INDIAN INSTITUTE OF MANAGEMENT CALCUTTA

WORKING PAPER SERIES

WPS No. 713/ September 2012

Algorithms for Rotation Symmetric Boolean Functions

by

Subrata Das
Assistant Professor, Department of Information Technology,
Academy of Technology, West Bengal

Satrajit Ghosh
Assistant Professor, Department of Computer Science,
APC College, West Bengal

Parthasarathi Dasgupta
Professor, IIM Calcutta, Diamond Harbour Road, Joka, Kolkata 700104, India

&

Samar Sensarma
Professor, Department of Computer Science & Engineering University of Calcutta

Algorithms for Rotation Symmetric Boolean Functions

Subrata Das[1], Satrajit Ghosh[2]

exhaustively is exponential. Thus, in order to look for RSBFs, it is imperative to have an idea about the number of orbits(i.e. partitions) in rotation symmetric functions.

In this paper, we propose three simple algorithms for generating RSBF s of a given number of variables and implement upto 26 variables.

Rest of the paper is organized as follows. Section 2 reviews some related recent works. Section 3 introduces some terminologies to be used in subsequent discussions and Section 4 proposes three algorithms A, B and C for generating RSBF s of n variables. Section 5 briefly discusses the implementation of the algorithms A, B and C. Finally, Section 6 concludes the chapter and briefly states the future scopes of work.

2 Literature Review

An extensive study of symmetric Boolean functions, especially of their cryptographic properties has been done in [8]. In [2] Pieprzyk and Qu have studied a said to form an orbit. Figure 1 illustrates the set of 14 orbits for a function of 6 variables.

f (000000)g						orbit 0
f (000001)	(000010)	(000100)	(001000)	(010000)	(100000)g	orbit 1
f (000011)	(000110)	(001100)	(011000)	(110000)	(100001)g	orbit 2
f (000101)	(001010)	(010100)	(101000)	(010001)	(100010)g	orbit 3
f (000111)	(001110)	(011100)	(111000)	(110001)	(100011)g	orbit 4
f (001001)	(010010)	(100100)g				orbit 5
f (001011)	(010110)	(101100)	(011001)	(110010)	(100101)g	orbit 6
f (001101)	(011010)	(110100)	(101001)	(010011)	(100110)g	orbit 7
f (001111)	(011110)	(111100)	(111001)	(110011)	(100111)g	orbit 8
f (010101)	(101010)g					orbit 9
f (010111)	(101110)	(011101)	(111010)	(110101)	(101011)g	orbit 10
f (011011)	(110110)	(101101)g				orbit 11
f (011111)	(111110)	(111101)	(111011)	(110111)	(101111)g	orbit 12
f (111111)g						orbit 13

Fig. 1. Orbits of RSBF for n = 6

Each orbit is also known as a partition . Number of partitions ...

Observation 3 Product of the internal period and the number of substrings within a string yields the number of variables of the string.

For instance, in Figure 1, for Orbit 9, number of substrings is 3, internal period is 2 and the number of variables is 6.

3.1 Algebraic Normal forms and RSBF

The classical approach to the analysis, synthesis or testing of a switching circuit is based on the description by the Boolean algebra operators. A description of a switching circuit based on Modulo-2 arithmetic (the simplest case of the Galois field algebra [10]) is inherently redundancy-free, and is implemented as the multi-level tree of XOR (addition operator over $GF(2)$) gates.

Definition 5. An n-variable Boolean function $f(x_{n-1}, \ldots, x_1, x_0)$ can be ex-

4 Proposed Algorithms

The proposed algorithm starts with a string of n zeros, which forms the rst orbit. Subsequent representative strings are formed by the odd numbers whose binary representation has

number 9 is deleted from the AVL tree (Figure 3(d)) For $k = 2$, $11 \quad 2^k > 31$. Rotate 01101 (=13) two to four times to yield respectively 01101 (=13), 11010 (=26), and 10101 (=21). The odd values 13 and 21 are stored in the AVL tree (Figure 3(e)).

Orbit 7

4.1 An improved Algorithm

The proposed Algorithm A is improved with a minor modification. We note the following observation:

Consider the bit string (for an odd number) having 1 at the right-most position $\{0^{n-1}1\}$. Let this bit string be right-rotated right by 1-bit (i.e., n-bits left-rotate) to form a new bit string P, say $P = \{1^1 0^{n-1}\}$. The starting bit-string of each orbit is an odd number, generated by simply adding 2 to the starting string of the previous orbit. In the previous algorithm, we had to check all these starting strings in an AVL tree to avoid repetitive occurrences of numbers.

Lemma 7 If the starting string of an orbit is $P + 1$, then the numbers between $P + 1$ and the number generated by one-bit right-rotation of $(P - 1)$ may be ignored for generating subsequent orbits.

Proof. From Algorithm A it is clear that the starting string of the orbits must be odd number. From the previous lemma 6 it is clear that the last rotation cousins are the consecutive numbers if starting strings of the orbits are consecutive odd numbers. When some number misses in the orbit then its corresponding last rotation cousin do no appear. So as soon as the starting string of the orbit becomes $P+1$ which comes already as the last cousin of some orbit we do not cosier from this to the last cousin of $P - 1$.

The above lemma shows that the n-bit left-rotation (= 1-bit right rotation) of successive odd numbers results in successive numbers. Thus, whenever the odd number becomes $(P + 1)$, all the successive numbers up to the number which is RR of $(P - 1)$ already appears.

A formal description of the proposed Algorithm B is given in Figure 4.

Following result is clear from the description Algorithm B.

Lemma 8 Worst-case time complexity of Algorithm B is 2^n.

Lemma 9 The proposed Algorithm B requires a maximum space of $\{2^{n-1}\}_n - a$, where $a = RR(2^{n-1} + 1) - (2^{n-1} + 1)$.

Proof. Follows from Lemma 6.

4.2 A further improved Algorithm

In both the algorithms proposed above, an AVL tree is used to reduce certain iterations. The following observation helps in getting rid of this auxiliary data structure and its associated operations.

Observation 4 If the rotation cousin of an odd starting number of an orbit is also odd, and is greater than the value of the next starting string of the next orbit, then this starting string may be discarded.

A formal description of Algorithm C is given in Figure 5.

Following result is clear from the description Algorithm C.

Algorithm C

Data structures: Cntr :# of orbits, Result

2. J. Pieprzyk and C. X. Qu, Fast hashing and Rotatio-symmetric functions, Journal of Universal Computer Science, pp. 20-31, vol. 5, no. 1, 1999
3. P. Stanica and S. Maitra, Rotation Symmetric Boolean functions - Count and Cryptographic properties, Discrete Applied Mathematics, vol. 156, no. 10, May, 2008.
4. P. Stanica and S. Maitra and J A Clark, Results on Rotation s ymmetric Bent and Correlation immune Boolean functions in B. Roy and W. Meier (Eds.), FSE 2004, LNCS 3017, pp. 161-177, 2004, International Assoc. for Cryptologic Research.
5. M.Hell, A. Maximov,and S. Maitra, On e cient implementation of search stratgey for RSBF, 9^{th}