Representations of Elementary Functions Using Binary Moment Diagrams

Tsutomu Sasao

Department of Computer Science and Electronics,
Kyushu Institute of Technology
Iizuka 820-8502, Japan

Shinobu Nagayama

Department of Computer Engineering,
Hiroshima City University
Hiroshima 731-3194, Japan

Abstract

This paper considers representations for elementary functions such as polynomial, trigonometric, logarithmic, square root, and reciprocal functions. These real valued functions are converted into integer functions by using fixed-point representation, and they are represented by using binary moment diagrams (BMDs). Elementary functions are represented compactly by applying the arithmetic transform to the functions. For polynomial functions, upper bounds on the numbers of nodes in BMDs and multi-terminal binary decision diagrams (MTBDDs) are derived. These results show that for polynomial functions, BMDs require fewer nodes than MTBDDs. Experimental result for 16*-bit precision* $\sin(x)$ *function shows that the BMD requires only* 20% *of the nodes for the MTBDD.*

1. Introduction

Binary decision diagram (BDD) [1, 11] is the most popular decision diagram, and is extensively used in logic synthesis, verification, logic simulation, etc. BDDs can represent many practical logic functions compactly, but cannot represent the multiplier function with reasonable size. To represent such functions compactly, the arithmetic transform decision diagram (ACDD) [16], the binary moment diagram (BMD) [2], the multiplicative BMD (*BMD) [2], the Kronecker *BMD (K*BMD) [7], and the Taylor expansion diagram (TED) [3] have been proposed. ACDD and BMD represent a given integer function using the arithmetic transform expansion, while the multi-terminal BDD (MTBDD) [4] represents the function using the Shannon expansion. Decision diagrams based on the arithmetic transform represent integer functions, such as adder and multiplier, with polynomial sizes of the number of input variables [2, 16]. In this paper, elementary functions [13] are converted into integer functions, and they are represented by BMDs. Theoretical and experimental results show that BMDs represent elementary functions compactly.

2. Preliminaries

Definition 1 *Let* $B = \{0,1\}$, $P = \{0,1,\ldots,p-1\}$ *where* $p \geq 2$, *and* R *be the set of real numbers. An* n*-input* m*-output* ***logic function (or multiple-output logic function)*** *is a mapping:* $B^n \rightarrow B^m$, *an* ***integer function*** *is* $B^n \rightarrow P$, *and a* ***real function*** *is* $R \rightarrow R$.

Definition 2 *The* ***binary fixed-point representation*** *of a value* r *has the form*

$$d_{n\_int-1}\ d_{n\_int-2}\ \ldots\ d_0.\ d_{-1}\ \ldots\ d_{-n\_frac}, \qquad (1)$$

where $d_i \in \{0,1\}$, n_int *is the number of bits for the integer part, and* n_frac *is the number of bits for the fractional part of* r. *The representation in (1) is two's complement, and so*

$$r = -2^{n\_int-1} d_{n\_int-1} + \sum_{i=-n\_frac}^{n\_int-2} 2^i d_i.$$

In this paper, when we consider only non-negative number r, *its fixed-point representation excludes sign-bit.*

Definition 3 ***Precision*** *is the total number of bits for a binary fixed-point representation. Specially,* ***n-bit precision*** *specifies that* n *bits are used to represent the number; that is,* $n = n\_int + n\_frac$. *In this paper, an* ***n-bit precision function*** $f(X)$ *means that the input variable* X *is* n*-bit precision.*

By fixed-point representation, we can convert n-bit precision real valued functions into n-input m-output logic functions. The multiple-output functions can be converted into integer functions by considering m-bit binary vectors as integers. That is, we can convert n-bit precision real valued functions into integer functions: $B^n \rightarrow P$, where $P = \{0,1,\ldots,2^m-1\}$. In this paper, we convert elementary functions into integer functions by using n-bit fixed-point representation. Real valued functions in this paper are converted into integer functions, unless stated otherwise.

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

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

*The **Kronecker product** of A and B is defined as the following $(n^2 \times n^2)$ matrix:*

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

3. Arithmetic Transform

This section describes the arithmetic transform and the arithmetic spectrum. For details, see [16].

Definition 5 *Let $A(n)$ be the **arithmetic transform matrix** defined by*

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

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

$$A_f = A(n)F.$$

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

Example 1 *Consider the integer function $f(x_1, x_2) = x_1 + x_2$. The function-vector is $F = [0,1,1,2]^t$. The arithmetic spectrum is*

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

(End of Example)

Definition 6 *Let $A^{-1}(n)$ be the **inverse arithmetic transform matrix** defined by*

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

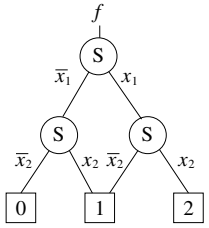
*The matrix $A^{-1}(n)$ has the same form as the **Reed-Muller transform matrix** $R(n)$. Thus, it is also called the **integer Reed-Muller matrix***

Definition 7 *In a symbolic representation,*

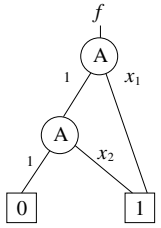
$$A^{-1}(1) = [\ 1 \ \ x_i\ ].$$

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

$$f = X_a A_f, \qquad X_a = \bigotimes_{i=1}^{n} [\ 1 \ \ x_i\ ].$$



(a) MTBDD for f.



(b) BMD for f.

Figure 1. MTBDD and BMD for $f = x_1 + x_2$.

Example 2 *From the inverse arithmetic transform and the arithmetic spectrum obtained in Example 1, the integer function f is represented as follows:*

$$f = X_a A_f = [\ 1 \ \ x_2 \ \ x_1 \ \ x_1x_2\ ] \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} = x_1 + x_2.$$

(End of Example)

Definition 8 *Using $A^{-1}(1)$ and $A(1)$, an integer function f is represented as follows:*

$$\begin{aligned} f &= A^{-1}(1)A(1)F = [\ 1 \ \ x_i\ ] \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \end{bmatrix} \\ &= [\ 1 \ \ x_i\ ] \begin{bmatrix} f_0 \\ f_1 - f_0 \end{bmatrix} = f_0 + x_i(f_1 - f_0), \end{aligned} \tag{2}$$

*where $f_0 = f(x_i = 0)$, $f_1 = f(x_i = 1)$. (2) is the **arithmetic transform expansion** (also called A-expansion or moment decomposition). The **arithmetic expression** for f is obtained by the arithmetic transform expansion. The arithmetic coefficients correspond to coefficients of the arithmetic expression for f.*

4. BMD (Binary Moment Diagram)

Definition 9 *A **binary moment tree (BMT)** is obtained by applying the arithmetic transform expansion $f = f_0 + x_i(f_1 - f_0)$ to a given integer function f recursively. A **binary moment diagram (BMD)** is derived from the BMT by using the following reduction rules:*

1. *Share equivalent sub-graphs.*
2. *If the outgoing edge of a node v labeled with x_i points to the constant zero, then delete the node v and connect the edges pointing v to the other outgoing edge of v directly.*

Example 3 *Fig. 1(a) shows the MTBDD for the integer function f in Example 1. Fig. 1(b) shows the BMD for f. In the figures, the nodes labeled with S denote the Shannon expansion, while the nodes labeled with A denote the arithmetic transform expansion.* *(End of Example)*

Terminal nodes in a BMD represent the arithmetic spectrum A_f for a function f, while terminal nodes in an MTBDD represent the function-vector F of f. Thus, the number of terminal nodes in a BMD is equal to the number of distinct arithmetic coefficients. On the other hand, in an MTBDD, it is equal to the number of distinct function values.

For X^k and kth-order polynomial functions, we can compute the numbers of non-zero arithmetic coefficients and nodes in BMDs from the values of precision n and polynomial order k. The following lemma and theorem give the number of non-terminal nodes in the BMD for the n-bit precision function $f(X) = X^k$, and the upper bound on the number of nodes in a BMD for an n-bit precision kth-order polynomial function.

Lemma 1 *For the n-bit precision function $f(X) = X^k$, the number of non-zero arithmetic coefficients is*

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

(Proof) See Appendix.

Lemma 2 *For an n-bit precision kth-order polynomial function $f(X) = c_0 + c_1X + c_2X^2 + \ldots + c_kX^k$, the number of non-zero arithmetic coefficients is at most*

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

(Proof) See Appendix.

Lemma 3 *For an n-bit precision kth-order polynomial function $f(X) = c_0 + c_1X + c_2X^2 + \ldots + c_kX^k$, when $c_i > 0$ and $X \geq 0$, the number of non-zero arithmetic coefficients is*

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

(Proof) See Appendix.

Lemma 4 *For the n-bit precision function $f(X) = X^k$, the number of non-terminal nodes in the BMD is*

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

(Proof) See Appendix.

In [7], similar problem has been considered. But it shows only an upper bound, and is not tight. On the other hand, Lemma 4 gives the exact number.

Lemma 5 *For an n-bit precision kth-order polynomial function $f(X) = c_0 + c_1X + c_2X^2 + \ldots + c_kX^k$, the number of non-terminal nodes in the BMD is at most*

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

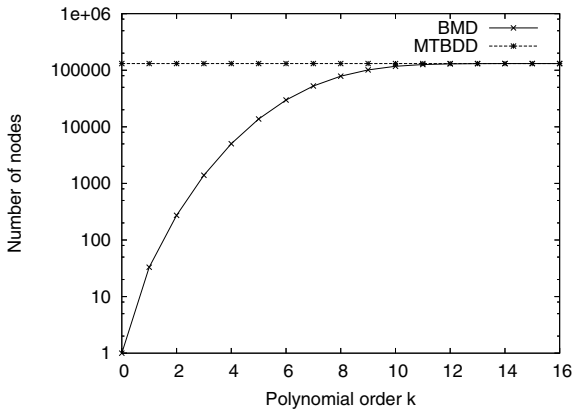


Figure 2. Number of nodes in BMDs and MTBDDs for 16-bit precision kth-order polynomial functions.

(Proof) See Appendix.

Theorem 1 *For an n-bit precision kth-order polynomial function $f(X) = c_0 + c_1X + c_2X^2 + \ldots + c_kX^k$, the total number of non-terminal nodes and terminal nodes in the BMD is at most*

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

(Proof) See Appendix.

Lemma 6 *For an n-bit precision 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 number of nodes in the MTBDD for $f(X)$ is $2^{n+1} - 1$.*

(Proof) See Appendix.

Corollary 1 *For an n-bit precision kth-order polynomial function $f(X) = c_0 + c_1X + c_2X^2 + \ldots + c_kX^k$, the number of nodes in the MTBDD for $f(X)$ is at most $2^{n+1} - 1$. When $c_i > 0$ and $X \geq 0$, the number of nodes in the MTBDD for $f(X)$ is $2^{n+1} - 1$.*

Example 4 *Fig. 2 compares the upper bounds on the number of nodes in BMDs and MTBDDs for 16-bit precision kth-order polynomial functions. Fig. 3 compares the upper bounds on the number of nodes in BMDs and MTBDDs for n-bit precision 3rd-order polynomial functions. (End of Example)*

When the precision n is fixed, the upper bound on the number of nodes in BMD for kth-order polynomial function increases with k. On the other hand, in an MTBDD, the upper bound on the nodes is $2^{n+1} - 1$ independently of k. Thus, when k is small, the upper bound on nodes in a BMD is smaller than that in an MTBDD.

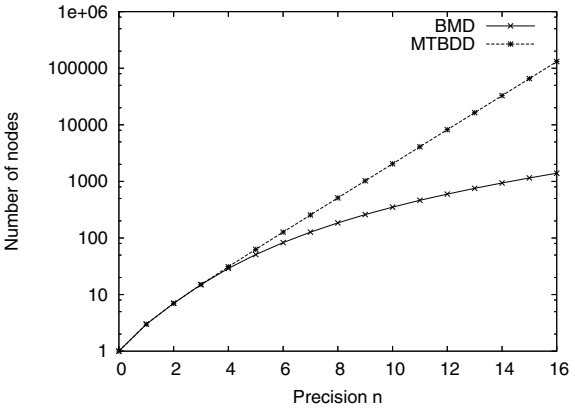


Figure 3. Number of nodes in BMDs and MTBDDs for n-bit precision 3rd-order polynomial functions.

Table 1. Numbers of non-zero arithmetic coefficients for randomly generated n-bit precision 4th-order polynomial functions.

Precision	Number of arithmetic coefficients		
n	Upper bound	Non-zero	Distinct
7	99	99	78
8	163	163	111
9	256	256	152
10	386	386	202
11	562	562	262
12	794	794	333
13	1093	1093	416
14	1471	1471	512
15	1941	1941	622
16	2517	2517	747

Function values are not rounded (i.e error-free), and have more bits than n.

When the polynomial order k is fixed, the upper bound on the number of nodes in BMD for n-bit precision polynomial function increases more slowly than that in MTBDD with n. Furthermore, Corollary 1 shows that the upper bound for the MTBDD is tight when all the coefficients c_i are positive. Thus, when n is large, BMDs require many fewer nodes than MTBDDs.

From the above observations, we can see that for n-bit precision kth-order polynomial functions, BMDs require fewer nodes than MTBDDs when k is smaller than n. Usually, the polynomial order k is smaller than precision n. Thus, for practical polynomial functions, BMDs are more compact than MTBDDs.

5. Experimental Results

5.1. Number of Arithmetic Coefficients

Polynomial Functions: To verify the tightness of the upper bound in Lemma 2, we randomly generated n-bit precision 4th-order polynomial functions $f(X) = c_4X^4 + c_3X^3 + c_2X^2 + c_1X + c_0$ where $X \geq 0$ (i.e., we generated 5 uniform random numbers for coefficients, where each coefficient has 16-bit precision: $|c_i| \leq 2^{15}$). Table 1 compares the number of non-zero arithmetic coefficients for f with the upper bound. In Table 1, the columns labeled with "Non-zero" and "Upper bound" show the numbers of non-zero arithmetic coefficients for $f(X)$ and their upper bounds given by Lemma 2, respectively. The column "Distinct" shows the number of distinct arithmetic coefficients for $f(X)$. For each precision n, we randomly generated 10 polynomial functions. For all of the generated functions, the numbers of non-zero arithmetic coefficients are equal to the upper bounds. This fact verifies the theoretical result (Lemma 2). Table 1 shows that for polynomial functions, many arithmetic coefficients are 0, and many non-zero coefficients have identical values as well.

Table 2. Numbers of distinct arithmetic coefficients for 16-bit precision elementary functions.

Elementary functions	Number of distinct values		Ratio [%]
	Function	Coefficients	
$2^x - 1$	59895	148	0.25
$1/\sqrt{x+1} - 0.707$	19196	174	0.90
$\ln(x+1)$	45427	165	0.36
$\log_2(x+1)$	59895	160	0.27
$\sqrt{x+1} - 1$	27147	138	0.50
$2/(x+1) - 1$	54292	180	0.33
$\sin(x)$	55147	141	0.25

Function values are rounded to 16-bit precision.
Domain of the functions is $0 \leq x < 1$.
Ratio = Coefficients / Function values $\times$ 100.

Non-polynomial Functions: In addition to the polynomial functions, we represented the non-polynomial elementary functions shown in Table 2. Table 2 compares the numbers of distinct function values and distinct arithmetic coefficients for 16-bit precision elementary functions. For each elementary function, its domain is $0 \leq x < 1$ and the function values are rounded to 16-bit precision. Table 2 shows that the elementary functions are transformed into the compact arithmetic spectrum. For $\frac{1}{\sqrt{x+1}} - 0.707$ and $\sqrt{x+1} - 1$, the numbers of distinct function values are smaller than other functions, since their range is smaller than the others. On the other hand, the numbers of distinct arithmetic coefficients are much smaller than that of function values independently of the range of functions. The numbers of distinct function values and distinct arithmetic coefficients correspond to the numbers of terminal nodes in the MTBDD and the BMD, respectively. Thus, BMDs require many fewer terminal nodes than MTBDDs to represent the elementary functions.

5.2. Number of Nodes in BMD

Polynomial Functions: Table 3 compares the numbers of nodes in BMDs and in MTBDDs for randomly gener-

Table 3. Numbers of nodes in BMDs and MTBDDs for randomly generated n-bit precision 4th-order polynomial functions.

n	MTBDD		BMD		Ratio
	Bound	#Nodes	Bound	#Nodes	[%]
7	255	255	197	175	69
8	511	511	325	272	53
9	1023	1023	511	406	40
10	2047	2047	771	586	29
11	4095	4095	1123	822	20
12	8191	8191	1587	1125	14
13	16383	16383	2185	1507	9
14	32767	32767	2941	1981	6
15	65535	65535	3881	2561	4
16	131071	131071	5033	3262	2

Function values are not rounded (i.e error-free).
Ratio = (BMD nodes / MTBDD nodes) $\times$ 100.

Table 4. Numbers of nodes in BMDs and MTBDDs for 16-bit precision elementary functions.

Elementary functions	Number of nodes		Ratio
	MTBDD	BMD	[%]
$2^x - 1$	122659	29634	24
$1/\sqrt{x+1} - 0.707$	58412	28446	49
$\ln(x+1)$	100880	28442	28
$\log_2(x+1)$	122542	29553	24
$\sqrt{x+1} - 1$	73406	26149	36
$2/(x+1) - 1$	114093	28348	25
$\sin(x)$	115450	22638	20

Function values are 16-bit precision.
Domain of the functions is $0 \leq x < 1$.
Ratio = (BMD / MTBDD) $\times$ 100.

ated n-bit precision 4th-order polynomial functions $f(X)$. In Table 3, the columns labeled with "MTBDD" and "BMD" show the numbers of nodes in MTBDDs and BMDs for $f(X)$, respectively. The sub-columns "#Nodes" and "Bound" show the number of nodes for $f(X)$ and its upper bound, respectively. The upper bounds for BMD and MTBDD are derived by Theorem 1 and Corollary 1, respectively. As shown in Table 1, for the polynomial functions, many arithmetic coefficients are 0, and many non-zero coefficients have identical values. Thus, the numbers of nodes in BMDs for the polynomial functions are smaller than the upper bounds in Theorem 1 by the reduction rule for BMDs. On the other hand, in MTBDDs for the polynomial functions, the numbers of nodes are $2^{n+1} - 1$, since for all the generated polynomial functions, Lemma 6 holds.

Non-polynomial Functions: Table 4 compares the numbers of nodes in BMDs with that in MTBDDs for 16-bit precision elementary functions. As shown in Table 2, for the elementary functions, BMDs require many fewer terminal nodes than MTBDDs. Thus, the total numbers of nodes in BMDs are smaller than that of MTBDDs. Especially, for an important elementary function $\sin(x)$, the BMD requires only 20% of the nodes for the MTBDD.

6. Conclusion and Comments

This paper considered BMD and MTBDD representations for elementary functions such as polynomial, trigonometric, logarithmic, square root, and reciprocal functions. We derived the number of nodes in BMDs for kth-order polynomial functions, and confirmed that the BMDs require fewer nodes than the MTBDDs. Especially, when the precision n is larger than the polynomial order k, BMDs require many fewer nodes than MTBDDs. Experimental result using 16-bit precision $\sin(x)$ function shows that the BMD requires only 20% of the nodes for the MTBDD.

This paper showed that the arithmetic transforms represent elementary functions compactly. We conjecture that other decision diagrams based on the arithmetic transform (e.g., ACDD, *BMD, K*BMD) also represent elementary functions efficiently. In the past, many people consider the arithmetic transforms [5, 6, 8, 9, 10, 12, 14, 17, 18, 19]. However, to the best of the authors' knowledge, this paper first considered representations of elementary functions by arithmetic transform.

Since BMDs represent elementary functions compactly, BMDs are promising for verification of hardware for elementary functions, and for the alternative implementation of embedded RAM on FPGA for the function tables.

Acknowledgments

This research is partly supported 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 Kitakyushu innovative cluster project.

References

[1] R. E. Bryant, "Graph-based algorithms for boolean function manipulation," *IEEE Trans. on Comput.*, Vol. C-35, No. 8, pp. 677–691, Aug. 1986.

[2] R. E. Bryant and Y-A. Chen, "Verification of arithmetic circuits with binary moment diagrams," *Design Automation Conference*, pp. 535–541, 1995.

[3] M. J. Ciesielski, P. Kalla, Z. Zheng, and B. Rouzeyre, "Taylor expansion diagrams: A compact canonical representation with applications to symbolic verification," *Design, Automation and Test in Europe (DATE2002)*, pp. 285–289, 2002.

[4] E. M. Clarke, K. L. McMillan, X. Zhao, and M. Fujita, "Spectral transforms for extremely large Boolean functions," *IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, pp. 86–90, Sept. 1993.

[5] B. J. Falkowski, "A note on the polynomial form of Boolean functions and related topics," *IEEE Trans. on Comput.*, Vol. 48, No. 8, pp. 860–864, Aug. 1999.

[6] K. D. Heidtmann, "Arithmetic spectrum applied to fault detection for combinational networks," *IEEE Trans. on Comput.*, Vol. 40, No. 3, pp. 320–324, Mar. 1991.

[7] S. Höreth and R. Drechsler, "Formal verification of word-level specifications," *Design, Automation and Test in Europe (DATE1999)*, pp. 52–58, 1999.

[8] S. L. Hurst, D. M. Miller, and J. C. Muzio, *Spectral Techniques in Digital Logic*, Academic Press, Bristol, 1985.

[9] J. Jain, "Arithmetic transform of Boolean functions," Chapter 6 in [15].

[10] S. K. Kumar and M. A. Breuer, "Probabilistic aspects of Boolean switching functions via a new transform," *Journal of the ACM*, 28(3), pp. 502–520, 1981.

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

[12] V.-D. Malyugin, *Paralleled Calculations by Means of Arithmetic Polynomials*, Physical and Mathematical Publishing Company, Russian Academy of Sci., Moscow, Russia, 1997.

[13] J.-M. Muller, *Elementary Function: Algorithms and Implementation*, Birkhauser Boston, Inc., Secaucus, NJ, 1997.

[14] S. G. Papaioannou and W. A. Barrett, "The real transform of a Boolean function and its application," *Computer and Electronic Engineering*, Vol. 2, pp. 215–224, Pergamon Press, 1975.

[15] T. Sasao and M. Fujita (eds.), *Representations of Discrete Functions*, Kluwer Academic Publishers, 1996.

[16] R. Stankovic, T. Sasao, and C. Moraga, "Spectral transform decision diagrams," Chapter 3 in [15].

[17] R. Stankovic and T. Sasao, "A discussion on the history of research in arithmetic and Reed-Muller expressions," *IEEE Trans. on CAD*, Vol. 20, No. 9, pp. 1177–1179, Sept. 2001.

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

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

Appendix

Proof for Lemma 1 From the property of the arithmetic transform, the number of non-zero arithmetic coefficients is equal to the number of terms in the expression that is obtained by expanding and rearranging the following:

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

where $x_i^2 = x_i$ ($i = 0,1,2,\ldots,n-1$) because x_i is a Boolean variable. In the expression expanded and rearranged, 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_ix_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 A.1 *Consider the* 4*-bit precision function* $f(X) = X^2$. *From* $X = -8x_3 + 4x_2 + 2x_1 + x_0$, *we have*

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

Note that this expression has 10 *terms. The number of non-zero arithmetic coefficients obtained by Lemma 1 is* $\binom{4}{1} + \binom{4}{2} = 4 + 6 = 10$, *and it is identical to the number of terms in the above expression. (End of Example)*

Proof for Lemma 2 From the proof for Lemma 1, 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. Therefore, Lemma 2 holds. ∎

Proof for Lemma 3 From the proof for Lemma 1, when $X \geq 0$, all the non-zero arithmetic coefficients for X^i ($i = 1,2,\ldots,k$) are positive. Since $c_i > 0$, the number of non-zero arithmetic coefficients for $f(X)$ is equal to its upper bound. ∎

Proof for Lemma 4 In a BMT for X^k, let the variable order be $x_0, x_1, \ldots, x_{n-1}$, from the root node to terminal nodes. From the proof for Lemma 1, it is clear that the arithmetic expression for X^k consists of terms with at most k literals. The arithmetic expression for X^k can be represented by a tree structure representing all the terms in the expression.

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

$$\beta(n,k) = 1 + \beta(n-1,k) + \beta(n-1,k-1), \qquad \text{(A.1)}$$

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 sub-treee and the right sub-tree, respectively.

We show that $\alpha(n,k)$ satisfies the relation (A.1).

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

Therefore, we have the lemma. ∎

Proof for Lemma 5 From the proof for Lemma 2, it is clear that the arithmetic expression for $f(X)$ consists of terms with at most k literals. The arithmetic expression for $f(X)$ can be represented by tree structure representing the all terms in the expression. From the property of the BMT, a BMT always requires a constant term independently of the value of the constant term. Therefore, BMD for $f(X)$ has at most $\sum_{i=1}^{k}\binom{n}{i}$ non-terminal nodes. ∎

Proof for Theorem 1 From Lemma 2, the number of terminal nodes in BMD for $f(X)$ is at most $\sum_{i=0}^{k}\binom{n}{i}$. From Lemma 5, the number of non-terminal nodes in BMD for $f(X)$ is at most $\sum_{i=1}^{k}\binom{n}{i}$. Therefore, we have the theorem. ∎

Proof for Lemma 6 When the number of distinct values of X is 2^n, the number of distinct values of $f(X)$ is also 2^n. Then, the MTBDD for $f(X)$ is the complete binary tree, and the number of nodes in the tree is $2^{n+1} - 1$. ∎