Sensitivity of Boolean Functions with Low Polynomial Degree

Igors Stepanovs
whiteCryption

MSc thesis at University of Latvia
Supervisor: prof. Andris Ambainis

September 29, 2012

Section 1

Preliminaries

Preliminaries: Outline

- We study the relationship between the degree and the sensitivity of Boolean functions
- We start by defining these complexity measures

The Degree of a Boolean Function

- Any Boolean function $f \colon \{0,1\}^n \to \{0,1\}$ can be represented by some multilinear polynomial $p : \{0,1\}^n \to \{0,1\}$ such that for all possible inputs $x \in \{0,1\}^n$ it holds that $f(x) = p(x)$
- Define $\deg(f) \stackrel{\text{def}}{=} \deg(p)$

The Degree of a Boolean Function - II

- Consider Nisan and Szegedy function $f\colon \{0,1\}^3 \to \{0,1\}$ such that $f(x) \overset{\text{def}}{=} 1$ if and only if $|x| \in \{1,2\}$
- Boolean formula:
  $f(x) = \neg(x_1 \wedge x_2 \wedge x_3) \wedge \neg(\neg x_1 \wedge \neg x_2 \wedge \neg x_3)$
- Representing polynomial:
  $p(x) = x_1 + x_2 + x_3 - x_1x_2 - x_1x_3 - x_2x_3$
- Hence, $\deg(f) = \deg(p) = 2$

The Sensitivity of a Boolean Function

- The sensitivity of a Boolean function over an input x is the number of input bits x_i such that flipping the value of x_i also flips the result of this function
- We denote it by $s_x(f)$
- We define the sensitivity of a Boolean function $s(f)$ as the maximum value of $s_x(f)$ over all possible inputs x

The Sensitivity of a Boolean Function - II

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

The Sensitivity of a Boolean Function - II

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$	$s_x(f)$
0	0	0	0	?
0	0	1	1	
0	1	0	1	
0	1	1	1	
1	0	0	1	
1	0	1	1	
1	1	0	1	
1	1	1	0	

The Sensitivity of a Boolean Function - II

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$	$s_x(f)$
0	0	0	0	?
0	0	1	1	
0	1	0	1	
0	1	1	1	
1	0	0	1	
1	0	1	1	
1	1	0	1	
1	1	1	0	

Choose $x = (0, 0, 0)$
Then $f(0, 0, 0) = 0$

The Sensitivity of a Boolean Function - II

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$	$s_x(f)$
0	0	0	0	?
0	0	1	1	
0	1	0	1	
0	1	1	1	
1	0	0	1	
1	0	1	1	
1	1	0	1	
1	1	1	0	

Choose $x = (0,0,0)$
Then $f(0,0,0) = 0$

$f(1,0,0) = 1$
$f(0,1,0) = 1$
$f(0,0,1) = 1$

The Sensitivity of a Boolean Function - II

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$	$s_x(f)$
0	0	0	0	$?$
0	0	1	1	
0	1	0	1	
0	1	1	1	
1	0	0	1	
1	0	1	1	
1	1	0	1	
1	1	1	0	

Choose $x = (0,0,0)$
Then $f(0,0,0) = 0$

$f(1,0,0) = 1$
$f(0,1,0) = 1$
$f(0,0,1) = 1$

$\Rightarrow s_x(f) = 3$

The Sensitivity of a Boolean Function - II

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$	$s_x(f)$
0	0	0	0	3
0	0	1	1	?
0	1	0	1	
0	1	1	1	
1	0	0	1	
1	0	1	1	
1	1	0	1	
1	1	1	0	

The Sensitivity of a Boolean Function - II

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$	$s_x(f)$
0	0	0	0	3
0	0	1	1	?
0	1	0	1	
0	1	1	1	
1	0	0	1	
1	0	1	1	
1	1	0	1	
1	1	1	0	

Choose $x = (0, 0, 1)$
Then $f(0, 0, 1) = 1$

The Sensitivity of a Boolean Function - II

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$	$s_x(f)$
0	0	0	0	3
0	0	1	1	?
0	1	0	1	
0	1	1	1	
1	0	0	1	
1	0	1	1	
1	1	0	1	
1	1	1	0	

Choose $x = (0, 0, 1)$
Then $f(0, 0, 1) = 1$

$f(1, 0, 1) = 1$
$f(0, 1, 1) = 1$
$f(0, 0, 0) = 0$

The Sensitivity of a Boolean Function - II

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$	$s_x(f)$
0	0	0	0	3
0	0	1	1	?
0	1	0	1	
0	1	1	1	
1	0	0	1	
1	0	1	1	
1	1	0	1	
1	1	1	0	

Choose $x = (0,0,1)$

Then $f(0,0,1) = 1$

$f(1,0,1) = 1$

$f(0,1,1) = 1$

$f(0,0,0) = 0$

$\Rightarrow s_x(f) = 1$

The Sensitivity of a Boolean Function - II

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$	$s_x(f)$
0	0	0	0	3
0	0	1	1	1
0	1	0	1	
0	1	1	1	
1	0	0	1	
1	0	1	1	
1	1	0	1	
1	1	1	0	

The Sensitivity of a Boolean Function - II

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$	$s_x(f)$
0	0	0	0	3
0	0	1	1	1
0	1	0	1	1
0	1	1	1	1
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	0	3

The Sensitivity of a Boolean Function - II

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$	$s_x(f)$
0	0	0	0	3
0	0	1	1	1
0	1	0	1	1
0	1	1	1	1
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	0	3

$$s(f) = \max_x s_x(f) = 3$$

Section 2

The Goal of the Project

The Goal of the Project: Outline

- Consider the gap between $\deg(f)$ and $s(f)$ as $\log_{s(f)} \deg(f)$
- We want to find a Boolean function f for which the distance between $\deg(f)$ and $s(f)$ is as big as possible, i.e.

$$\log_{s(f)} \deg(f) \longrightarrow \min$$

The Motivation of the Project

- We want to find relations between the complexity measures of Boolean functions
- The sensitivity of a Boolean function serves as a lower bound for its deterministic query complexity and communication complexity
- Finding a Boolean function with a big gap between $\deg(f)$ and $s(f)$ might be a first step to finding a Boolean function for which the quantum query complexity would be more than twice smaller than the deterministic query complexity

The Studied Cases

The previously known bounds on the gap:

f	$\deg(f)$	$s(f)$	The Gap
Nisan and Szegedy	2	3	$\log_3 2 \approx 0.63$
Kushilevitz	3	6	$\log_6 3 \approx 0.61$
Nisan and Szegedy*	4	9	$\log_9 4 \approx 0.63$

(*) The Nisan and Szegedy function can be generalized to get Boolean functions with $\deg(f) = 2^k$, $s(f) = 3^k$ and hence $\log_{3^k} 2^k \approx 0.63$

The Studied Cases - II

- For $\deg(f) = 5$ there is no known strict upper bound on $s(f)$
- In order to improve the best known gap between $\deg(f)$ and $s(f)$ we want to achieve $\log_{s(f)} 5 < \log_6 3$
- The desired sensitivity is $s(f) \geq 14$

Section 3

The Initial Approach

The Initial Approach: Outline

- There are 2^{2^n} different Boolean functions of *n* variables
- We map all Boolean functions to a small number of different polynomials which depend on particular properties of these functions
- Instead of studying Boolean functions themselves we work with the corresponding characteristic polynomials

Symmetrization Polynomial

- Consider a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$
- For an arbitrary input x we have $|x| \in \{0, ..., n\}$
- For each value of $|x|$ we count the number of times such that $f(x) = 1$
- We write $r(k) = m$ when there are exactly m cases with $f(x) = 1$ amongst all possible inputs x with $|x| = k$
- We call r a "symmetrization polynomial"

Symmetrization Polynomial - II

- We normally represent a symmetrization polynomial r by a table

k	0	1	...	n
$r(k)$	a_0	a_1	...	a_n

  for some values $a_i = r(i)$
- Alternatively, we write

$$r(k) = [a_0, a_1, ..., a_n]$$

Symmetrization Polynomial - III

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

k	0	1	2	3
$r_f(k)$				

Symmetrization Polynomial - III

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$ ↓
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0
			$\sum=0$

k	0	1	2	3
$r_f(k)$	?			

Symmetrization Polynomial - III

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

k	0	1	2	3
$r_f(k)$	0			

Symmetrization Polynomial - III

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$ ↓
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0
			$\sum = 3$

k	0	1	2	3
$r_f(k)$	0	?		

Symmetrization Polynomial - III

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

k	0	1	2	3
$r_f(k)$	0	3		

Symmetrization Polynomial - III

Consider the Nisan and Szegedy function f:

↓

x_1	x_2	x_3	$f(x)$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

$\sum=3$

k	0	1	2	3
$r_f(k)$	0	3	?	

Symmetrization Polynomial - III

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

k	0	1	2	3
$r_f(k)$	0	3	3	

Symmetrization Polynomial - III

Consider the Nisan and Szegedy function f:

↓

x_1	x_2	x_3	$f(x)$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

$\sum = 0$

k	0	1	2	3
$r_f(k)$	0	3	3	?

Symmetrization Polynomial - III

Consider the Nisan and Szegedy function f:

x_1	x_2	x_3	$f(x)$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

k	0	1	2	3
$r_f(k)$	0	3	3	0

Generating the Symmetrization Polynomials

Simple observations:

- $\deg(r_f) \leq \deg(f)$
- $r_f = [0, k, ...] \Rightarrow s(f) \geq k$
- Any f with $s(f) = k$ can be relabeled so that its $r_f = [0, k, ...]$
- In order to find all r_f which might correspond to $\deg(f) = d$ and $s(f) = k$, we interpolate r_f in $i \leq d + 1$ different points such that $r_f = [0, k, ...]$

We find that for $\deg(f) = 5$ the sensitivity upper bound is $s(f) \leq 15$, we focus on this case

Splitting the Polynomials

- We can split any representing polynomial by assigning $x_i = 0$ or $x_i = 1$ for any $i \in \{1, ..., n\}$
- For the Nisan and Szegedy representing polynomial we toggle the value of x_3:
  $p = x_1 + x_2 + x_3 - x_1x_2 - x_1x_3 - x_2x_3$
  $p_0 = x_1 + x_2 - x_1x_2$
  $p_1 = 1 - x_1x_2$
- Then $p = (1 - x_3) \cdot p_0 + x_3 \cdot p_1$
- The symmetrization polynomial of p is also getting split:

k	0	1	2	3
$r(k)$	0	3	3	0
$r_0(k)$	0	2	1	
$r_1(k)$	1	2	0	

Splitting the Polynomials

- We can split any representing polynomial by assigning $x_i = 0$ or $x_i = 1$ for any $i \in \{1, ..., n\}$
- For the Nisan and Szegedy representing polynomial we toggle the value of x_3:
  $p = x_1 + x_2 + x_3 - x_1x_2 - x_1x_3 - x_2x_3$
  $p_0 = x_1 + x_2 - x_1x_2$
  $p_1 = 1 - x_1x_2$
- Then $p = (1 - x_3) \cdot p_0 + x_3 \cdot p_1$
- The symmetrization polynomial of p is also getting split:

k	0	1	2	3
$r(k)$	0	3	3	0
$r_0(k)$	0	2	1	
$r_1(k)$	1	2	0	

$$r(0) = r_0(0)$$

Splitting the Polynomials

- We can split any representing polynomial by assigning $x_i = 0$ or $x_i = 1$ for any $i \in \{1, ..., n\}$
- For the Nisan and Szegedy representing polynomial we toggle the value of x_3:
  $p = x_1 + x_2 + x_3 - x_1x_2 - x_1x_3 - x_2x_3$
  $p_0 = x_1 + x_2 - x_1x_2$
  $p_1 = 1 - x_1x_2$
- Then $p = (1 - x_3) \cdot p_0 + x_3 \cdot p_1$
- The symmetrization polynomial of p is also getting split:

k	0	1	2	3
$r(k)$	0	3	3	0
$r_0(k)$	0	2	1	
$r_1(k)$	1	2	0	

$$r(1) = r_0(1) + r_1(0)$$

Splitting the Polynomials

- We can split any representing polynomial by assigning $x_i = 0$ or $x_i = 1$ for any $i \in \{1, \ldots, n\}$
- For the Nisan and Szegedy representing polynomial we toggle the value of x_3:
  $p = x_1 + x_2 + x_3 - x_1x_2 - x_1x_3 - x_2x_3$
  $p_0 = x_1 + x_2 - x_1x_2$
  $p_1 = 1 - x_1x_2$
- Then $p = (1 - x_3) \cdot p_0 + x_3 \cdot p_1$
- The symmetrization polynomial of p is also getting split:

k	0	1	2	3
$r(k)$	0	3	3	0
$r_0(k)$	0	2	1	
$r_1(k)$	1	2	0	

$$r(2) = r_0(2) + r_1(1)$$

Splitting the Polynomials

- We can split any representing polynomial by assigning $x_i = 0$ or $x_i = 1$ for any $i \in \{1, ..., n\}$
- For the Nisan and Szegedy representing polynomial we toggle the value of x_3:
  $p = x_1 + x_2 + x_3 - x_1x_2 - x_1x_3 - x_2x_3$
  $p_0 = x_1 + x_2 - x_1x_2$
  $p_1 = 1 - x_1x_2$
- Then $p = (1 - x_3) \cdot p_0 + x_3 \cdot p_1$
- The symmetrization polynomial of p is also getting split:

k	0	1	2	3
$r(k)$	0	3	3	0
$r_0(k)$	0	2	1	
$r_1(k)$	1	2	0	

$r(3) = r_1(2)$

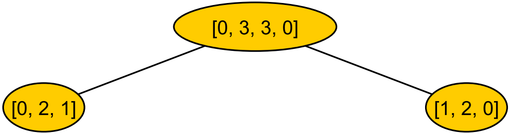
Splitting the Polynomials

- We can split any representing polynomial by assigning $x_i = 0$ or $x_i = 1$ for any $i \in \{1, ..., n\}$
- For the Nisan and Szegedy representing polynomial we toggle the value of x_3:

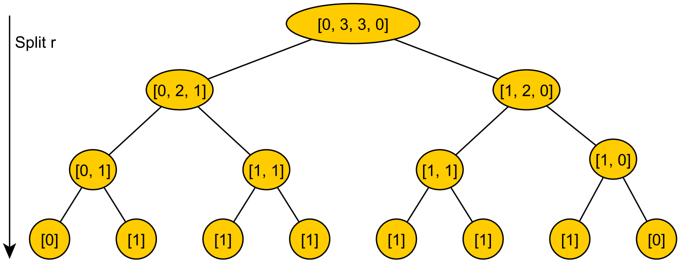
  $p = x_1 + x_2 + x_3 - x_1x_2 - x_1x_3 - x_2x_3$

  $p_0 = x_1 + x_2 - x_1x_2$

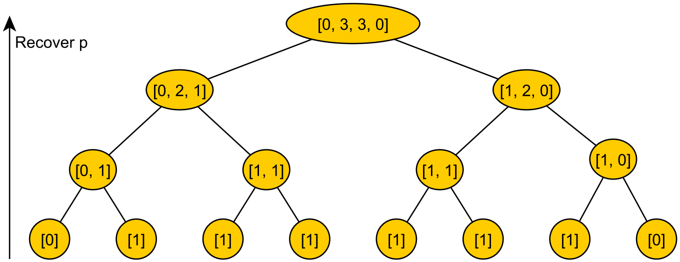
  $p_1 = 1 - x_1x_2$
- Then $p = (1 - x_3) \cdot p_0 + x_3 \cdot p_1$
- The symmetrization polynomial of p is also getting split:



Splitting the Polynomials - II



Splitting the Polynomials - II



Splitting the Polynomials - III

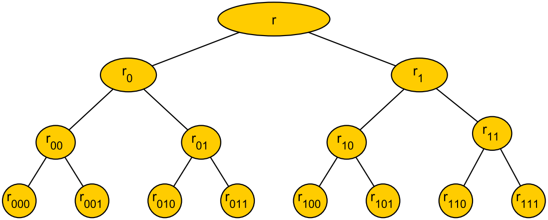
The current solution is very inefficient:

- We need to generate and every r which might correspond to the desired p
- Too many ways to split r
- Too many ways to recover p after finishing to split r

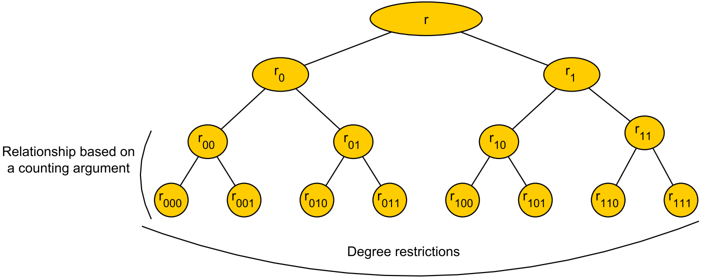
Section 4

Further Improvements

Further Improvements: Outline



Further Improvements: Outline



Relationship Within a Single Division Level

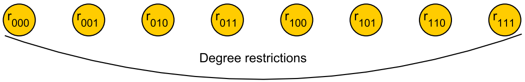
Express the representing polynomial p in the following way:

$$A + Bx_n + Cx_{n-1} + Dx_{n-2} + Ex_nx_{n-1} + Fx_nx_{n-2} + Gx_{n-1}x_{n-2} + Hx_nx_{n-1}x_{n-2}$$

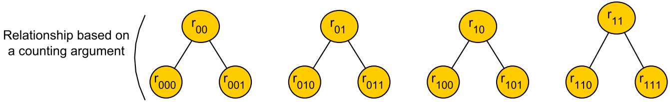
where $x_i \notin A, B, C, D, E, F, G, H$ for all $i \in \{n-2, n-1, n\}$

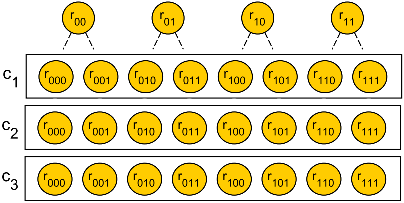
If $\deg(p) \leq d$, then the following must hold:

- $p_{000} = A$
- $p_{001} = A + D$
- $p_{100} = A + B$
- $p_{101} = A + B + D + F$
- Compute $F = p_{101} - p_{100} - p_{001} + p_{000}$
- We require $\deg(F) \leq d - 2$, hence $\deg(r_F) \leq d - 2$



Relationship Between Different Division Levels





- We use r_b^i to denote the symmetrization polynomial which corresponds to a new Boolean function after we set $x_i = b$ in the current Boolean function
- $\sum_{i=1}^{n} r_0^i(k) = \binom{n-k}{1} r(k)$
- $\sum_{i=1}^{n} r_1^i(k) = \binom{k+1}{1} r(k+1)$

Section 5

Results

Results

- Implemented the described approach in C++
- Managed to use the program to build Nisan and Szegedy ($\deg = 2$) and Kushilevitz ($\deg = 3$) functions
- Managed to show that there exists no Boolean function f with $\deg(f) = 5$ and $s(f) \geq 15$.
- For $\deg(f) = 5$ and $s(f) = 15$ had to process only 165 different ways to split some initial symmetrization polynomials
- The current method is too inefficient to process other cases (e.g. for $s(f) = 14$)

Thank you!