Introduction to the Complexity of Boolean Functions

Jiyou Li
lijiyou@sjtu.edu.cn

Shanghai Jiao Tong University

Oct. 13rd, 2013
Outline

1. Introduction to Boolean Functions
2. Complexity Aspects of Boolean Functions
3. Our Recent Work
A Boolean function is a map from \( \{0, 1\}^n \) to \( \{0, 1\} \).
A Boolean function is a map from \( \{0, 1\}^n \) to \( \{0, 1\} \). We denote \( \{0, 1\} \) by \( \mathbb{F}_2 \).
A Boolean function is

- A circuit;
A Boolean function is

- A circuit;
- A decision problem;
A Boolean function is

- A circuit;
- A decision problem;
- A subset of the hypercube of dimension n;
A Boolean function is

- A circuit;
- A decision problem;
- A subset of the hypercube of dimension n;
- A binary code;
A Boolean function is

- A circuit;
- A decision problem;
- A subset of the hypercube of dimension $n$;
- A binary code;
- A family of subsets in $[n]$;
A Boolean function is

- A circuit;
- A decision problem;
- A subset of the hypercube of dimension $n$;
- A binary code;
- A family of subsets in $[n]$;
- A graph property;
A Boolean function is

- A circuit;
- A decision problem;
- A subset of the hypercube of dimension $n$;
- A binary code;
- A family of subsets in $[n]$;
- A graph property;
- A set of integers;
A Boolean function is

- A circuit;
- A decision problem;
- A subset of the hypercube of dimension $n$;
- A binary code;
- A family of subsets in $[n]$;
- A graph property;
- A set of integers;
- A voting scheme;
A Boolean function is

- A circuit;
- A decision problem;
- A subset of the hypercube of dimension $n$;
- A binary code;
- A family of subsets in $[n]$;
- A graph property;
- A set of integers;
- A voting scheme;
- …
# Representation 1—Truth Table

<table>
<thead>
<tr>
<th>$x$</th>
<th>$f(x)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>0</td>
</tr>
<tr>
<td>001</td>
<td>0</td>
</tr>
<tr>
<td>010</td>
<td>1</td>
</tr>
<tr>
<td>011</td>
<td>1</td>
</tr>
<tr>
<td>111</td>
<td>1</td>
</tr>
<tr>
<td>110</td>
<td>0</td>
</tr>
<tr>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>101</td>
<td>1</td>
</tr>
</tbody>
</table>
Representation 2–Conjunctive Normal Form (CNF)

$$f(x_1, x_2, \ldots, x_n) = \bigwedge_{j=1}^{k} C_j,$$

where each clause $C_j = \bigvee_{k=1}^{s} y_{j,k}$, $y_{j,k} = x_i$ or $y_{j,k} = \overline{x_i}$ are the literals.
A typical formula of 2-CNF

\[ f(x_1, x_2, x_3) = (x_1 \lor x_2) \land (\overline{x_1} \lor x_3). \]
Representation 2–Disjunctive Normal Form (DNF)

\[ f(x_1, x_2, x_3) = (x_1 \land x_3) \lor (\overline{x_1} \land x_2) \lor (x_2 \land x_3). \]
In $F_2$, $x \land y = x \cdot y$, $\overline{x} = 1 + x$ and $x \lor y = x \cdot y + x + y$, one has

$$f(x_1, x_2, \ldots, x_n) = \sum_{l \subseteq [n]} a_l x^l,$$

where $a_l \in F_2$ and for $l = \{i_1, i_2, \ldots, i_k\}$, $x^l$ denotes $x_{i_1} x_{i_2} \cdots x_{i_k}$. 
In $\mathbb{F}_2$, $x \land y = x \cdot y$, $\overline{x} = 1 + x$ and $x \lor y = x \cdot y + x + y$, one has

$$f(x_1, x_2, \ldots, x_n) = \sum_{l \subseteq [n]} a_l x^l,$$

where $a_l \in \mathbb{F}_2$ and for $l = \{i_1, i_2, \ldots, i_k\}$, $x^l$ denotes $x_{i_1} x_{i_2} \cdots x_{i_k}$. The algebraic degree is defined as the maximal size of $l$ such that $a_l \neq 0$. 
In $\mathbb{F}_2$, $x \land y = x \cdot y$, $\overline{x} = 1 + x$ and $x \lor y = x \cdot y + x + y$, one has

$$f(x_1, x_2, \ldots, x_n) = \sum_{I \subseteq [n]} a_I x^I,$$

where $a_I \in \mathbb{F}_2$ and for $I = \{i_1, i_2, \ldots, i_k\}$, $x^I$ denotes $x_{i_1} x_{i_2} \cdots x_{i_k}$. The algebraic degree is defined as the maximal size of $I$ such that $a_I \neq 0$.

$$f(x) = x_1 x_2 + x_1 x_3 + x_3.$$
Representation 4–Set system

\[ f = \{\{2\}, \{2, 3\}, \{1, 3\}, \{1, 2, 3\}\}. \]
Representation 5–2-coloring of the n-Hypercube
Problems in Boolean functions

- How to compute a Boolean function efficiently?
Problems in Boolean functions

- How to compute a Boolean function efficiently?
- What is a good function?
Problems in Boolean functions

- How to compute a Boolean function efficiently?
- What is a good function?
- What is a complicated function?
Problems in Boolean functions

- How to compute a Boolean function efficiently?
- What is a good function?
- What is a complicated function?
- What is a fair function?
Problems in Boolean functions

- How to compute a Boolean function **efficiently**?
- What is a **good** function?
- What is a **complicated** function?
- What is a **fair** function?
- What is a **pseudorandom** function?
Problems in Boolean functions

- How to compute a Boolean function efficiently?
- What is a good function?
- What is a complicated function?
- What is a fair function?
- What is a pseudorandom function?
- What is the number of Boolean functions satisfying some properties?
Stream Ciphers

Block Cipher

Stream Cipher

Keystream Generator
S-box
Examples of special Boolean functions

- Constant;
Examples of special Boolean functions

- Constant;
- Dictatorship;
Examples of special Boolean functions

- Constant;
- Dictatorship;
- k-junta;

\[ \chi_S(x) = \sum_{i \in S} x_i, \quad S \subseteq \{1, \ldots, n\} \]
Examples of special Boolean functions

- Constant;
- Dictatorship;
- k-junta;
- Majority;
Examples of special Boolean functions

- Constant;
- Dictatorship;
- k-junta;
- Majority;
- Parity function $\chi_S(x) = \sum_{i \in S} x_i$, $S \subseteq [n]$;
Examples of special Boolean functions

- Constant;
- Dictatorship;
- k-junta;
- Majority;
- Parity function $\chi_S(x) = \sum_{i \in S} x_i, \ S \subseteq [n]$;
Examples of special Boolean functions

- Constant;
- Dictatorship;
- k-junta;
- Majority;
- Parity function $\chi_S(x) = \sum_{i \in S} x_i$, $S \subseteq [n]$;
- ...
Definition (monotone function)

A Boolean function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ is said to be monotone if $f(x) \leq f(y)$ whenever $\forall i \in [n], x_i \leq y_i$, where $x, y \in \{0, 1\}^n$. 
A Boolean function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ is said to be symmetric if for any $\tau \in S_n$,

$$f(x_1, x_2, \ldots, x_n) = f(\tau(x)) = f(x_{\tau(1)}, x_{\tau(2)}, \ldots, x_{\tau(n)}).$$
Definition (transitive function)

A Boolean function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ is said to be transitive if for any $1 \leq i, j \leq n$, there is $\tau \in Aut(f)$ such that $i = \tau(j)$, where $Aut(f) \subseteq S_n$ is defined by

$$Aut(f) = \{ \tau \in S_n \mid f(x_{\tau(1)}, x_{\tau(2)}, \ldots, x_{\tau(n)}) = f(x_1, x_2, \ldots, x_n) \}.$$
Parameters of Boolean functions

- Weight
Parameters of Boolean functions

- Weight
- Algebraic degree
Parameters of Boolean functions

- Weight
- Algebraic degree
- Nonlinearity
Parameters of Boolean functions

- Weight
- Algebraic degree
- Nonlinearity
- Query complexity
Parameters of Boolean functions

- Weight
- Algebraic degree
- Nonlinearity
- Query complexity
- Certificate complexity
- Circuit complexity
Parameters of Boolean functions

- Weight
- Algebraic degree
- Nonlinearity
- Query complexity
- Certificate complexity
- Circuit complexity
- Influence and total influence (energy)
Parameters of Boolean functions

- Weight
- Algebraic degree
- Nonlinearity
- Query complexity
- Certificate complexity
- Circuit complexity
- Influence and total influence (energy)
- Sensitivity and block sensitivity
Parameters of Boolean functions

- Weight
- Algebraic degree
- Nonlinearity
- Query complexity
- Certificate complexity
- Circuit complexity
- Influence and total influence (energy)
- Sensitivity and block sensitivity
- Algebraic symmetries
Parameters of Boolean functions

- Weight
- Algebraic degree
- Nonlinearity
- Query complexity
- Certificate complexity
- Circuit complexity
- Influence and total influence (energy)
- Sensitivity and block sensitivity
- Algebraic symmetries
- Combinatorial structures
- ...
P and NP

**P**: There is a deterministic Turing Machine M and a polynomial \( p(n) \) such that for each input \( w \), the running time \( \leq p(|w|) \);

**NP**: There is a nondeterministic Turing Machine M and a polynomial \( p(n) \) such that for each input \( w \) the running time \( \leq p(|w|) \).
Boolean satisfiability problem (SAT)

Problem

*Given a **CNF** formula \( f \), is there an assignment to make the formula evaluate to **TRUE**?*
Theorem (Cook and Levin, 1970s)

3-SAT is \textbf{NP}-complete.
A deterministic decision tree $D_f$ for $f(x)$ takes $x = (x_1, \ldots, x_n)$ as an input and determines the value of $f(x_1, \ldots, x_n)$ using queries of the form

"is $x_i = 1$?"

Let $C(D_f, x)$ denote the maximal number of queries made by $D_f$ on input $x$. The deterministic decision tree complexity of $f$ is defined as

$$D(f) = \min_{D_f} \max_{x \in \mathbb{F}_2^n} C(D_f, x).$$
Definition

f is called evasive if $D(f) = n$. 
A decision tree
Definition

A graph property is a (transitive) Boolean function on $\binom{n}{2}$ edges as variables.
A Boolean function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ is said to be **monotone** if $f(x) \leq f(y)$ whenever $\forall i \in [n] \ x_i \leq y_i$ where $x, y \in \{0, 1\}^n$. 

**Definition (monotone function)**
Conjecture (Aanderaa Rosenberg Conjecture, proved by Rivest and Vuillemin in 1976)

There exists $\delta > 0$ such that for all nontrivial, monotone graph properties $f$,

$$D(f) \geq \delta n^2.$$
Conjecture (Karp Conjecture)

For all nontrivial, monotone graph properties $f$ one has

$$D(f) = \binom{n}{2}.$$
Theorem (Rivest and Vuillemin, 1976)

All nontrivial, monotone transitive Boolean functions of prime power variables (so are graph properties) are evasive.
**Theorem (Rivest and Vuillemin, 1976)**

All nontrivial, monotone transitive Boolean functions of prime power variables (so are graph properties) are evasive.

**Theorem (Rivest and Vuillemin, 1976)**

A random Boolean function is evasive.
Theorem (Kahn, Saks and Sturtevant, Combinatoria, 1984)

For all nontrivial, monotone graph properties $f$, 

$$D(f) \geq \frac{1}{4} n^2 - o(n^2).$$
Theorem (Kahn, Saks and Sturtevant, Combinatorica, 1984)

For all nontrivial, monotone graph properties \( f \),

\[
D(f) \geq \frac{1}{4} n^2 - o(n^2).
\]

Theorem (Kahn, Saks and Sturtevant, Combinatorica, 1984)

All nontrivial, monotone graph properties of prime power number vertices are evasive.
Theorem (A. Yao, FOCS, 1987)

Every nontrivial, monotone bipartite graph property is evasive.
Theorem (Korneffel and Triesch, 2010)

For all nontrivial, monotone graph properties $f$,

$$D(f) \geq \frac{8}{25} n^2 - o(n^2).$$
Theorem (Scheidweiler and Triesch, 2013)

For all nontrivial, monotone graph properties $f$,  

$$D(f) \geq \frac{1}{3} n^2 - o(n^2).$$
Certificate Complexity

**Definition**

Let $f(x)$ be a Boolean function of $n$ variables. If $f(x) = 0$, then a 0-certificate for $x$ is a sequence of bits in $x$ that proves $f(x) = 0$. The definition for 1-certificate is similar.

We define the certificate complexity $Certi(f)$ of $f(x)$ as

$$Certi(f) = \max_{x} \{\text{number of bits in the smallest 0/1- certificate for } x\}$$
Certificate Complexity
Theorem

For every Boolean function $f$, 

$$Certi(f) \leq D(f) \leq Certi(f)^2.$$
A circuit
Circuit Complexity

Definition

A circuit is a directed acyclic graph. The inputs are the sources of the graph and the remaining vertices are the gates $G(i)$ which are labeled by the type $\omega_i \in \{\lor, \land, \neg\}$.

$G_i = G_i(G_1, G_2, \ldots, G_{i-1})$. $f$ is said to be computed by this circuit if there is some $j$ such that $f = G_j$. 
Circuit Complexity

**Definition**

A **circuit** is a directed acyclic graph. The inputs are the sources of the graph and the remaining vertices are the gates $G(i)$ which are labeled by the type $\omega_i \in \{\lor, \land, \neg\}$. $G_i = G_i(G_1, G_2, \ldots, G_{i-1})$. $f$ is said to be computed by this circuit if there is some $j$ such that $f = G_j$.

Circuit depth $D(f)$: the depth from the sources to the gate
Circuit size $C(f)$: the number of the gates.
A **circuit** is a directed acyclic graph. The inputs are the sources of the graph and the remaining vertices are the gates $G(i)$ which are labeled by the type $\omega_i \in \{\lor, \land, \neg\}$. 

$G_i = G_i(G_1, G_2, \ldots, G_{i-1})$. $f$ is said to be computed by this circuit if there is some $j$ such that $f = G_j$.

**Circuit depth** $D(f)$: the depth from the sources to the gate

**Circuit size** $C(f)$: the number of the gates.

**P/poly**: A circuit with polynomial size.
Definition

A circuit is a directed acyclic graph. The inputs are the sources of the graph and the remaining vertices are the gates \( G(i) \) which are labeled by the type \( \omega_i \in \{ \lor, \land, \neg \} \).

\[ G_i = G_i(G_1, G_2, \ldots, G_{i-1}) \]

\( f \) is said to be computed by this circuit if there is some \( j \) such that \( f = G_j \).

Circuit depth \( D(f) \): the depth from the sources to the gate
Circuit size \( C(f) \): the number of the gates.
\( \textbf{P/poly} \): A circuit with polynomial size.
Examples.
Some upper bounds

n-binary addition can be computed by a circuit of size $5n - 3$ and depth $2n - 1$;
n-binary **addition** can be computed by a circuit of size $5n - 3$ and depth $2n - 1$;

n-binary **multiplication** can be computed by a circuit of size $O(n^2)$ and depth $O(\log n)$, or a circuit of size $O(n^{\log 3})$ and depth $O(\log n)$, of size $O(n \log n \log \log n)$ and depth $O(\log n)$;
Some upper bounds

n-binary **addition** can be computed by a circuit of size $5n - 3$ and depth $2n - 1$; 
n-binary **multiplication** can be computed by a circuit of size $O(n^2)$ and depth $O(\log n)$, or a circuit of size $O(n^{\log^3})$ and depth $O(\log n)$, of size $O(n \log n \log \log n)$ and depth $O(\log n)$; 
n-binary **division** can be computed by a circuit of size $O(n^2)$ and depth $O(n \log n)$, or a circuit of size $O(n^2 \log n)$ and depth $O(\log 2n)$;
Some upper bounds

- n-binary addition can be computed by a circuit of size $5n - 3$ and depth $2n - 1$;
- n-binary multiplication can be computed by a circuit of size $O(n^2)$ and depth $O(\log n)$, or a circuit of size $O(n^{\log 3})$ and depth $O(\log n)$, of size $O(n \log n \log \log n)$ and depth $O(\log n)$;
- n-binary division can be computed by a circuit of size $O(n^2)$ and depth $O(n \log n)$, or a circuit of size $O(n^2 \log n)$ and depth $O(\log 2n)$;
- $det_n$ can be computed by a circuit of size $3n^3 + n^2 - 4n$ of depth $O(\log n)$. 
Theorem

The function $f$ computed by a Turing machine in $T$ steps can also be computed by a circuit whose size and depth satisfy the following bounds

$$C(f) = O(T^2),$$
$$D(f) = O(T \log T).$$
Theorem

The function $f$ computed by a Turing machine in $T$ steps can also be computed by a circuit whose size and depth satisfy the following bounds

$$C(f) = O(T^2),$$
$$D(f) = O(T \log T).$$

Since $P \subseteq P/poly$, by the theorem of Karp-Lipton,

$P \neq NP \iff \text{NP} \not\subseteq P/poly$!
Theorem (Shannon)

For $n \geq 100$, almost all Boolean functions on $n$ variables require circuits of size at least $\frac{2^n}{10n}$. 
For an input $x = (x_1, x_2, \ldots, x_n)$, the sensitivity on $x$ is defined by

$$s(f, x) = \sum_{i=1}^{n} \left| f(x) - f(x \oplus i) \right|,$$

where $x \oplus i$ is equal to $x$ but with the $i^{th}$ bit flipped.
For an input \( x = (x_1, x_2, \ldots, x_n) \), the sensitivity on \( x \) is defined by

\[
s(f, x) = \sum_{i=1}^{n} \left| f(x) - f(x \oplus i) \right|,
\]

where \( x \oplus i \) is equal to \( x \) but with the \( i^{th} \) bit flipped. The sensitivity is defined as

\[
s(f) = \max_{x \in \mathbb{F}_2^n} s(f, x).
\]
For an input $x = (x_1, x_2, \ldots, x_n)$, the block sensitivity on $x$ is defined by

$$bs(f, x) = \max_{\text{partitions } S_1, \ldots, S_k \text{ of } [n]} \sum_{i=1}^{k} \left| f(x) - f(x \oplus S_i) \right|,$$

where $x \oplus S_i$ is equal to $x$ but with all the bits in $S_i$ flipped.
For an input $x = (x_1, x_2, \ldots, x_n)$, the block sensitivity on $x$ is defined by

$$bs(f, x) = \max_{\text{partitions } S_1, \ldots, S_k \text{ of } [n]} \sum_{i=1}^{k} \left| f(x) - f(x \oplus S_i) \right|,$$

where $x \oplus S_i$ is equal to $x$ but with all the bits in $S_i$ flipped. The block sensitivity is defined as

$$bs(f) = \max_{x \in \mathbb{F}_2^n} bs(f, x).$$
Theorem (Simon, 1983)

If $f(x)$ depends on all $n$ variables, then

$$s(f) \geq \frac{1}{2} \log n - \frac{1}{2} \log \log n + \frac{1}{2}.$$
Sensitivity and Block Sensitivity

Lemma

\[ s(f) \leq bs(f) \leq D(f). \]
Sensitivity and Block Sensitivity

**Theorem (Nisan, 1989)**

\[ D(f) \leq bs(f)^4. \]
Conjecture (Nisan and Szegedy: Sensitivity Conjecture)

For every Boolean function $f$,

$$D(f) \leq \text{poly}(s(f)).$$
Conjecture (Nisan and Szegedy: Sensitivity Conjecture)

For every Boolean function $f$,

$$D(f) \leq \text{poly}(s(f)).$$

Equivalently,

$$bs(f) \leq \text{poly}(s(f)).$$
The influence of the $i^{th}$ variable is defined as

$$I_i(f) = \Pr_x[f(x) \neq f(x^{\oplus i})]$$

where $x^{\oplus i}$ is equal to $x$ but with the $i^{th}$ bit flipped.
**Definition**

The **influence** of the $i^{th}$ variable is defined as

$$l_i(f) = \Pr_x[f(x) \neq f(x \oplus i)],$$

where $x \oplus i$ is equal to $x$ but with the $i^{th}$ bit flipped. We then define the **total influence** of $f$, also called the **average sensitivity** of $f$ as:

$$l(f) = \sum_{i=1}^{n} l_i(f).$$
Theorem (J. Kahn, G. Kalai, and N. Linial, FOCS, 1988)

If $f$ is a balanced Boolean function, then $\exists i$ such that

$$I_i(f) = \Omega\left(\frac{\log n}{n}\right).$$
Influence and Total Influence (Energy)

Fourier Analysis over Reals

Definition

Let $f(x)$ be a boolean function from $\{0, 1\}^n$ to $\{-1, 1\}$. The Fourier coefficient of $f(x)$ at $u$ is defined by

$$\hat{f}(u) = \frac{1}{2^n} \sum_{x \in \{0,1\}^n} (-1)^{u \cdot x} f(x).$$

$$f(x) = \sum_{u \in \{0,1\}^n} \hat{f}(u)(-1)^{u \cdot x}.$$
Influence and Total Influence (Energy)

\[ l_i(f) = \sum_{u \in \{0,1\}^n, u_i = 1} \hat{f}(u)^2. \]

\[ l(f) = \sum_{u \in \{0,1\}^n} \text{wt}(u) \hat{f}(u)^2. \]
Hypercontractivity


∀1 ≤ p ≤ q ≤ ∞, ∀|α| ≤ \( \sqrt{\frac{p-1}{q-1}} \), then

\[ \| T_{\alpha} f \|_q \leq \| f \|_p, \]

where \( T_{\alpha} \) is a noise operator.
Theorem

Let $f$ be a function that has a depth $d$ circuit of size $s$. Then $I(f) \leq O(\log^{d-1}(s))$. 
A Boolean function defined by weighted sum

For $x = (x_1, x_2, \ldots, x_n) \in \{0, 1\}^n$, denote

$$s(x) = \sum_{k=1}^{n} kx_k(\mod p), 1 \leq s(x) \leq p.$$  

Define

$$f(x) = \begin{cases} 
  x_{s(x)}, & 1 \leq s(x) \leq n; \\
  x_1, & \text{otherwise.}
\end{cases}$$
Conjecture (Shparlinski, IPL, 2007)

\[
\max_{x \in \{0,1\}^n} |\hat{f}(u)| \leq 2^{(-0.5 + o(1))n}.
\]

\[
l(f) \geq (0.5 + o(1))n.
\]
**Theorem (Shparlinski, IPL, 2007)**

\[
\max_{u \in \{0, 1\}^n} |\hat{f}(u)| \leq 2^{(-0.1587 + o(1))n}.
\]

\[
I \geq (0.0575 + o(1)) n.
\]

\[
I = \sum_{u \in \{0, 1\}^n} wt(u)\hat{f}(u)^2.
\]
Theorem (J. Li, IPL, 2012)

\[ I(f) = (0.5 + o(1)) n. \]

Proof.
Subsets counting.
Theorem (J. Li and Q. Xiang, 2013)

\[ |\hat{f}(u)| \leq 2^{(-0.5+o(1))n} \]

holds for almost all \( u \in \{0, 1\}^n \).
A parity counting problem

**Problem**

For a set $T \subseteq [p]$, evaluate

$$bias(T) = N_e(D, b) - N_o(D, b),$$

where

$$N_e(D, b) = \#\{D \subseteq \mathbb{F}_p \mid \sum_{x \in D} x = b, |D \cap T| \equiv 0 \mod 2\}$$

and

$$N_o(D, b) = \#\{D \subseteq \mathbb{F}_p \mid \sum_{x \in D} x = b, |D \cap T| \equiv 1 \mod 2\}.$$
Any Question?
Any Question?

Thank you for your attention!