ on an n-bit pattern $x = x_0 \ldots x_{n-1}$ selected by an n-bit pattern $\omega = \omega_0 \ldots \omega_{n-1}$ is defined (Clark et al. 2003) as:

$$L_\omega(x) = \omega_0 \cdot x_0 \oplus \ldots \oplus \omega_{n-1} \cdot x_{n-1} = \bigoplus_{i=0}^{n-1} \omega_i \cdot x_i \quad (1)$$

The *Hamming weight* of a bit pattern x, denoted by $wt(x)$, is equal to the number of **1**'s in x. The amount by which x and y differ, as mentioned in Table 1, equals $wt(x \oplus y)$.

Variables and Domains

We now define the elements of the (X, D, C)-based CSP model for the general case of designing a nonlinear and non-invertible $n \times m$ S-box. Our concrete examples are for 6×4 S-boxes such as those used in 3DES. Note that invertible S-boxes can be obtained when $n = m$ by simply adding an `alldiff` constraint, which makes the function one to one.

	$i_1 i_2 i_3 i_4$							
$i_0 i_5$	0	1	2	3	...	13	14	15
0	x_0	x_2	x_4	x_6	$\ldots$	x_{26}	x_{28}	x_{30}
1	x_1	x_3	x_5	x_7	$\ldots$	x_{27}	x_{29}	x_{31}
2	x_{32}	x_{34}	x_{36}	x_{38}	$\ldots$	x_{58}	x_{60}	x_{62}
3	x_{33}	x_{35}	x_{37}	x_{39}	$\ldots$	x_{59}	x_{61}	x_{63}

Figure 2: Diagrammatic relationship between the defined CSP variables and 6×4 S-box entries

To model our nonlinearity criteria, we define the set X of 2^n variables $X = \{x_0, x_1, \ldots, x_{2^n-1}\} = \{x_i | i \in \mathbb{Z}_{2^n}\}$, each representing an entry in the S-box. The domain in D of each variable is $\mathbb{Z}_{2^m} = \{0, 1, \ldots, 2^m - 1\}$.

To adapt the CSP for our case study of $n \times m$ S-boxes, the i^{th} variable x_i specifies the m-bit S-box output for an n-bit input i. Using the variables in X, a 6×4 S-box such as the ones used in 3DES, is organized as shown in Figure 2, addressed by incrementing the input. In Figure 2, a 6-bit input $i, 0 \leq i \leq 63$ is represented by the bit pattern $i_0 i_1 i_2 i_3 i_4 i_5$ for clarity. Criterion **S-1** in Table 1 is already satisfied based on our choice of variables.

Nonlinearity Metrics for Variable Assignments

Since for each input i the S-box returns the value of x_i, therefore the nonlinearity of the S-box can be stated as a nonlinearity between each index i and the value of x_i. The ability of expressing each bit of an m-bit value $e \in \mathbb{Z}_{2^m}$ in the assignment $x_i = e$, as a linear combination of the bits in the n-bit subscript $i \in \mathbb{Z}_{2^n}$ (Matsui 1994; Heys 2002), is now examined. Here, we use this measure as the score of a solution (to be optimized) and extend the definition to a partial assignment.

Consider an n-bit subscript $i = i_0 \ldots i_{n-1}$ of a variable x_i, and a corresponding assignment to x_i of a value from $\mathbb{Z}_{2^m}$. The linear combinations to be checked for equality are obtained by selecting bits in i and the value assigned to x_i using selectors a and b respectively, $\forall a, b, 0 \leq a < 2^n$ and $0 \leq b < 2^m$. We denote, by $L_\omega(x_i)$, the application of the function L_ω of Equation 1 on the value assigned to the CSP variable x_i. For a complete assignment Φ with all variables in X assigned, let $N_X^\Phi(a, b)$, quantifying the *success of linearization* of the relation between i to x_i using coefficients a and b, be:

$$N_X^\Phi(a,b) = \#\{i | x_i \in X; L_a(i) = L_b(x_i)\} \tag{2}$$

Observe that $0 \leq N_X^\Phi(a, b) \leq 2^n$.

Given a partial-assignment Φ' resulting from a partial instantiation of variables $X' \subseteq X$, we further define the *partial success of linearization* $N_{X'}^{\Phi'}(a, b)$ as follows:

$$N_{X'}^{\Phi'}(a,b) = \#\{i | x_i \in X'; L_a(i) = L_b(x_i)\} \tag{3}$$

Besides the properties for $N_X^\Phi(a, b)$ (Matsui 1994), the following properties are also inferred directly from the definition of $N_{X'}^{\Phi'}(a, b)$.

Property 1 $\forall a,\ b,\ X',\ \Phi',\ 0 \leq N_{X'}^{\Phi'}(a, b) \leq |X'|$.

Property 2 $\forall a, b, u, X', \Phi',$ *and* $u \in X \setminus X'$,

$$N_{X' \cup \{u\}}^{\Phi'}(a, b) - N_{X'}^{\Phi'}(a, b) \in \{0, 1\}.$$

Proof. Property 2 states the immediate observation that the consideration of each additional input can raise the number of correctly linearized inputs by at most 1. This reasoning applied consecutively to each variable in X' is used to explain Property 1.

Q.E.D.

These two properties are used to design heuristics that improve the efficiency of search for solutions to satisfy the nonlinearity constraint **S-2**. They place a bound on the partial success of linearization of a partially instantiated S-box and are used in the proof of the next property.

The following two paragraphs introduce two concepts used to quantify the nonlinearity of an S-box, and to define and compute the soft constraint proposed in the next section that maximizes this nonlinearity quantification .

Nonlinearity as a Probability Measure For *each* variable x_i corresponding to input i in a complete assignment Φ, given selectors a and b defined as above, let $p(a, b)$ denote the fraction of cases when $L_a(i) = L_b(x_i)$, computed as:

$$p(a,b) = \frac{N_X^\Phi(a,b)}{2^n} \tag{4}$$

$p(a, b) = 1$ is the condition where the linear combination of the bits in the value assigned to x_i selected by b equals a linear combination of the bits in i selected by a, i.e., $\forall i, L_a(i) = L_b(x_i)$. If $p(a, b)$ is equal to zero, the linear combination of the output bits selected by b is always equal to the negation of the linear combination of input bits selected by a. According to the nonlinearity requirement **S-2**, $p(a, b)$ should be near $\frac{1}{2}$.

Linear Approximation Table (LAT) The Linear Approximation Table (Matsui 1994) for a complete assignment is a $2^n \times 2^m$ matrix. Its rows are headed by selector a, $0 \leq a < 2^n$, and columns by selector b, $0 \leq b < 2^m$ (see Table 2). Each entry specifies the quantity $N_X^\Phi(a, b) - \frac{|X|}{2}$, with one entry in row a and column b representing an offsetted measure of the correlation between the bits of x_i selected by b and the bits of i selected by a. As an example, for the 3DES S-box S_8, the first and last two rows of its LAT are in Table 2. The LAT of a solution is formed by an arithmetic accumulation of individual contributions due to *each* variable assignment $x_i = e, i \in \mathbb{Z}_{2^n}, e \in \mathbb{Z}_{2^m}$. A contribution arising from an assignment $x_i = e$ is equal to $L_a(i) \oplus L_b(e) \oplus 1, a \in \mathbb{Z}_{2^n}, b \in \mathbb{Z}_{2^m}$, that is, 0 or 1. The offset quantity $\frac{|X|}{2}$ is subtracted from each entry in the LAT of a solution.

A quantification for the nonlinearity of an S-box is now introduced.

b	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
a																
0	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
62	0	-8	4	0	-2	2	-2	6	10	6	2	2	0	0	-4	0
63	0	-8	0	4	2	-2	-10	-2	-6	6	-2	6	-4	4	-4	0

Table 2: The Linear Approximation Table for the S-box S_8 of Figure 1

The Score of an Assignment The most effective linear approximation of a complete assignment Φ containing $|X|$ variables is obtained if, for some a and b, $|N_X^{\Phi}(a,b) - \frac{|X|}{2}|$ is maximal. To reduce the weakest point of the assignment Φ, we use the so-called *effectiveness of linearization* (O'Connor 1995) as the optimization score:

$$\sigma_X(\Phi) = \max_{a,b}\{|N_X^{\Phi}(a,b) - \frac{|X|}{2}| : 1 \le a < |X|; 1 \le b < |D|\} \tag{5}$$

A complete assignment with a smaller score is considered better. We look for $\underset{\Phi}{\mathrm{argmin}}(\sigma_X(\Phi))$. The score $\sigma_{X'}$, $X' \subseteq X$, of a partial assignment Φ' is defined as:

$$\sigma_{X'}(\Phi') = \max_{a,b}\{|N_{X'}^{\Phi'}(a,b) - \frac{|X|}{2}| : 1 \le a < |X|; 1 \le b < |D|\} \tag{6}$$

The Nonlinearity Global Constraint

The straightforward modeling of the nonlinearity requirement leads to a soft constraint that minimizes $\sigma_X(\Phi)$. When used as a hard constraint for a threshold τ, it becomes:

$$\sigma_X(\Phi) \le \tau \tag{7}$$

The following property of a partial assignment allows for projection of Equation 7 to lower-arity constraints.

Property 3 (Projections) *A partial assignment Φ' with values for variables in $X', X' \subseteq X$, cannot be extended to a solution with score better than a threshold τ if the following inequality is not satisfied:*

$$|X'| - \tau - \frac{|X|}{2} \le \max_{a,b} N_{X'}^{\Phi'}(a,b) \le \frac{|X|}{2} + \tau \tag{8}$$

Proof. During projection, the goal is for the score of S-box Φ' to never exceed the maximum threshold τ:

$$\max_{a,b} |N_{X'}^{\Phi'}(a,b) - \frac{|X|}{2}| \le \tau \tag{9}$$

Figure 3 depicts the distribution of $N_{X'}^{\Phi'}(a,b)$ (Equation 3) for a partially instantiated S-box Φ'. The horizontal axis is the number of variables instantiated, ϕ. After $|X'|$ variables are instantiated at point A along the solid line, the dashed line at a 45-degree angle with the horizontal represents the pathological case where the count $N_{X'}^{\Phi'}(a,b)$ increases by one for every subsequent extension of Φ' up to point D, as per Property 2. The solid zig-zag lines connecting points A and C represents the corresponding, actual distribution of $N_{X'}^{\Phi'}(a,b)$ for the complete S-box Φ to attain the count equal to $N_X^{\Phi}(a,b)$ at point C. From this construction, we have $OF = N_{X'}^{\Phi'}(a,b)$, $OG = N_X^{\Phi}(a,b)$, $FH = BD = AB = |X| - |X'|$, and $OH = OF + FH = N_{X'}^{\Phi'}(a,b) + |X| - |X'|$.

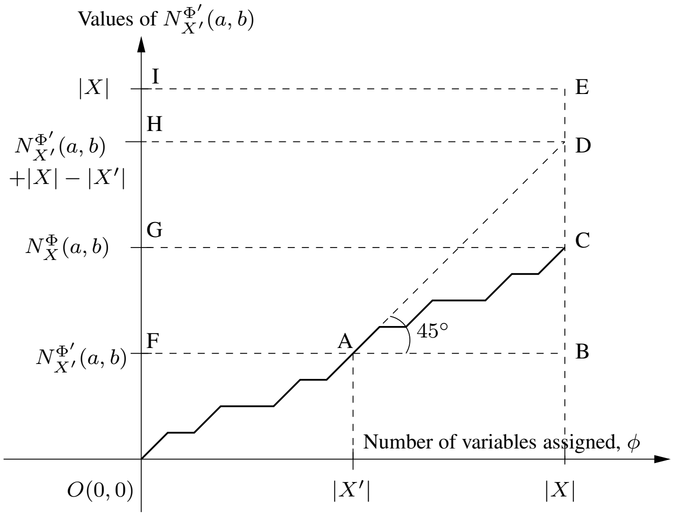

Figure 3: Evaluating partially instantiated S-boxes.

By construction, $(|X| - |X'|)$ remaining variables are to be instantiated in order to extend Φ' to Φ. To guarantee extensibility: $OG \le OH$, i.e.,

$$N_X^{\Phi}(a,b) \quad \le \quad N_{X'}^{\Phi'}(a,b) + |X| - |X'|$$

This is true for all selectors a and b, and in particular, holds for the maximum value of $N_X^{\Phi}(a,b)$ (resp. $N_{X'}^{\Phi'}(a,b)$) over all a, b:

$$\max_{a,b} N_X^{\Phi}(a,b) \le |X| - |X'| + \max_{a,b} N_{X'}^{\Phi'}(a,b) \tag{10}$$

From Equation 9, $\frac{|X|}{2} - \max_{a,b} N_X^{\Phi}(a,b) \le \tau$

$$\text{i.e. } \frac{|X|}{2} - \tau \quad \le \quad \max_{a,b} N_X^{\Phi}(a,b) \tag{11}$$

Combining Equation 10 and Equation 11,

$$\frac{|X|}{2} - \tau \le \max_{a,b} N_X^{\Phi}(a,b) \le |X| - |X'| + \max_{a,b} N_{X'}^{\Phi'}(a,b) \tag{12}$$

By transitivity and regrouping, $\max_{a,b} N_{X'}^{\Phi'}(a,b) \ge \frac{|X|}{2} - \tau - |X| + |X'|$

$$\text{i.e. } \max_{a,b} N_{X'}^{\Phi'}(a,b) \quad \ge \quad |X'| - \tau - \frac{|X|}{2} \tag{13}$$

Given a partial S-box assignment Φ' with variables in X', by the end of the construction of any solution Φ obtained by extending Φ', the following inequality holds: $OF \le OG$.

$$\text{i.e. } N_{X'}^{\Phi'}(a,b) \quad \le \quad N_X^{\Phi}(a,b) \tag{14}$$

0	3	5	6	9	10	15	12	7	4	14	13	2	1	8	11
3	0	6	5	10	9	12	15	4	7	13	14	1	2	11	8
3	15	0	12	5	6	9	10	4	8	7	11	14	13	2	1
0	12	3	15	9	10	5	6	7	11	4	8	2	1	14	13

Figure 4: A 6×4 S-box with score 8, generated by our CSP solver

This is true for all selectors a and b, and in particular, holds for the maximum value of $N_{X'}^{\Phi'}(a,b)$ (resp. $N_X^{\Phi}(a,b)$) over all a, b:

$$\max_{a,b} N_{X'}^{\Phi'}(a,b) \leq \max_{a,b} N_X^{\Phi}(a,b) \tag{15}$$

From Equation 9, $\max_{a,b} N_X^{\Phi}(a,b) - \frac{|X|}{2} \leq \tau$

$$\text{i.e. } \max_{a,b} N_X^{\Phi}(a,b) \quad \leq \quad \frac{|X|}{2} + \tau \tag{16}$$

Combining Equations 15 and 16,

$$max_{a,b} N_{X'}^{\Phi'}(a,b) \leq \max_{a,b} N_X^{\Phi}(a,b) \leq \frac{|X|}{2} + \tau \tag{17}$$

The result follows by combing Equation 13 and Equation 17. **Q.E.D.**

Results

The experimentation setup consists of an Intel Pentium Core-2 Duo 3-GHz CPU, 3.3 GB RAM and GNU/Linux with kernel version 2.6.28-11. The constraints are precompiled for DES criteria **S-3**, **S-4**, **S-5**, **S-6** and **S-7**. The precompiled constraints are fed to our implementation of a solver that supports Maintenance of Arc Consistency (MAC) with AC2001 (Bessière and Régin 2001). The soft constraint of Equation 7 modeling **S-2** is transformed into a hard constraint by setting the threshold value for τ. We experiment with $\tau = 16$ and $\tau = 10$.

Better-quality S-boxes based on the score The score for the standard 3DES S-box S_4 is found to be 10 (minimum), while the score for S_7 is 18 (maximum). *Our approach yielded S-boxes with score 8, superior in quality to any of the standard 3DES S-boxes.* Figure 4 reports one such S-box.

Performance Statistics The MAC solver is initially started only with the binary constraints. We test three heuristics for integrating the n-ary constraints in this solver. Recall that τ is the threshold value for the S-box score (Equation 7 and Equation 8), used in the notations that follow.

- *Complete, Non-incremental heuristic, H_S^τ.* This is the basic case where the n-ary constraint for **S-2** is checked only after all assignments, without using them in any domain-filtering.
- *Incomplete, Incremental heuristic, H_I^τ.* At each node in the search tree, incrementally assign and check if the constraint in Equation 7 is partially satisfied. On violation, abandon the assignment and proceed with the next one.
- *Complete, Incremental heuristic H_C^τ.* At each node in the search tree, project the constraint in Equation 7 by enforcing Property 3 on the current partial assignment.

Within the first hour, with a threshold $\tau = 10$ specified, the incomplete, incremental heuristic H_I^{10} found around $3,600$ 6×4 S-boxes with the "best" score equal to 8. This count went up to more than $13,500$ in the 5-hour run that Table 3 reports.

Although this heuristic yields S-boxes with the "best" score, it is not complete. In order to know whether we have found the optimal quality S-boxes we would have to exhaust the whole search space. If the search space is too large to be exhausted, we would like to at least know what fraction of this search space we have managed to explore, as a measure of how promising the current heuristic is (given that it may produce no other output) and as a weak estimation of the certainty that the best solution found so far is close to the optimal solution. Only heuristics traversing the same search space can be meaningfully compared in this way. This is the case for search heuristics proceeding using the same order on variables and values.

We therefore quantify the size of the search space, as the total number of potential S-boxes. As shown later, the search space of our problem instances is very large, and additional research is needed in order to be able to exhaust it. Assuming that the solver is systematic and chronological (visiting alternatives in lexicographic order), each partial or full assignment of values to all variables (whether it satisfies the constraints or not), and visited or skipped by the search tree, is defining a traversed distance (explored search space):

$$S_p^{(n \times m)} = \sum_{i=0}^{|X'|-1} x_i \cdot (2^m)^{|X|-i-1} \tag{18}$$

Here, $X' \subseteq X$ is a set of already-instantiated variables in the current partial assignment, $x_i \in X'$ is assigned a specific value from its domain D, and in Equation 18, x_i stands for the specific value assigned to the variable x_i.

With dynamic reordering of values and variables, Equation 18 still applies and one only has to use the current order.

For 6×4 S-boxes, $S_p^{(6 \times 4)}$ evaluates to 78-digit base-10 numbers. Given the large size of this search space, distances typically covered by the MAC solver in reasonable time differed only in their last few assignments (78-digit numbers differed in approximately the last 15 digits). Sometimes, certain constraints rule out much larger areas of the search space. To conveniently report this, we define a *search offset* metric S-box $S_{p_1}^{(n \times m)}$:

$$r^{(n \times m)} = \frac{S_p^{(n \times m)} - S_{p_1}^{(n \times m)}}{2^{m \times 2^n}} \tag{19}$$

Here, $S_{p_1}^{(n \times m)}$ denotes the value for $S_p^{(n \times m)}$ (determined from Equation 18) for the first S-box obtained by the solver. The solver has yielded $S_{p_1}^{(6 \times 4)} \approx \texttt{0x033} \times \texttt{16}^{\texttt{60}}$. (The hexadecimal form is for convenience and S_{p_1} could be alternatively written in decimal.) The difference between $S_{p_1}^{(6 \times 4)}$ for the incomplete and complete heuristics is $\approx 3 \times 16^{52}$

Time	$r^{(6\times4)}\times10^{49}$	**S-box Count**	
(hrs)		$\sigma_X(\Phi)=10$	$\sigma_X(\Phi)=8$
1	$355,940$	$8,562$	$3,583$
2	$572,810,000$	$17,827$	$4,999$
3	$646,070,000$	$27,875$	$7,836$
4	$688,140,000$	$37,875$	$10,883$
5	$1,030,000,000$	$47,671$	$13,602$

Table 3: Solver Performance Using Incomplete, Incremental Heuristic H_I^{10}

Time	**Non-incremental (H_S^{16})**		**Incremental (H_C^{16})**	
(hrs)	$r^{(6\times4)}\times10^{49}$	**S-box Count**	$r^{(6\times4)}\times10^{49}$	**S-box Count**
1	1.198	4	102,160	20,786
2	21.725	14	265,040	35,957
3	42.091	15	915,420	49,110
4	42.091	26	993,950	80,933
5	61.340	40	1,061,500	94,069

Table 4: Solver Performance Using Complete Heuristics, with S-box threshold $\tau = 16$.

even when they use the same value for τ (graphs not shown due to lack of space). Table 3 reports the (scaled) search offsets of the solver using incomplete heuristics.

Performance Analysis of the three Heuristics Table 3 reports performance of the incomplete, incremental heuristic, with threshold $\tau = 10$. Table 4 compares the non-incremental and complete, incremental heuristics. The quantities reported at each hour represent the (scaled) fraction r of Equation 19, and the number of S-boxes generated up to that point in each case. Based on a 5-hour run of the experiment, the complete, incremental heuristic is observed to relatively vary between a factor of 17 and 85 times faster than the non-incremental heuristic (in terms of size of explored search space). The number of S-boxes generated is observed to correspondingly increase by an average factor of over 3,300. We tried all values of τ and report in Table 4 only for $\tau = 16$.

Conclusion

The soft global nonlinearity n-ary constraint, is projected onto fewer variables and thereby applied for dynamic domain filtering during search. This heuristic yielded a 17–85-fold relative increase in 6×4 S-box generation efficiency. The technique can be used for the design of S-boxes of any size, an important step towards the automatic design of new cryptosystems guaranteed to resist to known attacks.

References

Bessière, C., and Régin, J.-C. 2001. Refining the basic constraint propagation algorithm. In Nebel, B., ed., *IJCAI*, 309–315. Morgan Kaufmann.

Clark, J.; Jacob, J.; Maitra, S.; and Stanica, P. 2003. Almost boolean functions: the design of boolean functions by spectral inversion. *Evolutionary Computation* 3:2173–2180 Vol.3.

Coppersmith, D. 1994. The data encryption standard (DES) and its strength against attacks. *IBM J. Res. Dev.* 38(3):243–250.

1988. Data encryption standard (DES). Federal Information Processing Standard 46-2.

Feistel, H. 1973. Cryptography and computer privacy. 228:15–23.

Heys, H. M. 2002. A tutorial on linear and differential cryptanalysis. *Cryptologia* XXVI(3):189–221.

Matsui, M. 1994. Linear cryptanalysis method for des cipher. In *EUROCRYPT '93: Workshop on the theory and application of cryptographic techniques on Advances in cryptology*, 386–397. Springer-Verlag.

O'Connor, L. 1995. Properties of linear approximation tables. 1008.

Ramamoorthy, V.; Silaghi, M.; Matsui, T.; Hirayama, K.; and Yokoo, M. 2011. The design of cryptographic S-Boxes using CSPs. In *CP*.

Rossi, F.; Beek, P. v.; and Walsh, T. 2006. *Handbook of Constraint Programming (Foundations of Artificial Intelligence).* New York, NY, USA: Elsevier Science Inc.

Shannon, C. E. 1949. A mathematical theory of communication. *Bell System Technical Journal* 28:656–715.