Bad and Good Ways of Post-Processing Biased Random Numbers

Markus Dichtl

Siemens AG

Corporate Technology

Overview

This talk comes in two parts:

- A bad way
- Good ways

Why Post-Processing?

Observation: All physical random numbers seem to deviate from the statistical ideal.

Post-processing is used to remove or reduce these deviations from the ideal.

The Most Frequent Statistical Problem

Bias: A deviation of the probability of 1-bits from the ideal value ½.

For statistically independent bits with probability p of 1-bits:

$$\text{Bias} \qquad \varepsilon = p - 1/2$$

The Bad Scheme

In their FSE 2005 paper, “Unbiased Random Sequences from Quasigroup String Transformations”, Markovski, Gligoroski, and Kocarev suggested this scheme for TRNG post-processing.

What is a Quasigroup? (I)

A **quasigroup** is a set Q with a mapping $*$ $Q \times Q \rightarrow Q$ such that all equations of the form

$a * x = b$ and $y * a = b$

are uniquely solvable for x and y for all a and b

What is a Quasigroup? (II)

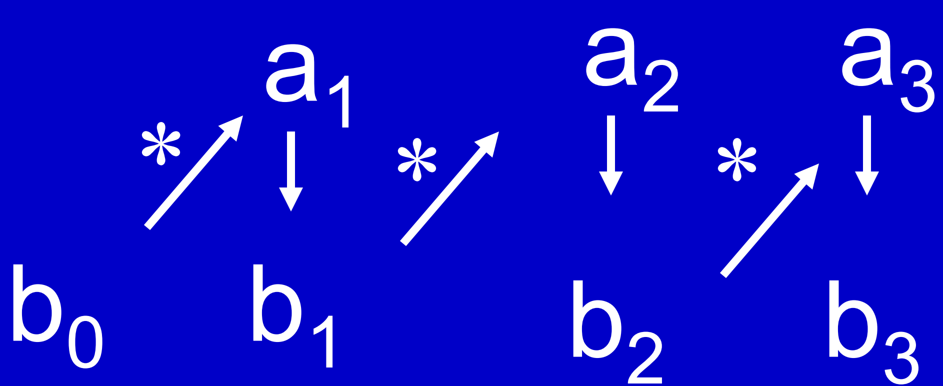
A function is a quasigroup iff its function table is a latin square.

*	0	1	2	3
0	2	1	0	3
1	3	0	1	2
2	1	2	3	0
3	0	3	2	1

The e-Transformation

The e-transformation maps a string $a_1a_2\ldots a_n$ and a „leader“ b_0 ($b_o * b_o \neq b_o$) to the string $b_1b_2\ldots b_n$ by

$$b_i = b_{i-1} * a_i \text{ for } i = 1, \ldots, n$$



The E-Algorithm

E-algorithm : k-fold application of the e-transformation (fixed leader and quasigroup)

According to the recommendations of the original paper for highly biased input, we choose k=128 for a quasigroup of order 4.

The Good News about the Bad Scheme

As the quasigroup mapping is bijective, it can do no harm.

The entropy of the output is just the entropy of the input.

The HB TRNG

The authors of the quasigroup post-processing paper claim that it is suitable for highly biased input like

99.9 % 0-bits

0.1 % 1-bits (bias -0.499)

We call this generator HB (for High Bias)

Attack

We attack HB post-processed with the E-Algorithm based on a quasigroup of order 4 and k=128.

As almost all inputs bits are 0, we guess them to be 0 and determine the output by applying the E-Algorithm.

The probability to guess two bits correctly is 0.998001

If we guess wrongly, we use the inverse E-Algorithm to determine the correct input for continuing the attack.

Attack with Quasigroup Unknown

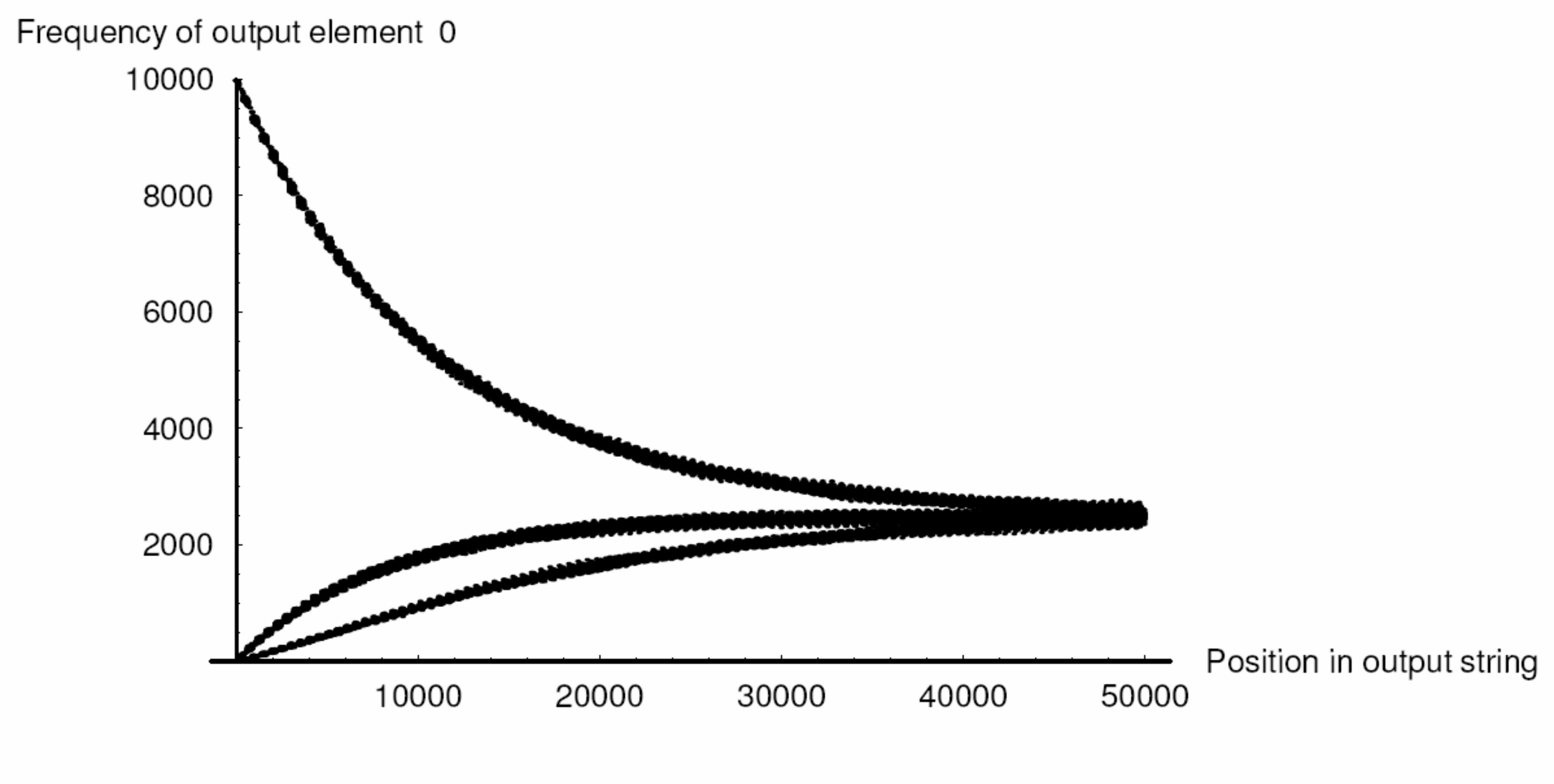
It does not help too much to keep quasigroup and leader secret, as there are only 1728 choices of quasigroups of order 4 and leader.

Simplified attack suggested by an anonymous reviewer:

Apply the inverse E-algorithms for the 1728 choices, the correct one is identified by many 0-bits in the output.

What is Going on in the E-Algorithm?

Bias is replaced with dependency, and this is achieved very slowly



And now for something quite different

One anonymous FSE 2007 reviewer:

The paper needs to be much more up-front about the fact that you are demolishing apples while promoting the virtues of oranges.

We have to give up the idea of bijective post-processing (apples) of random numbers and look at compressing functions instead (oranges).

Von Neumann Post-Processing

John von Neumann (1951)

~~00~~

01 $\longrightarrow$ 0

10 $\longrightarrow$ 1

~~11~~

For statistically independent but biased input: perfect balanced and independent output

Problem: Unbounded latency

A Dilemma

Perfect output statistics

and

bounded latency

exclude each other.

Popular Examples for Bounded Latency Algorithms

XOR

Feeding the RNG-bits into a LFSR, reading output from the LFSR at a lower rate

Algorithms for Fixed Input/Output Rate

No perfect solution!

We consider the input/output rate 2.

For single bits: XOR is optimal!

Bias after XOR: $2\varepsilon^2$

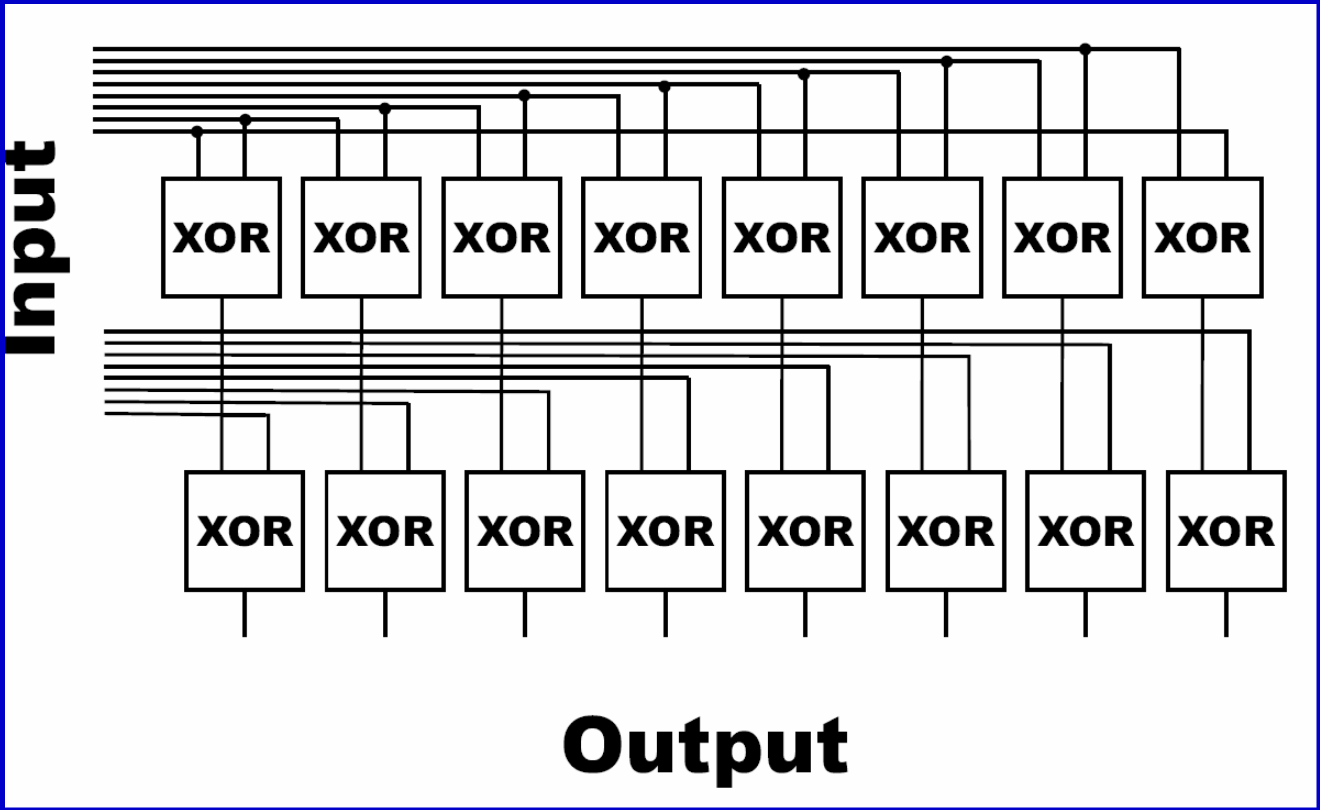
What we are Looking for

Input: 16 bits

Output: 8 bits

Input is assumed to be statistically independent, but biased. We cannot assume to know the numerical value of the bias ε.

The Function H



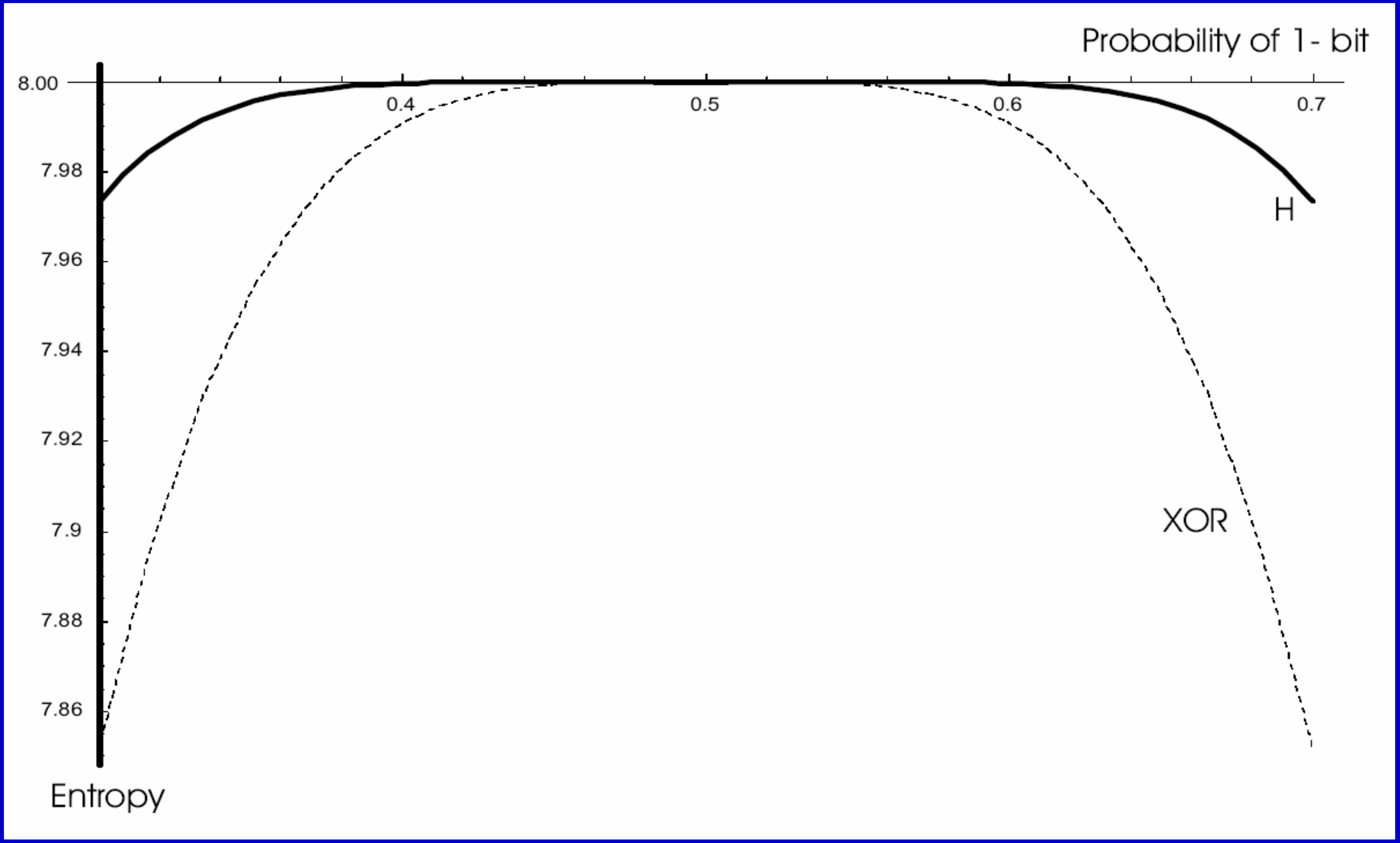
2 Bytes are mapped to 1.

The Function H in C

```
unsigned char H (unsigned char a, unsigned char b)
{
return ( a^rotateleft(a,1)^b);        /* ^ is XOR in C*/
}
```

Entropy Comparison: H and XOR

2 bytes are mapped to 1 byte.



What about Low Biases?

Probability of 1-bit: 0.51 (Bias 0.01)

Entropy of one output byte with XOR:

7.99999990766751

Entropy of one output byte with H:

7.99999999996305

which is 2499 times closer to 8.

Probabilities of Raw Bytes

w	byte probability for a raw data byte
0	$\frac{1}{256} - \frac{\epsilon}{16} + \frac{7\,\epsilon^2}{16} - \frac{7\,\epsilon^3}{4} + \frac{35\,\epsilon^4}{8} - 7\,\epsilon^5 + 7\,\epsilon^6 - 4\,\epsilon^7 + \epsilon^8$
1	$\frac{1}{256} - \frac{3\,\epsilon}{64} + \frac{7\,\epsilon^2}{32} - \frac{7\,\epsilon^3}{16} + \frac{7\,\epsilon^5}{4} - \frac{7\,\epsilon^6}{2} + 3\,\epsilon^7 - \epsilon^8$
2	$\frac{1}{256} - \frac{\epsilon}{32} + \frac{\epsilon^2}{16} + \frac{\epsilon^3}{8} - \frac{5\,\epsilon^4}{8} + \frac{\epsilon^5}{2} + \epsilon^6 - 2\,\epsilon^7 + \epsilon^8$
3	$\frac{1}{256} - \frac{\epsilon}{64} - \frac{\epsilon^2}{32} + \frac{3\,\epsilon^3}{16} - \frac{3\,\epsilon^5}{4} + \frac{\epsilon^6}{2} + \epsilon^7 - \epsilon^8$
4	$\frac{1}{256} - \frac{\epsilon^2}{16} + \frac{3\,\epsilon^4}{8} - \epsilon^6 + \epsilon^8$
5	$\frac{1}{256} + \frac{\epsilon}{64} - \frac{\epsilon^2}{32} - \frac{3\,\epsilon^3}{16} + \frac{3\,\epsilon^5}{4} + \frac{\epsilon^6}{2} - \epsilon^7 - \epsilon^8$
6	$\frac{1}{256} + \frac{\epsilon}{32} + \frac{\epsilon^2}{16} - \frac{\epsilon^3}{8} - \frac{5\,\epsilon^4}{8} - \frac{\epsilon^5}{2} + \epsilon^6 + 2\,\epsilon^7 + \epsilon^8$
7	$\frac{1}{256} + \frac{3\,\epsilon}{64} + \frac{7\,\epsilon^2}{32} + \frac{7\,\epsilon^3}{16} - \frac{7\,\epsilon^5}{4} - \frac{7\,\epsilon^6}{2} - 3\,\epsilon^7 - \epsilon^8$
8	$\frac{1}{256} + \frac{\epsilon}{16} + \frac{7\,\epsilon^2}{16} + \frac{7\,\epsilon^3}{4} + \frac{35\,\epsilon^4}{8} + 7\,\epsilon^5 + 7\,\epsilon^6 + 4\,\epsilon^7 + \epsilon^8$

Byte Probabilites for XOR

w	byte probability for the XOR of two raw data bytes
0	$\frac{1}{256} - \frac{\epsilon^2}{8} + \frac{7\,\epsilon^4}{4} - 14\,\epsilon^6 + 70\,\epsilon^8 - 224\,\epsilon^{10} + 448\,\epsilon^{12} - 512\,\epsilon^{14} + 256\,\epsilon^{16}$
1	$\frac{1}{256} - \frac{3\,\epsilon^2}{32} + \frac{7\,\epsilon^4}{8} - \frac{7\,\epsilon^6}{2} + 56\,\epsilon^{10} - 224\,\epsilon^{12} + 384\,\epsilon^{14} - 256\,\epsilon^{16}$
2	$\frac{1}{256} - \frac{\epsilon^2}{16} + \frac{\epsilon^4}{4} + \epsilon^6 - 10\,\epsilon^8 + 16\,\epsilon^{10} + 64\,\epsilon^{12} - 256\,\epsilon^{14} + 256\,\epsilon^{16}$
3	$\frac{1}{256} - \frac{\epsilon^2}{32} - \frac{\epsilon^4}{8} + \frac{3\,\epsilon^6}{2} - 24\,\epsilon^{10} + 32\,\epsilon^{12} + 128\,\epsilon^{14} - 256\,\epsilon^{16}$
4	$\frac{1}{256} - \frac{\epsilon^4}{4} + 6\,\epsilon^8 - 64\,\epsilon^{12} + 256\,\epsilon^{16}$
5	$\frac{1}{256} + \frac{\epsilon^2}{32} - \frac{\epsilon^4}{8} - \frac{3\,\epsilon^6}{2} + 24\,\epsilon^{10} + 32\,\epsilon^{12} - 128\,\epsilon^{14} - 256\,\epsilon^{16}$
6	$\frac{1}{256} + \frac{\epsilon^2}{16} + \frac{\epsilon^4}{4} - \epsilon^6 - 10\,\epsilon^8 - 16\,\epsilon^{10} + 64\,\epsilon^{12} + 256\,\epsilon^{14} + 256\,\epsilon^{16}$
7	$\frac{1}{256} + \frac{3\,\epsilon^2}{32} + \frac{7\,\epsilon^4}{8} + \frac{7\,\epsilon^6}{2} - 56\,\epsilon^{10} - 224\,\epsilon^{12} - 384\,\epsilon^{14} - 256\,\epsilon^{16}$
8	$\frac{1}{256} + \frac{\epsilon^2}{8} + \frac{7\,\epsilon^4}{4} + 14\,\epsilon^6 + 70\,\epsilon^8 + 224\,\epsilon^{10} + 448\,\epsilon^{12} + 512\,\epsilon^{14} + 256\,\epsilon^{16}$

Byte Probabilities for H (Part)

output byte probability for H
$\frac{1}{256} + \frac{\epsilon^3}{16} - \frac{\epsilon^4}{4} - \frac{3\,\epsilon^5}{4} + \frac{\epsilon^6}{2} + 3\,\epsilon^7 + 3\,\epsilon^8 - 4\,\epsilon^9 - 8\,\epsilon^{10}$
$\frac{1}{256} - \frac{\epsilon^3}{16} - \frac{\epsilon^4}{4} + \frac{3\,\epsilon^5}{4} + \frac{\epsilon^6}{2} - 3\,\epsilon^7 + 3\,\epsilon^8 + 4\,\epsilon^9 - 8\,\epsilon^{10}$
$\frac{1}{256} - \frac{\epsilon^3}{16} - \frac{\epsilon^5}{4} - \frac{\epsilon^6}{2} + 5\,\epsilon^7 - \epsilon^8 - 12\,\epsilon^9 + 8\,\epsilon^{10}$
$\frac{1}{256} + \frac{\epsilon^3}{16} + \frac{\epsilon^5}{4} - \frac{\epsilon^6}{2} - 5\,\epsilon^7 - \epsilon^8 + 12\,\epsilon^9 + 8\,\epsilon^{10}$
$\frac{1}{256} - \frac{\epsilon^3}{16} + \frac{\epsilon^4}{4} - \frac{\epsilon^5}{4} - \frac{3\,\epsilon^6}{2} + \epsilon^7 - 5\,\epsilon^8 + 20\,\epsilon^9 + 24\,\epsilon^{10} - 64\,\epsilon^{11}$
$\frac{1}{256} + \frac{3\,\epsilon^3}{16} + \frac{\epsilon^4}{4} + \frac{\epsilon^5}{4} + \frac{5\,\epsilon^6}{2} + 3\,\epsilon^7 - 5\,\epsilon^8 - 20\,\epsilon^9 - 40\,\epsilon^{10} - 32\,\epsilon^{11}$
$\frac{1}{256} + \frac{\epsilon^3}{16} - \frac{\epsilon^4}{4} - \frac{\epsilon^5}{4} + \frac{\epsilon^6}{2} - 3\,\epsilon^7 + 3\,\epsilon^8 + 20\,\epsilon^9 - 8\,\epsilon^{10} - 32\,\epsilon^{11}$
[illegible]

Why H is so Good and a New Challenge

That the lowest power of ε in the probabilties of H is ε^3 explains why H is better than XOR, which has ε^2 terms.

Challenge:

To make disappear further powers of ε!

The Functions H2 and H3 in C

```
unsigned char H2(unsigned char a, unsigned char b)
{
return ( a^rotateleft(a,1)^rotateleft(a,2)^b);
}

unsigned char H3(unsigned char a, unsigned char b)
{
return ( a^rotateleft(a,1)^rotateleft(a,2)^ rotateleft(a,4)^ b);
}
```

Properties of H2 and H3

Lowest ε-power in the byte probabilities:

H2: ε^4

H3: ε^5

Going Further

Of course, we also want to get rid of ε^5 !

It seems that linear methods cannot achieve this.

What must be done?

We must partition 2^{16} 16-bit-values into 256 sets of 256 elements each in such a way that in the sums of the probabilties of each set the powers ε^1 through ε^5 cancel out.

The probabilities of the 16-bit-values depend only on the Hamming weight w. Hence, there are 17 possibilities. The different Hamming weights occur with different frequencies.

Occurrences and Probabilities for 16-bit-values

w	Occurrences	Probability of 16 bit input with Hamming weight w
0	1	$\frac{1}{65536} - \frac{\epsilon}{2048} + \frac{15\,\epsilon^2}{2048} - \frac{35\,\epsilon^3}{512} + \frac{455\,\epsilon^4}{1024} - \frac{273\,\epsilon^5}{128} + \frac{1001\,\epsilon^6}{128} - \frac{715\,\epsilon^7}{32} + \frac{6435\,\epsilon^8}{128} - \frac{715\,\epsilon^9}{8} + \frac{1001\,\epsilon^{10}}{8} - \frac{273\,\epsilon^{11}}{2} + \frac{455\,\epsilon^{12}}{4} - 70\,\epsilon^{13} + 30\,\epsilon^{14} - 8\,\epsilon^{15} + \epsilon^{16}$
1	16	$\frac{1}{65536} - \frac{7\,\epsilon}{16384} + \frac{45\,\epsilon^2}{8192} - \frac{175\,\epsilon^3}{4096} + \frac{455\,\epsilon^4}{2048} - \frac{819\,\epsilon^5}{1024} + \frac{1001\,\epsilon^6}{512} - \frac{715\,\epsilon^7}{256} + \frac{715\,\epsilon^9}{64} - \frac{1001\,\epsilon^{10}}{32} + \frac{819\,\epsilon^{11}}{16} - \frac{455\,\epsilon^{12}}{8} + \frac{175\,\epsilon^{13}}{4} - \frac{45\,\epsilon^{14}}{2} + 7\,\epsilon^{15} - \epsilon^{16}$
2	120	$\frac{1}{65536} - \frac{3\,\epsilon}{8192} + \frac{\epsilon^2}{256} - \frac{49\,\epsilon^3}{2048} + \frac{91\,\epsilon^4}{1024} - \frac{91\,\epsilon^5}{512} + \frac{143\,\epsilon^7}{128} - \frac{429\,\epsilon^8}{128} + \frac{143\,\epsilon^9}{32} - \frac{91\,\epsilon^{11}}{8} + \frac{91\,\epsilon^{12}}{4} - \frac{49\,\epsilon^{13}}{2} + 16\,\epsilon^{14} - 6\,\epsilon^{15} + \epsilon^{16}$
3	560	$\frac{1}{65536} - \frac{5\,\epsilon}{16384} + \frac{21\,\epsilon^2}{8192} - \frac{45\,\epsilon^3}{4096} + \frac{39\,\epsilon^4}{2048} + \frac{39\,\epsilon^5}{1024} - \frac{143\,\epsilon^6}{512} + \frac{143\,\epsilon^7}{256} - \frac{143\,\epsilon^9}{64} + \frac{143\,\epsilon^{10}}{32} - \frac{39\,\epsilon^{11}}{16} - \frac{39\,\epsilon^{12}}{8} + \frac{45\,\epsilon^{13}}{4} - \frac{21\,\epsilon^{14}}{2} + 5\,\epsilon^{15} - \epsilon^{16}$
4	1820	$\frac{1}{65536} - \frac{\epsilon}{4096} + \frac{3\,\epsilon^2}{2048} - \frac{3\,\epsilon^3}{1024} - \frac{9\,\epsilon^4}{1024} + \frac{15\,\epsilon^5}{256} - \frac{11\,\epsilon^6}{128} - \frac{11\,\epsilon^7}{64} + \frac{99\,\epsilon^8}{128} - \frac{11\,\epsilon^9}{16} - \frac{11\,\epsilon^{10}}{8} + \frac{15\,\epsilon^{11}}{4} - \frac{9\,\epsilon^{12}}{4} - 3\,\epsilon^{13} + 6\,\epsilon^{14} - 4\,\epsilon^{15} + \epsilon^{16}$
5	4368	$\frac{1}{65536} - \frac{3\,\epsilon}{16384} + \frac{5\,\epsilon^2}{8192} + \frac{5\,\epsilon^3}{4096} - \frac{25\,\epsilon^4}{2048} + \frac{17\,\epsilon^5}{1024} + \frac{33\,\epsilon^6}{512} - \frac{55\,\epsilon^7}{256} + \frac{55\,\epsilon^9}{64} - \frac{33\,\epsilon^{10}}{32} - \frac{17\,\epsilon^{11}}{16} + \frac{25\,\epsilon^{12}}{8} - \frac{5\,\epsilon^{13}}{4} - \frac{5\,\epsilon^{14}}{2} + 3\,\epsilon^{15} - \epsilon^{16}$
6	8008	$\frac{1}{65536} - \frac{\epsilon}{8192} + \frac{5\,\epsilon^3}{2048} - \frac{5\,\epsilon^4}{1024} - \frac{9\,\epsilon^5}{512} + \frac{\epsilon^6}{16} + \frac{5\,\epsilon^7}{128} - \frac{45\,\epsilon^8}{128} + \frac{5\,\epsilon^9}{32} + \epsilon^{10} -$

Observation

If we add the probabilty of a 16-bit-tupel and the probability of ist bitwise complement, then all odd ε-powers cancel out. So, we add them to our sets only together.

⇒ Considerable simplification of the problem

The Simplified Problem

w	Occurrences	Probabilty of input + probability of complement
0	1	$\frac{1}{32768} + \frac{15\,\epsilon^2}{1024} + \frac{455\,\epsilon^4}{512} + \frac{1001\,\epsilon^6}{64} + \frac{6435\,\epsilon^8}{64} + \frac{1001\,\epsilon^{10}}{4} + \frac{455\,\epsilon^{12}}{2} + 60\,\epsilon^{14} + 2\,\epsilon^{16}$
1	16	$\frac{1}{32768} + \frac{45\,\epsilon^2}{4096} + \frac{455\,\epsilon^4}{1024} + \frac{1001\,\epsilon^6}{256} - \frac{1001\,\epsilon^{10}}{16} - \frac{455\,\epsilon^{12}}{4} - 45\,\epsilon^{14} - 2\,\epsilon^{16}$
2	120	$\frac{1}{32768} + \frac{\epsilon^2}{128} + \frac{91\,\epsilon^4}{512} - \frac{429\,\epsilon^8}{64} + \frac{91\,\epsilon^{12}}{2} + 32\,\epsilon^{14} + 2\,\epsilon^{16}$
3	560	$\frac{1}{32768} + \frac{21\,\epsilon^2}{4096} + \frac{39\,\epsilon^4}{1024} - \frac{143\,\epsilon^6}{256} + \frac{143\,\epsilon^{10}}{16} - \frac{39\,\epsilon^{12}}{4} - 21\,\epsilon^{14} - 2\,\epsilon^{16}$
4	1820	$\frac{1}{32768} + \frac{3\,\epsilon^2}{1024} - \frac{9\,\epsilon^4}{512} - \frac{11\,\epsilon^6}{64} + \frac{99\,\epsilon^8}{64} - \frac{11\,\epsilon^{10}}{4} - \frac{9\,\epsilon^{12}}{2} + 12\,\epsilon^{14} + 2\,\epsilon^{16}$
5	4368	$\frac{1}{32768} + \frac{5\,\epsilon^2}{4096} - \frac{25\,\epsilon^4}{1024} + \frac{33\,\epsilon^6}{256} - \frac{33\,\epsilon^{10}}{16} + \frac{25\,\epsilon^{12}}{4} - 5\,\epsilon^{14} - 2\,\epsilon^{16}$
6	8008	$\frac{1}{32768} - \frac{5\,\epsilon^4}{512} + \frac{\epsilon^6}{8} - \frac{45\,\epsilon^8}{64} + 2\,\epsilon^{10} - \frac{5\,\epsilon^{12}}{2} + 2\,\epsilon^{16}$
7	11440	$\frac{1}{32768} - \frac{3\,\epsilon^2}{4096} + \frac{7\,\epsilon^4}{1024} - \frac{7\,\epsilon^6}{256} + \frac{7\,\epsilon^{10}}{16} - \frac{7\,\epsilon^{12}}{4} + 3\,\epsilon^{14} - 2\,\epsilon^{16}$
8	6435	$\frac{1}{32768} - \frac{\epsilon^2}{1024} + \frac{7\,\epsilon^4}{512} - \frac{7\,\epsilon^6}{64} + \frac{35\,\epsilon^8}{64} - \frac{7\,\epsilon^{10}}{4} + \frac{7\,\epsilon^{12}}{2} - 4\,\epsilon^{14} + 2\,\epsilon^{16}$

The Solution S

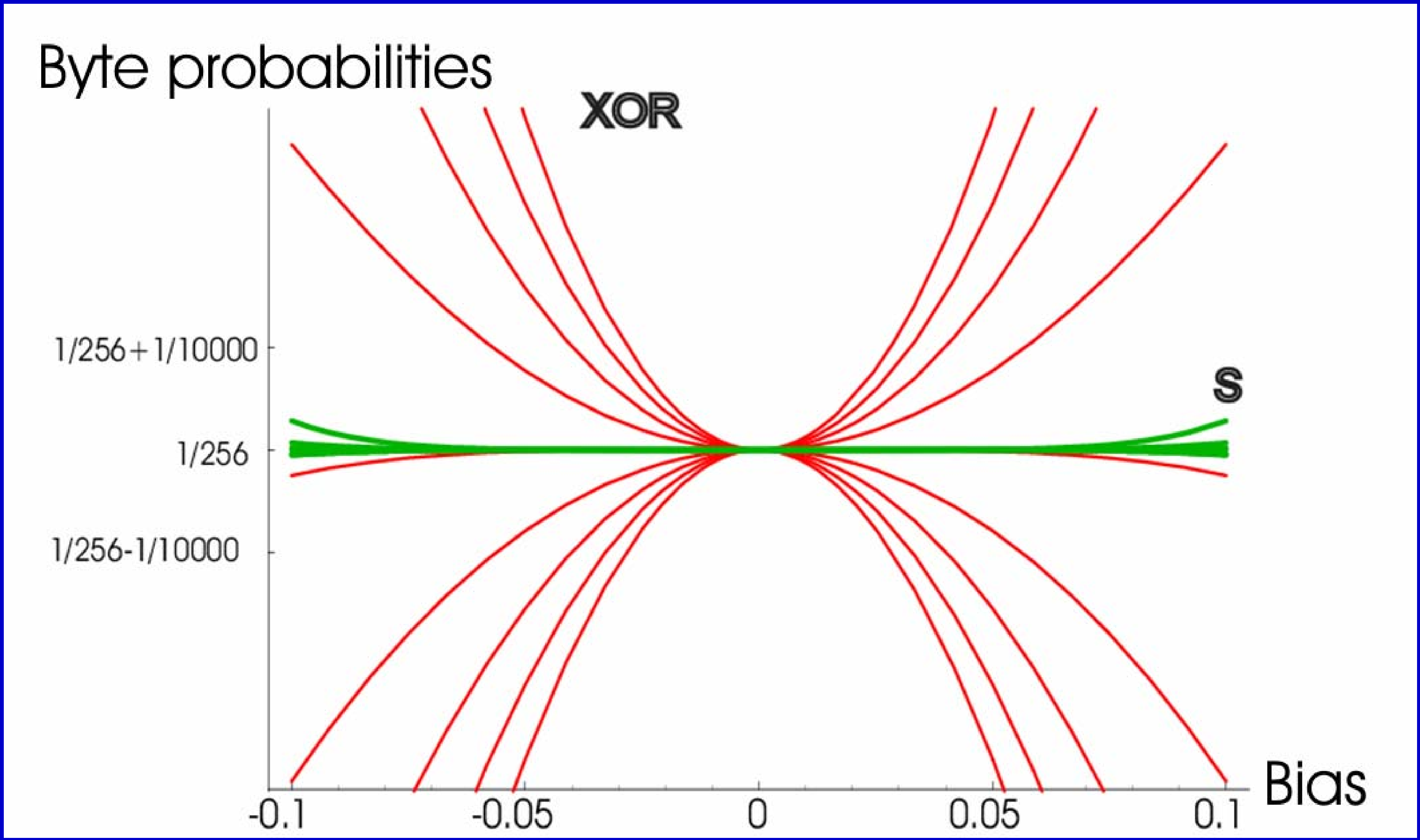
The 256 sets of the solutions S fall into 7 types:

Type	#	w =0	w =1	w =2	w =3	w =4	w =5	w =6	w =7	w =8
A	1	1						112		15
B	16		1				42		85	
C	46					14	28		36	50
D	60			2			37	16	43	30
E	112				5	7		58	43	15
F	4					13	30	8	2	75
G	17					20	4	24	60	20

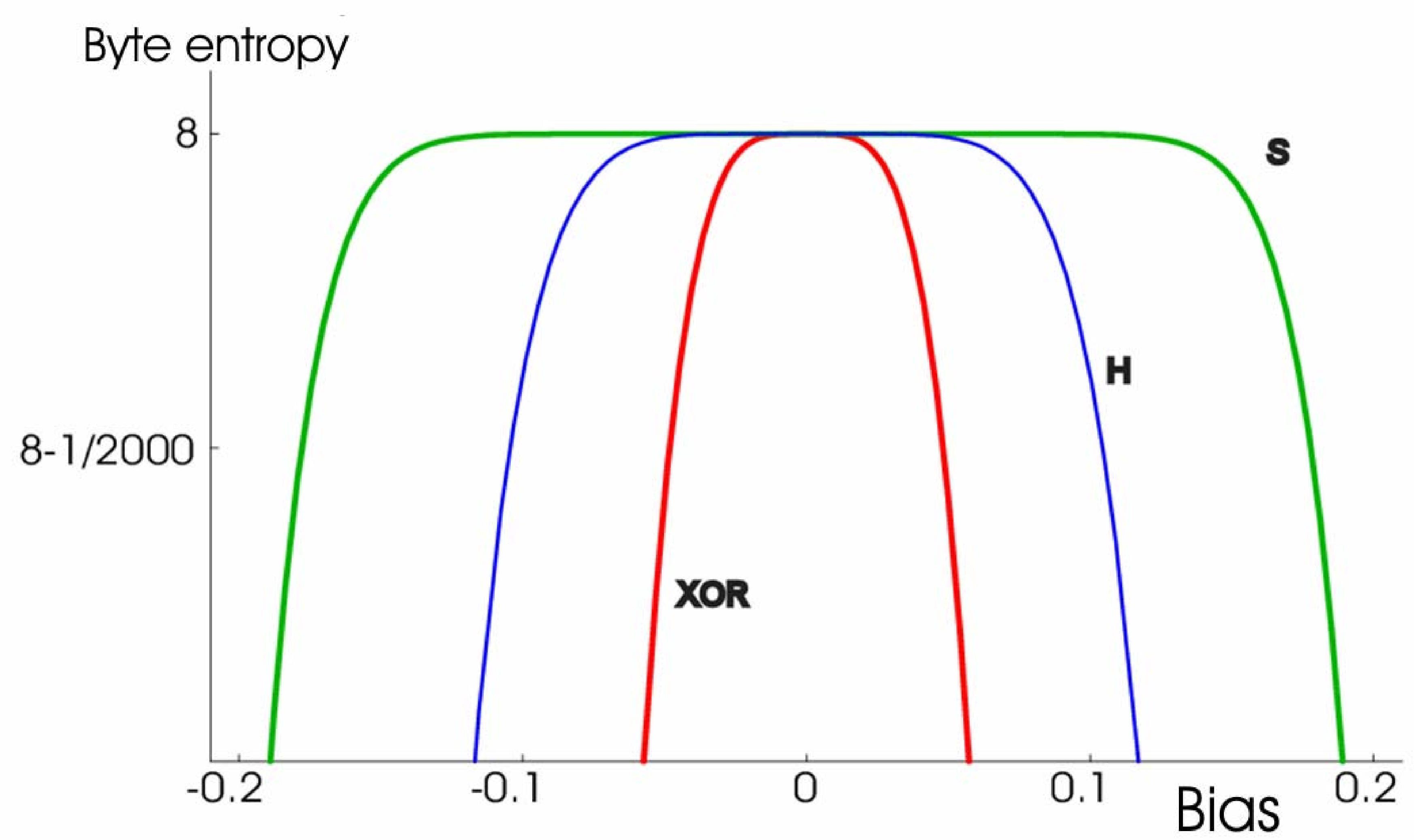
Byte Probabilities of S

Type	Probability of output byte
A	$\frac{1}{256} + 28\,\epsilon^6 + 30\,\epsilon^8 + 448\,\epsilon^{10} + 256\,\epsilon^{16}$
B	$\frac{1}{256} + 7\,\epsilon^6 - 112\,\epsilon^{10} - 256\,\epsilon^{16}$
C	$\frac{1}{256} - \frac{21\,\epsilon^6}{4} + 49\,\epsilon^8 - 168\,\epsilon^{10} + 224\,\epsilon^{12} - 64\,\epsilon^{14}$
D	$\frac{1}{256} + \frac{37\,\epsilon^6}{16} - \frac{33\,\epsilon^8}{4} - 78\,\epsilon^{10} + 312\,\epsilon^{12} - 112\,\epsilon^{14} - 64\,\epsilon^{16}$
E	$\frac{1}{256} + \frac{7\,\epsilon^6}{16} - \frac{87\,\epsilon^8}{4} + 134\,\epsilon^{10} - 248\,\epsilon^{12} + 48\,\epsilon^{14} + 64\,\epsilon^{16}$
F	$\frac{1}{256} - \frac{45\,\epsilon^6}{8} + \frac{111\,\epsilon^8}{2} - 212\,\epsilon^{10} + 368\,\epsilon^{12} - 288\,\epsilon^{14} + 128\,\epsilon^{16}$
G	$\frac{1}{256} - \frac{15\,\epsilon^6}{4} + 25\,\epsilon^8 - 24\,\epsilon^{10} - 160\,\epsilon^{12} + 320\,\epsilon^{14}$

Byte Probabilities of S and XOR



Entropy Comparison of S, H, and XOR



Negative Results

The ε^6-terms cannot be eliminated. (Proved by linear programming techniques.)

When considering mappings from 32 to 16 bits, the probabilities of the output values contain 9-th or lower powers of ε.

Conclusion

The quasigroup TRNG post-processing suggested by Markovski, Gligoroski, and Kocarev does not work. It is based on faulty mathematics.

The fixed input/output rate TRNG post-processing functions suggested in this talk are considerably better than the previously known algorithms. There are open questions concerning the systematic construction of such functions.