Towards cryptographic function distinguishers with evolutionary circuits

Statistical testing of cryptographic function output based on genetic programming

Petr Švenda, Martin Ukrop, Vashek Matyáš
{svenda,xukrop,matyas}@fi.muni.cz

CROCS
Centre for Research on Cryptography and Security

Overview

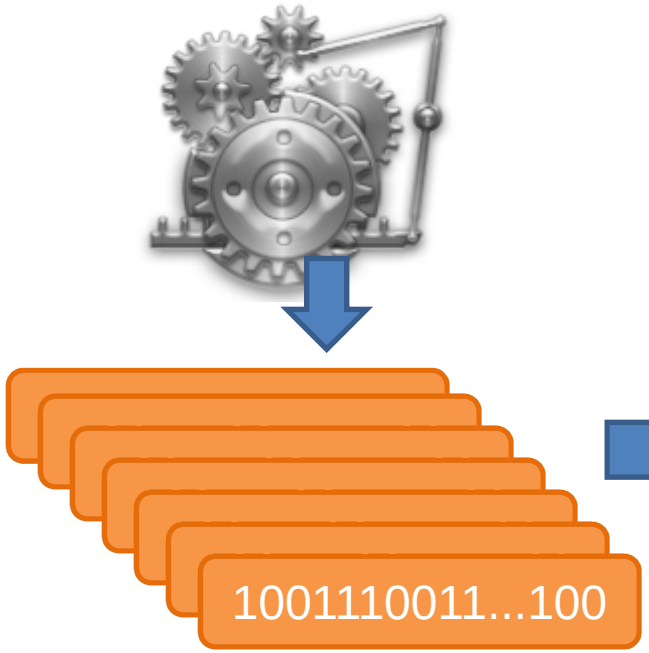
1. Randomness testing with STS NIST & Dieharder
   - Can we beat traditional approach? (Speed, input length.)
2. Random distinguisher based on software circuit
   - Our approach based on genetic programming
3. Results for selected eStream/SHA-3 candidates
   - How good is it?
4. Discussion, interesting observations

Why to test randomness of function output?

1. Building block for pseudorandom generator
2. Common requirement
   - AES, SHA-3 competition, FIPS-140
3. Significant deviances from uniform distribution and unpredictability indicate function defects
   - but no proof in opposite case

- Manual approach: human cryptanalysis
- Automated approach: statistical testing

Workflow with STS NIST/Dieharder



10^5-10^9 B

Tests

Count the 1s
Overlapping permutations
Runs tests
....

“null hypothesis” ⇒ p-values

p-value < α ⇒ fail

Test vectors

Quantum Random Bit Generator service

500x 1011010100...101

ECRYPT

500x 1001110011...100

Algorithm execution

Hypothesis: *If function output is somehow defective, we should be able to distinguish between the data produced by a function and truly random data.*

Results

1/0

1 => QRNG

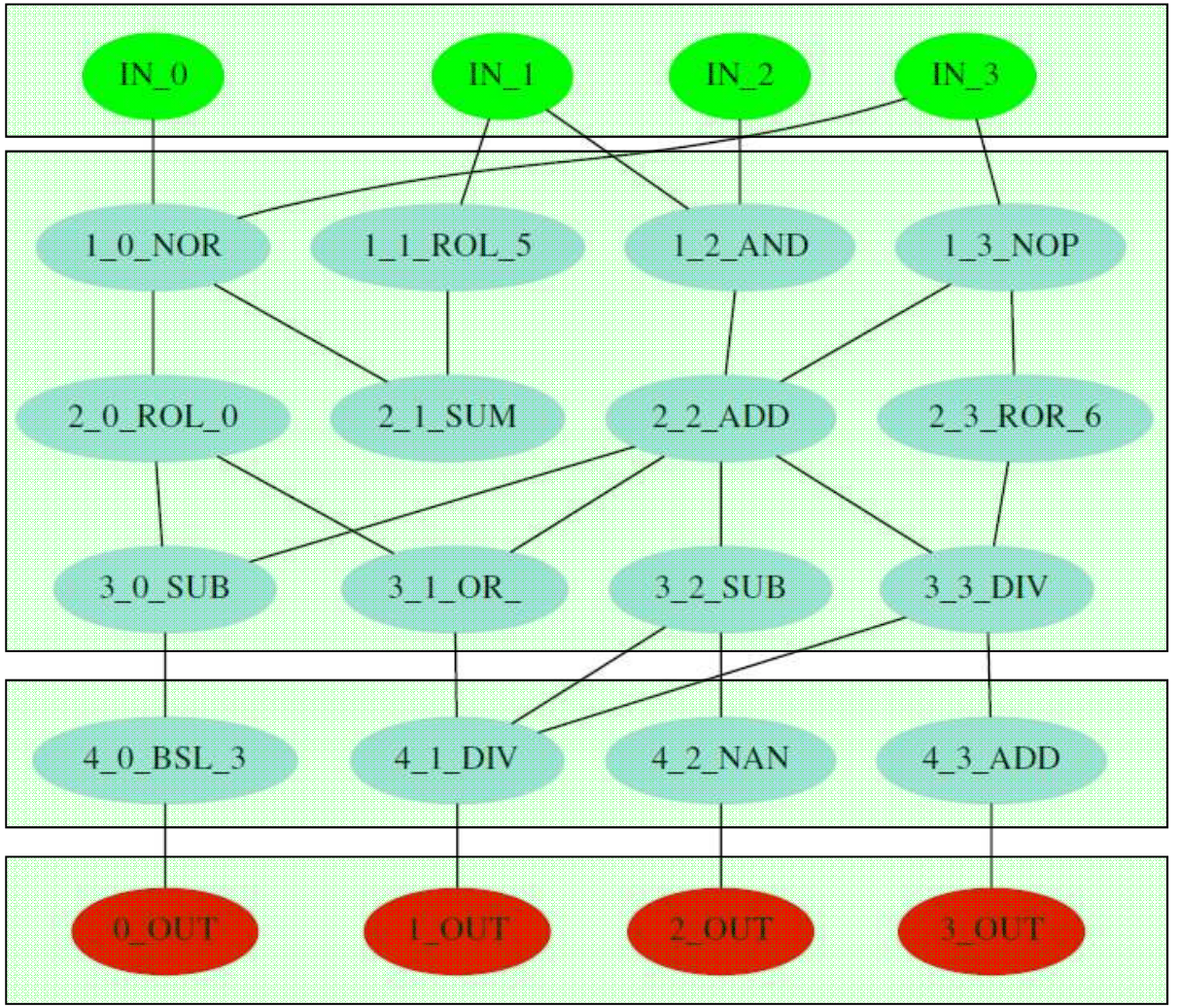
Proposed idea – software circuit

- Design test(s) automatically
  - test is algorithm ⇒ hardware-like circuit (next slide)

- Several issues:
  - Who will define null hypothesis? (random distinguisher)
  - Who will design the circuit? (genetic programming)
  - How to compare quality of candidates? (test vectors)

https://github.com/petrs/EACirc/

Software circuit (EACirc)



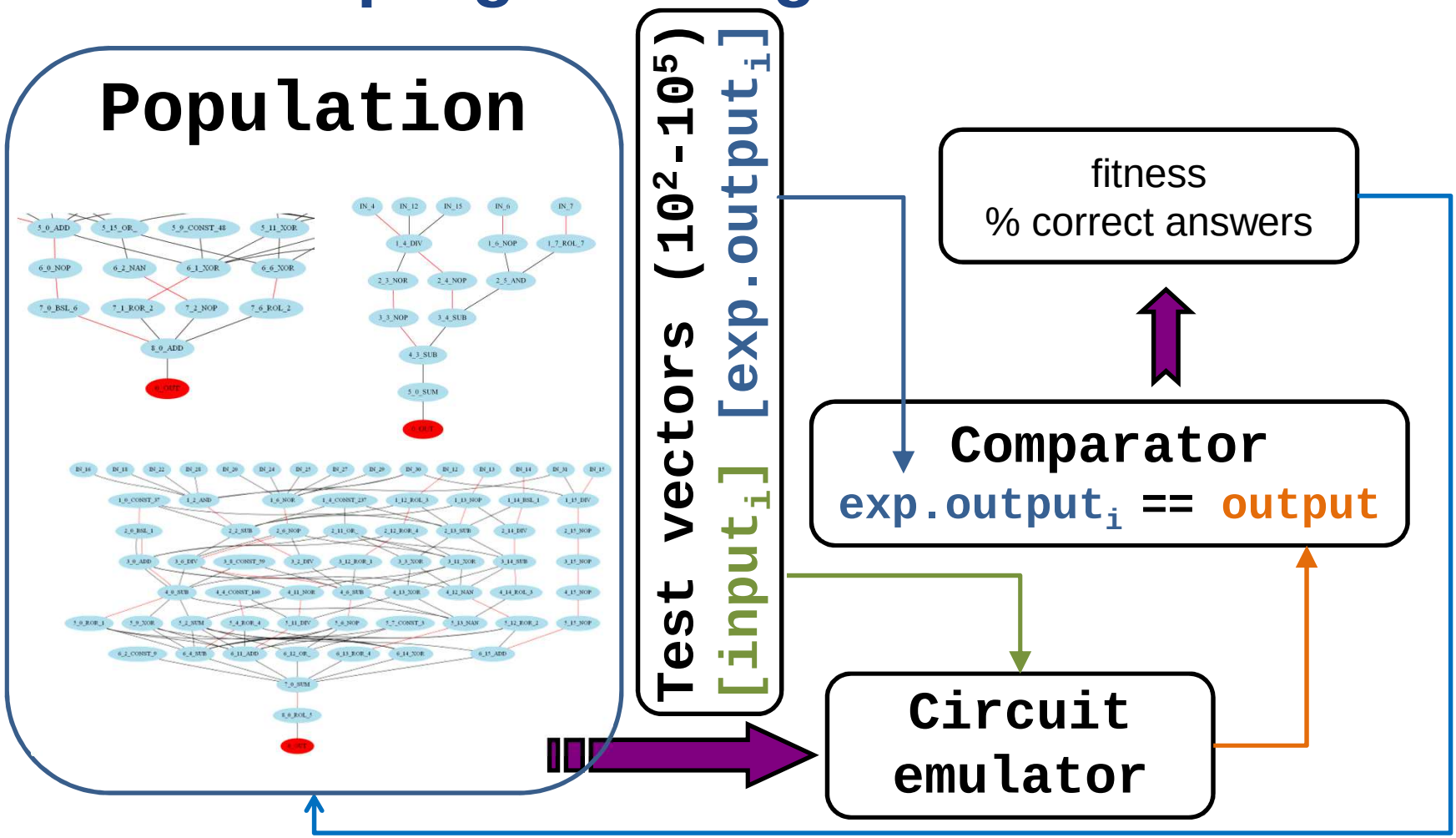
Input layer

Internal layers

Output layer

Outputs

Genetic programming of circuits



Test vectors

Quantum Random Bit Generator service

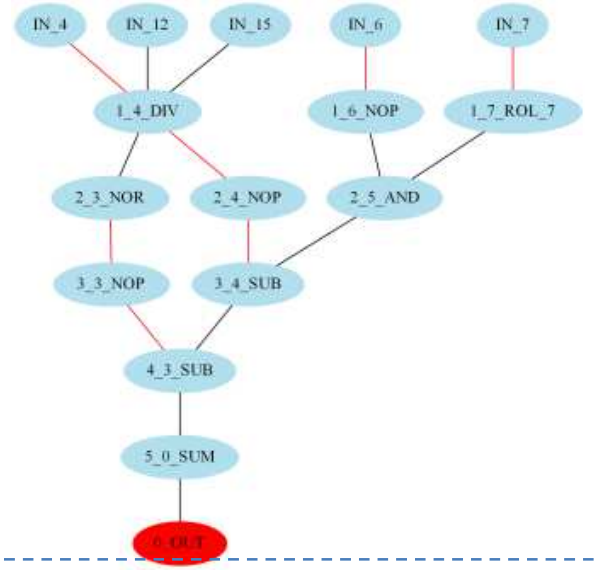
500x 1011010100...101

ECRYPT

500x 1001110011...100

Circuit execution

1011010100...101



Fitness

10110111

HW(10110111) > 4 => QRNG

Methodology

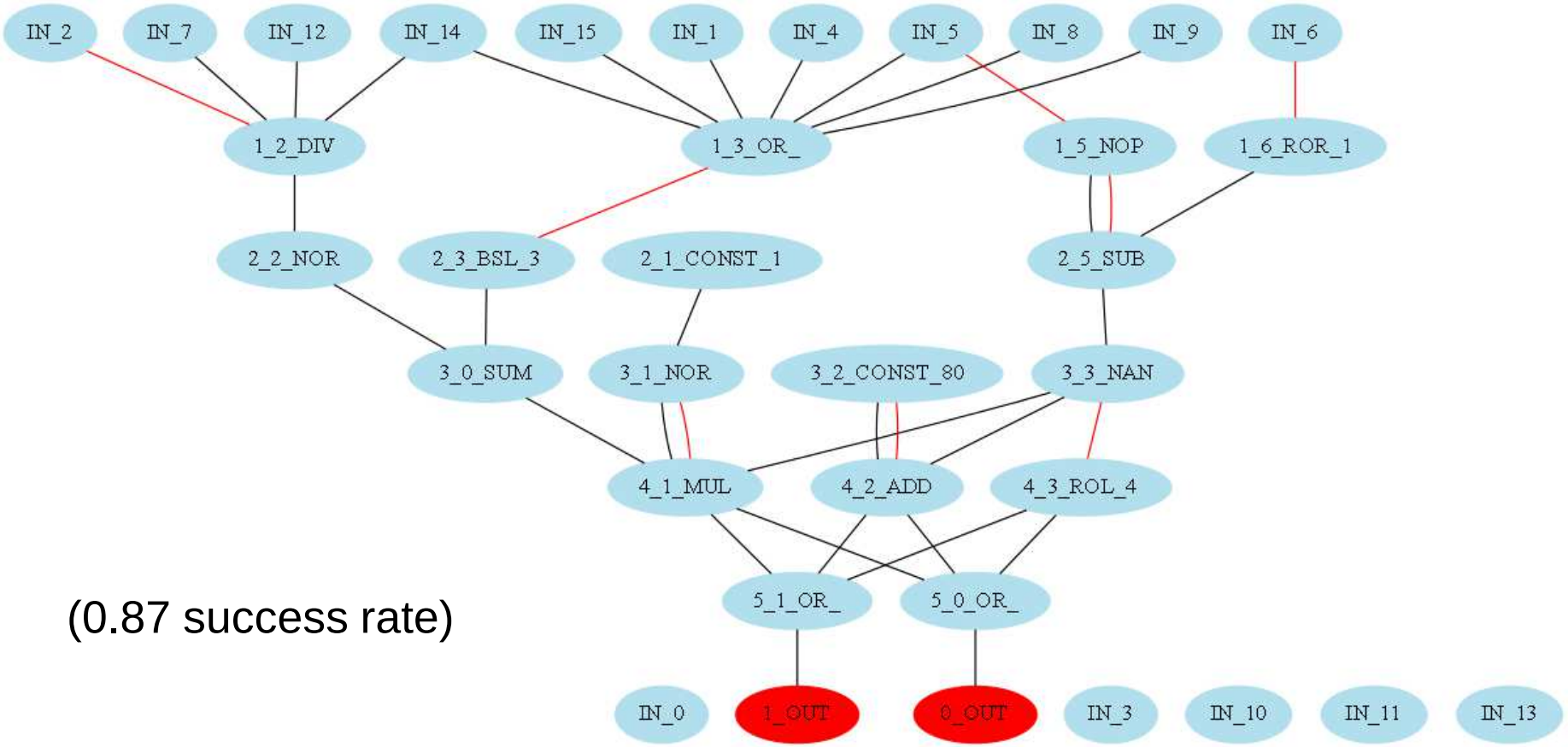
- Limit number of algorithm rounds
  - tested on 7 eStream and 18 SHA-3 candidates
- Generate & run STS NIST and Dieharder tests
- Prepare input data for EACirc
  - generate ½ test vectors from function (key change freq.)
  - generate ½ test vectors from truly random source (QRBGS http://random.irb.hr/)
- Generate & test software circuits (repeat, EA)

Were we successful?

- Definition of success?

- Better than random guessing?
- Better or at least as good as human-made batteries?
- Other advantages against statistical batteries?

Salsa20 – limited to two rounds



(0.87 success rate)

Test vectors – key change frequency

Key fixed for whole run (all generations)

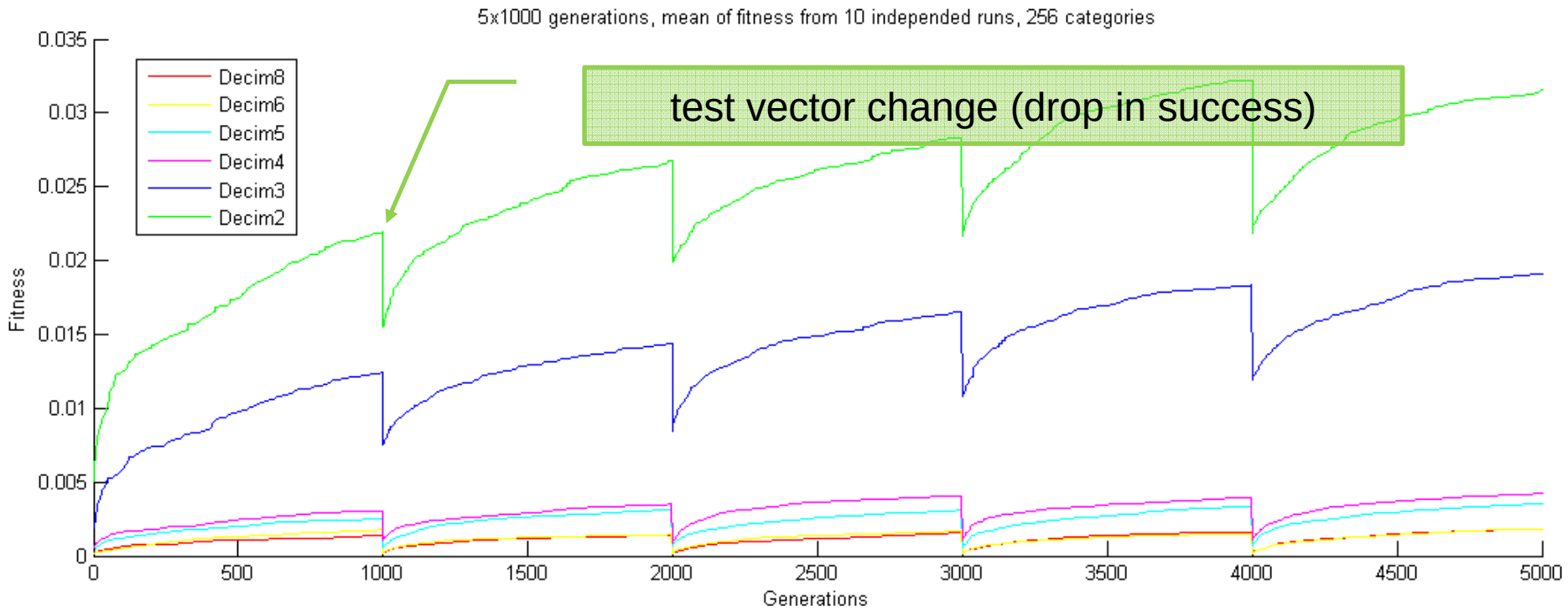
Key fixed only for one test set (e.g., 500 test vectors)

Key per every test vector (e.g., every 16 bytes)

Table 2: Results for Decim.

# of rounds	IV and key reinitialization								
	once for run			for each test set			for each test vector		
	Dieharder (x/20)	STS NIST (x/162)	EACirc	Dieharder (x/20)	STS NIST (x/162)	EACirc	Dieharder (x/20)	STS NIST (x/162)	EACirc
1	0.0	0	$n = 2681$	0.0	0	(0.85)	0.0	5	$n = 1431$
2	0.5	0	(0.54)	1.0	0	(0.54)	15.5	146	(0.52)
3	1.0	0	(0.53)	1.0	0	(0.53)	15.0	160	(0.52)
4	3.5	79	(0.52)	3.0	78	(0.52)	20.0	160	(0.52)
5	4.5	79	(0.52)	3.5	91	(0.52)	17.5	161	(0.52)
6	19.0	158	(0.52)	19.0	159	(0.52)	18.0	162	(0.52)
7	18.5	162	(0.52)	19.0	161	(0.52)	20.0	161	(0.52)
8	20.0	162	(0.52)	20.0	159	(0.52)	19.0	161	(0.52)

Decim – 6 out of 8 rounds (preliminary)



χ^2 difference between random/fnc histograms of categories

What is a function test then?

- One particular circuit?
  - circuit was evolved for particular function and key
  - sometimes, circuit works even when key is changed
  - (most probably) not useful for a different function

- Test = whole process with evolution of circuits!
  - Is evolution able to design a distinguisher in limited number of generations?
  - If so, then function output is defective!

Comparison to statistical batteries

- Advantages
  - new approach, no need for predefined pattern
  - dynamic construction of test for particular function
  - works on very short sequences (16 bytes only)
- Disadvantages
  - no proof of test quality or coverage (random search)
  - possibly hard to analyze the result (possibly automatic)
  - possibly longer test run time (learning period)

Questions ?

Thank you for your attention!

Questions