Complexities of Graph-Based Representations for Elementary Functions

Shinobu Nagayama, *Member*, *IEEE*, and Tsutomu Sasao, *Fellow*, *IEEE*

Abstract—This paper analyzes complexities of decision diagrams for elementary functions such as polynomial, trigonometric, logarithmic, square root, and reciprocal functions. These real functions are converted into integer-valued functions by using fixed-point representation. This paper presents the numbers of nodes in decision diagrams representing the integer-valued functions. First, complexities of decision diagrams for polynomial functions are analyzed, since elementary functions can be approximated by polynomial functions. A theoretical analysis shows that binary moment diagrams (BMDs) have low complexity for polynomial functions. Second, this paper analyzes complexity of edge-valued binary decision diagrams (EVBDDs) for monotone functions, since many common elementary functions are monotone. It introduces a new class of integer functions, $\mathrm{M}p$-monotone increasing function, and derives an upper bound on the number of nodes in an EVBDD for the $\mathrm{M}p$-monotone increasing function. A theoretical analysis shows that EVBDDs have low complexity for $\mathrm{M}p$-monotone increasing functions. This paper also presents the exact number of nodes in the smallest EVBDD for the n-bit multiplier function, and a variable order for the smallest EVBDD.

Index Terms—Decision diagrams, MTBDDs, EVBDDs, BMDs, elementary functions, kth-degree polynomial functions, $\mathrm{M}p$-monotone increasing functions.

1 Introduction

Elementary functions [17] are widely used in various applications, such as computer graphics, digital signal processing, communication systems, robotics, astrophysics, and fluid physics. In these applications, as well as addition and multiplication, elementary functions are extensively used as a basic operation. The computation of elementary functions has been studied for more than 150 years [31], and various algorithms such as COordinate Rotation DIgital Computer (CORDIC) [2], [29] have been proposed. Nowadays, most of high-level programming languages have a library for elementary functions (e.g., math.h), and the users can evaluate elementary functions without caring their computation methods. However, with rapid spread of digital systems in recent years, hardware accelerators of elementary functions (elementary function generators) attract a lot of attention again, and systematic design and verification methods for elementary function generators have become important.

For design and verification of typical digital circuits, systematic methods using decision diagrams (DDs) have been established. However, DDs are not robust enough to represent all the functions compactly. The size of DDs varies drastically, depending on classes of functions. For example, binary DDs (BDDs) [1], [3], [16] can represent many practical logic functions compactly but cannot represent multiplier and hidden weighted bit (HWB) functions with reasonable size [4]. To compactly represent integer-valued functions such as adder and multiplier, many word-level DDs such as arithmetic transform DD (ACDD) [25], binary moment diagram (BMD) [5], multiplicative BMD (*BMD) [5], Kronecker multiplicative BMD (K*BMD) [8], and Taylor expansion diagram (TED) [6] have been proposed. However, each of these word-level DDs can represent some classes of functions compactly but cannot represent other classes of functions with reasonable size [10].

Thus, choosing a DD appropriate to a given class of functions is important, and analyzing complexities of DDs for various classes of functions is helpful to find an appropriate DD. Complexities of DDs for various functions have been analyzed before [30]. However, as for representations of elementary functions, not enough results have been reported [20], [24], [27]. Hence, in this paper, we analyze complexities of DDs for elementary functions to find compact graph-based representations for them. Since elementary functions include a wide range of functions, we analyze complexities by two approaches. First, by using polynomial expansions, we analyze complexities of DDs for elementary functions. Second, by using monotonicity, we analyze complexity for monotone elementary functions.

Results of this paper are useful to design elementary function generators. In fact, based on the results in this paper, we developed a systematic design method for elementary function generators [21]. Fig. 1 shows an architecture for the elementary function generators based on edge-valued BDD (EVBDD). In an EVBDD, we can evaluate a function by traversing the diagram and accumulating the values of traversed edges. Thus, this architecture consists only of a memory to store an

- *S. Nagayama is with the Department of Computer and Network Engineering, Hiroshima City University, 3-4-1 Ozuka-Higashi, Asa-Minami-Ku, Hiroshima-shi, Hiroshima-ken 731-3194, Japan. E-mail: s_naga@hiroshima-cu.ac.jp.*
- *T. Sasao is with the Department of Computer Science and Electronics, Kyushu Institute of Technology, Iizuka-shi, Fukuoka-ken 820-8502, Japan. E-mail: sasao@cse.kyutech.ac.jp.*

Manuscript received 23 Nov. 2007; revised 18 May 2008; accepted 14 July 2008; published online 6 Aug. 2008.
Recommended for acceptance by I. Markov.
For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number TC-2007-11-0611.
Digital Object Identifier no. 10.1109/TC.2008.134.

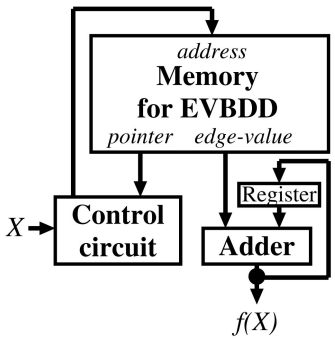


Fig. 1. EVBDD-based elementary function generator.

TABLE 1
Function Table for 3-Bit $\sin(X)$

X	$\sin(X)$
0.000	0.000
0.125	0.125
0.250	0.247
0.375	0.366
0.500	0.479
0.625	0.585
0.750	0.682
0.875	0.768

(a)

X	$f_b(X)$
0.000	0.000
0.001	0.001
0.010	0.010
0.011	0.011
0.100	0.100
0.101	0.101
0.110	0.101
0.111	0.110

(b)

X	$f(X)$
000	0
001	1
010	2
011	3
100	4
101	5
110	5
111	6

(c)

(a) Function table for $\sin(X)$. (b) Truth table for logic function $f_b(X)$. (c) Table for integer-valued function $f(X)$.

EVBDD, an accumulator (an adder and a register) for the edge values, and a control circuit to traverse the EVBDD. The complexity of EVBDD presented in this paper corresponds to the size of memory needed for this architecture. This architecture realizes various elementary functions easier and faster than CORDIC, a standard elementary function generator. Unlike function generators realizing approximate functions, the developed elementary function generators directly realize function tables, and thus, they are accurate and retain characteristics of elementary functions.

In this paper, we focus on complexities of DDs for elementary functions. Specifically, by improving the upper bounds on the number of nodes in DDs in [20] and [24], this paper derives the exact number of nodes for some functions and also proves new theorems. The results are useful for various applications using elementary functions in addition to the design methods for function generators.

This paper is organized as follows: Section 2 introduces a fixed-point representation, the arithmetic transform, and DDs used in this paper. Section 3 analyzes complexities of DDs for polynomial functions. This section also presents the number of nodes in the smallest EVBDD for the n-bit multiplier function and variable orders that produce the smallest EVBDDs. Section 4 analyzes complexity for monotone functions. Since many common elementary functions are monotone functions, this section introduces an Mp-monotone increasing function to show properties of elementary functions and presents complexity of EVBDDs for the Mp-monotone increasing function and its affine transformations.

2 Preliminaries

2.1 Number Representation

This section defines a number representation and describes how to convert real functions into integer-valued functions. First of all, we define various types of functions used in this paper.

Definition 1. *Let $B = \{0, 1\}$, Z be the set of the integers, and R be the set of the real numbers. An n-input m-output **logic function** is a mapping: $B^n \rightarrow B^m$, an **integer function** is $Z \rightarrow Z$, a (binary-input) **integer-valued function** is $B^n \rightarrow Z$, and a **real function** is $R \rightarrow R$.*

Definition 2. ***Elementary function** is a real function built from a combination of constants, a variable, four arithmetic operations, power functions, exponential functions, logarithmic functions, trigonometric functions, and inverse trigonometric functions.*

To represent real values, we use the following:

Definition 3. *A value X represented by the **binary fixed-point representation** is denoted by*

$$X = (x_{n\_int-1}\ x_{n\_int-2}\ \ldots\ x_1\ x_0.\ x_{-1}\ x_{-2}\ \ldots\ x_{-n\_frac})_2,$$

where $x_i \in \{0, 1\}\ \forall i$, n_int is the number of bits for the integer part, and n_frac is the number of bits for the fractional part of X. Note that

$$X = \sum_{i=-n\_frac}^{n\_int-1} 2^i x_i.$$

*Specially, n-**bit fixed-point representation** specifies that n bits are used to represent the value; that is, $n = n\_int + n\_frac$. In this paper, an n-**bit function** $f(X)$ means that the input variable X has n bits. And, $\{X\}$ denotes the unordered set of binary variables in X.*

By fixed-point representation, we can convert a real function into an n-input m-output logic function. The logic function can be converted into an integer-valued function by considering binary vectors as integers. That is, we can convert a real function into an integer-valued function: $B^n \rightarrow P_m$, where $P_m = \{0, 1, \ldots, 2^m - 1\}$. In this paper, elementary functions are converted into integer-valued functions by using fixed-point representation, unless stated otherwise. Specially, for polynomial functions, coefficients are also represented by fixed-point representation. And, for simplicity, x_0 denotes the least significant bit in the fixed-point representation of X.

Example 1. Table 1a is the function table for $\sin(X)$. By the 3-bit fixed-point representation, this function is converted into the logic function $f_b(X)$ in Table 1b. By representing the output vectors by integers, we have the integer-valued function $f(X)$ in Table 1c. In this paper, the 3-bit $\sin(X)$ denotes the integer-valued function $f(X)$ in Table 1c.

2.2 Arithmetic Transform

This section introduces the arithmetic transform and the arithmetic spectrum [25]. These are fundamentals of BMDs and are used to analyze complexities of BMDs.

First, define a matrix operation and some notations.

Definition 4. *Let $\boldsymbol{A}$ and $\boldsymbol{B}$ be $(n \times n)$ square matrices, where*

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix}.$$

*The **Kronecker product** of $\boldsymbol{A}$ and $\boldsymbol{B}$ is the $(n^2 \times n^2)$ matrix:*

$$\boldsymbol{A} \otimes \boldsymbol{B} = \begin{bmatrix} a_{11}\boldsymbol{B} & a_{12}\boldsymbol{B} & \dots & a_{1n}\boldsymbol{B} \\ a_{21}\boldsymbol{B} & a_{22}\boldsymbol{B} & \dots & a_{2n}\boldsymbol{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1}\boldsymbol{B} & a_{n2}\boldsymbol{B} & \dots & a_{nn}\boldsymbol{B} \end{bmatrix}.$$

Definition 5. *Given a matrix $\boldsymbol{M}$, the **transposed matrix $\boldsymbol{M^t}$** is obtained by interchanging the rows and columns. For an n-bit integer-valued function $f(X)$, the **function vector $\boldsymbol{F}$** is the column vector of the function values $\boldsymbol{F} = [f(0), f(1), \dots, f(2^n - 1)]^t$.*

Using these notations, we define the arithmetic transform and the arithmetic spectrum as follows:

Definition 6. *Let $\mathcal{A}(n)$ be the **arithmetic transform matrix** defined by*

$$\mathcal{A}(n) = \bigotimes_{i=1}^{n} \mathcal{A}(1), \qquad \mathcal{A}(1) = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix},$$

*where the addition and the multiplication are done in integer. For an integer-valued function f given by the function vector $\boldsymbol{F}$, the **arithmetic spectrum** $\mathcal{A}_f = [a_0, a_1, \dots, a_{2^n-1}]^t$ is*

$$\mathcal{A}_f = \mathcal{A}(n)\boldsymbol{F}.$$

*Each a_i in the spectrum is called the **arithmetic coefficient**.*

Example 2. Consider the 1-bit adder function $f(x_1, x_2) = x_1 + x_2$. The function vector is $\boldsymbol{F} = [0, 1, 1, 2]^t$. The arithmetic spectrum is

$$\mathcal{A}_f = \mathcal{A}(2)\boldsymbol{F} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}.$$

Similarly, we define the inverse arithmetic transform as follows:

Definition 7. *Let $\mathcal{A}^{-1}(n)$ be the **inverse arithmetic transform matrix** defined by*

$$\mathcal{A}^{-1}(n) = \bigotimes_{i=1}^{n} \mathcal{A}^{-1}(1), \qquad \mathcal{A}^{-1}(1) = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}.$$

Definition 8. *In a symbolic representation,*

$$\mathcal{A}^{-1}(1) = [\,1 \quad x_i\,].$$

*Therefore, the **inverse arithmetic transform** is defined as*

$$f = \boldsymbol{X_a}\mathcal{A}_f, \qquad \boldsymbol{X_a} = \bigotimes_{i=1}^{n} [\,1 \quad x_i\,].$$

Example 3. By the inverse arithmetic transform from the arithmetic spectrum obtained in Example 2, the integer-valued function f is represented as follows:

$$f = \boldsymbol{X_a}\mathcal{A}_f = [\,1 \quad x_2 \quad x_1 \quad x_1x_2\,] \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$
$$= x_1 + x_2.$$

From Definitions 7 and 8, we can see that an integer-valued function $f(X)$ can be represented by the arithmetic spectrum and the inverse arithmetic transform as follows:

Lemma 1. *Using $\mathcal{A}^{-1}(1)$ and $\mathcal{A}(1)$, an integer-valued function f is represented as follows:*

$$\begin{aligned} f &= \mathcal{A}^{-1}(1)\mathcal{A}(1)\boldsymbol{F} = [\,1 \quad x_i\,] \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \end{bmatrix} \\ &= [\,1 \quad x_i\,] \begin{bmatrix} f_0 \\ f_1 - f_0 \end{bmatrix} \\ &= f_0 + x_i(f_1 - f_0), \end{aligned} \tag{1}$$

*where $f_0 = f(x_i = 0)$, $f_1 = f(x_i = 1)$. Equation (1) is the **arithmetic transform expansion** (also called A-expansion or moment decomposition [5]). The **arithmetic expression** for f is obtained by the arithmetic transform expansion. The arithmetic coefficients correspond to coefficients of the arithmetic expression for f.*

2.3 Decision Diagrams

This section defines DDs used in this paper. For more details on definition and reduction rules of each DD, see [5], [22], [30], and [32].

Definition 9. *A **BDD** [1], [3], [16] is a rooted directed acyclic graph (DAG) representing a logic function. The BDD is obtained by repeatedly applying the Shannon expansion $f = \overline{x_i} f_0 + x_i f_1$ to the logic function, where $f_0 = f(x_i = 0)$, and $f_1 = f(x_i = 1)$. It consists of two terminal nodes representing function values 0 and 1, respectively, and nonterminal nodes representing input variables. Each nonterminal node has two unweighted outgoing edges, 0-edge and 1-edge, which correspond to the values of the input variables. Both terminal nodes have no outgoing edges. A reduced ordered BDD (ROBDD) is obtained by fixing the variable order in a BDD and by applying the following two reduction rules:*

1. *Share equivalent subgraphs.*
2. *Delete nonterminal nodes whose both outgoing edges point to the same node.*

In this paper, a BDD means an ROBDD.

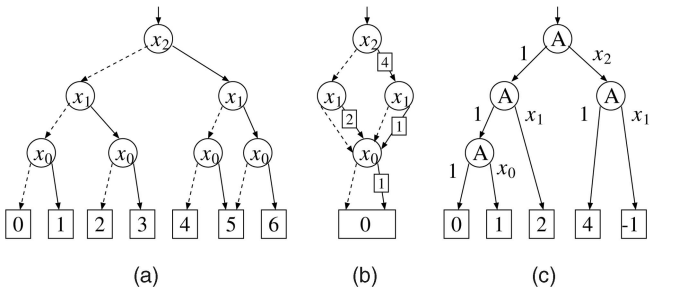


Fig. 2. Three types of DDs for 3-bit $\sin(X)$. (a) MTBDD. (b) EVBDD. (c) BMD.

Definition 10. *A **multiterminal BDD (MTBDD)** [7] is an extension of a BDD and represents an integer-valued function. In the MTBDD, the terminal nodes are labeled by integers.*

Definition 11. *An **EVBDD** [14] is a variant of the BDD and represents an integer-valued function. The EVBDD is obtained by repeatedly applying the expansion $f = \overline{x_i} f_0 + x_i(f_1' + \alpha)$ to the integer-valued function, where $f_0 = f(x_i = 0)$, $f_1 = f(x_i = 1) = f_1' + \alpha$, and α is the constant term of f_1. The EVBDD consists of only one terminal node representing 0 and nonterminal nodes with 1-edges having integer weights α. In the EVBDD, 0-edges always have zero weights, and the incoming edge into the root node can have a nonzero weight that represents the constant term of the original function. In the EVBDD, the following two reduction rules are applied:*

1. *Share equivalent subgraphs.*
2. *Delete nonterminal nodes whose both outgoing edges point to the same node and whose edge weight is $\alpha = 0$.*

In a reduced EVBDD, each node represents a distinct subfunction.

Definition 12. *A **BMD** [5] is a rooted DAG representing an integer-valued function. The BMD is obtained by repeatedly applying the arithmetic transform expansion $f = f_0 + x_i(f_1 - f_0)$ to the integer-valued function, where $f_0 = f(x_i = 0)$, and $f_1 = f(x_i = 1)$. f_0 and $x_i(f_1 - f_0)$ are called the **constant moment** and the **linear moment**, respectively. The BMD consists of terminal nodes representing the arithmetic coefficients and nonterminal nodes representing the arithmetic transform expansions. Each nonterminal node has two edges corresponding to two terms: constant moment and linear moment in the arithmetic transform expansion. In the BMD, the following two reduction rules are applied:*

1. *Share equivalent subgraphs.*
2. *Delete nonterminal nodes whose outgoing edge representing the linear moment points to the terminal node representing 0.*

Example 4. Figs. 2a, 2b, and 2c show the MTBDD, the EVBDD, and the BMD for the 3-bit $\sin(X)$ in Table 1c. In Figs. 2a and 2b, each nonterminal node labeled by an input variable represents the Shannon expansion with respect to the input variable. And, each nonterminal node has two edges: a 0-edge, denoted by a dashed line, and a 1-edge, denoted by a solid line. Note that the EVBDD has weighted 1-edges. In Fig. 2c, each node labeled by "A" represents the arithmetic transform expansion. And, each node has two edges: an edge for constant moment, labeled by "1," and an edge for linear moment, labeled by an input variable. In the MTBDD, to evaluate the function, we traverse the MTBDD from the root node to a terminal node according to the input values and obtain the function value (an integer) from the terminal node. In the EVBDD, we obtain the function value as the sum of the weights for the edges traversed from the root node to the terminal node. And, in the BMD, we obtain the function value by computing the arithmetic transform expansion $f = f_0 + x_i(f_1 - f_0)$ recursively at each nonterminal node.

3 Complexities of Design Diagrams for Polynomial Functions

Differentiable elementary functions can be expanded into polynomial functions such as Taylor series. To analyze complexities for elementary functions, this section shows upper bounds on the number of nodes in three types of DDs for kth-degree polynomial functions $c_k X^k + c_{k-1}X^{k-1} + \cdots + c_0$, where each coefficient c_i is an integer. By considering scaling factors of $X^i \; \forall i$ as polynomial coefficients, we can convert fixed-point polynomial functions into integer polynomial functions. For example, for a 2-bit polynomial function $f(X) = X^2 + X$, when $X = 0.75 = (0.11)_2$, $f(0.75) = 1.3125 = (1.0101)_2$. If the scaling factors are not considered when converting into an integer polynomial, then we have $X = (11)_2 = 3$, and $f(3) = 3^2 + 3 = 12 = (1100)_2$. It produces a wrong binary vector. To produce a correct binary vector, we can use an integer polynomial: $X^2 + 2^2 X$ considering the scaling factors. Thus, in this section, we analyze complexities for integer polynomial functions.

Example 5. Consider a polynomial function $f(X) = 5X^2 + 7X + 2$. Fig. 3a shows the function vector $\boldsymbol{F}$ and the arithmetic spectrum $\mathcal{A}_f$. Figs. 3b, 3c, and 3d show an MTBDD, an EVBDD, and a BMD for $f(X)$, respectively. In the MTBDD, terminal nodes represent the function vector. On the other hand, in the BMD, terminal nodes represent the arithmetic spectrum, and each path from the root node to a terminal node represents each term of the arithmetic expression for f:

$$320x_3x_2 + 160x_3x_1 + 80x_3x_0 + 376x_3 + 80x_2x_1 + 40x_2x_0 + 108x_2 + 20x_1x_0 + 32x_1 + 12x_0 + 2.$$

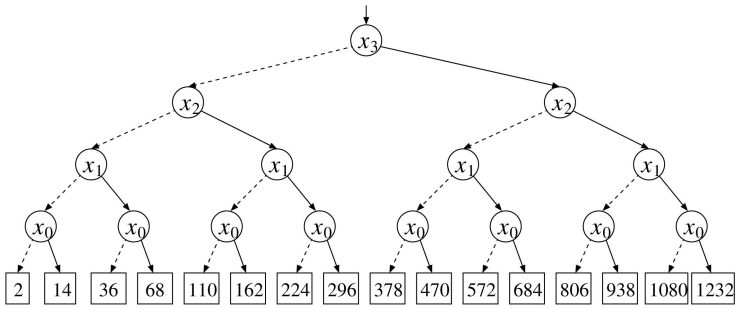
3.1 Complexity of MTBDDs for Polynomial Functions

Since terminal nodes in an MTBDD represent the distinct values in the function vector, the number of terminal nodes is equal to the number of distinct function values. Thus, an upper bound on the number of nodes in an MTBDD is derived from the following.

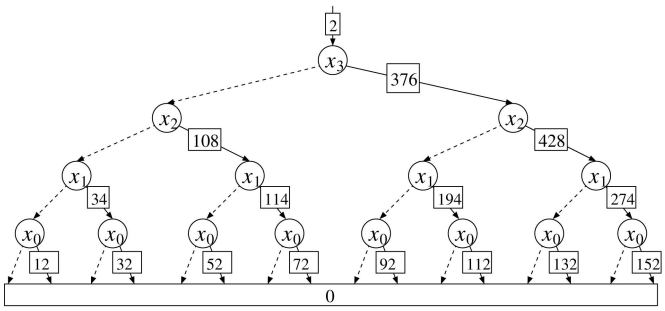
Lemma 2. *For an n-bit function $f(X)$, if $f(X)$ is an* injection, *i.e., the relation $\alpha \neq \beta \rightarrow f(\alpha) \neq f(\beta)$ holds on any α and β, then the total number of nonterminal nodes and terminal nodes*

X	$\boldsymbol{F}$	$\mathcal{A}_f$	X	$\boldsymbol{F}$	$\mathcal{A}_f$
0000	2	2	1000	378	376
0001	14	12	1001	470	80
0010	36	34	1010	572	160
0011	68	20	1011	684	0
0100	110	108	1100	806	320
0101	162	40	1101	938	0
0110	224	80	1110	1,080	0
0111	296	0	1111	1,232	0

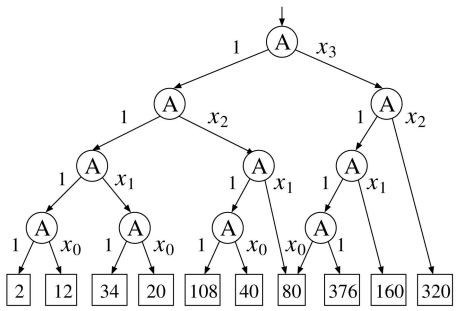
(a)



(b)



(c)



(d)

Fig. 3. Three types of DDs for $f(X) = 5X^2 + 7X + 2$. (a) Function vector and arithmetic spectrum. (b) MTBDD. (c) EVBDD. (d) BMD.

in the MTBDD for $f(X)$ is $2^{n+1} - 1$ independently of variable orders.

Proof. The number of distinct values of X is 2^n, and the number of distinct values of $f(X)$ is 2^n. Thus, the MTBDD for $f(X)$ is the complete binary tree, and the number of nodes in the tree is $2^{n+1} - 1$. □

From Lemma 2, we can see that an MTBDD for the function $f(X) = X$ requires exactly $2^{n+1} - 1$ nodes. On the other hand, if there exist two distinct values α and β satisfying $f(\alpha) = f(\beta)$, then the total number of nodes in the MTBDD for f is smaller than $2^{n+1} - 1$. Therefore, for polynomial functions, we have the following tight upper bound.

Theorem 1. *For an n-bit kth-degree polynomial function $f(X) = c_k X^k + c_{k-1} X^{k-1} + \cdots + c_0$, the total number of nodes in the MTBDD for $f(X)$ is at most $2^{n+1} - 1$. When $c_i > 0\ \forall i$, the number of nodes in the MTBDD for $f(X)$ is exactly $2^{n+1} - 1$, independently of variable orders. For also $f(X) = X^k$, the MTBDD requires exactly $2^{n+1} - 1$ nodes, independently of variable orders.*

3.2 Complexity of EVBDDs for Polynomial Functions

In this section, we first analyze the number of nodes in an EVBDD for the n-bit multiplier function $X \times Y$ (n bits $\times$ n bits). Then, by using the same approach, we derive the upper bounds on the number of nodes in EVBDDs for X^k and kth-degree polynomial functions.

To derive the number of nodes in an EVBDD for the n-bit multiplier function, we partition the EVBDD into two parts: the upper and the lower parts, as shown in Fig. 4. By enumerating the number of distinct cofactors for each part, we derive the exact number of nodes in the EVBDD.

The number of distinct cofactors for the upper part is obtained by the following:

Lemma 3. *For the n-bit multiplier function $X \times Y$, let P_u be an unordered set of u binary variables arbitrarily chosen from $\{X\} = \{x_{n-1}, x_{n-2}, \ldots, x_0\}$ and $\{Y\} = \{y_{n-1}, y_{n-2}, \ldots, y_0\}$, where $\{X\} \setminus P_u \neq \emptyset$ and $\{Y\} \setminus P_u \neq \emptyset$. Then, the number of distinct cofactors with respect to the binary variables in P_u is 2^u. And, the difference between any two cofactors: $f_i - f_j$ is not a constant function.*

Proof. Let cofactors with respect to two assignments to P_u be

$$\begin{aligned} f_i &= (A + X')(B + Y') \\ &= AB + AY' + BX' + X'Y', \end{aligned} \tag{2}$$

$$\begin{aligned} f_j &= (A' + X')(B' + Y') \\ &= A'B' + A'Y' + B'X' + X'Y', \end{aligned} \tag{3}$$

respectively, where A and A' are values assigned to $P_u \cap \{X\}$; B and B' are values assigned to $P_u \cap \{Y\}$;

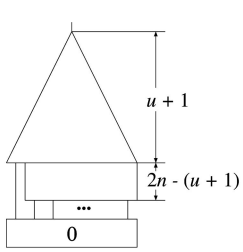


Fig. 4. Partition of EVBDD for n-bit multiplier function.

$\{X'\} = \{X\} \setminus P_u$; and $\{Y'\} = \{Y\} \setminus P_u$. Then, (2) and (3) are equal if and only if $A = A'$ and $B = B'$ (i.e., two assignments are identical). Thus, for each assignment, there exists a distinct cofactor. Since the number of distinct assignments to P_u is 2^u, the number of distinct cofactors is also 2^u. And, it is clear that the difference between (2) and (3) is not a constant function when $A \neq A'$ or $B \neq B'$ (i.e., either one pair of assignments is different). □

The number of distinct cofactors for the lower part is obtained by the following:

Lemma 4. *For the n-bit multiplier function $X \times Y$, let P_l be an unordered set of l binary variables arbitrarily chosen from $\{X\}$ and $\{Y\}$, where $l \geq n$, and either $\{X\} \subseteq P_l$ or $\{Y\} \subseteq P_l$ holds. If constant terms of cofactors are ignored, then the number of distinct cofactors with respect to the binary variables in P_l is $2^n - 1$.*

Proof. We prove only the case where $\{X\} \subseteq P_l$. (The proof for the case where $\{Y\} \subseteq P_l$ is similar.) Let cofactors with respect to two distinct assignments to P_l be

$$A(B + Y') = AB + AY', \tag{4}$$

$$A(B' + Y') = AB' + AY', \tag{5}$$

respectively, where A is a value assigned to X, B and B' are values assigned to $P_l \cap \{Y\}$, $B \neq B'$, and $\{Y'\} = \{Y\} \setminus P_l$. Then, (4) and (5) are equal if constant terms AB and AB' are ignored. The number of distinct assignments to X is 2^n, but the cofactor is the constant 0 when $X = 0$. Therefore, the lemma holds. □

From Lemmas 3 and 4, we derive the exact number of nodes in an EVBDD for a given variable order.

Lemma 5. *Given a variable order of EVBDD for the n-bit multiplier function $X \times Y$, let P_u be the unordered set of u binary variables from the top (i.e., root) to the uth in the variable order, where u is the largest integer satisfying $\{X\} \setminus P_u \neq \emptyset$ and $\{Y\} \setminus P_u \neq \emptyset$. Then, the number of nodes in the EVBDD is $2^{u+1} + (2^n - 1)\{2n - (u + 1)\}$.*

Proof. Suppose that the EVBDD is partitioned into the upper and the lower parts as shown in Fig. 4, where the upper part has $u + 1$ binary variables. We begin with the upper part. From Lemma 3, the number of distinct cofactors for each i of $0 \leq i \leq u$ is 2^i. Since each cofactor is represented by a nonterminal node, the upper part has $2^0 + 2^1 + \cdots + 2^u = 2^{u+1} - 1$ nodes.

Next, we consider the lower part. For cofactors in the lower part, their constant terms are represented as edge weights in the EVBDD. Thus, from Lemma 4, the number of distinct cofactors represented by nonterminal nodes is $2^n - 1$ for each j of $u + 2 \leq j \leq 2n$. Therefore, the lower part has $(2^n - 1)\{2n - (u + 1)\}$ nodes.

By summing the number of nonterminal nodes in each part and one for the terminal node, we have the lemma. □

In Lemma 5, when $u = n - 1$, the number of nodes in an EVBDD is the minimum. Thus, by substituting $n - 1$ into u in Lemma 5, we have the following theorem.

Theorem 2. *For the n-bit multiplier function, the number of nodes in the smallest EVBDD is $2^n(n + 1) - n$.*

On the number of nodes in an EVBDD for the n-bit multiplier function, only an asymptotic lower bound has been presented [30]. As far as we know, the exact smallest number of nodes in the EVBDD is presented for the first time in Theorem 2.

As for the orderings of the variables, we have the following.

Lemma 6. *For the n-bit multiplier function, the number of different orderings of the input variables is $(2n)!$. They are partitioned into n groups, where variable orders in each group produce EVBDDs with the same number of nodes, and variable orders in different groups produce EVBDDs with different number of nodes.*

Proof. Since Lemma 5 is based on the unordered sets $\{X\}$, $\{Y\}$, and P_u, the number of nodes in an EVBDD is invariant under the permutations of the variables in $\{X\}$, $\{Y\}$, and P_u. It depends only on the value of u (i.e., the position where binary variables are partitioned into upper and lower parts in Fig. 4). From Lemma 5, the range of possible values for u is $n - 1 \leq u \leq 2n - 2$, and thus the number of distinct values of u is n. And, for different u, the numbers of nodes in the EVBDDs are different. Therefore, we have the lemma. □

From Lemma 6, we can find the smallest EVBDD for the n-bit multiplier function by checking only n variable orders where the values of u are different. And, the variable orders satisfying $u = n - 1$ produce the smallest EVBDD. When $u = n - 1$, $\{X\}$ and $\{Y\}$ are partitioned into upper and lower parts. In this case, the number of nodes is independent of the variable order in $\{X\}$ and the variable order in $\{Y\}$. Also, the multiplication is a commutative operation, and so the results do not change even if X and Y are interchanged. Thus, we have the following:

Theorem 3. *For the n-bit multiplier function, the number of different variable orders that produce the smallest EVBDD is $2(n!)^2$.*

One of the variable orders for the smallest EVBDD is $(x_{n-1}, x_{n-2}, \ldots, x_0, y_{n-1}, y_{n-2}, \ldots, y_0)$. On the other hand, when the variable order is $(x_{n-1}, y_{n-1}, x_{n-2}, y_{n-2}, \ldots, x_0, y_0)$, the number of nodes is the largest.

Similarly to the multiplier function, by enumerating the number of distinct cofactors, we derive the exact number of nodes for the function $f(X) = X^k$.

Lemma 7. *For the n-bit function $f(X) = X^k$ $(k > 1)$, let P_u be an unordered set of u binary variables arbitrarily chosen from $\{X\}$, where $u < n$. Then, the number of distinct cofactors with respect to the binary variables in P_u is 2^u. And, the difference between any two cofactors: $f_i - f_j$ is a nonconstant function.*

Proof. Let cofactors with respect to two distinct assignments to P_u be

$$f_i = (A + X')^k = A^k + \binom{k}{1} A^{k-1}X' + \cdots + X'^k,$$
$$f_j = (A' + X')^k = A'^k + \binom{k}{1} A'^{k-1}X' + \cdots + X'^k,$$

respectively, where A and A' are values assigned to P_u, $A \neq A'$, and $\{X'\} = \{X\} \setminus P_u$. Then, it is clear that these two cofactors are distinct, and the difference between them is not a constant function. Thus, for each assignment, there exists a distinct cofactor. Since the number of distinct assignments to P_u is 2^u, the number of distinct cofactors is also 2^u. □

Theorem 4. *For the n-bit function $f(X) = X^k$ $(k > 1)$, the number of nodes in the EVBDD is 2^n and is independent of variable orders.*

Proof. From Lemma 7, the number of distinct cofactors for each i of $0 \leq i \leq n-1$ is 2^i. Thus, the EVBDD consists of $2^n - 1$ nonterminal nodes and one terminal node. That is, the total number of nodes in the EVBDD is 2^n. □

By extending Theorem 4, we derive the tight upper bound for the polynomial function.

Theorem 5. *For an n-bit kth-degree polynomial function $f(X) = c_k X^k + c_{k-1}X^{k-1} + \cdots + c_0$, the number of nodes in the EVBDD is at most 2^n. When $k > 1$ and $c_i > 0$ $\forall i$, the number of nodes in the EVBDD for $f(X)$ is exactly 2^n and is independent of variable orders.*

Proof. Similarly to the proof for Lemma 7, we can prove that for polynomial functions, the number of distinct cofactors with respect to u binary variables is at most 2^u, where $u < n$. Thus, the EVBDD has at most 2^n nodes. When $c_i > 0$ $\forall i$, polynomial functions are injections. Thus, when $k > 1$, the number of distinct cofactors with respect to u binary variables is exactly 2^u, and the number of nodes in the EVBDD is exactly 2^n. □

Theorem 6. *For an n-bit first-degree polynomial function $f(X) = c_1 X + c_0$ $(c_1 \neq 0)$, the number of nodes in the EVBDD is exactly $n + 1$ and is independent of variable orders.*

Proof. It is clear from the definitions of fixed-point representation and EVBDDs. □

3.3 Complexity of BMDs for Polynomial Functions

Since the terminal nodes in a BMD represent the arithmetic spectrum, the number of terminal nodes corresponds to the number of distinct arithmetic coefficients. In this section, from the number of nonzero arithmetic coefficients, we derive upper bounds on the number of nodes in BMDs for X^k and for a kth-degree polynomial function.

First, we derive the exact number of nonzero arithmetic coefficients for $f(X) = X^k$.

Lemma 8. *For the n-bit function $f(X) = X^k$, the number of nonzero arithmetic coefficients is*

$$\sum_{i=1}^{k} \binom{n}{i}.$$

Proof. From the property of the arithmetic transform, the number of nonzero arithmetic coefficients is equal to the number of terms in the expression that is obtained by expanding and rearranging the following:

$$X^k = (2^{n-1}x_{n-1} + 2^{n-2}x_{n-2} + \cdots + 2^1 x_1 + 2^0 x_0)^k,$$

where $x_i^2 = x_i$ $(i = 0, 1, 2, \ldots, n-1)$ because x_i is a Boolean variable. In the expanded and rearranged expression, the number of terms with a single literal (i.e., terms of x_i) is $\binom{n}{1}$. And, the number of terms with two literals (i.e., terms of $x_i x_j$ $(i < j)$) is $\binom{n}{2}$. Similarly, the number of terms with k literals is $\binom{n}{k}$. Therefore, the total number of terms is $\sum_{i=1}^{k} \binom{n}{i}$. □

Example 6. Consider the 4-bit function $f(X) = X^2$. From $X = 8x_3 + 4x_2 + 2x_1 + x_0$, we have

$$\begin{aligned} X^2 &= (8x_3 + 4x_2 + 2x_1 + x_0)^2 \\ &= \left(64x_3^2 + 16x_2^2 + 4x_1^2 + x_0^2\right) \\ &\quad + 2(32x_2x_3 + 16x_1x_3 + 8x_0x_3 \\ &\qquad + 8x_1x_2 + 4x_0x_2 + 2x_0x_1) \\ &= 64x_3 + 16x_2 + 4x_1 + x_0 + 64x_2x_3 + 32x_1x_3 \\ &\quad + 16x_0x_3 + 16x_1x_2 + 8x_0x_2 + 4x_0x_1. \end{aligned}$$

Note that this is the arithmetic expression for $f(X)$ and has 10 terms. The number of nonzero arithmetic coefficients obtained by Lemma 8 is $\binom{4}{1} + \binom{4}{2} = 4 + 6 = 10$, which is the same as the number of terms in the above expression.

By adding one for the zero arithmetic coefficient to Lemma 8, we have an upper bound on the number of terminal nodes in a BMD for $f(X) = X^k$. For $k = 1$, it is *tight* and *exactly* $n + 1$. However, for $k > 1$, it is *not tight* because arithmetic coefficients with the same values exist, and they are represented by the same terminal node due to the reduction rule for BMDs. To derive the exact number of terminal nodes, enumeration of the number of *distinct* nonzero arithmetic coefficients is necessary. Improving Lemma 8, we derive the following three theorems for X^2, X^3, and X^4.

Theorem 7. *For the n-bit function $f(X) = X^2$, when $n \geq 2$, the number of* distinct *nonzero arithmetic coefficients is $2(n-1)$.*

Proof. We prove the theorem by the mathematical induction. In this proof, X_n denotes an n-bit variable $X = (x_{n-1}\ x_{n-2}\ \ldots\ x_0)_2$. When $n = 2$, we have

$$X_2^2 = (2^1 x_1 + 2^0 x_0)^2 = 2^2 x_1 + x_0 + 2^2 x_1 x_0,$$

and so the theorem holds. Next, we assume that the theorem holds for $n = i$, and we prove that the theorem holds for $n = i + 1$.

When $n = i+1$, $\{X_{i+1}\} = \{x_i\} \cup \{X_i\}$. To focus only on the product terms including x_i, we consider the following expression:

$$\begin{aligned} X_{i+1}^2 - X_i^2 &= (X_{i+1} - X_i)(X_{i+1} + X_i) \\ &= 2^i x_i (2^i x_i + 2X_i) = 2^{2i} x_i + 2^{i+1} x_i X_i \\ &= 2^{2i} x_i + 2^{i+1} x_i (2^{i-1} x_{i-1} + 2^{i-2} x_{i-2} \\ &\qquad + 2^{i-3} x_{i-3} + \cdots + 2^0 x_0) \\ &= 2^{2i} x_i + 2^{2i} x_i x_{i-1} + 2^{2i-1} x_i x_{i-2} \\ &\quad + 2^{i-1} 2^{i-1} x_i x_{i-3} + \cdots + 2^{i-1} 2^2 x_i x_0. \end{aligned}$$

From this expression, it is clear that when $n = i+1$, two distinct nonzero arithmetic coefficients, 2^{2i} and 2^{2i-1}, appear as new coefficients. From the hypothesis of the mathematical induction, when $n = i$, the number of distinct nonzero arithmetic coefficients is $2(i-1)$. Thus, when $n = i+1$, the number of distinct nonzero arithmetic coefficients is $2(i-1)+2 = 2\{(i+1)-1\}$. Therefore, the theorem holds for $n \geq 2$. □

Theorem 8. *For the n-bit function $f(X) = X^3$, when $n \geq 3$, the number of* distinct *nonzero arithmetic coefficients is*

$$\frac{n^2 + 7n - 16}{2}.$$

Proof. We prove the theorem by the mathematical induction. When $n = 3$, we have

$$\begin{aligned} X_3^3 &= (2^2 x_2 + 2^1 x_1 + 2^0 x_0)^3 \\ &= 2^6 x_2 + 2^3 x_1 + x_0 \\ &\quad + 3\{(2^5 + 2^4) x_2 x_1 + (2^4 + 2^2) x_2 x_0 + (2^2 + 2) x_1 x_0\} \\ &\quad + 6 \cdot 2^3 x_2 x_1 x_0, \end{aligned}$$

and the number of distinct nonzero arithmetic coefficients is 7. Since

$$\frac{3^2 + 7 \times 3 - 16}{2} = 7,$$

the theorem holds when $n = 3$. Next, we assume that the theorem holds for $n = i$, and we prove that the theorem holds for $n = i+1$.

To focus only on the product terms including x_i, we consider the following expression:

$$\begin{aligned} X_{i+1}^3 - X_i^3 &= (X_{i+1} - X_i)\left(X_{i+1}^2 + X_{i+1} X_i + X_i^2\right) \\ &= 2^i x_i \left\{ (2^i x_i + X_i)^2 + (2^i x_i + X_i) X_i + X_i^2 \right\} \\ &= 2^i x_i \left(2^{2i} x_i + 3 \cdot 2^i x_i X_i + 3 X_i^2\right) \\ &= 2^{3i} x_i + 3 \cdot 2^{2i} x_i X_i + 3 \cdot 2^i x_i X_i^2 \\ &= 2^{3i} x_i + 3\{(2^{3i-1} + 2^{3i-2}) x_i x_{i-1} \\ &\qquad + (2^{3i-2} + 2^{3i-4}) x_i x_{i-2} \\ &\qquad + \cdots + (2^{2i} + 2^i) x_i x_0\} \\ &\quad + 6(2^{3i-3} x_i x_{i-1} x_{i-2} + 2^{3i-4} x_i x_{i-1} x_{i-3} \\ &\qquad + \cdots + 2^{i+1} x_i x_1 x_0). \end{aligned}$$

From this expression, it is clear that when $n = i+1$, all the coefficients of product terms with a single literal or two literals newly appear as distinct nonzero arithmetic coefficients. As for the coefficients of product terms with three literals, only three coefficients larger than $6 \cdot 2^{3i-6}$ newly appear. This is because coefficients not larger than $6 \cdot 2^{3i-6} = 6 \cdot 2^{3(i-1)-3}$ already appear for $n = i$. In total, $i+4$ distinct nonzero arithmetic coefficients newly appear. Thus, when $n = i+1$, the number of distinct nonzero arithmetic coefficients is

$$\frac{i^2 + 7i - 16}{2} + i + 4 = \frac{(i+1)^2 + 7(i+1) - 16}{2}.$$

Therefore, the theorem holds for $n \geq 3$. □

Similarly to Theorems 7 and 8, we have the following:

Theorem 9. *For the n-bit function $f(X) = X^4$, when $n \geq 4$, the number of* distinct *nonzero arithmetic coefficients is*

$$\frac{n^3 + 29n - 90}{6}.$$

Next, we consider the number of nonterminal nodes.

Lemma 9. *For the n-bit function $f(X) = X^k$, the number of nonterminal nodes in the BMD is at most*

$$\alpha(n,k) = \sum_{i=1}^{k} \binom{n}{i}.$$

Proof. From the proof for Lemma 8, it is clear that the arithmetic expression for X^k consists of terms with at most k literals. All the terms in the arithmetic expression for X^k can be represented by a binary tree structure.

Let $\beta(n,k)$ be the number of nodes in the tree. Then, $\beta(n,k)$ satisfies the following relation:

$$\beta(n,k) = 1 + \beta(n-1,k) + \beta(n-1,k-1), \tag{6}$$

where 1 is the number of root node, and $\beta(n-1,k)$ and $\beta(n-1,k-1)$ are the numbers of nodes in the left subtree and the right subtree, respectively.

We show that $\alpha(n,k)$ satisfies the relation (6), i.e.,

$$\begin{aligned} &1 + \alpha(n-1,k) + \alpha(n-1,k-1) \\ &\quad = 1 + \sum_{i=1}^{k} \binom{n-1}{i} + \sum_{i=1}^{k-1} \binom{n-1}{i} \\ &\quad = \sum_{i=1}^{k} \binom{n-1}{i} + \sum_{i=0}^{k-1} \binom{n-1}{i} \\ &\quad = \sum_{i=1}^{k} \left[\binom{n-1}{i} + \binom{n-1}{i-1} \right] = \sum_{i=1}^{k} \binom{n}{i} = \alpha(n,k). \end{aligned}$$

Therefore, we have the lemma. □

Lemma 9 just gives an upper bound, because it considers only the second reduction rule for BMDs in Definition 12 (the rule for zero suppression). To prove that this upper bound is tight, we show the necessary and sufficient condition to share nonterminal nodes.

Lemma 10. *Consider an arithmetic expression of an integer-valued function. Let p and q be two distinct terms with the same arithmetic coefficient. Let x_i be a common variable in p and q. And, let $p = x_i p'$ and $q = x_i q'$. In a BMD, nonterminal*

nodes can be shared if and only if there exists at least one x_i that yields p' and q' with the same arithmetic coefficient.

Proof. In a BMD, terminal nodes represent arithmetic coefficients, and paths from the root node to terminal nodes represent terms in an arithmetic expression. Thus, the following two conditions are equivalent:

1. Arithmetic coefficients of p and q are equal, and arithmetic coefficients of p' and q' are equal.
2. Nonterminal nodes for x_i can be shared.

□

Using Lemma 10, we have the exact number of nonterminal nodes for X, X^2, X^3, and X^4.

Theorem 10. *For the n-bit functions $f(X) = X^k$ where $k = 1, 2, 3, 4$, the numbers of nonterminal nodes in the BMDs are independent of variable orders and are*

$$\sum_{i=1}^{k} \binom{n}{i}.$$

Proof. Since it is clear that the theorem holds for $k = 1$, we prove that there is no x_i satisfying the condition of Lemma 10 for $k = 2, 3, 4$.

When $k = 2$, terms have at most two literals. When both p and q have only one literal, it is clear that no common literal x_i exists in p and q. When p has one literal and q has two literals, p' that is produced by removing x_i from p is a constant term, and q' has a literal. Since the constant term for X^2 is zero and arithmetic coefficient of q' is nonzero, there is no x_i satisfying the condition of Lemma 10 in these terms. When both p and q have two literals, both p' and q' have one literal. The arithmetic coefficients of p' and q' are equal if and only if $p' = q'$, that is, $p = q$. Therefore, when $k = 2$, there is no x_i satisfying the condition of Lemma 10.

When $k = 3$, as shown in the proof of Theorem 8, all terms with one or two literals have distinct arithmetic coefficients. Therefore, when $k = 3$, there is no x_i satisfying the condition of Lemma 10.

When $k = 4$, similarly to $k = 3$, all terms with one, two, or three literals have distinct arithmetic coefficients. Therefore, when $k = 4$, there is no x_i satisfying the condition of Lemma 10.

From the above, for $k = 1, 2, 3, 4$, no nonterminal nodes are shared, and thus X^k is represented by a binary tree. Since this proof is independent of the order of variables, we have the theorem. □

For $k \geq 5$, we generated many BMDs for X^k with various number of bits and variable orders. All the generated BMDs had the same number of nonterminal nodes as Lemma 9. Therefore, we have the following.

Conjecture 1. *For the n-bit function $f(X) = X^k$ for any $k \geq 1$, the number of nonterminal nodes in the BMD is independent of variable orders and is*

$$\sum_{i=1}^{k} \binom{n}{i}.$$

From Lemma 8 and Theorems 7, 8, 9, and 10, we obtain the following:

Corollary 1. *For the n-bit functions $f(X) = X^k$, where $k = 1, 2, 3, 4$, the total numbers of nodes in the BMDs are independent of variable orders and are exactly*

$$\begin{cases} 2n+1 & (\text{when } k=1), \\ \dfrac{n^2+5n-2}{2} & (\text{when } k=2 \text{ and } n \geq 2), \\ \dfrac{n^3+3n^2+26n-42}{6} & (\text{when } k=3 \textit{ and } n \geq 3), \\ \dfrac{(n-2)(n^3+4n^2+19n+168)}{24} & (\text{when } k=4 \text{ and } n \geq 4), \end{cases}$$

respectively.

In [12], a similar problem has been considered. However, it shows only an upper bound and is not tight. On the other hand, Theorem 10 and Corollary 1 give the exact numbers.

Similar to the analysis for X^k, we derive upper bounds on the numbers of nonzero arithmetic coefficients and BMD nodes for kth-degree polynomial functions.

Lemma 11. *For an n-bit kth-degree polynomial function $f(X) = c_k X^k + c_{k-1} X^{k-1} + \cdots + c_0$, the number of nonzero arithmetic coefficients is at most*

$$\sum_{i=0}^{k} \binom{n}{i}.$$

When $c_i > 0 \ \forall i$, the number of nonzero arithmetic coefficients is exactly

$$\sum_{i=0}^{k} \binom{n}{i}.$$

Proof. From the proof of Lemma 8, it is clear that the sets of terms obtained by expanding and rearranging X^i $(i = 1, 2, \ldots, k-1)$ are proper subsets of the set of terms for X^k if coefficients of the terms are ignored. When $c_0 \neq 0$, the arithmetic coefficient for the constant term $\binom{n}{0}$ must be considered. Thus, we have the upper bound. Since all the nonzero arithmetic coefficients for X^i $(i = 1, 2, \ldots, k)$ are positive, when $c_i > 0 \ \forall i$, the number of nonzero arithmetic coefficients for $f(X)$ is equal to the upper bound. Therefore, Lemma 11 holds. □

Lemma 12. *For an n-bit kth-degree polynomial function $f(X) = c_k X^k + c_{k-1} X^{k-1} + \cdots + c_0$, the number of nonterminal nodes in the BMD is at most*

$$\sum_{i=1}^{k} \binom{n}{i}.$$

Proof. When all terms in the arithmetic expression for $f(X)$ are represented by a binary tree structure, the number of nonterminal nodes in a BMD is the maximum. In the tree, terminal nodes represent the nonzero arithmetic coefficients. In a binary tree, the number of nonterminal nodes is one less than the number of terminal nodes [11]. Therefore, from Lemma 11, we have the lemma. □

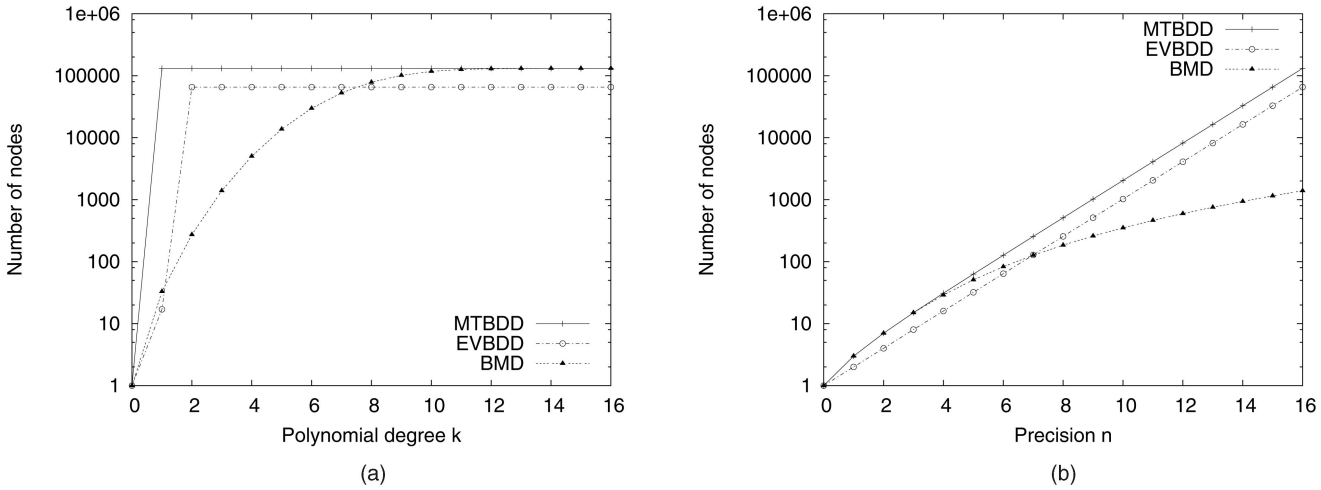


Fig. 5. Upper bounds on number of nodes in MTBDDs, EVBDDs, and BMDs. (a) For 16-bit kth-degree polynomial functions. (b) For n-bit third-degree polynomial functions.

By summing up the results of Lemmas 11 and 12, we obtain the following:

Theorem 11. *For an n-bit kth-degree polynomial function $f(X) = c_k X^k + c_{k-1} X^{k-1} + \cdots + c_0$, the total number of nodes in the BMD is at most*

$$2\sum_{i=0}^{k} \binom{n}{i} - 1.$$

Example 7. Figs. 5a and 5b compare the upper bounds on the number of nodes in MTBDDs, EVBDDs, and BMDs for 16-bit kth-degree polynomial functions and n-bit third-degree polynomial functions, respectively.

When the number of bits n is fixed, the upper bound on the number of nodes in BMD for kth-degree polynomial function increases with k. On the other hand, in an MTBDD and an EVBDD, when $k > 1$, the upper bounds on the nodes are $2^{n+1} - 1$ and 2^n, respectively, independently of k. Thus, when k is small, the upper bound on the nodes in a BMD is smaller than those in MTBDD and EVBDD.

When the polynomial degree k is fixed, the upper bound on the number of nodes in a BMD for n-bit polynomial function increases more slowly than those in MTBDD and EVBDD with n. Furthermore, Corollary 1 and Theorem 5 show that the upper bounds for the MTBDD and the EVBDD are tight when all the coefficients c_i are positive. Thus, when n is large, BMDs require many fewer nodes than MTBDDs and EVBDDs.

From the above observations, we can see that for n-bit kth-degree polynomial functions, BMDs tend to require fewer nodes than MTBDDs and EVBDDs when k is sufficiently smaller than n.

4 Complexity of EVBDDs for Monotone Functions

In this section, by using monotonicity of functions, we analyze complexity of EVBDDs for them. Since many common elementary functions are monotone functions, this section introduces an Mp-monotone increasing function to show the properties of elementary functions and derives an upper bound on the number of nodes in an EVBDD for the Mp-monotone increasing function.

4.1 Introduction of Mp-Monotone Increasing Function

We define an Mp-monotone increasing function as follows:

Definition 13. *Let I be a set of integers including 0, and let p be an integer. An integer function $f(X) : I \to Z$ such that $0 \le f(X+1) - f(X) \le p$ and $f(0) = 0$ is an* **Mp-monotone increasing function** *on I. That is, an* Mp-monotone *increasing function $f(X)$ satisfies $f(0) = 0$, and the increment of X by one increases the value of $f(X)$ by at most p.*

Definition 14. *Let $f(X)$ be an integer function.* ***Affine transformation*** *of $f(X)$ is defined as $af(X) + b$, where a and b are integers.*

Example 8. For the 3-bit $\sin(X) = [0, 1, 2, 3, 4, 5, 5, 6]^t$, the increment of X by one increases the function value by at most $p = 1$. Thus, this function is an M1-monotone increasing function. For the 3-bit $\frac{1}{X+1} = [8, 7, 6, 6, 5, 5, 5, 4]^t$, the increment of X by one *decreases* the function value by at most one. Thus, this function is *not* an M1-monotone increasing function, but it is an affine transformation of an M1-monotone increasing function $f(X) = [0, 1, 2, 2, 3, 3, 3, 4]^t : -1 \times f(X) + 8$. The polynomial function shown in Fig. 3a is also an affine transformation of an M76-monotone increasing function $g(X) = [0, 6, 17, 33, 54, 80, 111, 147, 188, 234, 285, 341, 402, 468, 539, 615]^t : 2 \times g(X) + 2$.

Let $f(X)$ be an Mp-monotone increasing function. Then, we have the following relation:

$$p = \max_{X \in I} \left(\frac{f(X+1) - f(X)}{(X+1) - X} \right).$$

This is the maximum *average rate of change* for the integer function $f(X)$. Note that the maximum *differential coefficient* of the original elementary function $\tilde{f}$ is an upper bound on p. More precisely, we have the following:

Theorem 12. *Let $\tilde{f}(X_r)$ and $f(X)$ be a real function and its integer (fixed-point) function, respectively, where $f(X)$ and X*

have the same number of fractional bits. When $f(X)$ is an Mp-monotone *increasing function, the following holds:*

$$p \le \left\lceil \max_{X_r \in R} \left(\tilde{f}'(X_r) \right) \right\rceil,$$

where R is a domain of the real variable X_r and corresponds to the domain I of the integer variable X.

Proof. Assume that $f(X)$ and X are represented by fixed-point numbers. Let n be the number of fractional bits for $f(X)$ and X. Because of the rounding errors [17], we have

$$\left| \tilde{f}(X_r) - f(X) \right| \le 2^{-(n+1)}.$$

Then,

$$p \le \max_{X_r \in R} \left(\frac{\tilde{f}(X_r + 2^{-n}) + 2^{-(n+1)} - \tilde{f}(X_r) + 2^{-(n+1)}}{2^{-n}} \right)$$
$$= \max_{X_r \in R} \left(\frac{\tilde{f}(X_r + 2^{-n}) - \tilde{f}(X_r)}{2^{-n}} \right) + 1.$$

Note that the rounding errors occur only if $\max\left(\frac{\tilde{f}(X_r + 2^{-n}) - \tilde{f}(X_r)}{2^{-n}} \right)$ is not an integer. Thus, we have

$$p \le \left\lceil \max_{X_r \in R} \left(\frac{\tilde{f}(X_r + 2^{-n}) - \tilde{f}(X_r)}{2^{-n}} \right) \right\rceil.$$

From the mean value theorem on differential coefficient, there exists X_s such that $X_r \le X_s \le X_r + 2^{-n}$ and that satisfies the following relation:

$$\frac{\tilde{f}(X_r + 2^{-n}) - \tilde{f}(X_r)}{2^{-n}} = \tilde{f}'(X_s).$$

Since $\max\left(\tilde{f}'(X_s)\right) \le \left\lceil \max_{X_r \in R} \left(\tilde{f}'(X_r) \right) \right\rceil$, we have the theorem. □

Thus, we can estimate the value of p by Theorem 12.

Example 9. Let $\tilde{f}_1(X_r) = \sin(X_r)$. Then, $\tilde{f}_1'(X_r) = \cos(X_r)$, and $\left\lceil \max_{X_r \in [0,1)} \left(\tilde{f}_1'(X_r) \right) \right\rceil = 1$. Note that $\sin(X_r)$ is monotone increasing for $X_r \in [0, 1)$. Thus, $\sin(X_r)$ is converted into an M1-monotone increasing function.

Let $\tilde{f}_2(X_r) = \tan(X_r)$. Then, $\tilde{f}_2'(X_r) = \frac{1}{\cos(X_r)}$, and $\left\lceil \max_{X_r \in [0,1)} \left(\tilde{f}_2'(X_r) \right) \right\rceil = \lceil 3.4255 \rceil = 4$. Note that $\tan(X_r)$ is monotone increasing for $X_r \in [0, 1)$. Thus, $\tan(X_r)$ is converted into an M4-monotone increasing function.

When $f(X)$ is an affine transformation of an Mp-monotone increasing function, we have the following relation:

$$a \times p + f(0) = \max_{X \in I} (f(X + 1) - f(X)),$$

where a is an integer common factor of all function values and is negative when $f(X)$ is monotone decreasing.

Table 2 shows the relations between various elementary functions and Mp-monotone increasing functions. As shown in Table 2, many common elementary functions can be converted into Mp-monotone increasing functions or their affine transformations. When the differential coefficient is large, the value of p is large. In Table 2, $\sin^{-1}(X)$, $\cos^{-1}(X)$, $\cosh^{-1}(X)$, and $\tanh^{-1}(X)$ are such examples.

Before analyzing complexity of EVBDDs, we derive the number of distinct Mp-monotone increasing functions. To derive that, we first define a $(p+1)$-valued 0-preserving function.

TABLE 2
Relations between 16-Bit Elementary Functions and Mp-Monotone Increasing Functions

Elementary functions	p	Type of functions
2^X	2	A.T. of M2-monotone increasing.
e^X	3	A.T. of M3-monotone increasing.
$\ln(X+1)$	1	M1-monotone increasing.
$\log_2(X+1)$	2	M2-monotone increasing.
$\frac{1}{X+1}$	1	A.T. of M1-monotone increasing.
$\sqrt{X+1}$	1	A.T. of M1-monotone increasing.
$\frac{1}{\sqrt{X+1}}$	1	A.T. of M1-monotone increasing.
$\sin(X)$	1	M1-monotone increasing.
$\cos(X)$	1	A.T. of M1-monotone increasing.
$\tan(X)$	4	M4-monotone increasing.
$\sin^{-1}(X)$	150	M150-monotone increasing.
$\cos^{-1}(X)$	150	A.T. of M150-monotone increasing.
$\tan^{-1}(X)$	1	M1-monotone increasing.
$\sinh(X)$	2	M2-monotone increasing.
$\cosh(X)$	2	A.T. of M2-monotone increasing.
$\tanh(X)$	1	M1-monotone increasing.
$\sinh^{-1}(X)$	1	M1-monotone increasing.
$\cosh^{-1}(X+1)$	362	M362-monotone increasing.
$\tanh^{-1}(X)$	22,714	M22714-monotone increasing.

A.T.: Affine transformation
Number of fractional bits for function values is 16.
Domain of the functions is $0 \le X < 1$.

Definition 15. *A two-valued input multivalued output function $h : B^n \to \{0, 1, \ldots, p\}$ such that $h(0, 0, \ldots, 0) = 0$ is a $(p+1)$-**valued 0-preserving function**. This is an extension of the 0-preserving function for logic function [23].*

Lemma 13. *The number of distinct n-bit* Mp-monotone *increasing functions is $(p+1)^{2^n - 1}$.*

Proof. Let $h(Y)$ be an n-bit input $(p+1)$-valued 0-preserving function, where $Y = (y_{n-1}\, y_{n-2} \ldots y_0)_2$. For each $h(Y)$, there exists an n-bit Mp-monotone increasing function

$$f(X) = \sum_{Y=0}^{X} h(Y).$$

Conversely, for any given n-bit Mp-monotone increasing function $f(X)$, there exists a $(p+1)$-valued 0-preserving function. The number of different h's is $(p+1)^{2^n - 1}$. Therefore, we have the lemma. □

Example 10. Let $\boldsymbol{H}$ be the function vector of a 2-bit input two-valued 0-preserving function h. And, let $\boldsymbol{F}$ be the function vector of a 2-bit M1-monotone increasing function f. The following shows all possible $\boldsymbol{F}$'s and the corresponding $\boldsymbol{H}$'s:

$\boldsymbol{H}$	$\boldsymbol{F}$	$\boldsymbol{H}$	$\boldsymbol{F}$
$[0,0,0,0]^t$	$[0,0,0,0]^t$	$[0,0,0,1]^t$	$[0,0,0,1]^t$
$[0,0,1,0]^t$	$[0,0,1,1]^t$	$[0,0,1,1]^t$	$[0,0,1,2]^t$
$[0,1,0,0]^t$	$[0,1,1,1]^t$	$[0,1,0,1]^t$	$[0,1,1,2]^t$
$[0,1,1,0]^t$	$[0,1,2,2]^t$	$[0,1,1,1]^t$	$[0,1,2,3]^t$

As shown above, the number of the different 2-bit M1-monotone increasing functions is $2^{2^2 - 1} = 8$.

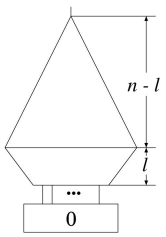


Fig. 6. Partition of EVBDD for $\mathrm{M}p$-monotone increasing function.

4.2 Complexity of EVBDDs for $\mathrm{M}p$-Monotone Increasing Functions

To analyze the complexity of EVBDDs for $\mathrm{M}p$-monotone increasing functions, we partition an EVBDD into two parts: the upper and the lower parts as shown in Fig. 6. To analyze the lower part, we introduce the following.

Definition 16. *A **shared EVBDD (SEVBDD)** is an extension of the EVBDD, and it has multiple root nodes to represent multiple integer functions. The SEVBDD is obtained by sharing equivalent subgraphs in EVBDDs for the integer functions.*

Lemma 14. *Let $\eta(l,p)$ be the number of nonterminal nodes in the SEVBDD representing all the l-bit $\mathrm{M}p$-monotone increasing functions, where the variable order of the SEVBDD is $x_{l-1}, x_{l-2}, \ldots, x_0$ (from the root nodes to the terminal node). Then,*

$$\eta(l,p) = \sum_{i=1}^{l} (p+1)^{2^i-1} - l.$$

Proof. We prove the lemma by the mathematical induction. When $l=1$, the function vectors of all the $\mathrm{M}p$-monotone increasing functions are $F=[0,0]^t, F=[0,1]^t, \ldots,$ and $F=[0,p]^t$. Since $F=[0,0]^t$ is the constant function 0, there is no nonterminal node representing it. As for the other p function vectors, there exists a nonterminal node for each function vector. Thus, when $l=1$, the lemma holds. Next, we assume that the lemma holds when $l=n$. And, we prove that the lemma holds when $l=n+1$.

Each nonterminal node in an EVBDD represents the following expansion: $f=\overline{x_i}f_0 + x_i(f_1' + \alpha)$. When $f_0 = f_1' + \alpha$, however, the nonterminal node for x_i is eliminated because of the reduction rules for EVBDD. Conversely, the nonterminal node for x_i is not eliminated when $f_0 \neq f_1' + \alpha$. When f is an $(n+1)$-bit $\mathrm{M}p$-monotone increasing function except for the constant function 0, $f_0 \neq f_1' + \alpha$ holds in the expansion with respect to x_n. From Lemma 13, the number of different $(n+1)$-bit $\mathrm{M}p$-monotone increasing functions except for the constant function 0 is $(p+1)^{2^{n+1}-1} - 1$. Thus, in an SEVBDD, there exist $(p+1)^{2^{n+1}-1} - 1$ nonterminal nodes representing the expansions with respect to x_n. Since f_0's and f_1''s produced by these expansions are the n-bit $\mathrm{M}p$-monotone increasing functions, from the hypothesis of induction for $l=n$, we have

$$\eta(n,p) = \sum_{i=1}^{n} (p+1)^{2^i-1} - n.$$

Therefore, when $l=n+1$, the number of nonterminal nodes is

$$\begin{aligned}
&\eta(n,p) + (p+1)^{2^{n+1}-1} - 1 \\
&\quad = \sum_{i=1}^{n} (p+1)^{2^i-1} - n + (p+1)^{2^{n+1}-1} - 1 \\
&\quad = \sum_{i=1}^{n+1} (p+1)^{2^i-1} - (n+1) = \eta(n+1,p).
\end{aligned}$$

Therefore, the lemma holds. □

Using this lemma, we derive the upper bound on the number of nodes as follows:

Theorem 13. *For an n-bit $\mathrm{M}p$-monotone increasing function $f(X)$, the number of nodes in the EVBDD is at most*

$$2^{n-l} + \sum_{i=1}^{l} (p+1)^{2^i-1} - l,$$

where l is the largest integer satisfying $2^{n-l} \geq (p+1)^{2^l-1}$, and the variable order of the EVBDD is $x_{n-1}, x_{n-2}, \ldots, x_0$ (from the root node to the terminal node).

Proof. Suppose that an EVBDD for $f(X)$ is partitioned into the upper and the lower parts as shown in Fig. 6. In this case, the lower part represents l-bit $\mathrm{M}p$-monotone increasing functions, and the upper part represents the selector function. The upper part has the maximum number of nodes when it forms a complete binary tree. That is, the maximum number of nodes in the upper part is $2^{n-l} - 1$. The lower part has the maximum number of nodes when it represents all the l-bit $\mathrm{M}p$-monotone increasing functions. From Lemma 14, the maximum number of nodes in the lower part is

$$\eta(l,p) = \sum_{i=1}^{l} (p+1)^{2^i-1} - l.$$

Thus, the number of nonterminal nodes in the EVBDD for $f(X)$ is at most

$$2^{n-l} + \sum_{i=1}^{l} (p+1)^{2^i-1} - l - 1.$$

By adding one that denotes the terminal node to this, we have the theorem. The number of $\mathrm{M}p$-monotone increasing functions that can be represented in the lower part is $(p+1)^{2^l-1}$. It does not exceed the number of functions that can be selected by the upper part: 2^{n-l}. Therefore, we have the relation: $(p+1)^{2^l-1} \leq 2^{n-l}$. □

Next, we prove that this upper bound holds for also affine transformations of an $\mathrm{M}p$-monotone increasing function.

Lemma 15. *Let $f(X)$ be an $\mathrm{M}p$-monotone increasing function, and let $g(X)$ be an affine transformation of $f(X)$: $g(X) = af(X) + b$, where a and b are integers. Then, the EVBDDs for $f(X)$ and $g(X)$ have the same number of nodes.*

Proof. In EVBDDs, the sum of weights of the traversed edges shows the function value. Thus, in the EVBDD for $f(X)$, multiplying each weight by a and adding b to the

TABLE 3
Upper Bounds on Number of Nodes in DDs for n-Bit Mp-Monotone Increasing Functions

n	MTBDD	EVBDD					
		M1 ($p=1$)		M2 ($p=2$)		M3 ($p=3$)	
		No. of nodes	Ratio [%]	No. of nodes	Ratio [%]	No. of nodes	Ratio [%]
16	131,071	8,327	6.35	10,406	7.94	16,450	12.55
17	262,143	16,519	6.30	18,598	7.09	32,833	12.52
18	524,287	32,903	6.28	34,982	6.67	49,217	9.39
19	1,048,575	65,670	6.26	67,750	6.46	81,985	7.82
20	2,097,151	98,438	4.69	133,286	6.36	147,521	7.03
21	4,194,303	163,974	3.91	264,358	6.30	278,593	6.64
22	8,388,607	295,046	3.52	526,502	6.28	540,737	6.45
23	16,777,215	557,190	3.32	1,050,790	6.26	1,065,025	6.35
24	33,554,431	1,081,478	3.22	2,099,366	6.26	2,113,601	6.30

Ratio = (EVBDD nodes) / (MTBDD nodes) $\times$ 100.

weight of edge to the root node can produce the EVBDD for $g(X)$. This conversion of EVBDDs does not change the number of nodes. □

Example 11. Table 3 compares the upper bounds on the number of nodes in MTBDDs and EVBDDs for n-bit Mp-monotone increasing functions. From Lemma 2, the upper bound for MTBDDs is $2^{n+1}-1$ independently of p. On the other hand, the upper bound for EVBDDs increases with p.

5 Conclusion

In this paper, we have analyzed the number of nodes of three types of DDs, MTBDD, EVBDD, and BMD, for elementary functions by two approaches. First, we analyzed the number of nodes for n-bit kth-degree polynomial functions. We showed that when the polynomial degree k is sufficiently smaller than the number of bits n, BMDs have low complexity for polynomial functions. Second, we introduced a new class of integer functions, called Mp-monotone increasing functions, and analyzed their complexities. We showed that the smaller p is, the lower complexity of EVBDDs is. We also derived the exact number of nodes in the smallest EVBDD for the n-bit multiplier function and variable orders that produce the smallest EVBDDs. All complexities derived in this paper are summarized in Tables 4 and 5.

TABLE 4
Numbers of Nonzero Arithmetic Coefficients for n-Bit Functions $f(X)$

$f(X)$	Total number of non-zero arithmetic coefficients		Number of distinct non-zero coefficients	
X	n	E.N.	n	E.N.
X^2	$\frac{1}{2}n(n+1)$	E.N.	$2(n-1)$	E.N.
X^3	$\frac{1}{6}n(n^2+5)$	E.N.	$\frac{1}{2}(n^2+7n-16)$	E.N.
X^4	$\frac{1}{24}n(n+1)(n^2-3n+14)$	E.N.	$\frac{1}{6}(n^3+29n-90)$	E.N.
X^k	$\sum_{i=1}^{k}\binom{n}{i}$	E.N.	–	
Poly.	$\sum_{i=0}^{k}\binom{n}{i}$	U.B.	–	
Poly+	$\sum_{i=0}^{k}\binom{n}{i}$	E.N.	–	

Poly.: polynomial functions
Poly+: polynomial functions, where all coefficients are positive.
E.N.: Exact number U.B.: Upper bound
–: Not considered in this paper.

Acknowledgments

This research is supported in part by the Grant-in-Aid for Scientific Research of the Japan Society for the Promotion of Science (JSPS), funds from Ministry of Education, Culture, Sports, Science, and Technology (MEXT) via the Knowledge Cluster Project, the MEXT Grant-in-Aid for Young Scientists (B), 200700051, 2008, and Hiroshima City University Grant for Special Academic Research (General Studies), 8108, 2008. The comments of the five reviewers were useful in improving this paper. This paper is an extended version of two papers [20], [24].

References

[1] S.B. Akers, "Binary Decision Diagrams," *IEEE Trans. Computers*, vol. 27, no. 6, pp. 509-516, June 1978.
[2] R. Andraka, "A Survey of CORDIC Algorithms for FPGA Based Computers," *Proc. ACM/SIGDA Sixth Int'l Symp. Field Programmable Gate Array (FPGA '98)*, pp. 191-200, Feb. 1998.
[3] R.E. Bryant, "Graph-Based Algorithms for Boolean Function Manipulation," *IEEE Trans. Computers*, vol. 35, no. 8, pp. 677-691, Aug. 1986.
[4] R.E. Bryant, "On the Complexity of VLSI Implementations and Graph Representations of Boolean Functions with Applications to Integer Multiplication," *IEEE Trans. Computers*, vol. 40, no. 2, pp. 205-213, Feb. 1991.
[5] R.E. Bryant and Y.-A. Chen, "Verification of Arithmetic Circuits with Binary Moment Diagrams," *Proc. 32nd Design Automation Conf. (DAC '95)*, pp. 535-541, 1995.

TABLE 5
Complexities of Three Types of DDs for n-Bit Functions

Functions $f(X)$	Total number of nodes					
	MTBDD		EVBDD		BMD	
X	$2^{n+1}-1$	E.N.	$n+1$	E.N.	$2n+1$	E.N.
X^2	$2^{n+1}-1$	E.N.	2^n	E.N.	$\frac{1}{2}(n^2+5n-2)$	E.N.
X^3	$2^{n+1}-1$	E.N.	2^n	E.N.	$\frac{1}{6}(n^3+3n^2+26n-42)$	E.N.
X^4	$2^{n+1}-1$	E.N.	2^n	E.N.	$\frac{1}{24}(n-2)(n^3+4n^2+19n+168)$	E.N.
X^k	$2^{n+1}-1$	E.N.	2^n	E.N.	$2\sum_{i=0}^{k}\binom{n}{i}-1$	U.B.
Polynomial	$2^{n+1}-1$	U.B.	2^n	U.B.	$2\sum_{i=0}^{k}\binom{n}{i}-1$	U.B.
Positive poly.	$2^{n+1}-1$	E.N.	2^n	E.N.	$2\sum_{i=0}^{k}\binom{n}{i}-1$	U.B.
Mp-monotone	$2^{n+1}-1$	U.B.	$2^{n-1}+\sum_{i=1}^{l}(p+1)^{2^i-1}-l$	U.B.	–	
Multiplier	–		$2^{u+1}+(2^n-1)\{2n-(u+1)\}$	E.N.	–	

Positive poly.: polynomial functions, where all coefficients are positive.
E.N.: Exact number U.B.: Upper bound –: Not considered in this paper.

[6] M.J. Ciesielski, P. Kalla, Z. Zheng, and B. Rouzeyre, "Taylor Expansion Diagrams: A Compact Canonical Representation with Applications to Symbolic Verification," *Proc. Design, Automation and Test in Europe (DATE '02),* pp. 285-289, 2002.

[7] E.M. Clarke, K.L. McMillan, X. Zhao, M. Fujita, and J. Yang, "Spectral Transforms for Large Boolean Functions with Applications to Technology Mapping," *Proc. 30th ACM/IEEE Design Automation Conf. (DAC '93),* pp. 54-60, June 1993.

[8] R. Drechsler, B. Becker, and S. Ruppertz, "K*BMDs: A New Data Structure for Verification," *Proc. European Design and Test Conf. (ED&TC '96),* pp. 2-8, 1996.

[9] R. Drechsler and B. Becker, *Binary Decision Diagrams: Theory and Implementation.* Kluwer Academic, 1998.

[10] R. Enders, "Note on the Complexity of Binary Moment Diagram Representations," *Proc. Workshop Applications of the Reed-Muller Expansion in Circuit Design,* pp. 191-197, Aug. 1995.

[11] M.T. Goodrich and R. Tamassia, *Data Structures and Algorithms in JAVA.* John Wiley & Sons, 1998.

[12] S. Höreth and R. Drechsler, "Formal Verification of Word-Level Specifications," *Proc. Design, Automation and Test in Europe (DATE '99),* pp. 52-58, 1999.

[13] T. Kam, T. Villa, R.K. Brayton, and A.L. Sangiovanni-Vincentelli, "Multi-Valued Decision Diagrams: Theory and Applications," *Multiple-Valued Logic: An Int'l J.,* vol. 4, nos. 1-2, pp. 9-62, 1998.

[14] Y-T. Lai and S. Sastry, "Edge-Valued Binary Decision Diagrams for Multi-Level Hierarchical Verification," *Proc. 29th ACM/IEEE Design Automation Conf. (DAC '92),* pp. 608-613, 1992.

[15] Y-T. Lai, M. Pedram, and S.B. Vrudhula, "EVBDD-Based Algorithms for Linear Integer Programming, Spectral Transformation and Functional Decomposition," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems,* vol. 13, no. 8, pp. 959-975, Aug. 1994.

[16] C. Meinel and T. Theobald, *Algorithms and Data Structures in VLSI Design: OBDD—Foundations and Applications.* Springer, 1998.

[17] J.-M. Muller, *Elementary Function: Algorithms and Implementation.* Birkhauser, 1997.

[18] S. Nagayama and T. Sasao, "Compact Representations of Logic Functions Using Heterogeneous MDDs," *IEICE Trans. Fundamentals,* vol. E86-A, no. 12, pp. 3168-3175, Dec. 2003.

[19] S. Nagayama and T. Sasao, "On the Optimization of Heterogeneous MDDs," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems,* vol. 24, no. 11, pp. 1645-1659, Nov. 2005.

[20] S. Nagayama and T. Sasao, "Representations of Elementary Functions Using Edge-Valued MDDs," *Proc. 37th Int'l Symp. Multiple-Valued Logic (ISMVL '07),* May 2007.

[21] S. Nagayama and T. Sasao, "Representations of Two-Variable Elementary Functions Using Edge-Valued MDDs and Their Applications to Function Generators," *Proc. 38th Int'l Symp. Multiple-Valued Logic (ISMVL '08),* May 2008.

[22] *Representations of Discrete Functions,* T. Sasao and M. Fujita, eds. Kluwer Academic, 1996.

[23] T. Sasao, *Switching Theory for Logic Synthesis.* Kluwer Academic, 1999.

[24] T. Sasao and S. Nagayama, "Representations of Elementary Functions Using Binary Moment Diagrams," *Proc. 36th Int'l Symp. Multiple-Valued Logic (ISMVL '06),* May 2006.

[25] R. Stankovic, T. Sasao, and C. Moraga, *Spectral Transform Decision Diagrams,* chapter 3 in [22].

[26] R. Stankovic and J. Astola, *Spectral Interpretation of Decision Diagrams.* Springer Verlag, 2003.

[27] R. Stankovic and J. Astola, "Remarks on the Complexity of Arithmetic Representations of Elementary Functions for Circuit Design," *Proc. Workshop Applications of the Reed-Muller Expansion in Circuit Design and Representations and Methodology of Future Computing Technology,* pp. 5-11, May 2007.

[28] M.A. Thornton, R. Drechsler, and D.M. Miller, *Spectral Techniques in VLSI CAD.* Springer, 2001.

[29] J.E. Volder, "The CORDIC Trigonometric Computing Technique," *IRE Trans. Electronic Computers,* vol. 8, no. 3, pp. 330-334, Sept. 1959.

[30] I. Wegener, *Branching Programs and Binary Decision Diagrams: Theory and Applications.* SIAM, 2000.

[31] M.R. Williams, *History of Computing Technology.* IEEE CS Press, 1997.

[32] S.N. Yanushkevich, D.M. Miller, V.P. Shmerko, and R.S. Stankovic, *Decision Diagram Techniques for Micro- and Nanoelectronic Design.* CRC Press, Taylor and Francis Group, 2006.

Shinobu Nagayama received the BS and ME degrees from the Meiji University, Kanagawa, Japan, in 2000 and 2002, respectively, and the PhD degree in computer science from the Kyushu Institute of Technology, Iizuka, Japan, in 2004. He is currently a lecturer at the Hiroshima City University, Hiroshima, Japan. His research interest includes numerical function generators, decision diagrams, software synthesis, and embedded systems. He received the Outstanding Contribution Paper Award from the IEEE Computer Society Technical Committee on Multiple-Valued Logic (MVL-TC) in 2005 for a paper presented at the International Symposium on Multiple-Valued Logic in 2004, and the Excellent Paper Award from the Information Processing Society of Japan (IPS) in 2006. He is a member of the IEEE and the IEEE Computer Society.

Tsutomu Sasao received the BE, ME, and PhD degrees in electronics engineering from Osaka University, Osaka, Japan, in 1972, 1974, and 1977, respectively. He has held faculty/research positions at Osaka University, the IBM T.J. Watson Research Center, Yorktown Heights, New York, and the Naval Postgraduate School, Monterey, California. He is currently a professor in the Department of Computer Science and Electronics, Kyushu Institute of Technology, Iizuka, Japan. His research areas include logic design and switching theory, representations of logic functions, and multiple-valued logic. He has published more than nine books on logic design, including *Logic Synthesis and Optimization*, *Representation of Discrete Functions*, *Switching Theory for Logic Synthesis*, and *Logic Synthesis and Verification* (Kluwer Academic, 1993, 1996, 1999, and 2001, respectively). He has served as a program chairman for the IEEE International Symposium on Multiple-Valued Logic (ISMVL) many times. He was also the symposium chairman of the 28th ISMVL held in Fukuoka, Japan, in 1998. He received the NIWA Memorial Award in 1979, Distinctive Contribution Awards from the IEEE Computer Society MVL-TC for papers presented at ISMVLs in 1986, 1996, 2003, and 2004, and Takeda Techno-Entrepreneurship Award in 2001. He has served as an associate editor of the *IEEE Transactions on Computers*. He is a fellow of the IEEE and a member of the IEEE Computer Society.