数理解析研究所講究録
第1799巻 2012年 13-20

2011年度冬のLAシンポジウム [2]

Formula Decomposition into Ternary Majorities (多数決3分木への論理式分解)

上野 賢哉*

概要

Any self-dual Boolean function can be decomposed into compositions of 3-bit majority functions. In this paper, we define a notion of a ternary majority formula, which is a ternary tree composed of nodes labeled by 3-bit majority functions and leaves labeled by literals. We study their complexity in terms of formula size. In particular, we prove upper and lower bounds for ternary majority formula size of the parity, majority and recursive majority functions. To prove the lower bounds, we analyze the largest separation between ternary majority formula size and DeMorgan formula size.

1 Introduction

A class of Boolean functions closed under compositions is called a Boolean clone. There are systematic studies on the relationship among Boolean clones known as Post's lattice [10]. (See also a survey [2] on Post's lattice with its applications.) According to the theory of Post's lattice, any monotone self-dual Boolean function can be decomposed into compositions of 3-bit majority functions. In other words, the 3-bit majority function is the universal gate for the class of monotone self-dual Boolean functions. On the other hand, the 3-bit Boolean function denoted by $(x_1 \wedge \neg x_2) \vee (\neg x_2 \wedge \neg x_3) \vee (\neg x_3 \wedge x_1)$ is the universal gate for the class of self-dual Boolean functions. This Boolean function is also representable by the 3-bit majority function with negations. Therefore any self-dual Boolean function can be also decomposed into compositions of 3-bit majority functions with negations.

Ibaraki and Kameda [5] developed a decomposition theory of monotone self-dual Boolean functions for the data structure called coteries which realize mutual exclusions in distributed systems. The theory was further investigated for self-dual Boolean functions in general by Bioch and Ibaraki [1], who gave the decomposition scheme of the 3-bit parity function into compositions of 3-bit majority functions. We will fully utilize this decomposition scheme in our results.

There are two kinds of formula models studied in the literature, $\mathbf{U}_2$-formula (DeMorgan formula) and its extension $\mathbf{B}_2$-formula (full binary basis formula). In this paper, we study a formula model $\mathbf{MAJ}_3$-formula (ternary majority formula) besides $\mathbf{U}_2$-formula and $\mathbf{B}_2$-formula. Every node of a $\mathbf{MAJ}_3$-formula is labeled by the 3-bit majority function while every node of a $\mathbf{U}_2$-formula and $\mathbf{B}_2$-formula is labeled by a 2-bit Boolean function.

Independently from any choice of formula models, proving formula size lower bounds is one of the most important problems in computational complexity theory as a weaker version of the circuit size lower bound problem and $\mathbf{P} \neq \mathbf{NP}$. A superpolynomial formula size lower bound for a function

*京都大学 次世代研究者育成センター (白川プロジェクト), kenya@kuis.kyoto-u.ac.jp

	$\mathbf{B}_2$-formula	$\mathbf{MAJ}_3$-formula	$\mathbf{U}_2$-formula
Parity	$\Theta(n)$	$O(n^{1.7329})$	$\Theta(n^2)$ [6]
		$\Omega(n^{1.2618})$	
Majority	$O(n^{3.13})$ [8, 9]	$O(n^{3.7925})$	$O(n^{4.57})$ [8, 9]
	$\Omega(n \log n)$ [4]	$\Omega(n^{1.2618})$	$\Omega(n^2)$ [6]
Recursive Majority	$O(n^{1.4650})$	$\Theta(n)$	$O(n^{1.4650})$ [7]
	$\Omega(n)$		$\Omega(n^{1.2618})$ [7]

図 1: Formula Size Upper and Lower Bounds

in some complexity class (e.g., **NP**) including $\mathbf{NC}^1$ implies a separation between the two complexity classes (e.g., $\mathbf{NC}^1 \neq \mathbf{NP}$). The complexity class $\mathbf{NC}^1$ is defined in terms of logarithm circuit depth, which turns out to be equivalent to polynomial formula size [12]. Therefore, the effect of the basis for formula complexity is also significant from the viewpoint of logical circuit design.

In this paper, we will prove the $\mathbf{MAJ}_3$-formula size upper and lower bounds in Section 3 and 5, respectively, as summarized in Figure 1. After the completion of this paper, we have noticed that the lower bound for the parity function is weaker than 1.33 of Chokler and Zwick [3] using the random restriction technique. Still, our lower bound method has merit in the sense that it can be applied for any Boolean function. To prove the lower bounds, we will show that the largest separation between $\mathbf{MAJ}_3$-formula and $\mathbf{U}_2$-formula complexity is at most $O(n^{\log_2 3})$ in Section 4. It can be regarded as analogue of Pratt's result [11], which showed the largest separation between $\mathbf{B}_2$-formula complexity and $\mathbf{U}_2$-formula complexity is at most $O(n^{\log_3 10})$.

We hope that a new technical discovery to clarify $\mathbf{MAJ}_3$-formula complexity will also shed light on resolving the stiff barrier against formula complexity of the existing models.

2 Definitions

In this section, we summarize definitions concerned with Boolean functions and formula size. We assume that the readers are familiar with the basics of these concepts together with the notations of O, o, Ω, ω and Θ.

2.1 Boolean Functions

In this paper, we consider the following Boolean functions. Through the paper, n means the number of input bits.

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

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

The majority function $\mathbf{MAJ}_{2l+1} : \{0,1\}^{2l+1} \mapsto \{0,1\}$ *on odd number of input bits is defined by*

$$\mathbf{MAJ}_{2l+1}(x_1, \cdots, x_n) = \begin{cases} 1 & (\sum_{i=1}^{n} x_i \geq l+1), \\ 0 & (\sum_{i=1}^{n} x_i \leq l). \end{cases}$$

The recursive majority function $\mathbf{RecMAJ}_3^h : \{0,1\}^{3^h} \mapsto \{0,1\}$ *is defined by*

$$\mathbf{RecMAJ}_3^h(x_1,\cdots,x_{3^h}) = \mathbf{MAJ}_3(\mathbf{RecMAJ}_3^{h-1}(x_1,\cdots,x_{3^{h-1}}), \mathbf{RecMAJ}_3^{h-1}(x_{3^{h-1}+1},\cdots,x_{2\cdot 3^{h-1}}), \mathbf{RecMAJ}_3^{h-1}(x_{2\cdot 3^{h-1}+1},\cdots,x_{3^h}))$$

with $\mathbf{RecMAJ}_3^1 = \mathbf{MAJ}_3$.

We will define another Boolean function right before it will appear. The notions of monotone and self-dual for Boolean function are defined as follows.

Definition 2.2 (Monotone and Self-Dual). *For Boolean vectors* $\vec{x} = (x_1,\cdots,x_n)$ *and* $\vec{y} = (y_1,\cdots,y_n)$, *we define* $\vec{x} \leq \vec{y}$ *if* $x_i \leq y_i$ *for all* $i \in \{1,\cdots n\}$. *A Boolean function* f *is called monotone if* $\vec{x} \leq \vec{y}$ *implies* $f(\vec{x}) \leq f(\vec{y})$ *for any* $\vec{x},\vec{y} \in \{0,1\}^n$. *A Boolean function* f *is called self-dual if* $f(x_1,\cdots,x_n) = \neg f(\neg x_1,\cdots,\neg x_n)$ *where* $\neg$ *denotes the negation, which flips* 1 *to* 0, *and* 0 *to* 1.

2.2 Formula Size

In this paper, we consider the following three formula models. For each model, a literal means either a variable x_i or the a negated variable $\neg x_i$ for some index i. Each formula is called monotone if it does not have negated variables. In the definition, the nodes $\wedge$ and $\vee$ mean the logical conjunction and disjunction, respectively.

Definition 2.3 (Formula Models). *A* $\mathbf{U}_2$*-formula is a binary tree with leaves labeled by literals and internal nodes labeled by* $\wedge$ *and* $\vee$. *A* $\mathbf{B}_2$ *-formula is a binary tree with leaves labeled by literals and internal nodes labeled by any of 2-bit Boolean functions such as* $\wedge$, $\vee$ *and* $\oplus_2$. *A* $\mathbf{MAJ}_3$*-formula is a ternary tree with leaves labeled by literals and internal nodes labeled by* $\mathbf{MAJ}_3$.

If we allow 0 and 1 in leaves along with literals, $\mathbf{MAJ}_3$-formulas can compute all the Boolean functions because $\mathbf{MAJ}_3(x_1,x_2,0) = x_1 \wedge x_2$ and $\mathbf{MAJ}_3(x_1,x_2,1) = x_1 \vee x_2$. So the 3-bit majority function with 0 and 1 can be regarded as a kind of the universal gate for all the Boolean functions. In this sense, $\mathbf{MAJ}_3$-formula is yet another natural extension of $\mathbf{U}_2$-formula like $\mathbf{B}_2$-formula. Even if we do not allow 0 and 1 in leaves $\mathbf{MAJ}_3$-formulas can compute all the self-dual Boolean functions. Furthermore, even if we allow only variables without negations, they can compute all the monotone self-dual Boolean functions.

The formula size for each formula model is defined as follows. For the convenience, we will not distinguish a Boolean function f and a formula computing f. We should note that $L_{\mathbf{MAJ}_3}(f)$ is defined only for self-dual Boolean functions while $L_{\mathbf{B}_2}(f)$ and $L_{\mathbf{U}_2}(f)$ are defined for all Boolean functions.

Definition 2.4 (Formula Size). *The size of a formula is its number of leaves for any formula model. We define the formula size of a Boolean function* f *as the size of the smallest formula computing* f. *We denote the size of* $\mathbf{U}_2$*-formula,* $\mathbf{B}_2$*-formula and* $\mathbf{MAJ}_3$*-formula of a Boolean function* f *by* $L_{\mathbf{B}_2}(f)$, $L_{\mathbf{U}_2}(f)$ *and* $L_{\mathbf{MAJ}_3}(f)$, *respectively. We will sometimes abbreviate* $L_{\mathbf{U}_2}(f)$ *to* $L(f)$ *for simplicity.*

Since we consider a ternary tree, the number of leaves is at most three times the number of nodes. So the $\mathbf{MAJ}_3$-formula model with 0 and 1 leaves is similar with a formula model with gates $\wedge$, $\vee$ and $\mathbf{MAJ}_3$. While this paper focuses on the $\mathbf{MAJ}_3$-formula model without 0 and 1, it might be interesting to ask the minimum number of 0 and 1 of $\mathbf{MAJ}_3$-formula as the measure of a kind of distance to self-dual Boolean functions for future work.

3 Ternary Majority Formula Size Upper Bounds

In this section, we prove $\mathbf{MAJ}_3$-formula size upper bounds of the parity and majority function. In both cases, the upper bounds are shown by utilizing the decomposition scheme of Bioch and Ibaraki [1] for the 3-bit parity function as

$$\oplus_3(x_1, x_2, x_3) = [1, [\bar{1}, \bar{2}, \bar{3}], [\bar{1}, 2, 3]]$$

where we use notations $[i, j, k] = \mathbf{MAJ}_3(x_i, x_j, x_k)$, $i = x_i$ and $\bar{i} = \neg x_i$. From the decomposition scheme, we obtain $L_{\mathbf{MAJ}_3}(\oplus_3) \leq 7$. We show that $\mathbf{MAJ}_3$-formula complexity is intermediate between $\mathbf{B}_2$-formula complexity and $\mathbf{U}_2$-complexity for both functions.

3.1 The Parity Function

In the case of $\mathbf{U}_2$-formula, we can construct a 2-bit parity formula $(x_1 \wedge \neg x_2) \vee (\neg x_1 \wedge x_2)$. By a similar construction, we can construct a $2n$-bit parity formula

$$\oplus_{2n}(x_1, \cdots, x_{2n}) = (f_1 \wedge \neg f_2) \vee (\neg f_1 \wedge f_2)$$

from two sorts of n-bit parity formulas $f_1 = \oplus_n(x_1, \cdots, x_n)$ and $f_2 = \oplus_n(x_{n+1}, \cdots, x_{2n})$ in general. Therefore we can prove an upper bound $L(\oplus_n) \leq n^2$ where $n = 2^h$ from a recursive inequality $L(\oplus_{2n}) \leq 4 \cdot L(\oplus_n)$.

In the case of $\mathbf{MAJ}_3$-formula, we can decompose the 3^h-bits parity function into a composition of a 3-bit parity function and three 3^{h-1}-bits parity functions. Thus we have a recursive inequality $L_{\mathbf{MAJ}_3}(\oplus_{3^h}) \leq 7 \cdot L_{\mathbf{MAJ}_3}(\oplus_{3^{h-1}})$ from the decomposition scheme of the 3-bit parity function. Solving this inequality straightforwardly, we can show an upper bound $L_{\mathbf{MAJ}_3}(\oplus_{3^h}) \in O(n^{\log_3 7}) \subseteq O(n^{1.7712})$. Actually we can give a better upper bound as follows.

Theorem 3.1 (See also [3]). $L_{\mathbf{MAJ}_3}(\oplus_{2l+1}) \in O(n^{1.7329})$ *where* $n = 2l + 1$.

Proof. For some constant α, we consider decomposition of the $(2\alpha + 1) \cdot m$-bit parity function into a composition of a 3-bit parity function with a m-bit parity function and two $\alpha \cdot m$-bit parity functions as follows.

$$\begin{aligned}
&\oplus_{(2\alpha+1)\cdot m}(x_1, \cdots, x_{(2\alpha+1)\cdot m}) = \\
&\quad \oplus_3(\oplus_m(x_1, \cdots, x_m), \\
&\qquad \oplus_{\alpha\cdot m}(x_{m+1}, \cdots, x_{(\alpha+1)\cdot m}), \\
&\qquad \oplus_{\alpha\cdot m}(x_{(\alpha+1)\cdot m+1}, \cdots, x_{(2\alpha+1)\cdot m})).
\end{aligned}$$

Here we can assume that $\alpha \cdot m$ is an odd integer by increasing or decreasing it at most 1. In this case, $(2\alpha + 1) \cdot m$ becomes also an odd integer if m is odd.

Let $S(n) = L_{\mathbf{MAJ}_3}(\oplus_n)$ and assume $S(n) \leq \beta \cdot n^\gamma$ for some constants $\beta, \gamma > 0$ for any odd number n. By increasing the value of β, the slight modification which makes $\alpha \cdot m$ be an odd integer can be ignored for the following estimation of γ. By using decomposition scheme of the 3-bit parity function,

$$\begin{aligned}
&S((1 + 2\alpha) \cdot m) \\
\leq &3 \cdot S(m) + 2 \cdot S(\alpha \cdot m) + 2 \cdot S(\alpha \cdot m) \\
\leq &(3 + 4 \cdot \alpha^\gamma) \cdot \beta \cdot n^\gamma.
\end{aligned}$$

It suffices to show that the last expression is bounded by $(1+2\alpha)^\gamma \cdot \beta \cdot n^\gamma$. Therefore we consider the minimum value of γ which satisfies

$$3 + 4 \cdot \alpha^\gamma \leq (1 + 2\alpha)^\gamma$$

by eliminating $\beta \cdot m^\gamma$ from both sides. We can verify that this inequality is satisfied when $\alpha = 1.73896$ and $\gamma = 1.73282$. □

3.2 The Majority Function

Our $\mathbf{MAJ}_3$-formula size upper bound for the majority function essentially relies on the general theory established by Paterson, Pippenger and

Zwick [8]. Their idea is based on construction of a carry save adder from a full adder of fixed size as building blocks. Here we consider a full adder $\mathbf{FA}_3$ from 3 bits to 2 bits. The first and second output bits y_1, y_2 of $\mathbf{FA}_3$ are the 3-bit parity and majority function, respectively.

In the case of $\mathbf{U}_2$-formula, the full adder $\mathbf{FA}_3$ can be constructed by $y_1 = (x_1 \wedge ((\neg x_2 \vee x_3) \wedge (x_2 \vee \neg x_3))) \vee (\neg x_1 \wedge ((x_2 \wedge \neg x_3) \vee (\neg x_2 \wedge x_3)))$ and $y_2 = (x_1 \wedge x_2) \vee ((x_1 \vee x_2) \wedge x_3)$. They defined the notion of the occurrence matrix. It summarizes the information of the number of occurrence in the formula. For example, the occurrence matrix of the above case is $M = \begin{pmatrix} 2 & 4 & 4 \\ 1 & 2 & 2 \end{pmatrix}$. In the first and second row of the matrix, each entry counts the number of occurrence of each variable in the first and second formula, respectively.

From the construction of an arbitrary fixed size full adder and its corresponding occurrence matrix, Paterson, Pippenger and Zwick [8] gave the following general upper bound method.

Theorem 3.2 ([8]). *Let M be an occurrence matrix of some full adder for some fixed basis and some Boolean function f. Let $\epsilon(M)$ be the maximum value of $\frac{1}{\gamma}$ such that $\|\vec{x}\|_\gamma \leq \|M \cdot \vec{x}\|_\gamma$ for any vector $\vec{x} \geq \vec{0}$ where $\|x\|_\gamma = (\sum_i |x_i|^\gamma)^{1/\gamma}$. Then $O(n^{\epsilon(M)+o(1)})$ gives a formula size upper bound for f on the fixed basis.*

By the theorem, we can derive a $\mathbf{U}_2$-formula size upper bound of $O(n^{4.70})$. Paterson and Zwick [9] gave a construction of the full adder from 11 bits to 4 bits and an improved upper bound of $O(n^{4.57})$.

In the case of $\mathbf{B}_2$-formula, Paterson, Pippenger and Zwick [8] proved a $\mathbf{B}_2$-formula size upper bound of $O(n^{3.21})$ improved to $O(n^{3.13})$ by Paterson and Zwick [9].

In the case of $\mathbf{MAJ}_3$-formula, the full adder $\mathbf{FA}_3$ can be constructed by $y_1 = [1, [\bar{1}, \bar{2}, \bar{3}], [\bar{1}, 2, 3]]$ and $y_2 = [1, 2, 3]$. So the corresponding occurrence matrix is $M = \begin{pmatrix} 3 & 2 & 2 \\ 1 & 1 & 1 \end{pmatrix}$. From this, we can obtain the following $\mathbf{MAJ}_3$-formula size upper bound for the majority function.

Theorem 3.3. $L_{\mathbf{MAJ}_3}(\mathbf{MAJ}_{2l+1}) \in O(n^{3.7925})$ *where* $n = 2l + 1$.

Proof. For the occurrence matrix, $M = \begin{pmatrix} 3 & 2 & 2 \\ 1 & 1 & 1 \end{pmatrix}$, the inequality $\|\vec{x}\|_\gamma \leq \|M \cdot \vec{x}\|_\gamma$ which appears in the theorem of Paterson, Pippenger and Zwick [8] can be interpreted as $p(a, b, c, \gamma) = (3 \cdot a + 2 \cdot b + 2 \cdot c)^\gamma + (a + b + c)^\gamma - a^\gamma - b^\gamma - c^\gamma \geq 0$. If $L_{\mathbf{MAJ}_3}(\mathbf{MAJ}_n) \in O(n^\gamma)$, there exists $a, b, c > 0$ such that $p(a, b, c, \gamma) < 0$. We set $a = 0.729608$, $b = c = 1$ and $1/\gamma = 3.7925$. Then, we have $p(a, b, c, \gamma) \approx -0.0000256657 < 0$. This certifies that the maximum value of $1/\gamma$ which satisfies $\|\vec{x}\|_\gamma \leq \|M \cdot \vec{x}\|_\gamma$ for any vectors $\vec{x} \geq \vec{0}$ is less than 3.7925. Thus we have obtained the upper bound. □

The optimality of the value γ can be confirmed by numerical analysis. That is, the minimum value of $p(a, b, c, \gamma) > 0$ for $\gamma = 3.7924$.

The best monotone $\mathbf{U}_2$-formula size upper bound for the majority function is $O(n^{5.3})$ by a probabilistic construction of Valiant [13]. Following the analysis of Valiant's construction replaced by balanced compositions of the 3-bit majority function with random variables, we can construct a monotone $\mathbf{MAJ}_3$-formula whose size is $O(n^{4.2945})$ ($\supseteq O(n^{\log_{3/2} 3 + \log_2 3})$). The size of its conversion into a monotone $\mathbf{U}_2$-formula is $O(n^{6.2913})$ ($\supseteq O(n^{\log_{3/2} 5 + \log_2 5})$) and larger than Valiant's bound.

4 Translation from Ternary Majority Formulas to De-Morgan Formulas

In this section, we analyze the relation between $\mathbf{MAJ}_3$-formula complexity and $\mathbf{U}_2$-formula complexity. The results in this section will be useful to derive a $\mathbf{MAJ}_3$-formula size lower bound from a $\mathbf{U}_2$-formula size lower bound for the same function as shown in Section 5. We begin with the following simple proposition.

Proposition 4.1. *$L_{\mathbf{MAJ}_3}(\mathbf{RecMAJ}_3^h) = n$ where $n = 3^h$ is the number of input bits.*

Proof. The upper bound $L_{\mathbf{MAJ}_3}(\mathbf{RecMAJ}_3^h) \leq n$ follows from the same construction as the definition. The lower bound $L_{\mathbf{MAJ}_3}(\mathbf{RecMAJ}_3^h) \geq n$ is also immediate because it depends on all the variables. □

From a majority formula $L(\mathbf{MAJ}_3) \leq 5$, we can recursively construct a formula for the recursive majority function whose size is 5^h. Therefore we have an upper bound $L(\mathbf{RecMAJ}_3^h) \leq 5^h$, i.e., $L(\mathbf{RecMAJ}_3^h) \in O(L_{\mathbf{MAJ}_3}(f)^{1.4650})$. Similarly, the best upper bound we know for $\mathbf{B}_2$-formula is also $L_{\mathbf{B}_2}(\mathbf{RecMAJ}_3^h) \leq 5^h$. The quantum adversary bound [7], which is useful to prove $\mathbf{U}_2$-formula size lower bounds, has a nice composition property written as $\mathbf{ADV}(f \cdot g) \geq \mathbf{ADV}(f) \cdot \mathbf{ADV}(g)$. It implies a formula size lower bound $4^h \leq L(\mathbf{RecMAJ}_3^h)$, i.e. $L(\mathbf{RecMAJ}_3^h) \in \Omega(L_{\mathbf{MAJ}_3}(f)^{1.2618})$.

We call the value γ an expansion factor from a $\mathbf{MAJ}_3$-formula into $\mathbf{U}_2$-formula for an arbitrary self-dual Boolean function f if $L(f) \in O((L_{\mathbf{MAJ}_3}(f))^\gamma)$. In the case of the recursive majority function, we can prove $\gamma \geq \log_3 5$ by solving $5 \cdot a^\gamma \leq (3a)^\gamma$ where $L_{\mathbf{MAJ}_3}(f_1) = L_{\mathbf{MAJ}_3}(f_2) = L_{\mathbf{MAJ}_3}(f_3) = a$. At first glance, the recursive majority function seems to have the largest expansion factor $\log_3 5$ from a $\mathbf{MAJ}_3$-formula into a $\mathbf{U}_2$-formula among all the $\mathbf{MAJ}_3$-formulas. Surprisingly, this is not true as we prove in the next lemma.

Lemma 4.2. *For any self-dual Boolean function f, $L(f) \in O(L_{\mathbf{MAJ}_3}(f)^{\log_2 3}) \subseteq O(L_{\mathbf{MAJ}_3}(f)^{1.5850})$.*

Proof. We are looking for the largest formula expansion from a $\mathbf{MAJ}_3$-formula into a $\mathbf{U}_2$-formula. Differently from the recursive majority function, the same variable might appear more than once in a ternary majority formula for an arbitrary Boolean function f. In this case, the expanded $\mathbf{U}_2$-formula can shrink more. So we can concentrate on the case in which all the variable appear exactly once. That is, $L_{\mathbf{MAJ}_3}(f) = n$.

We assume that $L(f) \leq \beta \cdot L_{\mathbf{MAJ}_3}(f)^\gamma$ for any self-dual Boolean function and consider an inductive argument. The expansion factor γ must satisfy an inequality $L(f) \leq 2 \cdot \beta \cdot (L_{\mathbf{MAJ}_3}(f_1))^\gamma + 2 \cdot \beta \cdot (L_{\mathbf{MAJ}_3}(f_2))^\gamma + \beta \cdot (L_{\mathbf{MAJ}_3}(f_3))^\gamma \leq \beta \cdot (L_{\mathbf{MAJ}_3}(f))^\gamma$ by looking at a formula expansion from a $\mathbf{MAJ}_3$-formula $f = \mathbf{MAJ}_3(f_1, f_2, f_3)$ into a $\mathbf{U}_2$-formula $f = (f_1 \wedge f_2) \vee ((f_1 \vee f_2) \wedge f_3)$. This expansion is processed from leaves to the root in a recursive way.

We can assume that $L_{\mathbf{MAJ}_3}(f_1) \leq L_{\mathbf{MAJ}_3}(f_2) \leq L_{\mathbf{MAJ}_3}(f_3)$ without loss of generality. We set $L_{\mathbf{MAJ}_3}(f_1) = a - b$, $L_{\mathbf{MAJ}_3}(f_2) = a + b$ and $L_{\mathbf{MAJ}_3}(f_3) = a + c$ where $a > b \geq 0$ and $c \geq b \geq 0$. In this case, we need to find the minimum value of γ which always satisfies $2 \cdot (a-b)^\gamma + 2 \cdot (a+b)^\gamma + (a+c)^\gamma \leq (3a+c)^\gamma$. So we set $p(a, b, c, \gamma) = (3a+c)^\gamma - (a+c)^\gamma - 2 \cdot (a+b)^\gamma - 2 \cdot (a-b)^\gamma$ and seek the minimum value of γ such that $p(a, b, c, \gamma) \geq 0$ for any $a > b \geq 0$ and $c \geq b > 0$.

First we fix a, b and γ and consider $q(\alpha) = (3+\alpha)^\gamma - (1+\alpha)^\gamma$ where $\alpha = \frac{c}{a}$ $(0 < \alpha)$. By the derivative $y' = \gamma \cdot x^{\gamma-1}$ of $y = x^\gamma$, $(3+\alpha)^\gamma$

increases more than $(1+\alpha)^\gamma$ whenever α slightly increases. So $q(\alpha)$ monotonically increases as α increases. To minimize $p(a,b,c,\gamma)$ for fixed γ, we would like to minimize $q(\alpha)$ and had better α be as small as possible. Hence we set $c = b$ because $c \geq b$.

Next we consider $r(\alpha,\gamma) = \frac{p(a,b,b,\gamma)}{a^\gamma} = (3+\alpha)^\gamma - 3\cdot(1+\alpha)^\gamma - 2\cdot(1-\alpha)^\gamma$ where $\alpha = \frac{b}{a}$ $(0 \leq \alpha < 1)$. Since $r(\alpha,\gamma) \geq 0$ for any α $(0 \leq \alpha < 1)$ implies $p(a,b,c,\gamma) \geq 0$ for any a, b and c, it suffices to seek the minimum γ which satisfies this condition.

It is easy to see that $\log_3 5$ is not the largest expansion factor because $r(1,\log_3 5) \approx -0.660928 < 0$ while $r(1,\log_3 5) = 0$. On the other hand, $\log_2 3$ seems to be a good candidate which is very near to the largest expansion factor because $r(0,\log_2 3) \approx 0.704522 > 0$ and $r(1,\log_2 3) \approx 1.77636 \times 10^{-15} > 0$. To confirm $r(0,\log_2 3) \geq 0$ for $0 \leq \alpha < 1$, it is sufficient to draw the graph of $r(\alpha,\log_2 3)$ $(0 \leq \alpha < 1)$ as shown in Figure 2. (Strictly speaking, it requires a rigorous analysis on $r(\alpha,\log_2 3)$, but we omit it in this paper.)

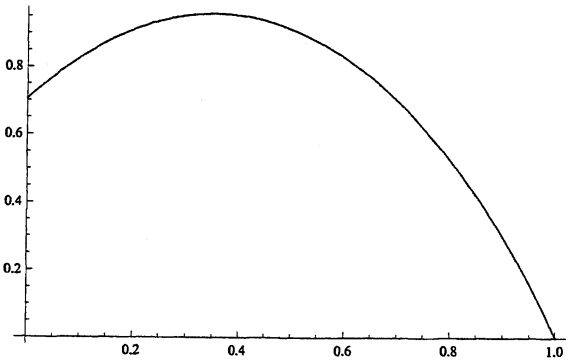

図 2: $r(\alpha,\log_2 3) = (3+\alpha)^{\log_2 3} - 3\cdot(1+\alpha)^{\log_2 3} - 2\cdot(1-\alpha)^{\log_2 3}$ $(0 \leq \alpha < 1)$.

Therefore the largest expansion factor is at most $\log_2 3$, which is given for a $\mathbf{MAJ}_3$-formula with $a - 1 = b = c$, i.e., $L_{\mathbf{MAJ}_3}(f_1) = 1$ and $L_{\mathbf{MAJ}_3}(f_2) = L_{\mathbf{MAJ}_3}(f_3)$ for each subtree. □

Pratt [11] proved $L_{\mathbf{U}_2}(f) \in O((L_{\mathbf{B}_2}(f))^{\log_3 10}) \subseteq O((L_{\mathbf{B}_2}(f))^{2.096})$. The exponent $\log_3 10$ is derived from the $\mathbf{U}_2$-formula size of 10 for the 3-bit parity function. The above lemma can be seen as an analogue of Pratt's bound [11] for the relation between $\mathbf{MAJ}_3$-formulas and $\mathbf{U}_2$-formulas.

5 Ternary Majority Formula Size Lower Bounds

In general, we can derive a $\mathbf{MAJ}_3$-formula size lower bound for an arbitrary Boolean function from a $\mathbf{U}_2$-formula size lower bound of the same function using Lemma 4.2 as follows.

Theorem 5.1. *For any self-dual Boolean function f such that $L(f) \in \Omega(n^c)$, $L_{\mathbf{MAJ}_3}(f) \in \Omega(n^{c/\log_2 3})$.*

Proof. By Lemma 4.2, an upper bound for $\mathbf{U}_2$-formula size expanded from a $\mathbf{MAJ}_3$-formula of size N is at most $O(N^{\log_2 3})$. This size must be not smaller than the formula size lower bound $L(f) \in \Omega(n^c)$. Therefore we have obtained the theorem. □

From $\mathbf{U}_2$-formula size lower bounds of $L(\oplus_n) \in \Omega(n^2)$ and $L(\mathbf{MAJ}_n) \in \Omega(n^2)$ by Khrapchenko [6], we have the following corollaries.

Corollary 5.2. $L_{\mathbf{MAJ}_3}(\oplus_{2l+1}) \in \Omega(n^{1.2618})$ *and* $L_{\mathbf{MAJ}_3}(\mathbf{MAJ}_{2l+1}) \in \Omega(n^{1.2618})$ *where* $n = 2l+1$.

Since $2/\log_2 3 = \log_3 4$, these lower bounds are equal to the $\mathbf{U}_2$-formula size lower bound for the recursive majority function accidentally. It seems to be difficult to give a matching $\mathbf{MAJ}_3$-formula size upper and lower bounds even for the parity function while we can obtain those for $\mathbf{U}_2$-formula and $\mathbf{B}_2$-formula. Both of them seem to be not tight and have room for further improvements.

6 Concluding Remarks

In this paper, we have introduced the notion of $\mathbf{MAJ}_3$-formula and have shown the upper and lower bounds for $\mathbf{MAJ}_3$-formula size of several Boolean functions. $\mathbf{MAJ}_3$-formula can be regarded as the most simplified form of threshold circuits as well as neural networks. Therefore there are possibilities to utilize techniques related with them. We hope that developing a new stream of studies on $\mathbf{MAJ}_3$-formulas will contribute a new progress revealing the complexity of itself as well as other existing formula models.

参考文献

[1] J. C. Bioch and T. Ibaraki. Decompositions of positive self-dual boolean functions. *Discrete Mathematics*, 140(1-3):23–46, 1995.

[2] E. Böhler, N. Creignou, S. Reith, and H. Vollmer. Playing with boolean blocks, part I: Post's lattice with applications to complexity theory. *ACM SIGACT News*, 34(4):38–52, 2003.

[3] H. Chockler and U. Zwick. Which bases admit non-trivial shrinkage of formulae? *Computational Complexity*, 10(1):28–40, 2001.

[4] M. J. Fischer, A. R. Meyer, and M. S. Paterson. $\Omega(n \log n)$ lower bounds on length of Boolean formulas. *SIAM Journal on Computing*, 11(3):416–427, Aug. 1982.

[5] T. Ibaraki and T. Kameda. A theory of coteries: Mutual exclusion in distributed systems. *IEEE Transactions on Parallel and Distributed Computing*, PDS-4(7):779–794, July 1993.

[6] V. M. Khrapchenko. Complexity of the realization of a linear function in the case of π-circuits. *Mathematical Notes*, 9:21–23, 1971.

[7] S. Laplante, T. Lee, and M. Szegedy. The quantum adversary method and classical formula size lower bounds. *Computational Complexity*, 15(2):163–196, 2006.

[8] M. S. Paterson, N. Pippenger, and U. Zwick. Optimal carry save networks. In *Boolean function complexity*, volume 169 of *London Mathematical Society Lecture Note Series*, pages 174–201. Cambridge University Press, 1992.

[9] M. S. Paterson and U. Zwick. Shallow circuits and concise formulae for multiple addition and multiplication. *Computational Complexity*, 3(3):262–291, 1993.

[10] E. L. Post. *The two-valued iterative systems of mathematical logic*, volume 5 of *Annals Mathematical Studies*. Princeton University Press, 1941.

[11] V. R. Pratt. The effect of basis on size of Boolean expressions. In *Proceedings of the 16th Annual Symposium on Foundations of Computer Science (FOCS 1975)*, pages 119–121. IEEE, 13–15 Oct. 1975.

[12] P. Spira. On time-hardware complexity tradeoffs for boolean functions. In *Proceedings of the 4th Hawaii Symposium on System Sciences*, pages 525–527, 1971.

[13] L. G. Valiant. Short monotone formulae for the majority function. *Journal of Algorithms*, 5(3):363–366, Sept. 1984.