Additive and Multiplicative Complexities of Circuits

Luís T. A. N. Brandão · René Peralta · Meltem Sönmez Turan — Computer Security Division

INFORMATION TECHNOLOGY LABORATORY

NIST National Institute of Standards and Technology U.S. Department of Commerce

Boolean Circuits

A Boolean circuit computes a function using basic Boolean operators (gates). Such circuits underpin the implementation of many computations.

NIST research. The "circuit complexity" project explores designs for Boolean circuits of interest. Two metrics are of particular relevance:

- (+) additive complexity: # XORs and XNORs
- (×) multiplicative complexity (MC): # ANDs

Circuit	Gate count					Depth	
	All	AND	XOR	XNOR	NOT	All	AND
AES S-Box	113	32	77	4	0	27	6
AES S-Box^{-1}	121	34	83	4	0	21	4
AES-128(k,m)	28 600	6400	21 356	844	0	326	60
SHA-256(m)	115 882	22 385	89 248	3894	355	5403	1604

Table produced in cllaboration with Ç. Çalık. Legend: k: AES key; m: message (128-bit for AES; 512-bit for SHA-256)

Email: `circuit_complexity@nist.gov`

Webpage: `https://csrc.nist.gov/projects/circuit-complexity`

Data repository: `https://github.com/usnistgov/Circuits`

Poster produced for the NIST-ITL Science Day 2021

(L. Brandão is at NIST as a Contractor from Strativia)

MC of Cubic Boolean Functions

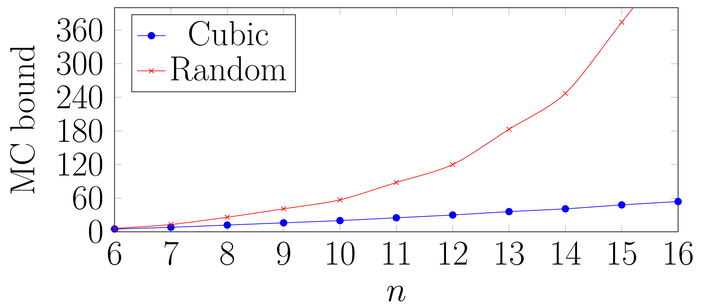
Aim: Construct efficient circuits for cubic Boolean functions in terms of # `AND` gates.

MC	Equivalence classes
2	`123, 123+14`
3	`123+45, 134+125, 123+24+15, 34+134+125`
4	`23+134+125, 24+34+134+125+35`

MC distribution of cubic Boolean functions, $n \leq 5$

Constructing n-bit cubic Boolean function f

1. Decompose f as $f = x_n f_1 + f_2$
   - f_1: $(n-1)$-bit function with degree ≤ 2
   - f_2: $(n-1)$-bit function with degree ≤ 3
2. Optimally implement f_1 with at most $\lfloor \frac{n-1}{2} \rfloor$ `AND` gates.
3. Apply this method to implement f_2.
4. Combine f_1 and f_2 circuits using one more `AND` gate.



Optimizing Linear Circuits

Scope: Reduce # XOR gates for linear functions.

- **MDS matrices**
- **Diffusion layer of ciphers**

Trivial implementation: A binary linear system ($n \times m$ matrix of bits) can be trivially implemented with $D = W - n$ XORs, where W is the matrix weight (number of 1's); and n is the number of rows.

Metric of interest: What improvement ratio can we get, compared with trivial implementations (D-XOR)? Across 57 circuits (20 S-Boxes; 37 MDS matrices), we (jointly with Ç. Çalık and T. Yalçın) are comparing synthesis tools vs. best known heuristics.

Hypothesis/evidence: Synthesis tools can be improved with heuristic techniques from the literature.