Boolean Functions Of Low Polynomial Degree

Pēteris Ledinš, Rihards Opmanis[1]

Department of Computer Science, University of Latvia,
29 Raina boulevard, Riga, Latvia
ledins@latnet.lv, rixix@navigators.lv

Abstract. Boolean functions of high deterministic query complexity and low degree of representing polynomial have different applications in theoretical computer science, yet not many are known with such properties. We analyze some previously known and construct several new functions with such properties.
Keywords. Boolean functions, polynomial degree, decision trees.

1. Introduction

Boolean functions being quite simple in definition as returning results from $\{a,b\}^n$ to $\{a,b\}$, where $a \neq b$ can be chosen, are a subject to careful investigation as they can be used in proving different characteristics of different computational models.

For example, Boolean functions can be used to prove different results considering quantum decision tree complexity. If we use $deg(f)$ as the degree of multilinear polynomial representing Boolean function f, and $D(f)$ as the deterministic decision tree complexity of f then we have $D(f) \geq deg(f)$ [1]. For exact quantum decision tree complexity there exists a result $Q_E(f) \geq \frac{deg(f)}{2}$ [6].

For use in quantum computing the challange is to find a function with high $D(f)$ and low $deg(f)$, that could give some advantages in proving bound-related problems.

2. Notation

As noted before we will use $D(f)$, but it is easier to determine the maximum sensitivity of a Boolean function - $s(f)$. Sensitivity of f on input $(x_1, x_2, \ldots, x_n)$ is the number of of variables x_i with property that $f(x_1, x_2, \ldots, x_i, \ldots, x_n) \neq f(x_1, x_2, \ldots, 1 - x_i, \ldots, x_n)$

It has been proved that $s(f) \leq D(f)$ [2]. So, if we know $s(f)$ then we know that $D(f)$ is at least the same.

To describe Boolean functions we use a description in form of tables with each row for different type of input and each column for each variable. Each cell contains "1", "0" or "$-$" in each cell. A table is constructed to show all possible inputs giving either 1 or 0 as the result of function. "0" or "1" in ith column symbolizes that input defined by the row must have $x_i = 0$ or $x_i = 1$ resp. while "$-$" means that the value of x_i is not significant for the input.

[1]Research supported by Grant No.01.0354 from the Latvian Council of Science and by the European Commission, Contract IST-1999-11234 (QAIP).

3. Existing results

Currently there are several functions providing $D(f) > deg(f)$ that are used to create (e.g. iterations and other various operations) others.

1. Function $f_1(x_1, x_2, x_3)$ being 0 iff all variables are equal [1]. $D(f_1) = 3$, $deg(f_1) = 2$
2. Function $f_2(x)$ that equals 1 iff $x = x_1x_2x_3x_4$ equals 0011, 0100, 0101, 0111, 1000, 1010, 1011, or 1100 with $deg(f_2) = 2$ and $D(f_2) = 3$ [3].
3. Function $f_3(x_1, x_2, x_3, x_4, x_5, x_6)$ with $deg(f_3) = 3$ and $D(f_3) = 6$ by Kushilevitz, cited by [4]. f_3 equals 0 when sum of x_i equals 0, 4 or 5 or 3 and one of the following is true: $x_1 = x_2 = x_3 = 1$, $x_2 = x_3 = x_4 = 1$, $x_3 = x_4 = x_5 = 1$, $x_4 = x_5 = x_1 = 1$, $x_5 = x_1 = x_2 = 1$, $x_1 = x_3 = x_6 = 1$, $x_1 = x_4 = x_6 = 1$, $x_2 = x_4 = x_6 = 1$, $x_2 = x_5 = x_6 = 1$, $x_3 = x_5 = x_6 = 1$. Otherwise the value is 1.
4. Function $f_4(x_1, x_2, x_3, x_4, x_5, x_6, x_7)$ with $D(f) = 7$ and $deg(f) = 4$ from [5]. The value of function is 1 iff input is defined in Table 1.

x_1	x_2	x_3	x_4	x_5	x_6	x_7
–	1	–	0	–	0	0
–	–	0	–	0	0	1
–	0	1	0	0	–	–
0	–	0	0	–	1	–
1	0	0	–	–	–	0
0	0	–	–	1	0	–
0	–	–	1	0	–	0

Table 1: Function f_4 from [5]

4. Construction of Boolean functions

In our search for Boolean functions of specified qualities we use the previously mentioned way of description and analyze regularities. For example, some kind of regularities can be seen in Table 1.

4.1. Hadamard matrices

A Hadamard matrix is a square matrix containing only $1s$ and $-1s$ such that when any two columns or rows are placed side by side, half the adjacent cells are the same sign and half the other (excepting from the count an L-shaped "half-frame" bordering the matrix on two sides which is composed entirely of $1s$) [7]. It is easy to see that one can change the order of columns and rows, still keeping the property of Hadamard matrices.

We use Hadamard matrix of order 8 from [8] shown in Table 2.

1	1	1	1	1	1	1	1
−1	−1	−1	1	−1	1	1	1
−1	1	−1	−1	1	−1	1	1
−1	1	1	−1	−1	1	−1	1
−1	1	1	1	−1	−1	1	−1
−1	−1	1	1	1	−1	−1	1
−1	1	−1	1	1	1	−1	−1
−1	−1	1	−1	1	1	1	−1

Table 2: Hadamard matrix had.8.1 from [8]

4.2. Analysis of f_4

Table 1 can be constructed easily from Table 2 - first column and first row are to be removed, matrix has to be read from the right side, $-1s$ replaced with $0s$ and $1s$ with "$-$" signs (as in Table 3). After that we only have to insert $1s$ in appropriate positions.

x_1	x_2	x_3	x_4	x_5	x_6	x_7
−	−	−	0	−	0	0
−	−	0	−	0	0	−
−	0	−	0	0	−	−
0	−	0	0	−	−	−
−	0	0	−	−	−	0
0	0	−	−	−	0	−
0	−	−	−	0	−	0

Table 3: Building function f_4

First we want the definition of function to satisfy the following property:

Property 1 (Rihard's property) *If there exists* 1 *in cell* (i, j) *and* 1 *in cell* (k, l) *then exactly one cell from* $\{(i, l), (k, j)\}$ *must hold* 0 *and exactly one -* "$-$".

This property is a way to:

1. Allow each input be defined with exactly one row
2. Have some kind of minimal distance between each two rows, thus having a small amount of $1s$ and $0s$ used in the definition.

As one can see the definition of f_4 satisfies Property 1. Of course, there are several ways to put the $1s$ in the matrix of a function. A very regular way is moving the first row of Table 3 to the bottom and putting $1s$ on the diagonal as in Table 4.

x_1	x_2	x_3	x_4	x_5	x_6	x_7
$-$	$-$	0	$-$	0	0	1
$-$	0	$-$	0	0	1	$-$
0	$-$	0	0	1	$-$	$-$
$-$	0	0	1	$-$	$-$	0
0	0	1	$-$	$-$	0	$-$
0	1	$-$	$-$	0	$-$	0
1	$-$	$-$	0	$-$	0	0

Table 4: Function f_5

Lemma 1 $s(f_5) = 7$

Lemma 1 is provided by sensitivity on zero input.

Theorem 1 $D(f_5) = 7$

Theorem 3 straight from Lemma 1.

Theorem 2 $deg(f_5) = 4$

Theorem 2 is proved by construction.

The number of members of the representing polynomial for f_5 is the same as for f_4 and equals 56 that shows how similar both the functions are.

4.3. More Boolean functions

Using the method showed previously we construct two other Boolean functions with similar properties representing the inputs giving 1 as result in Table 5 and Table 6. It must be reminded that f_6 is the same function as f_1.

x_1	x_2	x_3
0	$-$	1
$-$	1	0
1	0	$-$

Table 5: Function f_6

Theorem 3 $D(f_7) = 11,\ deg(f_7) = 6$

Deterministic query complexity comes from sensitivity and degree is found by construction.

Using the same method we can construct Boolean functions for other numbers of variables - 19, 23, 31, 43, 47 - using Hadamard matrices available from [8]. However,

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}
0	–	0	0	0	–	–	–	0	–	1
–	0	0	0	–	–	–	0	–	1	0
0	0	0	–	–	–	0	–	1	0	–
0	0	–	–	–	0	–	1	0	–	0
0	–	–	–	0	–	1	0	–	0	0
–	–	–	0	–	1	0	–	0	0	0
–	–	0	–	1	0	–	0	0	0	–
–	0	–	1	0	–	0	0	0	–	–
0	–	1	0	–	0	0	0	–	–	–
–	1	0	–	0	0	0	–	–	–	0
1	0	–	0	0	0	–	–	–	0	–

Table 6: Function f_7

we have not constructed polynomials to find the degree of theese matrices, but still we can do some estimates using the method described in [9]. That gives the degree of representing polynomial $deg(f) = \frac{n+1}{2}$ for these functions.

5. Future work

The new functions found do have characteristics demanded, but still for these functions $deg(f) > \frac{n}{2}$, when $D(f) = n$, where n - the number of variables. A result needed is some function f_x with $deg(f) < \frac{n}{2}$ that could possibly give some quantum query algorithm that had better advantages against classical counterpart than any now known.

Besides that Hadamard matrices and the likes could provide with interesting Boolean functions and thus need deeper investigation.

References

[1] N. Nisan and M. Szegedy, On the degree of Boolean functions as real polynomials Computational Complexity, 4(4), pp. 301-313, 1994. Earlier version in STOC'92.

[2] H. Buhrman and R. de Wolf. Complexity Measures and Decision Tree Complexity: A Survey. In Theoretical Computer Science, 288:21-43, 2002.

[3] A. Ambainis, Polynomial degree vs. quantum query complexity, Proceedings of FOCSŠ2003, pages 230-239

[4] N. Nisan and A. Wigderson, On rank vs. communication complexity, Proc. of FOCSŠ94, pp. 841-836, 1994.

[5] A. Ambainis, R. M. Freivalds, Boolean function with a low polynomial degree, Proceedings of the Latvian Academy of Sciences, vol. 57, 2003, pages 74-77.

[6] R. Beals, H. Buhrman, R. Cleve, M. Mosca, R. de Wolf. Quantum lower bounds by polynomials. Journal of ACM, 48: 778-797, 2001.

[7] E. W. Weisstein, Hadamard Matrix From MathWorld - A Wolfram Web Resource. http://mathworld.wolfram.com/HadamardMatrix.html

[8] N. J. A. Sloane, A Library of Hadamard Matrices,
http://www.research.att.com/~njas/hadamard/
[9] A. Belovs, A Way Of Constructing Functions With A Low Polynomial Degree, 2004