On the Design of S-box Constructions with Genetic Programming

Stjepan Picek
Delft University of Technology
The Netherlands

Domagoj Jakobovic
University of Zagreb,
Faculty of electrical engineering and computing
Croatia

ABSTRACT

In this paper, we try to combine the best from the world of heuristics and algebraic constructions for the design of S-boxes: we evolve algebraic constructions that produce S-boxes with as low as possible differential uniformity. Our approach is novel yet very simple and is allowing us to obtain constructions valid for any S-box size of practical interest.

CCS CONCEPTS

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

KEYWORDS

Cryptography, S-boxes, Genetic Programming

ACM Reference Format:
Stjepan Picek and Domagoj Jakobovic. 2019. On the Design of S-box Constructions with Genetic Programming. In *Genetic and Evolutionary Computation Conference Companion (GECCO '19 Companion), July 13–17, 2019, Prague, Czech Republic.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3319619.3322040

1 INTRODUCTION

In modern cryptography, a common type of cryptographic algorithms (commonly known as ciphers) are block ciphers [5]. To build the nonlinear part in block ciphers, a common option in modern designs is to use one or more Substitution Boxes (S-boxes, also known as vectorial Boolean functions, (n, m)-functions). Not all S-boxes provide the same resilience against cryptanalysis and today, there are numerous properties describing the resilience of S-boxes against certain types of attacks like the differential cryptanalysis [1]. A key property connected with the resilience against differential cryptanalysis is called differential uniformity. With differential uniformity, the lower the value, the better the resilience. S-boxes that have minimal possible differential uniformity (and thus have the best possible resilience against differential cryptanalysis) are called Almost Perfect Nonlinear (APN) functions. To construct such S-boxes, we know only several algebraic constructions. Such algebraic constructions are deterministic (always giving the same result) and producing a class (or infinite class) of constructions, which means we can use the same construction to obtain solutions for different S-box sizes. Besides algebraic constructions, we could also use heuristics to obtain APN functions. To that end, we propose to use genetic programming (GP) to evolve algebraic constructions resulting in S-boxes with good differential uniformity. To the best of our knowledge, this is the first work considering something like that. The only (similar) approach is given by Picek and Jakobovic where they used GP to evolve secondary constructions resulting in bent Boolean functions [7].

GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6748-6/19/07…$15.00
https://doi.org/10.1145/3319619.3322040

2 S-BOXES AND THEIR PROPERTIES

Let n be a positive integer, i.e., $n \in \mathbb{N}^+$. We denote by $\mathbb{F}_2^n$ the n-dimensional vector space over $\mathbb{F}_2$ and by $\mathbb{F}_{2^n}$ the finite field with 2^n elements, where $\mathbb{F}_2$ is the Galois field (GF) with two elements. The addition of elements of the finite field $\mathbb{F}_{2^n}$ is denoted with "+", as usual in mathematics. Let F be a function from $\mathbb{F}_2^n$ into $\mathbb{F}_2^n$ with $a \in \mathbb{F}_2^n$ and $b \in \mathbb{F}_2^n$. We denote:

$$D_F(a, b) = \left\{ x \in \mathbb{F}_2^n : F(x) + F(x + a) = b \right\}. \quad (1)$$

The entry at the position (a, b) corresponds to the cardinality of the delta difference table $D_F(a, b)$ and is denoted as $\delta(a, b)$. The differential uniformity δ_F is then defined as [6]:

$$\delta_F = \max_{a \neq 0, b} \delta(a, b). \quad (2)$$

Functions that have differential uniformity equal to 2 are called the Almost Perfect Nonlinear (APN) functions. When discussing the differential uniformity property for permutations (i.e., where the input and output dimension of an S-box are the same), the best possible value is 2 for any odd n and also for $n = 6$. For n even and larger than 6, this is an open question. For a long time, the only examples of constructions of APN functions were the power functions. Today, besides power functions, we know of only a few more constructions resulting in APN functions that are not equivalent to power functions, see e.g., [2, 4]. For further information about S-boxes and their properties, we refer interested readers to [3].

3 GP APPROACH AND RESULTS

The function set consists of several functions one can find in algebraic constructions: 1) trace function ($Tr(x) = x + x^2 + \ldots + x^{2^{n-1}}$), 2) addition in the field, 3) multiplication in the field, and 4) exponentiation in the field. GP uses a 3-tournament selection and the mutation operator is executed once on a given individual with a probability 0.3. The variation operators are simple tree crossover, uniform crossover, size fair, one-point, and context preserving

Table 1: Statistical results for S-box constructions

Size	Min	Max	Average	Std Dev
Multiple dimensions				
N/A	0	99.73	0.38	1.37
Single dimension				
3×3	0	6	0.36	1.31
4×4	0	14	0.75	2.92
5×5	0	30	1.71	6.46
6×6	0	62	3.43	13.4
7×7	0	126	4.47	20.98
8×8	0	254	12.2	51.29
9×9	0	510	18.73	89.02

crossover (selected at random), and subtree mutation [8]. The initial population is created at random, population size equals 100, and every experiment is repeated 30 times. The stopping condition is the number of generations without improvement, which we set to 100. We minimize the following fitness function:

$$fitness = |\delta_F - 2|. \quad (3)$$

There are two cases we investigate:

(1) **Algebraic constructions valid for multiple S-box sizes**. Genetic Programming evolves constructions for size n and for any power exponent d. This means, for each solution (construction), we iterate over all d values in order to evaluate the results. For each construction where the fitness value equals 0, we also test that construction for $n^{'}$, where $n^{'} > n$ and the final fitness value is the fitness obtained for the construction for every tested dimension. Once the testing for larger dimensions starts, we only use the value d that resulted in correct function in the original dimension n.

(2) **Algebraic constructions valid for a single S-box size**. We look for algebraic constructions valid only for size n. There, we give additional constraint that the evolved function cannot use any known APN exponents. These experiments are done for sizes in the range $[3, 9]$.

All the results are given in Table 1. The first scenario we consider is evolving constructions that produce S-boxes with required differential uniformity in multiple sizes. First, we see that the minimal value show GP is able to find constructions working in multiple dimensions. When looking at the average value, we can furthermore observe that the evolution process was very successful in most of the runs. Finally, we explain the maximal value behavior. Very large values indicate that on some occasions the constructions found for size 3 do not generalize to one or more larger sizes. In such instances, the fitness can grow very fast since the differential uniformity values are in the range $[2, 2^n]$, which means that only a single dimension where the construction does not work can degrade the final fitness value significantly.

Next, we investigate how difficult is it to evolve a construction that works in only a single dimension. Naturally, already from the previous results one can deduce that this should not be difficult. Indeed, if we are able to find constructions working for multiple dimensions, then we should be also able to find constructions working in a single dimension. Still, there are two additional considerations. First, we do not allow the use of known APN exponents. Second, we evolve constructions for each dimension. Before, it was enough to work with the smallest size (3×3) and find a construction that generalizes but now we directly work in search spaces that are much larger. We see that we are able to find APN construction for each dimension since all minimum values are 0, which is the global optimum. The maximal values are of the form $2^n - 2$, which means they represent linear functions that have the worst possible differential uniformity (2^n). Both average and standard deviations increase with the increase in the S-box size, which indicate that (as one would intuitively assume) that the difficulty of the problem increases with the size. It is interesting to notice the quick degrade in the average value for the 7×7 case (that still seems to be easy) and 8×8 case that seems to significantly increase in difficulty.

Our results show that GP is able to find constructions regardless of whether we are working on multiple S-box sizes or only a single size. Finally, we report extremely good results where we are able to reach the goal in every experiment. When considering APN functions working in a single dimension, we observe interesting results. For instance, for the 3×3 size, we found the solution:

$multiply(add(add\_one(X), trace(X)), exp(add\_one(X), d))$

with $d = 2$. This construction is producing APN function that is bijective. What is more, exponent value 2 is not possible exponent value for APN power functions. This means that the function we constructed is not equivalent to APN power function, which makes it a very interesting result.

4 CONCLUSIONS AND FUTURE WORK

In this paper, we investigate whether it is possible to use GP to evolve algebraic constructions for S-boxes. All our results show that GP is consistently able to find such solutions. Still, many equivalent solutions appear or are already known constructions. Despite that, our experiments revealed several interesting examples that warrant to be further studied. In future work, we plan to investigate what happens if we try to find constructions working in multiple odd or even dimensions only.

REFERENCES

[1] Eli Biham and Adi Shamir. 1991. Differential Cryptanalysis of DES-like Cryptosystems. In *Proceedings of the 10th Annual International Cryptology Conference on Advances in Cryptology (CRYPTO '90)*. Springer-Verlag, London, UK, UK, 2–21.

[2] Lilya Budaghyan, Claude Carlet, and Gregor Leander. 2009. Constructing new APN functions from known ones. *Finite Fields and Their Applications* 15, 2 (2009), 150 – 159. https://doi.org/10.1016/j.ffa.2008.10.001

[3] 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, NY, USA, 398–469.

[4] Y. Edel, G. Kyureghyan, and A. Pott. 2006. A new APN function which is not equivalent to a power mapping. *IEEE Transactions on Information Theory* 52, 2 (Feb 2006), 744–747. https://doi.org/10.1109/TIT.2005.862128

[5] Lars R. Knudsen and Matthew Robshaw. 2011. *The Block Cipher Companion*. Springer. I–XIV, 1–267 pages.

[6] 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.

[7] Stjepan Picek and Domagoj Jakobovic. 2016. Evolving Algebraic Constructions for Designing Bent Boolean Functions. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016 (GECCO '16)*. ACM, New York, NY, USA, 781–788. https://doi.org/10.1145/2908812.2908915

[8] 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).