IEICE TRANS. INF. & SYST., VOL.E98–D, NO.3 MARCH 2015

PAPER Special Section on Foundations of Computer Science—New Spirits in Theory of Computation and Algorithm—

Candidate Boolean Functions towards Super-Quadratic Formula Size

Kenya UENO[†a)], ***Member***

SUMMARY In this paper, we explore possibilities and difficulties to prove super-quadratic formula size lower bounds from the following aspects. First, we consider recursive Boolean functions and prove their general formula size upper bounds. We also discuss recursive Boolean functions based on exact 2-bit functions. We show that their formula complexity are at least $\Omega(n^2)$. Hence they can be candidate Boolean functions to prove super-quadratic formula size lower bounds. Next, we consider the reason of the difficulty of resolving the formula complexity of the majority function in contrast with the parity function. In particular, we discuss the structure of an optimal protocol partition for the Karchmer-Wigderson communication game.

key words: *Boolean function, computational complexity, formula complexity*

1. Introduction

Proving formula size lower bounds is a fundamental problem in complexity theory as a weaker version of the circuit size lower bound problem and $\mathbf{P} \neq \mathbf{NP}$. A super-polynomial formula size lower bound for a function in **NP** implies $\mathbf{NC}^1 \neq \mathbf{NP}$. As extensions of the classical result of Khrapchenko [1] proving the matching n^2 formula size lower bound for the parity function, there are a lot of techniques studied to improve formula size lower bounds. The current best formula size lower bound of $n^{3-o(1)}$ is proven by Håstad [2] for the Andreev function [3]. We do not know any Boolean function with a super-quadratic formula size lower bound except the Andreev function.

In recent studies [4]–[6], it has been revealed that the LP bound of Karchmer, Kushilevitz and Nisan [7] subsumes most of known techniques such as Khrapchenko [1], its extension by Koutsoupias [8], a key lemma used in the proof of Håstad [2], and the quantum adversary bound [4], [9]. However, Karchmer, Kushilevitz and Nisan have also shown that their technique cannot prove a formula size lower bound larger than $4n^2$. Therefore, all these lower bound techniques as extensions of Khrapchenko's method have limitation and cannot prove super-quadratic formula size lower bounds. Our recent work [10], [11] devised stronger versions of the LP bound, which are potentially strong enough to prove super-polynomial formula size lower bounds. However, there are still obscure barriers even against proving super-quadratic formula size lower bounds.

One of the main difficulties to prove a super-quadratic formula size lower bound is how to find candidates which are potentially hard enough to have super-quadratic formula size. Of course, hard Boolean functions which are defined with difficult problems (e.g., matching and clique) are definitely good candidates to prove strong lower bounds [12]. However, these functions are extremely hard to analyze. For the purpose of attacking the quadratic barrier, we need candidate Boolean functions which are relatively easier to analyze. At the same time, they must be hard enough to have super-quadratic formula complexity.

When we consider tractability of Boolean function analysis, two important notions are "symmetric" and "recursive". The n-bit parity function is both symmetric and recursive in the sense that it is invariant under permutations of the input bits (i.e., symmetric) and can be constructed by composing the 2-bit parity function with the $\lfloor n/2 \rfloor$-parity function and the $\lceil n/2 \rceil$-parity function (i.e., recursive). This is the main reason why we can prove its matching formula size lower bound easily.

This paper consists of two parts, which consider the difficulty of proving super-quadratic formula size lower bounds in terms of candidate Boolean functions as follows.

In the former part, we define recursive Boolean functions, and show that naive and recursive construction for some of them provides formulas of super-quadratic size. They are not always symmetric in general. Since it is widely believed that such naive and recursive construction provides an optimal formula in most cases [13]–[15], we conclude that the recursive Boolean functions are good candidates for a super-quadratic formula size lower bound. Moreover, we particularly pick up an explicit recursive Boolean function based on exact 2-bit functions, and propose the function as the best candidate. Their formula size upper bounds are $O(n^{\log_2 5})$ or $O(n^{\log_2 6})$ depending on the size of the base functions. We prove that their formula size lower bounds are at least $\Omega(n^2)$ by reducing them to the parity function.

In the latter part, we will discuss difficulties of resolving the formula complexity of the majority function. It is symmetric, but not recursive. The current best formula size upper and lower bounds of the majority function are $O(n^{4.57})$ [16] and $\Omega(n^2)$ [1], [10], respectively. Since there is a large gap between its formula size upper and lower bounds, it has been a main target to prove a super-quadratic formula size lower bound, even for the monotone case [17]. We propose a hypothesis named singleton cell hypothesis on

Manuscript received March 27, 2014.
Manuscript revised July 22, 2014.

[†]The author is with the Hakubi Center for Advanced Research and Graduate School of Informatics, Kyoto University, Kyoto-shi, 606–8501 Japan.
a) E-mail: kenya@i.kyoto-u.ac.jp

DOI: 10.1587/transinf.2014FCP0011

matrices to analyze formula complexity. We show that the hypothesis is true for the functions such that some of known quadratic lower bounds are optimal. We also observe that the hypothesis is related to the difficulty of proving super-quadratic formula size lower bounds of the majority function.

We hope that our analysis explaining the difficulties of resolving formula complexity will be useful to overcome the stiff barrier against proving super-quadratic formula size lower bounds.

2. Preliminaries

We assume that the readers are familiar with the basics of Boolean functions. Throughout the paper, n means as the number of input variables on Boolean functions. We will use the following Boolean functions.

Definition 2.1 (Boolean Functions)**.** *The parity function* $\mathbf{PAR}_n : \{0,1\}^n \mapsto \{0,1\}$ *is defined by*

$$\mathbf{PAR}_n(x_1, \cdots, x_n) = \begin{cases} 1 & (\sum_{i=1}^{n} x_i \equiv 1 \mod 2), \\ 0 & (\sum_{i=1}^{n} x_i \equiv 0 \mod 2). \end{cases}$$

The exact 2-bit function $\mathbf{EXACT}_n^2 : \{0,1\}^n \mapsto \{0,1\}$ *is defined by*

$$\mathbf{EXACT}_n^2(x_1, \cdots, x_n) = \begin{cases} 1 & (\sum_{i=1}^{n} x_i = 2), \\ 0 & (\textit{otherwise}). \end{cases}$$

The majority function $\mathbf{MAJ}_n : \{0,1\}^n \mapsto \{0,1\}$ *is defined by*

$$\mathbf{MAJ}_n(x_1, \cdots, x_n) = \begin{cases} 1 & (\sum_{i=1}^{n} x_i \geq \lceil n/2 \rceil), \\ 0 & (\textit{otherwise}). \end{cases}$$

A threshold function $\mathbf{TH}_n^t : \{0,1\}^n \mapsto \{0,1\}$ *as a generalization of the majority function is defined by*

$$\mathbf{TH}_n^t(x_1, \cdots, x_n) = \begin{cases} 1 & (\sum_{i=1}^{n} x_i \geq t), \\ 0 & (\textit{otherwise}). \end{cases}$$

for any fixed integer t $(0 \leq t \leq n)$.

For Boolean vectors $\mathbf{x} = (x_1, \cdots, x_n)$ and $\mathbf{y} = (y_1, \cdots, y_n)$, we define $\mathbf{x} \leq \mathbf{y}$ if $x_i \leq y_i$ for all $i \in \{1, \cdots n\}$. A Boolean function f is called monotone if $\mathbf{x} \leq \mathbf{y}$ implies $f(\mathbf{x}) \leq f(\mathbf{y})$ for all $\mathbf{x}, \mathbf{y} \in \{0,1\}^n$.

Formula sizes for Boolean functions are defined as follows.

Definition 2.2 (Formula Size)**.** *A formula is a binary tree with each leaf labeled by a literal and each internal node labeled by either of the binary connectives* $\wedge$ *and* $\vee$*. A literal is either a variable or its negation. The size of a formula is its number of literals. We define formula size* $L(f)$ *of a Boolean function* f *as the size of the smallest formula computing* f*. We also define* $L_m(f)$ *as the monotone formula size of a monotone Boolean function* f *where a monotone formula is a formula without negations.*

3. Formula Complexity of Recursive Boolean Functions

3.1 Definitions

In this section, we consider recursive Boolean functions, which can be divided by 2 cases, the balanced case and the unbalanced case as illustrated in Fig. 1. Throughout the paper, we assume $\alpha, \beta \geq 2$. First we define the balanced recursive functions as follows.

Definition 3.1 (Balanced Recursive Functions)**.** *From a base Boolean function* f *on* α *input bits, we recursively define a Boolean function* $\mathbf{REC}_\beta^h[f]$ *with* $\beta = \alpha$ *on* n *input variables by*

$$\mathbf{REC}_\beta^1[f](\mathbf{x}_1) = f(\mathbf{x}_1)$$

for $h = 1$*, and*

$$\mathbf{REC}_\beta^h[f](\mathbf{x}_h) = f(\mathbf{REC}_\beta^{h-1}[f](\mathbf{x}_{h-1}^1), \cdots, \mathbf{REC}_\beta^{h-1}[f](\mathbf{x}_{h-1}^\beta))$$

for $h \geq 2$*. Here* $\mathbf{x}_1, \mathbf{x}_h, \mathbf{x}_{h-1}^i (i = 1, \cdots, \beta)$ *denote Boolean vectors.*

Besides the notion of balanced recursive functions, we add the following notion of unbalanced recursive functions.

Definition 3.2 (Unbalanced Recursive Functions)**.** *From a base Boolean function* f *on* α *input bits, we recursively define a Boolean function* $\mathbf{REC}_\beta^h[f]$ *with* $\beta < \alpha$ *on* n *input*

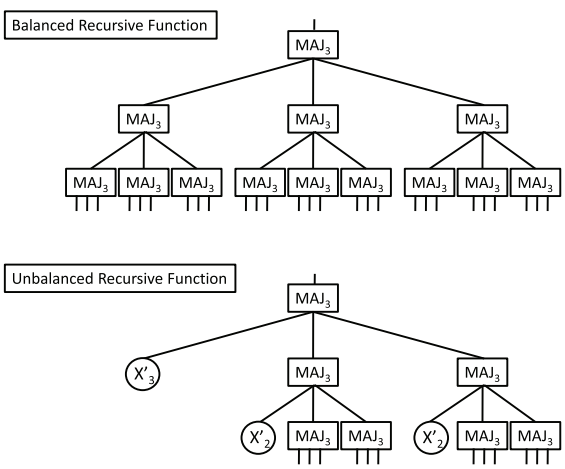

Fig. 1 Comparison between a balanced recursive function ($\alpha = \beta = 3$) and an unbalanced recursive function ($\alpha = 3$ and $\beta = 2$) where the base function is $\mathbf{MAJ}_3$ and $h = 3$ (Variables at the bottom level are omitted).

variables by

$$\mathbf{REC}^1_\beta[f](\mathbf{x}_1) = f(\mathbf{x}_1)$$

for $h = 1$*, and*

$$\mathbf{REC}^h_\beta[f](\mathbf{x}_h) = f(\mathbf{x}'_h, \mathbf{REC}^{h-1}_\beta[f](\mathbf{x}^1_{h-1}), \cdots, \mathbf{REC}^{h-1}_\beta[f](\mathbf{x}^\beta_{h-1}))$$

for $h \geq 2$*. Here* $\mathbf{x}_1, \mathbf{x}_h, \mathbf{x}'_h, \mathbf{x}^i_{h-1} (i = 1, \cdots, \beta)$ *denote Boolean vectors.*

We distinguish 2 definitions of balanced and unbalanced recursive Boolean functions $\mathbf{REC}^h_\beta[f]$ by looking at the condition whether $\beta = \alpha$ or $\beta < \alpha$.

The number of branches β is the number of occurrences of sub-functions $\mathbf{REC}^{h-1}_\beta(\mathbf{x}_{h-1})$ in the recursive definition. It implies that

$$\alpha = |\mathbf{x}'_h| + \beta,$$

and

$$|\mathbf{x}_h| = |\mathbf{x}'_h| + \beta \cdot |\mathbf{x}^1_{h-1}|.$$

We assume that the number α and β are constants and do not depend on the input length n.

We will use the following definition in our analysis.

Definition 3.3 (Partial Formula Size)**.** *From the input variables of a Boolean function* f*, we choose arbitrary* k *variables*

$$X_k = \{x_{i_1}, x_{i_2}, \cdots, x_{i_k}\} \subseteq \{x_1, x_2, \cdots, x_n\}$$

and count the number of their occurrences $L^k(X_k, F)$ *in a formula* F *computing* f*. By taking minimization over all the subsets of variables* X_k *and formulas computing* F *and we define*

$$L^k(f) = \min_{X_k, F} L^k(X_k F)$$

In the monotone case, $L^k_m(f)$ *is similarly defined by replacing formulas with monotone formulas.*

This notion is an extension of the definition of formula size because $L(f) = L^n(f)$.

3.2 General Upper Bounds for Recursive Boolean Functions

In this section, we discuss limitation and perspective of proving formula size lower bounds for recursive Boolean functions in general based on the following theorem.

Theorem 3.4. *For any Boolean function* f *depending on all the* α *input variables,*

$$L(\mathbf{REC}^h_\beta[f]) \in O(n^\gamma)$$

where $\gamma = \log_\beta(2^{\beta+1} - 2)$.

Proof. To prove the theorem, we would like to prove

$$L^k(f) \leq 2^{k+1} - 2.$$

For this purpose, we assume that the fixed k variables are $x_1, x_2, \cdots, x_k$. By Shannon's expansion,

$$\begin{aligned} &f(x_1, x_2, \cdots, x_k, \cdots, x_n) \\ =&(x_1 \wedge f(1, x_2, \cdots, x_k, \cdots, x_n)) \\ &\vee (\neg x_1 \wedge f(0, x_2, \cdots, x_k, \cdots, x_n)). \end{aligned}$$

Iterating this process for $i = 1, \cdots, k$, we can obtain a formula in which a variable x_i $(1 \leq i \leq k)$ appears 2^i times. Therefore,

$$\begin{aligned} L^k(f) &\leq 2 + 4 + \cdots + 2^k \\ &= \sum_{i=1}^{k} 2^i = 2^{k+1} - 2. \end{aligned}$$

To construct a formula for $\mathbf{REC}^h_\beta[f](\mathbf{x}_h)$, we take a formula computing f in which some β variables appears at most $2^{\beta+1} - 2$ times in total. Then, we replace the β variables by $\mathbf{REC}^{h-1}_\beta[f](\mathbf{x}^1_{h-1}), \cdots, \mathbf{REC}^{h-1}_\beta[f](\mathbf{x}^\beta_{h-1})$, respectively, and the other $n - \beta$ variables by $\mathbf{x}'_h$.

To estimate the general formula size upper bound l_h $(\geq L(\mathbf{REC}^h_\beta[f]))$, we consider a recursive inequality

$$l_h \leq (2^{\beta+1} - 2) \cdot l_{h-1} + \delta$$

where the constant δ depends on the construction of the formula which achieves $L^\beta(f)$. Since α and β are independent from h, δ is also independent from h. We can transform the inequality as follows.

$$l_h + \frac{\delta}{2^{\beta+1} - 3} \leq (2^{\beta+1} - 2) \cdot \left(l_{h-1} + \frac{\delta}{2^{\beta+1} - 3}\right)$$

Solving the inequality, we can obtain

$$l_h \leq (2^{\beta+1} - 2)^{h-1} \cdot \left(l_1 + \frac{\delta}{2^{\beta+1} - 3}\right) - \frac{\delta}{2^{\beta+1} - 3}.$$

for $h \geq 2$. Taking sufficiently large constant c which is independent from h, we have

$$l_h \leq c \cdot (2^{\beta+1} - 2)^h.$$

Let n_h be the the number of input bits $n_h = |\mathbf{x}_h|$ for the recursive Boolean function $\mathbf{REC}^h_\beta[f]$. What we would like to prove is

$$c \cdot (2^{\beta+1} - 2)^h \leq c' \cdot {n_h}^\gamma$$

for some constant c' to show l_h is bounded by $O(n^\gamma_h)$.

To estimate n_h, we consider

$$n_h = \alpha - \beta + \beta \cdot n_{h-1}.$$

Solving it based on $n_1 = \alpha$, we obtain

$$n_h = \frac{\alpha - 1}{\beta - 1} \cdot \beta^h - \frac{\alpha - \beta}{\beta - 1}$$

Table 1 Exponents of Formula Size Upper Bounds in terms of the Number of Branches (The second and third lines show non-monotone and monotone cases, respectively).

β	2	3	4	5	6	7	8	9	10
$\log_\beta(2^{\beta+1}-2)$	2.5850	2.4022	2.4535	2.5644	2.6992	2.8457	2.9982	3.1539	3.3110
$\log_\beta(2^\beta-1)$	1.5850	1.7713	1.9535	2.1337	2.3124	2.4895	2.6648	2.8383	3.0099

for $h \geq 2$. For fixed constants α and β, we have

$$\frac{\alpha-1}{\beta-1} \geq 1$$

from $\alpha \geq \beta$. Thus, we have

$$n_h \geq \beta^h$$

for sufficiently large h. Therefore

$$n_h^\gamma \geq (\beta^h)^\gamma = (2^{\beta+1}-2)^h$$

where $\gamma = \log_\beta(2^{\beta+1}-2)$. □

In the monotone case, we have the following theorem. We can prove it by a similar argument as in the case for non-monotone Boolean functions.

Theorem 3.5. *For any monotone Boolean function* $\mathbf{REC}_\alpha^h$ *whose number of branches is* $\beta \geq 2$*, we have*

$$L_m(\mathbf{REC}_\alpha^h) \in O(n^\gamma)$$

where $\gamma = \log_\beta(2^\beta - 1)$

Proof. The proof is based on a similar argument as in the case for non-monotone Boolean functions. The difference is that we use the monotone version of Shannon's expansion as follows.

$$\begin{aligned} &f(x_1, x_2, \cdots, x_k, \cdots, x_n) \\ =&(x_1 \wedge f(1, x_2, \cdots, x_k, \cdots, x_n)) \\ &\vee f(0, x_2, \cdots, x_k, \cdots, x_n). \end{aligned}$$

In this case, we have

$$\begin{aligned} L_m^k(f) &\leq 1 + 2 + \cdots + 2^{k-1} \\ &= \sum_{i=1}^{k} 2^i = 2^k - 1. \end{aligned}$$

and obtain the general upper bound of monotone formula size. □

In Table 1, we list the exponents of formula size upper bounds for recursive Boolean functions in terms of the number of branches. A counterintuitive fact is that the upper bound for $\beta = 2$ is larger than that for $\beta = 3, 4, 5$. Therefore, it might be better to focus on the case of $\beta = 2$ to seek candidate Boolean functions for super-quadratic formula size.

3.3 Recursive Boolean Functions Based on Exact 2-bit Functions

In this subsection, we consider recursive Boolean functions composed of exact 2-bit functions. First, we observe that the balanced recursive function $\mathbf{REC}_3^h[\mathbf{EXACT}_3^2]$ is too weak to obtain super-quadratic formula complexity.

Proposition 3.6.

$$L(\mathbf{REC}_3^h[\mathbf{EXACT}_3^2]) \in O(n^{\log_3 8}) \subset O(n^{1.90}).$$

Proof. We can construct a formula for $\mathbf{EXACT}_3^2$ as

$$(\neg x_1 \wedge x_2 \wedge x_3) \vee (x_1 \wedge \neg x_2 \wedge x_3) \vee (x_1 \wedge x_2 \wedge \neg x_3),$$

and shrink it as follows:

$$(((\neg x_1 \wedge x_2) \vee (x_1 \wedge \neg x_2)) \wedge x_3) \vee (x_1 \wedge x_2 \wedge \neg x_3).$$

This formula size is 8. Therefore, we can obtain a formula size upper bound of $O(n^{1.90})$ for the balanced recursive function by recursive compositions of the formula. □

This kind of subquadratic formula size upper bounds is widely applicable for many balanced recursive functions rather than a special case for the exact 2-bit function. Currently, we know few Boolean functions on a constant number of input bits with super-quadratic formula complexity. Therefore, we do not know any candidate base function to achieve our goal by analyzing balanced recursive Boolean functions.

On the other hand, unbalanced recursive Boolean functions can be good candidates to prove super-quadratic formula size lower bounds. As an example, we consider the following unbalanced recursive Boolean function with its formula size upper bound.

Proposition 3.7.

$$L(\mathbf{REC}_2^h[\mathbf{EXACT}_3^2]) \in O(n^{\log_2 5}) \subset O(n^{2.33}).$$

Proof. In the above construction of formula for $\mathbf{EXACT}_3^2$, the numbers of appearances of the variables x_2 and x_3 are 3 and 2, respectively. Hence we can recursively construct a formula for $\mathbf{REC}_2^h[\mathbf{EXACT}_3^2]$ with 5 sub-formulas for $\mathbf{REC}_2^{h-1}[\mathbf{EXACT}_3^2]$. By using a similar argument as in the proof of Theorem 3.4, we can prove the upper bound. □

It is widely believed that naive recursive constructions of Boolean formula are optimal in most cases. The previous studies [13]–[15] show some supporting evidence for this conjecture by proving almost tight lower bounds for compositions of the universal relation ($\{0,1\}^n \times \{0,1\}^n$), which includes all the communication matrices of Boolean functions (Definition 4.1).

Similarly, we also conjecture that the formula size upper bound for $\mathbf{REC}_2^h[\mathbf{EXACT}_3^2]$ is tight. One nice property

of the exact 2-bit function is that it contains the parity function as its sub-function. So we can prove the following formula size lower bound.

Theorem 3.8.

$$L(\mathbf{REC}_2^h[\mathbf{EXACT}_3^2]) \in \Omega(n^2).$$

Proof. If we assign $x_1 = 1$ as

$$\begin{aligned}\mathbf{EXACT}_3^2(1, x_2, x_3) &= (\neg x_2 \wedge x_3) \vee (x_2 \wedge \neg x_3) \\ &= \mathbf{PAR}_2(x_2, x_3).\end{aligned}$$

for each level of the recursive definition of the recursive function $\mathbf{REC}_2^h[\mathbf{EXACT}_3^2]$, it becomes the 2^h-bit parity function. We know the formula size lower bound of $(2^h)^2$ for it by Khrapchenko's bound [1]. As a result, $L(\mathbf{REC}_2^h[\mathbf{EXACT}_3^2])$ cannot be less than it. □

We can prove the same lower bound by the weighted version of the quantum adversary bound [9]. In general, we can give the condition in which formula size lower bounds of the recursive Boolean functions are bounded by $\Omega(n^2)$.

Theorem 3.9. *We fix $\alpha - 2$ bits of the α input bits of a Boolean function f. If the Boolean function on the remaining 2 input bits becomes either the 2-bit parity function or its negation, then we have*

$$L(\mathbf{REC}_2^h[f]) \in \Omega(n^2).$$

Proof. This theorem can be proven by the same argument as the proof of the previous theorem. □

We can prove a larger bound for a larger base function $\mathbf{EXACT}_4^2$.

Proposition 3.10.

$$L(\mathbf{REC}_2^h[\mathbf{EXACT}_4^2]) \in O(n^{\log_2 6}) \subset O(n^{2.59}).$$

Proof. We can construct a formula for $\mathbf{EXACT}_4^2$ as

$$\begin{aligned}&(\neg x_1 \wedge x_2 \wedge x_3 \wedge x_4) \vee (x_1 \wedge \neg x_2 \wedge x_3 \wedge x_4) \\ &\vee (x_1 \wedge x_2 \wedge \neg x_3 \wedge x_4) \vee (x_1 \wedge x_2 \wedge x_3 \wedge \neg x_4),\end{aligned}$$

and shrink it as follows:

$$\begin{aligned}&(((\neg x_1 \wedge x_2) \vee (x_1 \wedge \neg x_2)) \wedge x_3 \wedge x_4) \\ &\vee (x_1 \wedge x_2 \wedge ((\neg x_3 \wedge x_4) \vee (x_3 \wedge \neg x_4))).\end{aligned}$$

Its formula size is 12. Therefore, we have obtained the formula size upper bound. □

It matches the general upper bound for $\beta = 2$ and seems to be one of the best candidates with smaller branches ($\beta \leq 5$) towards super-quadratic formula size lower bounds because of $L^2(\mathbf{EXACT}_4^2) = 6$.

4. Parity versus Majority with Respect to Formula Size

In this section, we try to give some reason behind the difficulty of proving a super-quadratic formula size lower bound for the majority function. Before going to the results, we review some notions to prove formula size lower bounds.

4.1 Communication Matrix and Singleton Cell Hypothesis

Karchmer and Wigderson [18] characterize the formula size of any Boolean function in terms of a communication game. In the game, given a Boolean function f, Alice gets an input $\mathbf{x}$ such that $f(\mathbf{x}) = 1$ and Bob gets an input $\mathbf{y}$ such that $f(\mathbf{y}) = 0$. The goal of the game is to find an index i such that $x_i \neq y_i$. Here, x_i and y_i denote the i-th bits of $\mathbf{x}$ and $\mathbf{y}$, respectively. From the game, we consider the following matrix called the communication matrix.

Definition 4.1 (Communication Matrix)**.** *Given a Boolean function f, its communication matrix is defined as a matrix whose rows and columns are indexed by $X = f^{-1}(1)$ and $Y = f^{-1}(0)$, respectively. Each cell of the matrix contains indices i such that $x_i \neq y_i$. A combinatorial rectangle is a direct product $X' \times Y'$ where $X' \subseteq X$ and $Y' \subseteq Y$. A combinatorial rectangle $X' \times Y'$ is called monochromatic if every cell $(\mathbf{x}, \mathbf{y}) \in X' \times Y'$ contains the same index i. To describe it simply, we define a relation $R_f \subseteq X \times Y \times \{1, 2, \cdots, n\}$ as $R_f = \{(\mathbf{x}, \mathbf{y}, i) \mid \mathbf{x} \in X,\ \mathbf{y} \in Y,\ x_i \neq y_i\}$. We can also define the monotone version of the communication matrix and the relation associated with a monotone Boolean function f as $R_f^m = \{(\mathbf{x}, \mathbf{y}, i) \mid \mathbf{x} \in X,\ \mathbf{y} \in Y,\ (x_i = 1) \wedge (y_i = 0)\}$.*

When we interpret behavior of a communication protocol as a tree, the number of leaves in a best communication protocol for the Karchmer-Wigderson game is equivalent to the following bound.

Definition 4.2 (Protocol Partition Number)**.** *For any combinatorial rectangle $X' \times Y'$, we call its partition a pair of $X'_1 \times Y'$ and $X'_2 \times Y'$ where $X' = X'_1 \cup X'_2$ and $X'_1 \cap X'_2 = \emptyset$, or a pair of $X' \times Y'_1$ and $X' \times Y'_2$ where $Y' = Y'_1 \cup Y'_2$ and $Y'_1 \cap Y'_2 = \emptyset$. The protocol partition number $C^P(R_f)$ is defined by the minimum number of disjoint monochromatic rectangles which recursively partition the communication matrix $X \times Y$ where $R_f = \{(\mathbf{x}, \mathbf{y}, i) \mid \mathbf{x} \in X,\ \mathbf{y} \in Y,\ x_i \neq y_i\}$. In the monotone case, it is similarly defined by replacing R_f with R_f^m.*

Then, the theorem of Karchmer and Wigderson can be stated as follows.

Theorem 4.3 ([18])**.** *For any Boolean function f, $C^P(R_f) = L(f)$ and $C^P(R_f^m) = L_m(f)$.*

To prove a lower bound, we sometimes restrict rows and columns of the communication matrix as $R_f(X', Y') = \{(\mathbf{x}, \mathbf{y}, i) \mid \mathbf{x} \in X', \mathbf{y} \in Y', x_i \neq y_i\}$, and $R_f^m(X', Y') = \{(\mathbf{x}, \mathbf{y}, i) \mid \mathbf{x} \in X', \mathbf{y} \in Y',\ (x_i = 1) \wedge (y_i = 0)\}$ for some $X' \subseteq f^{-1}(1)$ and $Y' \subseteq f^{-1}(0)$. In particular, we will consider the following restriction.

Definition 4.4 (Minterms and Maxterms)**.** *For a monotone Boolean function f, an input $\mathbf{x}$ is called a minterm if $f(\mathbf{x}) =$*

1 *and* $(\mathbf{y} \leq \mathbf{x}) \wedge (\mathbf{x} \neq \mathbf{y})$ *implies* $f(\mathbf{y}) = 0$ *for any* $\mathbf{y}$ *and is called a maxterm if* $f(\mathbf{x}) = 0$ *and* $(\mathbf{x} \leq \mathbf{y}) \wedge (\mathbf{x} \neq \mathbf{y})$ *implies* $f(\mathbf{y}) = 1$ *for any* $\mathbf{y}$ *where* $\mathbf{x} \leq \mathbf{y}$ *if* $x_i \leq y_i$ *for all* $i \in \{1, \cdots n\}$. *Let* $\min(f)$ *and* $\max(f)$ *be the sets of all the minterms and maxterms for* f*, respectively.*

Protocol partition numbers for monotone communication matrices are invariant even when we restrict their rows and columns to minterms and maxterms, respectively, as follows.

Theorem 4.5 ([18])**.** *For any Boolean function* f*,*

$$C^P(R_f^m(\min(f), \max(f))) = L_m(f).$$

In the communication matrix of $X \times Y$, we consider the following sets of cells

$$S = \{(\mathbf{x}, \mathbf{y}) \mid \mathbf{x} \in X,\ \mathbf{y} \in Y,\ d_H(\mathbf{x}, \mathbf{y}) = 1\}.$$

where $d_H(\mathbf{x}, \mathbf{y})$ denotes the Hamming distance, which is the number of different bits, between $\mathbf{x}$ and $\mathbf{y}$. These cells in S are called singletons. We now give the following hypothesis.

Hypothesis 4.6 (Singleton Cell Hypothesis for f)**.** *Any leaf of an optimal protocol partition of the non-monotone Karchmer-Wigderson game corresponding to* f *includes at least one singleton cell.*

4.2 Singleton Cell Hypothesis for the Parity Function

It is difficult to prove the hypothesis in general because we should know about an optimal protocol partition. On the other hand, we can prove that the hypothesis is true for any Boolean function such that Khrapchenko's bound [1] gives the optimal formula size lower bound.

Theorem 4.7. *The singleton cell hypothesis is true for any Boolean function* f *for which Khrapchenko's bound gives the optimal formula size lower bound.*

Proof. To prove the theorem, we review Khrapchenko's bound. For a Boolean function f, we let $X = f^{-1}(1)$, $Y = f^{-1}(0)$ and consider the set of singleton cells

$$S = \{(\mathbf{x}, \mathbf{y}) \mid \mathbf{x} \in X,\ \mathbf{y} \in Y,\ d_H(\mathbf{x}, \mathbf{y}) = 1\}.$$

Then, Khrapchenko's bound is given by

$$L(f) \geq \frac{|S|^2}{|X| \cdot |Y|}.$$

To see this, we consider an optimal partition of $X \times Y$ into monochromatic rectangles $R_1, \cdots, R_t$. Let $s_1, \cdots, s_t$ be the numbers of singleton cells in $R_1, \cdots, R_t$, respectively. For a monochromatic rectangle R_i, let j be its corresponding index. That is, $x_j \neq y_j$ for each $(\mathbf{x}, \mathbf{y}) \in R_i$. For each row $\mathbf{x}$, there is at most one singleton cell $(\mathbf{x}, \mathbf{y})$ in the monochromatic rectangle because $x_k = y_k$ for all $k \neq j$. Similarly, for each column $\mathbf{y}$, there is at most one singleton cell $(\mathbf{x}, \mathbf{y})$. Thus, we have $s_i^2 \leq |R_i|$ for each i where $|R_i|$ is the number of cells in R_i. So, we have

$$\begin{aligned} |S|^2 &= \left(\sum_{i=1}^{t} s_i\right)^2 \leq t \cdot \sum_{i=1}^{t} s_i^2 \leq t \cdot \sum_{i=1}^{t} |R_i| \\ &= t \cdot |X| \cdot |Y|. \end{aligned}$$

The first inequality comes from the Cauchy-Schwarz inequality.

If Khrapchenko's bound gives the optimal formula size lower bound, it implies that all the inequalities are saturated. Hence, we have $s_i^2 = |R_i|$. This implies that there is at least one singleton cell for each monochromatic rectangle. □

Thus, the singleton hypothesis is true for the parity function with 2^k input bits. Actually, it is easy to observe that the number of monochromatic rectangles which appear in any optimal protocol partition of the communication matrix for the parity function is the same. This makes easy to prove the optimal formula size lower bound for the parity function.

4.3 Singleton Cell Hypothesis for the Majority Function

We cannot apply Theorem 4.7 to the majority function because Ueno [10] proved that its formula size lower bound improved from Khrapchenko's bound [1].

In the rest of the section, we prove that if the formula size of the majority function is larger than that of the parity function, then the singleton cell hypothesis is false for the majority function. The proof will be done by combining the following two lemmas.

Lemma 4.8. *If the singleton cell hypothesis is true for a monotone Boolean function* f*, then we have*

$$L(f) = L_m(f) = C^P(R_f(\min(f), \max(f))).$$

Proof. We call a monochromatic rectangle positive if every cell $(\mathbf{x}, \mathbf{y})$ in the rectangle contains the same index i such that $x_i = 1$ and $y_i = 0$, and negative if every cell $(\mathbf{x}, \mathbf{y})$ in the rectangle contains the same index i such that $x_i = 0$ and $y_i = 1$. There are no monochromatic rectangles which are both positive and negative. Theorem 4.3 shows a correspondence between a protocol partition and a formula in the sense that positive and negative monochromatic rectangles of the protocol partition correspond to positive and negative literals (i.e., variables and their negations), respectively.

Let f be a monotone Boolean function. Then, we have $f(\mathbf{x}) = 1$, $f(\mathbf{y}) = 0$, $x_i = 1$ and $y_i = 0$ for every singleton cell $(\mathbf{x}, \mathbf{y})$ whose index is i. Otherwise, f is not monotone. If a monochromatic rectangle contains the singleton cell $(\mathbf{x}, \mathbf{y})$, its corresponding index is also i. Note that any other index cannot be the corresponding index of the monochromatic rectangle because $(\mathbf{x}, \mathbf{y})$ is singleton. That is, $x_j = y_j$ for all $j \neq i$. Thus, the monochromatic rectangle is positive. If the singleton hypothesis is true, any

monochromatic rectangle in an optimal partition is positive. This partition can be transformed into a formula without negations (i.e., a monotone formula). Hence, we have $L(f) = L_m(f) = C^P(R_f(\min(f), \max(f)))$ by using Theorem 4.5. □

Lemma 4.9. *For any threshold function f with n input bits,*

$$C^P(R_f(\min(f), \max(f))) \leq L(\mathbf{PAR}_n).$$

Proof. The restricted communication matrix (i.e., $R_f(\min(f), \max(f))$) for a threshold function f is a submatrix of the whole non-monotone communication matrix $R_{\mathbf{PAR}_n}$ of the parity function. Thus, its protocol partition number gives a lower bound for $L(\mathbf{PAR}_n)$. □

From these two lemmas, we can show the following theorem.

Theorem 4.10. *If the singleton cell hypothesis is true for a threshold Boolean function f, then we have*

$$L_m(f) \leq L(\mathbf{PAR}_n).$$

Proof. Since a threshold function f is monotone, we have

$$L_m(f) = C^P(R_f(\min(f), \max(f)))$$

from Lemma 4.8. On the other hand, we know

$$C^P(R_f(\min(f), \max(f))) \leq L(\mathbf{PAR}_n)$$

from Lemma 4.9. Thus we have the theorem. □

Corollary 4.11. *If $L(\mathbf{PAR}_n) < L_m(\mathbf{MAJ}_n)$, the singleton cell hypothesis is false for the majority function.*

Many arguments as extensions of Khrapchenko's bound [1] focus on the structure of singleton cells. This corollary implies that, if the optimal formula size lower bound for the majority function is larger than that of the parity function, it is not sufficient to focus on only singleton cells and necessary to analyze the other cells for resolving the formula complexity of the majority function. Conversely, if the hypothesis is true for the majority function, we have improvements of its monotone and non-monotone formula size upper bounds to $O(n^2)$.

The singleton cell hypothesis is true for majority functions on small number of input bits for which we can search an optimal formula by computation search. In the case of the 3-bit majority function, the number of singleton cells which appear in an optimal protocol partition is either 1 or 2. In the case of the 5-bit majority function, we know that it can be either 1, 2, 3 or 4 in several variations of the optimal protocol partition. These also explain another reason of the difficulty to prove larger formula size for the majority function.

5. Concluding Remarks

In this paper, we have discussed candidate Boolean functions to prove a super-quadratic formula size lower bound by studying the recursive definitions of Boolean functions. We have also discussed the reason why it is difficult to prove a super-quadratic formula size for the majority function from the viewpoint of some hypothesis for non-monotone Karchmer-Wigderson game.

To show super-cubic formula size lower bounds, our result seems to be negative at first glance because a 9-bit Boolean function itself is already difficult to analyze, and so are recursive Boolean functions based on it. We still have difficulties to prove even super-quadratic formula size lower bounds for Boolean functions excepts Andreev function as revealed in [5]–[7].

Speaking of super-quadratic lower bounds, we have hope by using Boolean functions recursively defined from a 3-bit Boolean function. There also remains possibility for super-cubic lower bounds to define an analogue of the Andreev function replacing the parity function by a harder Boolean function of super-quadratic formula size.

Acknowledgments

The author is grateful to anonymous referees for their helpful comments to improve the presentation of the paper. This work is supported by Grant-in-Aid for Young Scientists (B) (JSPS KAKENHI Grant Number 24700010), Grant-in-Aid for Scientific Research on Innovative Areas (MEXT KAKENHI Grant Number 24106006), and the Hakubi Project of Kyoto University.

References

[1] V.M. Khrapchenko, "Complexity of the realization of a linear function in the case of π-circuits," Mathematical Notes, vol.9, pp.21–23, 1971.
[2] J. Håstad, "The shrinkage exponent of De Morgan formulas is 2," SIAM J. Comput., vol.27, no.1, pp.48–64, Feb. 1998.
[3] A.E. Andreev, "On a method for obtaining more than quadratic effective lower bounds for the complexity of π-scheme," Moscow University Mathematics Bulletin, vol.42, no.1, pp.63–66, 1987.
[4] S. Laplante, T. Lee, and M. Szegedy, "The quantum adversary method and classical formula size lower bounds," Computational Complexity, vol.15, no.2, pp.163–196, 2006.
[5] T. Lee, "A new rank technique for formula size lower bounds," Proc. 24th Annual Symposium on Theoretical Aspects of Computer Science (STACS 2007), Lect. Notes Comput. Sci. 4393, pp.145–156, 2007.
[6] P. Hrubeš, S. Jukna, A. Kulikov, and P. Pudlák, "On convex complexity measures," Theor. Comput. Sci., vol.411, pp.1842–1854, 2010.
[7] M. Karchmer, E. Kushilevitz, and N. Nisan, "Fractional covers and communication complexity," SIAM J. Discrete Math., vol.8, no.1, pp.76–92, Feb. 1995.
[8] E. Koutsoupias, "Improvements on Khrapchenko's theorem," Theor. Comput. Sci., vol.116, no.2, pp.399–403, Aug. 1993.
[9] P. Høyer, T. Lee, and R. Špalek, "Negative weights make adversaries stronger," Proc. 39th Annual ACM Symposium on Theory of Computing (STOC 2007), pp.526–535, 2007.
[10] K. Ueno, "A stronger LP bound for formula size lower bounds via clique constraints," Theor. Comput. Sci., vol.434, pp.87–97, May 2012.
[11] K. Ueno, "Breaking the rectangle bound barrier against formula size

lower bounds," Proc. 35th International Symposium on Mathematical Foundations of Computer Science (MFCS 2010), Lect. Notes Comput. Sci., vol.6281, pp.665–676, 2010.
[12] R. Raz and A. Wigderson, "Monotone circuits for matching require linear depth," J. ACM, vol.39, no.3, pp.736–744, July 1992.
[13] J. Edmonds, R. Impagliazzo, S. Rudich, and J. Sgall, "Communication complexity towards lower bounds on circuit depth," Computational Complexity, vol.10, no.3, pp.210–246, 2001.
[14] J. Håstad and A. Wigderson, "Composition of the universal relation," Advances in Computational Complexity Theory, pp.119–134, 1993.
[15] M. Karchmer, R. Raz, and A. Wigderson, "Super-logarithmic depth lower bounds via the direct sum in communication complexity," Computational Complexity, vol.5, no.3/4, pp.191–204, 1995.
[16] M.S. Paterson, N. Pippenger, and U. Zwick, "Optimal carry save networks," in Boolean function complexity, London Mathematical Society Lecture Note Series, vol.169, pp.174–201, Cambridge University Press, 1992.
[17] J. Radhakrishnan, "Better lower bounds for monotone threshold formulas," J. Computer and System Sciences, vol.54, no.2, pp.221–226, April 1997.
[18] M. Karchmer and A. Wigderson, "Monotone circuits for connectivity require super-logarithmic depth," SIAM J. Discrete Math., vol.3, no.2, pp.255–265, May 1990.

Kenya Ueno was born on October 20, 1982. He received the Bachelor of Science and Doctor of Information Science and Technology degrees from the University of Tokyo in 2005, and 2010, respectively. He is currently an assistant professor of the Hakubi Center for Advanced Research, Kyoto University. His research interests include computation complexity.