Heuristic algorithms for obtaining Polynomial Threshold Functions with low densities

Can Eren Sezener   Erhan Oztop

Ozyegin University
Istanbul, Turkey

May 27, 2015

Boolean functions

We call f a Boolean function (BF) iff it has the form $f : \mathbf{B}^n \rightarrow \mathbf{B}$, where $\mathbf{B} = \{True, False\}$ and $n \in \mathbb{N}$.

Boolean functions

We call f a Boolean function (BF) iff it has the form $f : \mathbf{B}^n \rightarrow \mathbf{B}$, where $\mathbf{B} = \{True, False\}$ and $n \in \mathbb{N}$.
By associating -1 with True and $+1$ with False, a BF can be considered as a real valued function, $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$.

Boolean functions

We call f a Boolean function (BF) iff it has the form $f : \mathbf{B}^n \to \mathbf{B}$, where $\mathbf{B} = \{True, False\}$ and $n \in \mathbb{N}$.
By associating -1 with True and $+1$ with False, a BF can be considered as a real valued function, $f : \{-1, 1\}^n \to \{-1, 1\}$.
There are many ways to obtain functions that represent BFs.

Exact representation

It can be shown that there is a **unique** multilinear polynomial p_f that exactly represents f:

$$f(x_1, x_2, \cdots, x_n) = p_f(x_1, x_2, \cdots, x_n) = \sum_{i=1}^{2^n} s_i \prod_{k \in S_i} x_k$$

where S_i runs over all the subsets over $\{1, 2, \ldots, n\}$.

An example

x_1	x_2	f
1	1	1
1	-1	-1
-1	1	1
-1	-1	1

Table: The truth table of a BF f

An example

x_1	x_2	f
1	1	1
1	-1	-1
-1	1	1
-1	-1	1

Table: The truth table of a BF f

Exact reps. have the form $p_f(x_1, x_2) = a_1 + a_2x_2 + a_3x_1 + a_4x_2x_1$.

An example

x_1	x_2	f
1	1	1
1	-1	-1
-1	1	1
-1	-1	1

Table: The truth table of a BF f

Exact reps. have the form $p_f(x_1, x_2) = a_1 + a_2x_2 + a_3x_1 + a_4x_2x_1$.
And $\mathbf{a} = \begin{bmatrix} a_1 & a_2 & a_3 & a_4 \end{bmatrix}^T$

An example

x_1	x_2	f
1	1	1
1	-1	-1
-1	1	1
-1	-1	1

Table: The truth table of a BF f

Exact reps. have the form $p_f(x_1, x_2) = a_1 + a_2x_2 + a_3x_1 + a_4x_2x_1$.
And $\mathbf{a} = \begin{bmatrix} a_1 & a_2 & a_3 & a_4 \end{bmatrix}^T$
A short hand definition of f, $\mathbf{f} = \begin{bmatrix} 1 & -1 & 1 & 1 \end{bmatrix}^T$.

Obtaining exact representations

We can obtain the coefficient vector $\mathbf{a}$ as $\mathbf{a} = 2^{-n}\mathbf{D}_n\mathbf{f}$ where $\mathbf{D}_n$ is a Sylvester-type Hadamard Matrix.

Obtaining exact representations

We can obtain the coefficient vector $\mathbf{a}$ as $\mathbf{a} = 2^{-n}\mathbf{D}_n\mathbf{f}$ where $\mathbf{D}_n$ is a Sylvester-type Hadamard Matrix.

For $\mathbf{f} = \begin{bmatrix} 1 & -1 & 1 & 1 \end{bmatrix}^T$,

Obtaining exact representations

We can obtain the coefficient vector $\mathbf{a}$ as $\mathbf{a} = 2^{-n}\mathbf{D}_n\mathbf{f}$ where $\mathbf{D}_n$ is a Sylvester-type Hadamard Matrix.

For $\mathbf{f} = \begin{bmatrix} 1 & -1 & 1 & 1 \end{bmatrix}^T$,

$$\mathbf{a} = 2^{-2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}\begin{bmatrix} 1 \\ -1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1/2 \\ 1/2 \\ -1/2 \\ 1/2 \end{bmatrix}.$$

Obtaining exact representations

We can obtain the coefficient vector $\mathbf{a}$ as $\mathbf{a} = 2^{-n}\mathbf{D}_n\mathbf{f}$ where $\mathbf{D}_n$ is a Sylvester-type Hadamard Matrix.

For $\mathbf{f} = \begin{bmatrix} 1 & -1 & 1 & 1 \end{bmatrix}^T$,

$$\mathbf{a} = 2^{-2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1/2 \\ 1/2 \\ -1/2 \\ 1/2 \end{bmatrix}.$$

$$p_f(x_1, x_2) = 1/2 + 1/2x_2 + -1/2x_1 + 1/2x_2x_1 = f(x_1, x_2)$$

for all $x_1, x_2 \in \mathbf{B}$

Definition

We say a polynomial p_f sign-represents the Boolean function f iff

$$f(x_1, x_2, \cdots, x_n) = sign(p_f(x_1, x_2, \cdots, x_n))$$

for all $x_i \in \{-1, 1\}$.

An example

$$f(x_1, x_2) = 1/2 + 1/2x_1 - 1/2x_2 + 1/2x_1x_2$$

An example

$$f(x_1, x_2) = 1/2 + 1/2x_1 - 1/2x_2 + 1/2x_1x_2$$
$$= sign(-x_1 + x_2 + x_1x_2)$$

Threshold density

Definition. The minimum number of monomials that sign-represent f is called the *threshold density* of f and is denoted with $\pi(f)$.

Threshold density

Definition. The minimum number of monomials that sign-represent f is called the *threshold density* of f and is denoted with $\pi(f)$.

Definition. The maximum number of monomials that sign-represent any n-variable f is called the *maximum threshold density* of dimension n and is denoted with $\Pi(n)$.

Threshold density

Definition. The minimum number of monomials that sign-represent f is called the *threshold density* of f and is denoted with $\pi(f)$.

Definition. The maximum number of monomials that sign-represent any n-variable f is called the *maximum threshold density* of dimension n and is denoted with $\Pi(n)$.

Simply,

$$\Pi(n) = \max_{f \in \{B^n \to B\}} \pi(f)$$

Know results

1. $\Pi(n) \leq 2^n - \sqrt{2^n} + 1$ [Saks 1993; O'Donnell and Servedio 2003]
2. $\Pi(n) \leq 0.75 \times 2^n$ [Oztop 2006]
3. The 3-Quarters algorithm for $\Pi(n) \leq 0.75 \times 2^n$ [Oztop 2009]
4. $\Pi(n) \leq 0.617 \times 2^n$, for *almost* all BFs [Amano 2010]
5. Seems like $\Pi(n) \ll 0.5 \times 2^n$ [Sezener and Oztop In Press]

Heuristic Algorithms

- L-Heuristic [Sezener and Oztop In Press]
- B-Heuristic
- A Binary Genetic Algorithm

Motivation for the L-Heuristic

- Let **f** be a 5-variable BF

Motivation for the L-Heuristic

- Let **f** be a 5-variable BF
- We can obtain its spectrum as $\mathbf{a} = 2^{-n}\mathbf{D}_n\mathbf{f}$

Motivation for the L-Heuristic

- Let **f** be a 5-variable BF
- We can obtain its spectrum as $\mathbf{a} = 2^{-n}\mathbf{D}_n\mathbf{f}$
- We know that $f(x_1, \ldots, x_5) = a_1 + a_2x_1 + \cdots + a_{32}x_5x_4\ldots x_1$

Motivation for the L-Heuristic

- Let $\mathbf{f}$ be a 5-variable BF
- We can obtain its spectrum as $\mathbf{a} = 2^{-n}\mathbf{D}_n\mathbf{f}$
- We know that $f(x_1, \ldots, x_5) = a_1 + a_2x_1 + \cdots + a_{32}x_5x_4 \ldots x_1$
- $a_{32} := 0$ and check if $f(x_1, \ldots) = sign(a_1 + \cdots + a_{31}x_5 \ldots x_2)$

Motivation for the L-Heuristic

- Let **f** be a 5-variable BF
- We can obtain its spectrum as $\mathbf{a} = 2^{-n}\mathbf{D}_n\mathbf{f}$
- We know that $f(x_1, \ldots, x_5) = a_1 + a_2x_1 + \cdots + a_{32}x_5x_4 \ldots x_1$
- $a_{32} := 0$ and check if $f(x_1, \ldots) = sign(a_1 + \cdots + a_{31}x_5 \ldots x_2)$
- If it is OK, check for $a_{32} := 0$ and $a_{31} := 0$ and so on ...

Motivation for the L-Heuristic

- Let **f** be a 5-variable BF
- We can obtain its spectrum as $\mathbf{a} = 2^{-n}\mathbf{D}_n\mathbf{f}$
- We know that $f(x_1, \ldots, x_5) = a_1 + a_2x_1 + \cdots + a_{32}x_5x_4\ldots x_1$
- $a_{32} := 0$ and check if $f(x_1, \ldots) = sign(a_1 + \cdots + a_{31}x_5\ldots x_2)$
- If it is OK, check for $a_{32} := 0$ and $a_{31} := 0$ and so on ...
- Check all possible subsets for elimination to obtain $\pi(f)$

Motivation for the L-Heuristic

- Let **f** be a 5-variable BF
- We can obtain its spectrum as $\mathbf{a} = 2^{-n}\mathbf{D}_n\mathbf{f}$
- We know that $f(x_1, \ldots, x_5) = a_1 + a_2x_1 + \cdots + a_{32}x_5x_4 \ldots x_1$
- $a_{32} := 0$ and check if $f(x_1, \ldots) = sign(a_1 + \cdots + a_{31}x_5 \ldots x_2)$
- If it is OK, check for $a_{32} := 0$ and $a_{31} := 0$ and so on ...
- Check all possible subsets for elimination to obtain $\pi(f)$
- HOWEVER, the number of subsets $= 2^{32}$

Motivation for the L-Heuristic

- Let **f** be a 5-variable BF
- We can obtain its spectrum as $\mathbf{a} = 2^{-n}\mathbf{D}_n\mathbf{f}$

Motivation for the L-Heuristic

- Let $\mathbf{f}$ be a 5-variable BF
- We can obtain its spectrum as $\mathbf{a} = 2^{-n}\mathbf{D}_n\mathbf{f}$
- Let's check $|\mathbf{a}|$, 1×18, 1×14, 2×10, 6×6, 22×2.

Motivation for the L-Heuristic

- Let **f** be a 5-variable BF
- We can obtain its spectrum as $\mathbf{a} = 2^{-n}\mathbf{D}_n\mathbf{f}$
- Let's check $|\mathbf{a}|$, 1×18, 1×14, 2×10, 6×6, 22×2.
- Eliminating a_i is easier than a_j iff $|a_i| < |a_j|$ [Sezener and Oztop In Press]

Motivation for the L-Heuristic

- Let **f** be a 5-variable BF
- We can obtain its spectrum as $\mathbf{a} = 2^{-n}\mathbf{D}_n\mathbf{f}$
- Let's check $|\mathbf{a}|$, 1×18, 1×14, 2×10, 6×6, 22×2.
- Eliminating a_i is easier than a_j iff $|a_i| < |a_j|$ [Sezener and Oztop In Press]
- **Heuristic:** First eliminate 2's, then 4's, then 6's and so on.

Pseudocode for the L-Heuristic

Algorithm 1: L-Heuristic

```
m=1, E={};
Sort the monomials using their coefficients (|a|) as the sorting key;
while m < 2^n do
    if monomials from 1 to m that are not included in E can be eliminated then
        add m to E;
    m = m + 1;
return m
```

Advantage: Checks 2^n many subsets rather than 2^{2^n} many.

Visualization of the L-Heuristic

1. Let $\mathbf{a} = \begin{bmatrix} 4 & 12 & -6 & 2 & -4 & 2 & -8 & 4 \end{bmatrix}^T$

Visualization of the L-Heuristic

1. Let $\mathbf{a} = \begin{bmatrix} 4 & 12 & -6 & 2 & -4 & 2 & -8 & 4 \end{bmatrix}^T$
2. $sort(|\mathbf{a}|) = \begin{bmatrix} 2 & 2 & 4 & 4 & 4 & 6 & 8 & 12 \end{bmatrix}^T$

Visualization of the L-Heuristic

1. Let $\mathbf{a} = \begin{bmatrix} 4 & 12 & -6 & 2 & -4 & 2 & -8 & 4 \end{bmatrix}^T$
2. $sort(|\mathbf{a}|) = \begin{bmatrix} 2 & 2 & 4 & 4 & 4 & 6 & 8 & 12 \end{bmatrix}^T$
3. Attempt to eliminate $\begin{bmatrix} 2 \end{bmatrix}^T$

Visualization of the L-Heuristic

1. Let $\mathbf{a} = \begin{bmatrix} 4 & 12 & -6 & 2 & -4 & 2 & -8 & 4 \end{bmatrix}^T$
2. $sort(|\mathbf{a}|) = \begin{bmatrix} 2 & 2 & 4 & 4 & 4 & 6 & 8 & 12 \end{bmatrix}^T$
3. Attempt to eliminate $\begin{bmatrix} 2 \end{bmatrix}^T$
4. If successful, attempt $\begin{bmatrix} 2 & 2 \end{bmatrix}^T$

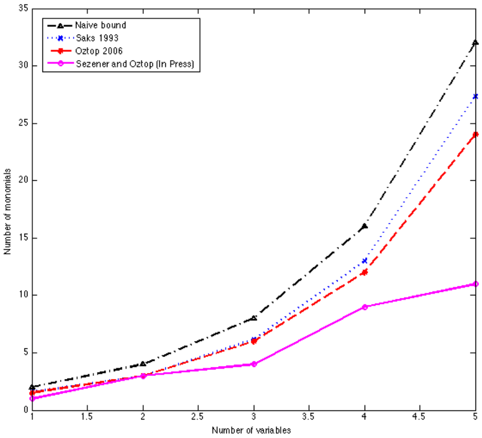
Visualization of the L-Heuristic

1. Let $\mathbf{a} = \begin{bmatrix} 4 & 12 & -6 & 2 & -4 & 2 & -8 & 4 \end{bmatrix}^T$
2. $sort(|\mathbf{a}|) = \begin{bmatrix} 2 & 2 & 4 & 4 & 4 & 6 & 8 & 12 \end{bmatrix}^T$
3. Attempt to eliminate $\begin{bmatrix} 2 \end{bmatrix}^T$
4. If successful, attempt $\begin{bmatrix} 2 & 2 \end{bmatrix}^T$
5. If not successful, attempt $\begin{bmatrix} 2 & 4 \end{bmatrix}^T$

Visualization of the L-Heuristic

1. Let $\mathbf{a} = \begin{bmatrix} 4 & 12 & -6 & 2 & -4 & 2 & -8 & 4 \end{bmatrix}^T$
2. $sort(|\mathbf{a}|) = \begin{bmatrix} 2 & 2 & 4 & 4 & 4 & 6 & 8 & 12 \end{bmatrix}^T$
3. Attempt to eliminate $\begin{bmatrix} 2 \end{bmatrix}^T$
4. If successful, attempt $\begin{bmatrix} 2 & 2 \end{bmatrix}^T$
5. If not successful, attempt $\begin{bmatrix} 2 & 4 \end{bmatrix}^T$
6. If successful, attempt $\begin{bmatrix} 2 & 4 & 4 \end{bmatrix}^T$

Results for $n \leq 5$



Pseudocode for the B-Heuristic

Algorithm 2: B-Heuristic

lo $= 1$, hi $= 2^n - 1$;
while $lo \leq hi$ **do**
    $m = floor((hi + lo)/2)$;
    if *first m monomials can be eliminated* **then**
        $lo = m + 1$;
    else
        $hi = m - 1$;
return m

Advantage: Checks $\log(2^n) = n$ many subsets rather than 2^n many.

Visualization of the L-Heuristic

1. Let $\mathbf{a} = \begin{bmatrix} 4 & 12 & -6 & 2 & -4 & 2 & -8 & 4 \end{bmatrix}^T$

Visualization of the L-Heuristic

1. Let $\mathbf{a} = \begin{bmatrix} 4 & 12 & -6 & 2 & -4 & 2 & -8 & 4 \end{bmatrix}^T$
2. $sort(|\mathbf{a}|) = \begin{bmatrix} 2 & 2 & 4 & 4 & 4 & 6 & 8 & 12 \end{bmatrix}^T$

Visualization of the L-Heuristic

1. Let $\mathbf{a} = \begin{bmatrix} 4 & 12 & -6 & 2 & -4 & 2 & -8 & 4 \end{bmatrix}^T$
2. $sort(|\mathbf{a}|) = \begin{bmatrix} 2 & 2 & 4 & 4 & 4 & 6 & 8 & 12 \end{bmatrix}^T$
3. Attempt to eliminate $\begin{bmatrix} 2 & 2 & 4 & 4 \end{bmatrix}^T$

Visualization of the L-Heuristic

1. Let $\mathbf{a} = \begin{bmatrix} 4 & 12 & -6 & 2 & -4 & 2 & -8 & 4 \end{bmatrix}^T$
2. $sort(|\mathbf{a}|) = \begin{bmatrix} 2 & 2 & 4 & 4 & 4 & 6 & 8 & 12 \end{bmatrix}^T$
3. Attempt to eliminate $\begin{bmatrix} 2 & 2 & 4 & 4 \end{bmatrix}^T$
4. If successful, attempt $\begin{bmatrix} 2 & 2 & 4 & 4 & 4 & 6 \end{bmatrix}^T$

Visualization of the L-Heuristic

1. Let $\mathbf{a} = \begin{bmatrix} 4 & 12 & -6 & 2 & -4 & 2 & -8 & 4 \end{bmatrix}^T$
2. $sort(|\mathbf{a}|) = \begin{bmatrix} 2 & 2 & 4 & 4 & 4 & 6 & 8 & 12 \end{bmatrix}^T$
3. Attempt to eliminate $\begin{bmatrix} 2 & 2 & 4 & 4 \end{bmatrix}^T$
4. If successful, attempt $\begin{bmatrix} 2 & 2 & 4 & 4 & 4 & 6 \end{bmatrix}^T$
5. If not successful, attempt $\begin{bmatrix} 2 & 2 & 4 & 4 & 4 \end{bmatrix}^T$

A Binary Genetic Algorithm

We used a binary genetic algorithm [Haupt and Haupt 2004].
Genotype **b** is a binary vector which determines which monomials to eliminate.

$$fitness = \begin{cases} sum(\mathbf{b}) \ if \ elimination \ is \ successful \\ 0 \ else \end{cases}$$

E.g., the fitness of $\mathbf{a} = \begin{bmatrix} 1/2 & -1/2 & 1/2 & 1/2 \end{bmatrix}$ is 0 and the fitness of $\mathbf{a}^* = \begin{bmatrix} 1 & 0 & 2 & 0 \end{bmatrix}$ is 2.

Average number of monomials

Algorithm	Avg. # monomials	Avg. computation time (s)
3-Quarters	8.2720	0.0007
L-Heuristic	4.9678	0.0654
B-Heuristic	5.8115	0.0199
GA	7.9941	4.2678

Table: Result comparison for 4 variable BFs

4-variable BFs

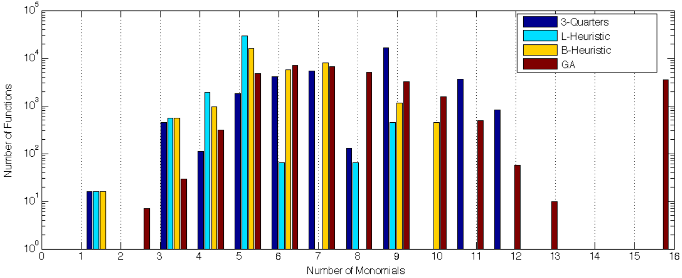


Figure: Distribution of the number of monomials to represent 4-variable BFs (log-lin scale)

References I

Amano, Kazuyuki (2010). "New Upper Bounds on the Average PTF Density of Boolean Functions". In: *Algorithms and Computation*. Ed. by Otfried Cheong, Kyung-Yong Chwa, and Kunsoo Park. Vol. 6506. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 304–315. ISBN: 978-3-642-17516-9. DOI: 10.1007/978-3-642-17517-6_28.

Haupt, Randy L. and Sue Ellen Haupt (2004). "The Binary Genetic Algorithm". In: *Practical Genetic Algorithms*. John Wiley & Sons, Inc., pp. 27–50. ISBN: 9780471671749. DOI: 10.1002/0471671746.ch2. URL: http://dx.doi.org/10.1002/0471671746.ch2.

References II

- O'Donnell, R. and R. Servedio (2003). "Extremal properties of polynomial threshold functions". In: *Eighteenth Annual Conference on Computational Complexity*, pp. 3–12.
- Oztop, Erhan (2006). "An Upper Bound on the Minimum Number of Monomials Required to Separate Dichotomies of $\{-1, 1\}^n$". In: *Neural Computation* 18.12, pp. 3119–3138.
- – (2009). "Sign-representation of Boolean Functions Using a Small Number of Monomials". In: *Neural Networks* 22.7, pp. 938–948. ISSN: 0893-6080. DOI: 10.1016/j.neunet.2009.03.016. URL: http://dx.doi.org/10.1016/j.neunet.2009.03.016.

References III

- Saks, M.E. (1993). "Slicing the Hypercube". In: *London Mathematical Society Lecture Note Series 187: Surveys in Combinatorics*. Ed. by K. Walker. Cambridge University Press, pp. 211–255.
- Sezener, Can Eren and Erhan Oztop (In Press). "Minimal sign representation of Boolean functions: algorithms and exact results for low dimensions". In: *Neural Computation*.

Questions?