HAL open science

Sensitivity, Block Sensitivity, and Certificate Complexity of Unate Functions and Read-Once Functions

Hiroki Morizumi

▶ To cite this version:

Hiroki Morizumi. Sensitivity, Block Sensitivity, and Certificate Complexity of Unate Functions and Read-Once Functions. 8th IFIP International Conference on Theoretical Computer Science (TCS), Sep 2014, Rome, Italy. pp.104-110, 10.1007/978-3-662-44602-7_9 . hal-01402033

HAL Id: hal-01402033

https://inria.hal.science/hal-01402033v1

Submitted on 24 Nov 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

Sensitivity, Block Sensitivity, and Certificate Complexity of Unate Functions and Read-Once Functions

Hiroki Morizumi

Interdisciplinary Graduate School of Science and Engineering, Shimane University, Shimane 690-8504, Japan
morizumi@cis.shimane-u.ac.jp

Abstract. Sensitivity, block sensitivity, and certificate complexity are complexity measures for Boolean functions. In this paper, we prove that these three complexity measures are equal to each other if a Boolean function is a unate function or a read-once function. We also prove $\sqrt{n}$ tight lower bounds for the three complexity measures of read-once functions. As an application of our results, the decision tree complexity of unate functions and read-once functions is upper bounded by the square of the sensitivity of the function.

1 Introduction

Sensitivity, block sensitivity, and certificate complexity of a Boolean function f, denoted by $s(f)$, $bs(f)$ and $C(f)$, respectively, are complexity measures for Boolean functions, and related to other complexity measures including the time complexity of CREW PRAMs and decision tree complexity. A long-standing open problem for these measures is whether or not block sensitivity can be polynomially upper bounded by sensitivity:

$$bs(f) \leq \mathrm{poly}(s(f))?$$

Although many efforts have been devoted to the open problem as we see later, it is still open. On the other hand, if a function f is a monotone function, it is known that $s(f) = bs(f) = C(f)$ [8]. Our main motivation of this paper is to seek other Boolean function classes such that $s(f) = bs(f) = C(f)$.

In this paper, we prove that $s(f) = bs(f) = C(f)$ for unate functions, which are generalized functions of monotone functions, and for read-once functions over the Boolean operators $\wedge$, $\vee$ and $\oplus$. We also prove that $\sqrt{n} \leq s(f)$ $(= bs(f) = C(f))$ for read-once functions which have n input variables, and the lower bound is tight.

Related works.

Rubinstein [9] exhibited a Boolean function f which has $bs(f) = \frac{1}{2}s(f)^2$. The result has been improved [10, 2], although the best known gap is still quadratic. Kenyon and Kutin [7] have proved that $bs(f) \leq \frac{e}{\sqrt{2\pi}}e^{s(f)}\sqrt{s(f)}$. The upper bound has been improved to $bs(f) \leq 2^{s(f)-1}s(f)$ by Ambainis et al. [1]. Survey papers [4, 5] include more background for this topic. On the average version of the sensitivity, Impagliazzo and Kabanets [6] have given the tight bound on the average sensitivity of read-once de Morgan formulas.

2 Preliminaries

2.1 Sensitivity, block sensitivity, and certificate complexity

Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function. For an input $x = (x_1, x_2, \ldots, x_n)$ of f and $S \subseteq [n] = \{1, 2, \ldots, n\}$, let x^S denotes the input obtained from x by flipping all the bits x_i such that $i \in S$. We abbreviate $x^{\{i\}}$ to x^i. Sensitivity, block sensitivity, and certificate complexity are defined as follows, respectively.

Definition 1. *The* sensitivity *of f on x, denoted by $s(f, x)$, is the number of indices i such that $f(x) \neq f(x^i)$. The* sensitivity *of f, denoted by $s(f)$, is $\max_x s(f, x)$. For $z \in \{0, 1\}$, the z-*sensitivity *of f, denoted by $s_z(f)$, is $\max_{x \in f^{-1}(z)} s(f, x)$.*

Definition 2. *The* block sensitivity *of f on x, denoted by $bs(f, x)$, is the maximum number of disjoint subsets $B_1, B_2, \ldots, B_b$ of $[n]$ such that $f(x) \neq f(x^{B_i})$ for all i. The* block sensitivity *of f, denoted by $bs(f)$, is $\max_x bs(f, x)$. For $z \in \{0, 1\}$, the z-*block sensitivity *of f, denoted by $bs_z(f)$, is $\max_{x \in f^{-1}(z)} bs(f, x)$.*

Definition 3. *A* certificate *of f on x is a subset $S \subseteq [n]$ such that $f(y) = f(x)$ whenever $y_i = x_i$ for all $i \in S$. The* size *of a certificate is $|S|$.*

The certificate complexity *of f on x, denoted by $C(f, x)$, is the size of a smallest certificate of f on x. The* certificate complexity *of f, denoted by $C(f)$, is $\max_x C(f, x)$. For $z \in \{0, 1\}$, the z-*certificate complexity *of f, denoted by $C_z(f)$, is $\max_{x \in f^{-1}(z)} C(f, x)$.*

We can easily show the following relation between $s(f)$, $bs(f)$ and $C(f)$.

Proposition 1. *For any Boolean function f,*

$$s(f) \leq bs(f) \leq C(f).$$

Proof. By the definitions of $s(f)$ and $bs(f)$, $s(f) \leq bs(f)$. For all x, since a certificate on x have to contain indices of at least one variable of each sensitive block, $bs(f, x) \leq C(f, x)$. Thus, $bs(f) \leq C(f)$. $\square$

Let $x_i, y_i \in \{0, 1\}$ for $1 \leq i \leq n$. A Boolean function is called *monotone* if $f(x_1, x_2, \ldots, x_n) \leq f(y_1, y_2, \ldots, y_n)$ whenever $x_i \leq y_i$ for all $1 \leq i \leq n$. Nisan [8] showed the following proposition for monotone functions.

Proposition 2 ([8]). *If f is a monotone function, then*

$$s(f) = bs(f) = C(f).$$

2.2 Unate functions and read-once functions

A Boolean function $f : \{0, 1\}^n \to \{0, 1\}$ is *positive unate* in x_i, $1 \leq i \leq n$, if

$$\begin{aligned} & f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n) \\ \leq\ & f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n) \end{aligned}$$

for all x_j, $j \neq i$, and is *negative unate* in x_i if

$$\begin{aligned} & f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n) \\ \geq\ & f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n) \end{aligned}$$

for all x_j, $j \neq i$. A function f is called *unate* if f is positive or negative unate in all x_i for $1 \leq i \leq n$. Monotone functions are a special case of unate functions such that a function is positive unate in all input variables.

A *Boolean formula* is a rooted binary tree in which each internal node is labeled by the Boolean operators $\wedge$, $\vee$, or $\oplus$ and each leaf is labeled by a Boolean variable or its negation. A Boolean formula computes a Boolean function in a natural way. A Boolean formula is called *read-once* if every variable appears exactly once. A *read-once* Boolean function is a Boolean function that can be represented by a read-once Boolean formula. Notice that we define read-once Boolean functions based on Boolean formulas which have the Boolean operator $\oplus$.

3 Unate functions

In this section, we prove the following theorem.

Theorem 1. *If f is a unate function, then*

$$s(f) = bs(f) = C(f).$$

$s(f)$, $bs(f)$ and $C(f)$ of a Boolean function f are not changed even if some input variables of f are flipped. More precisely, the following lemma holds.

Lemma 1. *Let $f(x)$ be a Boolean function, and let $S \subseteq [n]$. For any S, if $g(y)$ is defined as $f(y^S)$, then,*

$$s(f) = s(g),\ bs(f) = bs(g),\ C(f) = C(g).$$

Proof. It is obvious by the definitions of $s(f)$, $bs(f)$ and $C(f)$. □

Proof (of Theorem 1). Let $S = \{i | f \text{ is negative unate in } x_i, 1 \leq i \leq n\}$. We define $g(y)$ as $f(y^S)$, then $g(y)$ is monotone. By Lemma 1,

$$s(f) = s(g),\ bs(f) = bs(g),\ C(f) = C(g).$$

By Proposition 2,

$$s(g) = bs(g) = C(g).$$

Hence,

$$s(f) = bs(f) = C(f).$$

□

4 Read-once functions

In this section, we prove that $s(f) = bs(f) = C(f)$ for any read-once Boolean function (Theorem 2), and prove that $\sqrt{n} \leq s(f)$ (Corollary 1) and the $\sqrt{n}$ lower bound is tight.

4.1 Lemma

Consider a read-once Boolean formula F representing a read-once Boolean function. In F, two subformulas which are connected to a same node have no common input variables, since every variable appears exactly once in a read-once Boolean formula. This fact enables us to analyze the sensitivity and certificate complexity of functions computed at each node in F.

Lemma 2. *Let f_1 and f_2 be Boolean functions such that f_1 and f_2 have no common input variables, and f_1 and f_2 are not constant functions.*

If $f = f_1 \wedge f_2$, then

$$\begin{aligned} s_0(f) &= \max\{s_0(f_1), s_0(f_2)\}, \\ C_0(f) &= \max\{C_0(f_1), C_0(f_2)\}, \\ s_1(f) &= s_1(f_1) + s_1(f_2), \\ C_1(f) &= C_1(f_1) + C_1(f_2). \end{aligned}$$

If $f = f_1 \vee f_2$, then

$$\begin{aligned} s_0(f) &= s_0(f_1) + s_0(f_2), \\ C_0(f) &= C_0(f_1) + C_0(f_2), \\ s_1(f) &= \max\{s_1(f_1), s_1(f_2)\}, \\ C_1(f) &= \max\{C_1(f_1), C_1(f_2)\}. \end{aligned}$$

If $f = f_1 \oplus f_2$, then

$$\begin{aligned} s_0(f) &= \max\{s_0(f_1) + s_0(f_2), s_1(f_1) + s_1(f_2)\}, \\ C_0(f) &= \max\{C_0(f_1) + C_0(f_2), C_1(f_1) + C_1(f_2)\}, \\ s_1(f) &= \max\{s_0(f_1) + s_1(f_2), s_1(f_1) + s_0(f_2)\}, \\ C_1(f) &= \max\{C_0(f_1) + C_1(f_2), C_1(f_1) + C_0(f_2)\}. \end{aligned}$$

Proof. Assume that $f = f_1 \wedge f_2$. We consider that $s_0(f) = \max\{s_0(f_1), s_0(f_2)\}$. If $s_0(f_1) \geq s_0(f_2)$, we can assign input variables of f_2 so that $f_2 = 1$, and independently we can assign input variables of f_1. Thus, we can confirm that $s_0(f) = \max\{s_0(f_1), s_0(f_2)\}$.

Similarly, we can confirm all equations by the definitions of sensitivity and certificate complexity. □

4.2 Equality

Lemma 2 immediately gives the following lemma.

Lemma 3. *Let f_1 and f_2 be Boolean functions such that f_1 and f_2 have no common input variables, and f_1 and f_2 are not constant functions. If*

$$f = f_1 \wedge f_2, \;\; f = f_1 \vee f_2, \;\; \text{or } f = f_1 \oplus f_2,$$

and

$$s_0(f_1) = C_0(f_1), \;\; s_1(f_1) = C_1(f_1),$$

$$s_0(f_2) = C_0(f_2),\ s_1(f_2) = C_1(f_2),$$

then

$$s_0(f) = C_0(f),\ s_1(f) = C_1(f).$$

Now, we prove the following theorem.

Theorem 2. *If f is a read-once Boolean function, then*

$$s(f) = bs(f) = C(f).$$

Proof. Since $s(f) \leq bs(f) \leq C(f)$ for any Boolean function f by Proposition 1, we only need to prove $s(f) = C(f)$.

Let n be the number of input variables of f. We use induction on n and prove $s_0(f) = C_0(f)$ and $s_1(f) = C_1(f)$.

Base: $n = 1$. Then, $f = x_1$ or $f = \neg x_1$, and $s_0(f) = s_1(f) = 1$ and $C_0(f) = C_1(f) = 1$. Thus, $s_0(f) = C_0(f)$ and $s_1(f) = C_1(f)$.

Induction Step: Suppose $s_0(f') = C_0(f')$ and $s_1(f') = C_1(f')$ for every Boolean function f' such that the number of input variables of f' is less than n.

Let F be a read-once Boolean formula which computes f. Recall that we define Boolean formulas as rooted binary trees. Let f_1 and f_2 are Boolean functions computed by subformulas which are connected to the root node of F. Then, $f = f_1 \wedge f_2$, $f = f_1 \vee f_2$, or $f = f_1 \oplus f_2$, and the number of input variables of f_1 and f_2 is less than n, respectively. By the supposition, $s_0(f_1) = C_0(f_1)$, $s_1(f_1) = C_1(f_1)$, $s_0(f_2) = C_0(f_2)$ and $s_1(f_2) = C_1(f_2)$. Thus, by Lemma 3, $s_0(f) = C_0(f)$ and $s_1(f) = C_1(f)$, which mean $s(f) = C(f)$. □

4.3 Lower bound

Lemma 2 also gives a lower bound for the sensitivity of read-once functions.

Theorem 3. *If f is a read-once Boolean function of n input variables, then*

$$n \leq s_0(f)s_1(f).$$

Proof. We use induction on n.

Base: $n = 1$. Then, $f = x_1$ or $f = \neg x_1$, and $s_0(f)s_1(f) = 1$. Thus, $n \leq s_0(f)s_1(f)$.

Induction Step: Suppose $n' \leq s_0(f')s_1(f')$ for every Boolean function f' such that the number of input variables of f', denoted by n', is less than n.

Let F be a read-once Boolean formula which computes f. Recall that we define Boolean formulas as rooted binary trees. Let f_1 and f_2 are Boolean functions computed by subformulas which are connected to the root node of F, and let n_1 and n_2 are the number of input variables of f_1 and f_2, respectively. Then, $f = f_1 \wedge f_2$, $f = f_1 \vee f_2$, or $f = f_1 \oplus f_2$, and $n_1 < n$, $n_2 < n$, and $n_1 + n_2 = n$. By the supposition, $n_1 \leq s_0(f_1)s_1(f_1)$ and $n_2 \leq s_0(f_2)s_1(f_2)$.

If $f = f_1 \wedge f_2$, then, by Lemma 2,

$$\begin{aligned} s_0(f)s_1(f) &= \max\{s_0(f_1), s_0(f_2)\}s_1(f_1) + \max\{s_0(f_1), s_0(f_2)\}s_1(f_2) \\ &\geq s_0(f_1)s_1(f_1) + s_0(f_2)s_1(f_2) \\ &\geq n_1 + n_2 = n. \end{aligned}$$

Similarly, we can prove that $n \leq s_0(f)s_1(f)$ also for the cases that $f = f_1 \vee f_2$ and $f = f_1 \oplus f_2$. □

Recall that $s(f) = \max\{s_0(f), s_1(f)\}$.

Corollary 1. *If f is a read-once Boolean function of n input variables, then*

$$\sqrt{n} \leq s(f).$$

The lower bounds in Theorem 3 and Corollary 1 are tight, since we can easily confirm that the following read-once Boolean function f has $s_0(f) = n/m$ and $s_1(f) = m$. (We assume that m is a positive integer such that n/m becomes an integer.)

$$f = \bigvee_{i=1}^{n/m} \bigwedge_{j=1}^{m} x_{m(i-1)+j}.$$

5 Concluding Remarks

In this paper, we investigated the sensitivity, block sensitivity, and certificate complexity of unate functions and read-once functions. As the conclusion of this paper, we show an application of our results to decision tree complexity.

Let $D(f)$ denote the decision tree complexity of f, i.e., the depth of an optimal decision tree that computes f. Beals et al. [3] prove

Theorem 4 ([3]). *For any Boolean function f,*

$$D(f) \leq C_1(f)bs(f).$$

Recall that we proved that $s(f) = bs(f) = C(f)$ for any unate function f (Theorem 1) and for any read-once function f (Theorem 2), and $C_1(f) \leq C(f)$ by the definition. Thus, we obtain the following corollary.

Corollary 2. *If f is a unate function or a read-once function, then*

$$D(f) \leq s(f)^2.$$

Although Corollary 2 is meaningful for unate functions, we have to be attentive for read-once functions, since we can easily see that $D(f) = n$ for every read-once function. Thus, Corollary 2 is an alternating proof of Corollary 1 rather than an upper bound of $D(f)$. Notice that the alternating proof depends on Theorem 4 and cannot prove Theorem 3.

References

1. Ambainis, A., Gao, Y., Mao, J., Sun, X., Zuo, S.: New upper bound on block sensitivity and certificate complexity in terms of sensitivity. CoRR abs/1306.4466 (2013)
2. Ambainis, A., Sun, X.: New separation between $s(f)$ and $bs(f)$. Electronic Colloquium on Computational Complexity (ECCC) 18, 116 (2011)
3. Beals, R., Buhrman, H., Cleve, R., Mosca, M., de Wolf, R.: Quantum lower bounds by polynomials. J. ACM 48(4), 778–797 (2001)
4. Buhrman, H., de Wolf, R.: Complexity measures and decision tree complexity: a survey. Theor. Comput. Sci. 288(1), 21–43 (2002)
5. Hatami, P., Kulkarni, R., Pankratov, D.: Variations on the sensitivity conjecture. Theory of Computing, Graduate Surveys 2, 1–27 (2011)
6. Impagliazzo, R., Kabanets, V.: Fourier concentration from shrinkage. Electronic Colloquium on Computational Complexity (ECCC) 20, 163 (2013)
7. Kenyon, C., Kutin, S.: Sensitivity, block sensitivity, and l-block sensitivity of boolean functions. Inf. Comput. 189(1), 43–53 (2004)
8. Nisan, N.: CREW PRAMs and decision trees. SIAM J. Comput. 20(6), 999–1007 (1991)
9. Rubinstein, D.: Sensitivity vs. block sensitivity of boolean functions. Combinatorica 15(2), 297–299 (1995)
10. Virza, M.: Sensitivity versus block sensitivity of boolean functions. Inf. Process. Lett. 111(9), 433–435 (2011)