Evolving S-boxes Based on Cellular Automata with Genetic Programming

Stjepan Picek
KU Leuven, imec-COSIC
Kasteelpark Arenberg 10
3001 Leuven, Belgium
stjepan@computer.org

Luca Mariot
Alberto Leporati
luca.mariot@disco.unimib.it
alberto.leporati@unimib.it
DISCo, Università degli Studi di Milano-Bicocca
Viale Sarca 336/14
20126 Milano, Italy

Domagoj Jakobovic
University of Zagreb, Faculty of Electrical Engineering and Computing
Unska 3, Zagreb, Croatia
domagoj.jakobovic@fer.hr

ABSTRACT

The design of cryptographically strong Substitution Boxes (S-boxes) is an interesting problem from both a cryptographic perspective as well as the combinatorial optimization one. Here we introduce the concept of evolving cellular automata rules that can be then translated into S-boxes. With it, we are able to find optimal S-boxes for sizes from 4×4 up to 7×7. As far as we know, this is the first time a heuristic approach is able to find optimal S-boxes for sizes larger than 4.

CCS CONCEPTS

•**Computing methodologies** → **Discrete space search;** •**Security and privacy** → *Block and stream ciphers;*

KEYWORDS

Substitution boxes, Genetic Programming, Cellular automata, Cryptography

ACM Reference format:
Stjepan Picek, Luca Mariot, Alberto Leporati, and Domagoj Jakobovic. 2017. Evolving S-boxes Based on Cellular Automata with Genetic Programming. In *Proceedings of GECCO '17 Companion, Berlin, Germany, July 15-19, 2017,* 2 pages.
DOI: http://dx.doi.org/10.1145/3067695.3076084

1 INTRODUCTION

In the process of designing block ciphers, one well explored direction is to build the so-called Substitution-Permutation Network (SPN) cipher. Such ciphers usually consist of an XOR operation with the key/subkeys, a linear layer, and a substitution layer [3]. A standard way to build the substitution layer is to use one or more Substitution Boxes (S-boxes) where a number of properties need to be satisfied for an S-box to be useful in practice. An S-box, or (n, m) function, is a mapping from n inputs into m outputs.

From the cryptographic properties perspective, the bare minimum one would need to consider when designing S-boxes with the same number of inputs and outputs (as we consider in this paper) is for them to be bijective, with high nonlinearity, and low differential uniformity. When utilizing heuristics in the design of S-boxes, a common approach is to use the permutation encoding since it ensures the bijectivity property. However, already for the size 5×5 heuristics are not able to reach the optimal values of nonlinearity and differential uniformity and therefore algebraic constructions are the common method of choice [5]. However, there are several ciphers that use S-boxes not directly obtained by algebraic constructions or heuristics, but where the S-boxes are actually cellular automata (CA). The best known example is the Keccak sponge construction that is now part of the SHA-3 standard [1].

In this paper, we focus on the investigation of CA rules that are able to produce S-boxes with optimal cryptographic properties. In particular, we use Genetic Programming (GP) to evolve CA local rules in the form of Boolean functions, by viewing the corresponding CA as S-boxes. With our approach we are able to produce large quantities of S-boxes with optimal cryptographic properties, dependent on the chosen optimization criteria.

2 S-BOXES AND CELLULAR AUTOMATA

Let n, m be positive integers, i.e., $n, m \in \mathbb{N}^+$. The set of all n-tuples of elements in the field $\mathbb{F}_2$ is denoted as $\mathbb{F}_2^n$, where $\mathbb{F}_2$ is the Galois field with two elements. The inner product of two vectors a and b from $\mathbb{F}_2^n$ equals $a \cdot b = \bigoplus_{i=1}^n a_i b_i$ where "$\bigoplus$" represents addition modulo two. An (n, m)-function is any mapping F from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$. For any set S, we denote $S \backslash \{0\}$ by S^*.

An (n, m)-function F is *balanced* if it takes every value of $\mathbb{F}_2^m$ the same number 2^{n-m} of times.

The *nonlinearity* N_F of an (n, m)-function F equals the minimum nonlinearity of all its component functions $v \cdot F$, where $v \in \mathbb{F}_2^{m*}$ [2]:

$$N_F = 2^{n-1} - \frac{1}{2} \max_{\substack{a \in \mathbb{F}_2^n \\ v \in \mathbb{F}_2^{m*}}} |W_F(a, v)|. \tag{1}$$

Here, $W_F(a, v)$ is the Walsh-Hadamard transform of an (n, m)-function F, defined as:

$$W_F(a, v) = \sum_{x \in \mathbb{F}_2^n} (-1)^{v \cdot F(x) \oplus a \cdot x}, \quad a, v \in \mathbb{F}_2^m. \tag{2}$$

Let $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^m$, $a \in \mathbb{F}_2^n$ and $b \in \mathbb{F}_2^m$. We denote:

$$D_F(a, b) = \left\{ x \in \mathbb{F}_2^n : F(x) + F(x + a) = b \right\}. \tag{3}$$

The entry at the position (a, b) corresponds to the cardinality of the delta difference table $D_F(a, b)$. The *differential uniformity* δ_F is then defined as [4]:

$$\delta_F = \max_{a \neq 0, b} D_F(a, b). \tag{4}$$

Cellular automata (CA) are a parallel computational model which can be represented as a regular grid of cells, such that each cell synchronously updates its state according to a local rule, which depends on a specific number of neighboring cells. In this paper we consider a CA model where the cells are arranged on a finite periodic one-dimensional array and are described by a binary state, 0 or 1.

Formally, let $m, n \in \mathbb{N}$ with $m \geq n$, and let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ be a Boolean function. A *cellular automaton* (CA) of length m with *local rule* f and *periodic boundary condition* is a vectorial Boolean function $F : \mathbb{F}_2^m \to \mathbb{F}_2^m$ defined for all $x = (x_1, \cdots, x_m) \in \mathbb{F}_2^m$ as:

$$F(x_1, \cdots, x_m) = (f(x_1, x_2, \cdots, x_n), \cdots, f(x_m, x_1, \cdots, x_{n-1})) . \tag{5}$$

In the rest of the paper we assume $m = n$, and we identify F (the CA) with an S-box.

3 GP APPROACH AND RESULTS

We use GP to evolve Boolean functions of n variables, in the form of trees, which are used as CA local rules. In this process, we assume that the state of a CA is represented as a periodic one-dimensional binary array of size n. The elements of the binary array are used as GP terminals, where the terminal c_0 denotes the value that is being updated. The terminals $c_1, \ldots, c_{n-1}$ denote the cells to the right of the current cell. In our experiments, the neighborhood of a cell is formed by the cell itself and the $n-1$ cells to its right, so each value in the current state can be used in a local update rule. A candidate Boolean function obtained with GP is evaluated in the following manner: all the possible 2^n input states are considered, and for each state the same rule is applied in parallel to each of the bits to determine the next state. The obtained global rule represents a candidate S-box that is then evaluated according to the desired cryptographic criteria.

We use the following function set: NOT, which inverts its argument, XOR, NAND, NOR, each of which takes two input arguments. Additionally, we use the function IF, which takes three arguments and returns the second one if the first one evaluates to *true*, and the third one otherwise. In the evolution process, GP uses a 3-tournament selection and mutation with a probability of 0.5. The variation operators are simple tree crossover, uniform crossover, size fair, one-point, and context preserving crossover [6] (selected at random) and subtree mutation. The initial population is created at random with ramped half-and-half initialization; every experiment is repeated 50 times and the population size equals 2 000.

In the fitness function first the balancedness is verified, and if an S-box is balanced, we give it a value of zero, otherwise the value equals -1; this is denoted with the label BAL. If the S-box is balanced, we calculate the nonlinearity and differential uniformity (which is subtracted from the value 2^n, since we aim to minimize the value of that property) and **maximize** the resulting value:

$$fitness = BAL + \Delta_{BAL,0}(N_F + (2^n - \delta_F)). \tag{6}$$

Table 1: Statistical results and comparison.

S-box size	T_max	GP			Ours		Related work	
		Max	Avg	SD	N_F	δ_F	N_F	δ_F
4×4	16	**16**	16	0	4	4	4	4
5×5	42	**42**	41.73	1.01	**12**	**2**	10	4
6×6	86	**84**	80.47	4.72	**24**	**4**	22	6
7×7	182	**182**	155.07	8.86	**56**	**2**	48	6
8×8	364	318	281.87	13.86	82	20	**104**	**8**

$\Delta_{BAL,0}$ represents the Kronecker delta function that equals one when the function is balanced (i.e., $BAL = 0$) and zero otherwise.

In Table 1 we give the statistical results and compare the best values obtained here with the state-of-the-art results obtained with EC [5], where better values are in bold style. Statistical results are averaged over the best obtained values for each run, and column T_max denotes the theoretical maximal value of the fitness (or one that would correspond to the assumed theoretical maximal value). For the 4×4 size both our technique and related work can easily reach optimal values. However, for sizes 5×5 till 7×7 our approach yielded significantly improved results. On the other hand, for the 8×8 size, related work managed to obtain better nonlinearity and differential uniformity. Still, we emphasize that even those results are far from the best one where nonlinearity equals 112 and differential uniformity equals 4 (note that it is only assumed but not proven that the maximal nonlinearity equals 112).

4 CONCLUSIONS

Our approach shows great potential and marks the first time that heuristic techniques are able to find optimal S-boxes for sizes larger than 4×4. Moreover, our approach transforms a problem that has been up to now of extreme difficulty into a simpler problem for certain S-box sizes. Naturally, since not all S-boxes can be represented as a CA, our technique cannot be used to design all optimal S-boxes of the corresponding size. However, we believe that the corpus of obtainable functions is still large enough to give a sufficient diversity for future block cipher designs.

ACKNOWLEDGMENTS

This work has been supported in part by Croatian Science Foundation under the project IP-2014-09-4882.

REFERENCES

[1] Guido Bertoni, Joan Daemen, Michäel Peeters, and Gilles Van Assche. 2011. The Keccak reference. (January 2011). http://keccak.noekeon.org/.
[2] Claude Carlet. 2010. Vectorial Boolean Functions for Cryptography. In *Boolean Models and Methods in Mathematics, Computer Science, and Engineering* (1st ed.), Yves Crama and Peter L. Hammer (Eds.). Cambridge University Press, New York, USA, 398–469.
[3] Lars R. Knudsen and Matthew Robshaw. 2011. *The Block Cipher Companion.* Springer.
[4] Kaisa Nyberg. 1991. Perfect Nonlinear S-Boxes. In *Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings (Lecture Notes in Computer Science)*, Vol. 547. Springer, 378–386.
[5] Stjepan Picek, Marko Cupic, and Leon Rotim. 2016. A New Cost Function for Evolution of S-boxes. *Evolutionary Computation* (2016).
[6] Riccardo Poli, William B. Langdon, and Nicholas Freitag McPhee. 2008. *A field guide to genetic programming.* Published via `http://lulu.com` and freely available at `http://www.gp-field-guide.org.uk`. (With contributions by J. R. Koza).