Representation of Rotation Symmetric Multiple-Valued Functions Using Decision Diagrams

Shinobu Nagayama* Tsutomu Sasao† Jon T. Butler‡ Martin Lukac*

*Dept. of Computer and Network Eng., Hiroshima City University, Hiroshima, JAPAN
†Dept. of Computer Science, Meiji University, Kawasaki, JAPAN
‡Dept. of Electr. and Comp. Eng., Naval Postgraduate School, Monterey, CA USA

***Abstract*—Rotation symmetric functions are a variant of symmetric functions, such that function values are unchanged by any rotation of digits in input vectors. Since rotation symmetric functions are resistant to linear attacks due to their non-linear characteristics, they are promising for cryptographic applications. However, there are few studies on how to represent them. This paper focuses on representation using functional decomposition and decision diagrams, as a steppingstone to find a compact representation of rotation symmetric functions. Experimental results show that the presented representation is promising.**

***Keywords*-Rotation symmetric functions; multiple-valued functions; functional decomposition; decision diagrams.**

I. Introduction

Rotation symmetric (RotS) functions [15], [20], [29], [30] are a class of functions such that function values are unchanged by any rotation of digits (i.e., circular translation of indices) in input vectors. They were introduced in 1999 by Pieprzyk and Qu [20]. It is known that RotS functions are resistant to linear attacks due to their non-linear characteristics, similarly to totally symmetric (TotS) functions and bent functions [8], [13], [21], [22], [27], [32]. Thus, RotS functions are promising for cryptographic applications. Specifically, they can be used as components in the rounds of efficient hashing algorithms, such as MD4, MD5, SHA, and HAVAL [20], [29]. RotS functions have a wider range of applications than TotS functions since the set of RotS functions is a superset of the set of TotS functions, as will be shown later.

While many studies on TotS functions [2], [5], [6], [9], [12], [31], [34] and bent functions [21], [27], [32] have been reported, few studies on RotS functions have been reported. Their representation method and construction method, particularly, have not been studied yet [29], other than by simple truth table or logic expression. As far as we know, there is no study on decision diagrams for RotS functions. Since finding compact representation of RotS functions significantly benefits the above cryptographic applications, as its steppingstone, this paper focuses on representation using decision diagrams.

Although RotS functions are a variant of TotS functions, their size complexity is quite different. It is well-known that TotS functions are compactly represented by ordinary decision diagrams [1], [3], [10]. However, as we will show later, RotS functions belong to the same worst class as maximally *asymmetric* functions [7], [18] and random functions in terms of size of ordinary decision diagrams. Thus, in this paper, as another possibility, we consider functional decomposition along with decision diagrams.

The rest of this paper is organized as follows: Section II shows some definitions for rotation symmetric multiple-valued functions and decision diagrams. Section III shows some characteristics of RotS functions. Section IV presents a decomposition method of RotS functions, and decision diagrams for sub-functions obtained by the decomposition method. Section V shows the size of decision diagrams for the sub-functions, and Section VI concludes the paper.

II. Preliminaries

This section shows definitions of rotation symmetric multiple-valued functions [20], [29] and decision diagrams.

A. Rotation Symmetric Multiple-Valued Functions

Definition 1: For an n-variable r-valued function $f(x_{n-1},x_{n-2},\ldots,x_0) : \{0,1,\ldots,r-1\}^n \rightarrow \{0,1,\ldots,r-1\}$, assignments of values to the n variables are **input vectors** $\vec{X}$. When the r^n input vectors $(0,0,\ldots,0)\cdots(r-1,r-1,\ldots,r-1)$ are fed to the function f in ascending order, the vector of obtained function values is the **function vector** $\vec{F}$.

Definition 2: A function f is **totally symmetric** (TotS) iff its function vector $\vec{F}$ is unchanged by any permutation of variable values.

Definition 3: We define a k-digit rotation of an input vector $\vec{X}$ as follows:

$$Rot(\vec{X},k) = (x_{n-1-k},x_{n-2-k},\ldots,x_0,x_{n-1},x_{n-2},\ldots,x_{n-k}).$$

Corollary 1: Let X_d be an integer obtained by considering $\vec{X}$ as an n-digit base r number $(x_{n-1},x_{n-2},\ldots,x_0)_r$. Then, $Rot(\vec{X},k)$ can be also computed as follows:

$$Rot(\vec{X},k) = (X_d \mod r^{n-k}) \times r^k + \left\lfloor \frac{X_d}{r^{n-k}} \right\rfloor.$$

2378-2226/25/$31.00 ©2025 IEEE
DOI 10.1109/ISMVL64713.2025.00037

Table I
EXAMPLE OF TOTS FUNCTION f_T AND ROTS FUNCTION f_R.

x_2	x_1	x_0	f_T	f_R
0	0	0	2	0
1	1	1	0	1
2	2	2	1	2
0	0	1	1	0
0	1	0	1	0
1	0	0	1	0
0	0	2	0	2
0	2	0	0	2
2	0	0	0	2
0	1	1	1	1
1	1	0	1	1
1	0	1	1	1
0	2	2	1	0
2	2	0	1	0
2	0	2	1	0
1	1	2	0	1
1	2	1	0	1
2	1	1	0	1
1	2	2	0	2
2	2	1	0	2
2	1	2	0	2
0	1	2	2	0
1	2	0	2	0
2	0	1	2	0
0	2	1	2	1
2	1	0	2	1
1	0	2	2	1

Definition 4: An n-variable r-valued function f is **rotation symmetric** (RotS) iff the following holds:

$$\forall k, f(\vec{X}) = f(Rot(\vec{X},k)),$$

where $0 < k < n$.

Example 1: Table I shows examples of a three-variable ternary TotS function f_T and a RotS function f_R. In this table, input vectors are reordered and grouped in ones obtained by rotation. As shown in Table I, function values of f_T depend on combinations of input values. On the other hand, function values of f_R depend on rotations of $\vec{X}$. □

B. Decision Diagrams

Definition 5: A **multi-valued decision diagram (MDD)** [10] is a rooted directed acyclic graph (DAG) representing an r-valued function. The MDD is obtained by recursively applying the extended Shannon expansion to the r-valued function. It consists of r terminal nodes representing function values, 0 to $r-1$, and nonterminal nodes representing input r-valued variables. Each nonterminal node has r outgoing edges that correspond to r values of an input variable. Terminal nodes have no outgoing edges. In this paper, an MDD is obtained by fixing the variable order in an MDD, and by applying the following two reduction rules:

1) Coalesce equivalent sub-graphs.

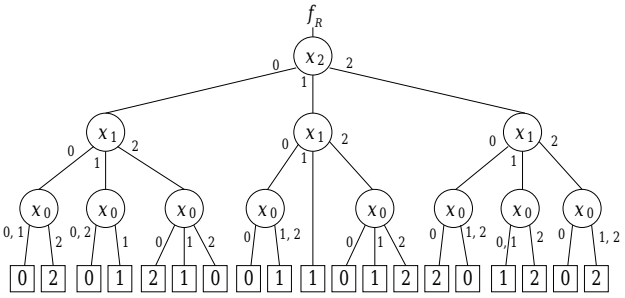

Figure 1. MDD for RotS Function f_R Defined in Table I

2) Delete nonterminal nodes v all whose outgoing edges point to the same node, and redirect edges pointing to v to its child node u.

Example 2: Fig. 1 shows an MDD for the RotS function f_R in Table I. In Fig. 1, for ensuring visibility, terminal nodes in the MDD are NOT shared. The number of nodes in this MDD is 15 (3 distinct terminal nodes and 12 distinct nonterminal nodes). □

Definition 6: A **vectorized EVMDD (VEVMDD)** [33] is a variant of an MDD and an edge-valued MDD (EVMDD) [17], and represents a multiple-output integer function. It consists of one terminal node representing $\vec{0}$ and nonterminal nodes with edges having vectors. While edges in EVMDDs have integer (scalar) values, edges in VEVMDDs can have vectors. The vectors consist of integers, and 0-edges always have the zero vector. Output vectors of the function are represented as a sum of vectors of edges traversed from the root node to the terminal node.

Definition 7: An **MDD with edge values for shifting** [19] is a variant of a VEVMDD, and its m-branch nonterminal nodes are based on the following extended Shannon expansion:

$$\begin{aligned} f &= x_i^0 f_0 + x_i^1 (f_1 << s_1(i)) + \ldots \\ &\quad + x_i^{r-1} (f_{r-1} << s_{r-1}(i)), \end{aligned}$$

where x_i is a multiple-valued variable represented by a nonterminal node, x_i^j is its literal given by

$$x_i^j = \begin{cases} 1 & (x_i = j) \\ 0 & (\text{Otherwise}), \end{cases}$$

$s_j(i)$ is an edge value, and f_j is a cofactor with $f(x_i = j)$ [23]. The terminal node represents a unit vector $\vec{e} = (0,0,\ldots,1)$. The unit vector $\vec{e}$ is shifted sequentially by values $s_j(i)$ of edges traversed from the root node to the terminal node. For convenience, we call this **shift EVMDD (SEVMDD)**.

Definition 8: A **multiple-terminal ZDD (MTZDD)** [19] is a variant of a zero-suppressed binary decision diagram (ZDD) [14] for representing a binary input $(r+1)$-valued output function, and it has $r+1$ terminal nodes. One of the

Table II
EQUIVALENCE CLASSES OF 4-VARIABLE TERNARY INPUT VECTORS.

$\vec{\alpha}$			$\vec{X}$			
α_2	α_1	α_0	x_3	x_2	x_1	x_0
0	2	2	0	0	1	1
			0	1	1	0
			1	1	0	0
			1	0	0	1
			0	1	0	1
			1	0	1	0
1	1	2	0	0	1	2
			0	1	2	0
			1	2	0	0
			2	0	0	1
			0	0	2	1
			0	2	1	0
			2	1	0	0
			1	0	0	2
			0	1	0	2
			1	0	2	0
			0	2	0	1
			2	0	1	0

$r+1$ terminal nodes represents the invalid value $\emptyset$, and the others represent valid function values: $0,1,\ldots,r-1$. Each nonterminal node represents each input binary variable, and has two unweighted outgoing edges, 0-edge and 1-edge, that correspond to two values of an input variable. In MTZDDs, the variable order is fixed, and the following two reduction rules are applied:

1) Coalesce equivalent sub-graphs.
2) Delete nonterminal nodes v whose 1-edge points to the terminal node representing $\emptyset$, and redirect edges pointing to v to its child node u pointed by v's 0-edge.

III. CHARACTERISTICS OF ROTS FUNCTIONS

A. Equivalence Classes of Input Vectors

For TotS functions, function values depend on combinations of input values. Thus, we can classify input vectors into equivalence classes, each of which has the same combination of input values. Then, we denote each equivalence class as follows:

$$\vec{\alpha} = (\alpha_{r-1}, \alpha_{r-2}, \ldots, \alpha_1, \alpha_0),$$

where α_i denotes the number of variables whose values are i, and $\sum_{i=0}^{r-1} \alpha_i = n$. We call those equivalence classes **α-equivalence classes** [16]. The number of input vectors belonging to each equivalence class is [18]:

$$\frac{n!}{\alpha_0!\alpha_1!\ldots\alpha_{r-1}!}.$$

Unlike TotS functions, function values of RotS functions depend on rotations of $\vec{X}$. Thus, $\vec{X}$ and its rotation $Rot(\vec{X},k)$ for $0 < k < n$ (i.e., up to n input vectors) can be classified into an equivalence class. We call the equivalence class **R-equivalence class**.

Example 3: Table II shows examples of α-equivalence classes and R-equivalence classes for 4-variable ternary input vectors. The α-equivalence class $\vec{\alpha} = (0,2,2)$ has 6 input vectors, and two R-equivalence classes. One of the two R-equivalence classes has only two input vectors: $(0,1,0,1)$ and $(1,0,1,0)$. The $\vec{\alpha} = (1,1,2)$ has 12 input vectors, and all the other R-equivalence classes have 4 input vectors. □

As shown in this example, an R-equivalence class is clearly a subset of an α-equivalence class. For an n-variable r-valued input vectors, the number of α-equivalence classes, N_α, is obtained by the following [7], [28]:

$$N_\alpha = \binom{n+r-1}{r-1}.$$

On the other hand, the number of R-equivalence classes is the same as the number of distinct circular permutations with repetition. Thus, from Burnside's lemma [4], the following is derived straightforwardly:

Theorem 1: For an n-variable r-valued input vectors, the number of R-equivalence classes, N_R, is

$$N_R = \frac{1}{n}\sum_{i=1}^{n} r^{\gcd(n,i)},$$

where $\gcd(n,i)$ is the greatest common divisor of n and i.

Similarly, the number of R-equivalence classes included in an α-equivalence class is also derived as follows:

Corollary 2: For an n-variable r-valued input vectors, the number of R-equivalence classes included in an α-equivalence class $\vec{\alpha} = (\alpha_{r-1}, \alpha_{r-2}, \ldots, \alpha_1, \alpha_0)$ is

$$\frac{1}{n}\sum_{i=1}^{n} \frac{s(i)}{\Pi_j^{r-1}\, t_j(i)!},$$

where

$$s(i) = \begin{cases} \gcd(n,i)! & (\sum_j^{r-1} \alpha_j \cdot \gcd(n,i) \mod n = 0) \\ 0 & (\text{Otherwise}), \end{cases}$$

and

$$t_j(i) = \frac{\alpha_j \cdot \gcd(n,i)}{n}.$$

From Corollary 2, α-equivalence classes $(0,2,2)$ and $(2,2,0)$ include the same number of R-equivalence classes. This means that the number of R-equivalence classes depends only on a *combination of α_i's values*.

B. Comparison Between TotS and RotS Functions

As shown in Example 1, any TotS function is rotation symmetric according to Definition 4. However, there exist RotS functions that are not totally symmetric. Thus, a set of TotS functions is a proper subset of RotS functions even though α-equivalence classes for TotS functions include R-equivalence classes for RotS functions. That is, TotS functions can be considered as a special case of RotS functions. It is well-known that TotS functions are compactly represented by MDDs [5]. However, as for RotS functions

Table III
AVERAGE NUMBER OF NODES IN MDD FOR VARIOUS FUNCTIONS.

n	r	TotS	RotS	MAF	Random	UB [17]
3	3	12	13	14	14	16
4	3	20	29	33	31	40
5	3	33	65	66	66	67
6	3	51	**148**	**148**	**148**	**148**
7	3	75	389	390	388	391
8	3	105	1,105	1,106	1,105	1,120
9	3	142	3,183	3,187	3,186	3,307
10	3	188	8,879	8,865	8,872	9,868
11	3	243	22,297	22,275	22,299	29,524
12	3	303	48,225	48,237	48,234	49,207
13	3	387	108,253	108,254	108,253	108,256
3	4	19	**25**	**25**	24	**25**
4	4	39	81	82	80	89
5	4	72	246	250	250	341
6	4	124	591	592	593	597
7	4	201	**1,621**	**1,621**	**1,621**	**1,621**
8	4	309	**5,717**	**5,717**	**5,717**	**5,717**
9	4	453	**22,101**	**22,101**	**22,101**	**22,101**
10	4	642	87,636	87,636	**87,637**	**87,637**
3	5	26	**36**	**36**	**36**	**36**
4	5	61	158	157	159	161
5	5	130	722	727	727	786
6	5	255	2,763	2,749	2,750	3,906
7	5	460	7,012	7,009	7,012	7,031
8	5	779	**22,656**	**22,656**	**22,656**	**22,656**
9	5	1,252	**100,781**	**100,781**	**100,781**	**100,781**

Boldfaced numbers show that the number of nodes in an MDD reaches at its upper bound.

that are a superset of TotS functions, the size of MDDs have not been studied yet.

Table III compares the size of MDDs for various n-variable r-valued functions. The size of MDDs is the average number of nodes in 10 MDDs for randomly generated functions. In Table III, the average is rounded to an integer. The column "TotS" shows the size of MDD for TotS functions, "RotS" shows the size for RotS functions, "MAF" shows the size for maximally asymmetric functions [18], and "Random" shows the size for functions whose values are generated by uniform random numbers. The column "UB" shows the upper bound on the number of nodes in an MDD for an n-variable r-valued function. It is obtained by

$$\frac{r^{n-l}-1}{r-1}+r^l,$$

where l is the largest integer satisfying $n-l \geq r^l$ [17]. The variable order for MDDs is the natural order (i.e., $x_{n-1}, x_{n-2}, \ldots, x_0$).

As shown in Table III, the size of MDDs for RotS functions is much larger than that for TotS functions even though RotS functions are *symmetric*. The size for RotS functions is almost equal to the upper bound, similarly to maximally *asymmetric* functions and random functions. Although the size of these MDDs is much smaller than the size of their truth tables [18], it has the largest size among MDDs.

Table IV
DECOMPOSING f_R BY INDICES OF $\vec{\alpha}$ AND R-EQUIV. CLASSES.

$\vec{\alpha}$			
α_2	α_1	α_0	i_{dx}
0	0	3	index_0
0	3	0	index_1
3	0	0	index_2
0	1	2	index_3
1	0	2	index_4
0	2	1	index_5
2	0	1	index_6
1	2	0	index_7
2	1	0	index_8
1	1	1	index_9

i_{dx}	Index of R-Equiv.	h
index_0	0	0
index_1	0	1
index_2	0	2
index_3	0	0
index_4	0	2
index_5	0	1
index_6	0	0
index_7	0	1
index_8	0	2
index$_9$	0	0
index$_9$	1	1

IV. DECOMPOSITION METHODS OF ROTS FUNCTIONS

As shown in the previous section, the size of MDDs for RotS functions is almost equal to the upper bound on the size of MDDs. As another possibility, in this section, we consider a functional decomposition method along with decision diagrams.

As described in Section III, R-equivalence classes are subsets of an α-equivalence class. Thus, by indexing each R-equivalence class in an α-equivalence class, we can represent a given RotS function without using input vectors directly. Then, similarly to TotS functions [19], we can decompose a given RotS function into the following functions:

1) $g(\vec{X})$ transforming $\vec{X}$ into $\vec{\alpha}$,
2) an index generation function $i_{dx}(\vec{\alpha})$ [24], [25] producing an index from $\vec{\alpha}$,
3) a function $g_{CF}(\vec{X})$ producing an index of R-equivalence class from $\vec{X}$, and
4) $h(i_{dx}, g_{CF})$ producing a function value of the original RotS function from both indices of $\vec{\alpha}$ and R-equivalence class.

Example 4: Table IV shows tables representing f_R shown in Table I using the above decomposition. □

The functions $g(\vec{X})$ and $i_{dx}(\vec{\alpha})$ can be represented compactly by a VEVMDD and an SEVMDD, respectively, in exactly the same way as in [19]. By representing both indices using the one-hot encoding, the function h can be represented compactly by an MTZDD.

Example 5: Fig. 2 shows an MTZDD for h shown in Table IV. □

Theorem 2: A function $h(i_{dx}, g_{CF})$ obtained by decomposition of an r-valued RotS function can be represented by an MTZDD having at most

$$N_R + r + 1$$

nodes, where N_R is the number of R-equivalence classes.

(Proof) Let a *valid-path* in an MTZDD be a sequence of edges and nodes leading from the root node to a terminal node representing a valid function value. Since a valid-path

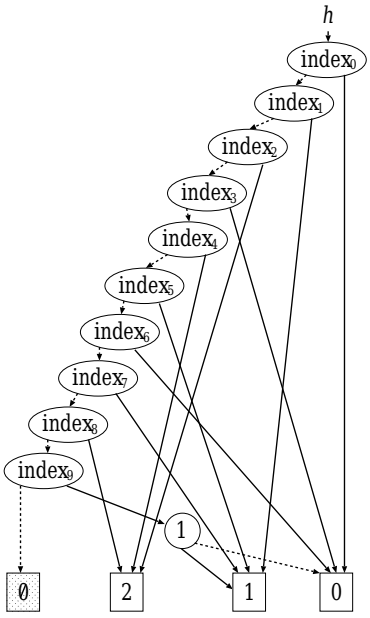

Figure 2. MTZDD for h

for h represents a tuple of an index of $\vec{\alpha}$, an index of R-equivalence class, and a function value of h, the number of distinct valid-paths is exactly N_R. On those valid-paths, N_α nodes for indices of $\vec{\alpha}$ can be shared among R-equivalence classes when an α-equivalence class has more than one R-equivalence class. Since R-equivalence classes are non-overlapping subsets of an α-equivalence class, their indices can be represented with one fewer nonterminal node than the number of R-equivalence classes. Thus, exactly N_R nonterminal nodes are needed to represent all indices. By adding $r+1$ terminal nodes to the number of nonterminal nodes, we have the theorem. ∎

The function $g_{CF}(\vec{X})$ producing an index of R-equivalence class from $\vec{X}$ is a classification function [26]. The complexity of classification functions roughly depends on the number of input vectors to be classified. To reduce the number of input vectors for g_{GF}, we delete input vectors in R-equivalence classes equal to α-equivalence classes. This is because in that case, function values can be specified by only α-equivalence classes, and we do not need to classify input vectors with R-equivalence classes.

In addition, we reduce the number of input vectors by merging α-equivalence classes having the same number of R-equivalence classes. For example, the three α-equivalence classes shown in Table V can be merged into one since we can choose a correct R-equivalence class in an α-equivalence class by converting input values. Such conversion of input values can be represented compactly by a VEVMDD, as shown in Fig. 3. In this figure, the triangle part branches by $\vec{\alpha}$, and then a rectangle part converts input values of $\vec{X}$ in

Table V
R-EQUIV. CLASSES OF 4-VARIABLE TERNARY INPUT VECTORS.

$\vec{\alpha}$			$\vec{X}$			
α_2	α_1	α_0	x_3	x_2	x_1	x_0
0	2	2	0	0	1	1
			0	1	1	0
			1	1	0	0
			1	0	0	1
			0	1	0	1
			1	0	1	0
2	0	2	0	0	2	2
			0	2	2	0
			2	2	0	0
			2	0	0	2
			0	2	0	2
			2	0	2	0
2	2	0	1	1	2	2
			1	2	2	1
			2	2	1	1
			2	1	1	2
			1	2	1	2
			2	1	2	1

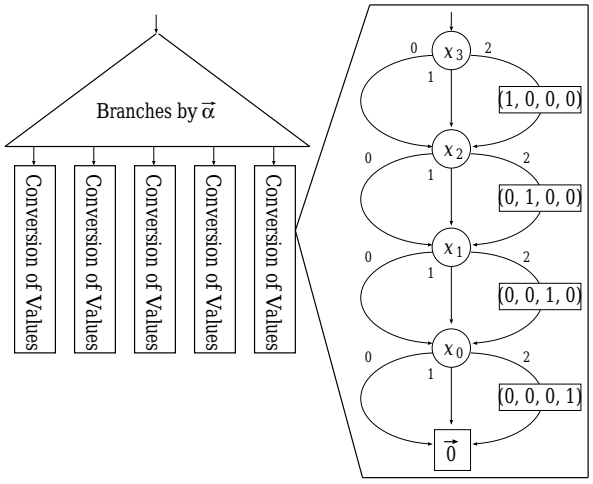

Figure 3. VEVMDD for Conversion of Input Values.

each α-equivalence class. The VEVMDD on the right side of Fig. 3 converts input vectors in $\vec{\alpha} = (2,2,0)$ into input vectors in $\vec{\alpha} = (0,2,2)$. When an α-equivalence class has only one R-equivalence class, the VEVMDD for conversion has only the terminal node.

Theorem 3: The upper bound on the number of nodes in an VEVMDD converting n-variable $\vec{X}$ according to $\vec{\alpha}$ is

$$N_\alpha(n+1),$$

where N_α is the number of α-equivalence classes.

(Proof) In Fig. 3, the triangle part chooses an α-equivalence class, and thus, it requires at most $N_\alpha - 1$ nonterminal nodes. Each α-equivalence class has a rectangle part having n nonterminal nodes. By adding one for the terminal node, we have the theorem. ∎

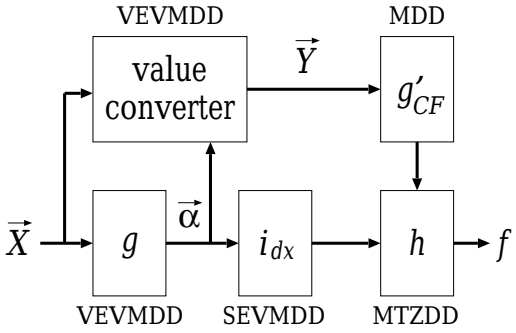

Figure 4. Decomposition Based on Equivalence Classes

Table VI
NUMBER OF NODES FOR SUB-FUNCTIONS IN DECOMPOSITION.

		Function Independent Parts					Ratio
n	r	g	i_{dx}	CV	g_{CF}	h	(%)
3	3	4	4	4	9	14.8	114
4	3	5	4	24	19	26.6	92
5	3	6	4	46	60	49.1	76
6	3	7	4	61	200	116.0	78
7	3	8	4	71	454	288.1	74
8	3	9	4	81	1,422	785.9	71
9	3	10	4	91	4,486	2,124.3	67
10	3	11	4	101	10,747	5,833.4	66
11	3	12	4	111	33,855	15,972.0	72
12	3	13	4	121	105,709	44,194.1	92
13	3	14	4	131	263,031	122,425.3	113
3	4	4	5	13	9	28.7	115
4	4	5	5	51	45	65.9	81
5	4	6	5	112	116	180.9	74
6	4	7	5	172	421	618.8	105
7	4	8	5	246	1,528	2,187.0	135
8	4	9	5	323	6,203	7,962.3	139
9	4	10	5	394	18,973	28,734.7	130
10	4	11	5	451	73,649	104,370.5	119
3	5	4	6	26	9	49.5	138
4	5	5	6	95	45	143.7	91
5	5	6	6	213	257	530.4	73
6	5	7	6	393	781	2,364.3	86
7	5	8	6	641	3,155	10,627.2	152
8	5	9	6	939	13,554	47,947.2	212
9	5	10	6	1,338	60,249	215,459.4	214

Ratio = h / (RotS in Table III) $\times$ 100

Fig. 4 shows the above decomposition of RotS functions and decision diagrams for each sub-function. In this decomposition, only the h depends on a given RotS function, while the others are independent of function values. That is, generating only the h for each RotS function is enough, and the others are the same (enough to generate them only once) for any RotS function.

V. EXPERIMENTAL RESULTS

Table VI shows the number of nodes in decision diagrams for sub-functions in decompositions of n-variable r-valued RotS functions. The column "g" shows the number of nodes in a VEVMDD for the sub-function g converting $\vec{X}$ into $\vec{\alpha}$. The column "i_{dx}" shows the number of nodes in an SEVMDD for i_{dx} producing a one-hot encoded index from $\vec{\alpha}$. The numbers of nodes for g and i_{dx} are $n+1$ and $r+1$, respectively [19]. The column "CV" shows the number of nodes in an VEVMDD for merging α-equivalence classes and converting input values $\vec{X}$ into $\vec{Y}$. The column "g'_{CF}" shows the number of nodes in an MDD for a classification function producing an index of R-equivalence class from $\vec{Y}$. The column "h" shows the average number of nodes in MTZDDs for producing function values of the 10 RotS functions same as Table III. And, the column "Ratio" shows the ratio of the size of MTZDD for h to the size of MDD in Table III. The variable order for decision diagrams is the natural order (e.g., $x_{n-1}, x_{n-2}, \ldots, x_0$).

The sub-functions g, i_{dx}, CV, and g'_{CF} are invariant for any RotS function. Thus, for each RotS function, constructing only the MTZDD for h is sufficient. As shown in Table VI, the size of MTZDDs for h can be smaller than the size of ordinary MDDs for whole RotS functions. However, as n or r increases, the number of R-equivalence classes significantly increases, and thus, the size of MTZDDs becomes larger. Along with that, the size of MDDs for classification functions g'_{CF} also becomes large. Thus, although the presented representation method based on functional decomposition is promising to construct decision diagrams for RotS functions efficiently, there is enough room for improvement to their compact representation. Since the size for g, i_{dx}, and CV is small enough, reducing the size for classification functions would be effective.

VI. CONCLUSION AND FUTURE WORK

This paper presented a decomposition method of a rotation symmetric (RotS) multiple-valued function for decision diagram-based representation. The presented decomposition method is based on equivalence classes of input vectors, and represents a RotS function using five sub-functions. Four of the five sub-functions are invariant for any RotS function. Thus, constructing a decision diagram for only one sub-function is sufficient. This paper derived some theorems on sizes of decision diagrams for the sub-functions. Experimental results using randomly generated RotS multiple-valued functions showed a possibility for compact representation of RotS functions.

When n or r is large, the size of MDD for a classification function in the decomposition method becomes large. Thus, reducing the size for classification functions would be effective for compact representation of RotS functions. That is our future work.

ACKNOWLEDGMENTS

This research was partly supported by the JSPS KAKENHI Grant (C), No.23K11038, 2024. Dr. M. Behrisch's comments at ISMVL 2023 motivated us to begin this study. The reviewers' comments were helpful in improving the paper.

REFERENCES

[1] S. B. Akers, "Binary decision diagrams," *IEEE Trans. Comput.*, Vol. C-27, No. 6, pp. 509–516, Jun. 1978.

[2] R. C. Born, "An iterative technique for determining the minimal number of variables for a totally symmetric function with repeated variables," *IEEE Trans. Comput.*, Vol. C-21, No. 10, pp. 1129–1131, 1972.

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

[4] W. Burnside, *Theory of Groups of Finite Order*, Project Gutenberg 2012.

[5] J. T. Butler, D. S. Herscovici, T. Sasao, and R. J. Barton III, "Average and worst case number of nodes in decision diagrams of symmetric multiple-valued functions," *IEEE Trans. Comput.*, Vol. 46, No. 4, pp. 491–494, Apr. 1997.

[6] J. T. Butler and T. Sasao, "On the properties of multiple-valued functions that are symmetric in both variable values and labels," *28th International Symposium on Multiple-Valued Logic 1998*, pp. 83-88, 1998.

[7] J. T. Butler and T. Sasao, "Maximally asymmetric multiple-valued functions," *49th International Symposium on Multiple-Valued Logic 2019*, pp. 188-193, 2019.

[8] A. Canteaut and M. Videau, "Symmetric Boolean functions," *IEEE Trans. Infor. Theory*, Vol. 51, No. 8, pp. 2791–2811, 2005.

[9] B. Dahlberg, "On symmetric functions with redundant variables - weighted functions," *IEEE Trans. Comput.*, Vol. C-22, No. 5, pp. 450–458, 1973.

[10] T. Kam, T. Villa, R. K. Brayton, and A. L. Sangiovanni-Vincentelli, "Multi-valued decision diagrams: Theory and applications," *Multiple-Valued Logic: An International Journal*, Vol. 4, No. 1-2, pp. 9–62, 1998.

[11] Z. Kohavi, *Switching and Finite Automata Theory*, McGraw-Hill Book Company, 1979.

[12] D. T. Lee and S. J. Hong, "An algorithm for transformation of an arbitrary switching function to a completely symmetric function," *IEEE Trans. Comput.*, Vol. C-25, No. 11, pp. 1117–1123, 1976.

[13] S. Maitra and P. Sarkar, "Maximum nonlinearity of symmetric Boolean functions on odd number of variables," *IEEE Trans. Infor. Theory*, Vol. 48, No. 9, pp. 2626–2630, 2002.

[14] S. Minato, "Zero-suppressed BDDs for set manipulation in combinatorial problems," *30th Design Automation Conference*, pp. 272–277, 1993.

[15] C. Moraga, R. S. Stanković, and J. T. Astola, "On the Reed-Muller-Fourier spectrum of multiple-valued rotation symmetric functions," *48th International Symposium on Multiple-Valued Logic*, pp.241-246, 2018.

[16] C. Moraga, M. Stanković, and R. S. Stanković, "On ternary symmetric bent functions," *50th International Symposium on Multiple-Valued Logic*, pp.76-81, 2020.

[17] S. Nagayama, T. Sasao, and J. T. Butler, "Analysis of multi-state systems with multi-state components using EVMDDs," *42nd International Symposium on Multiple-Valued Logic*, pp.122-127, May 2012.

[18] S. Nagayama, T. Sasao, and J. T. Butler, "On representation of maximally asymmetric functions based on decision diagrams," *Journal of Applied Logics – IFCoLog Journal of Logics and Their Applications*, Vol. 10, No. 6, pp. 1105-1130, Dec. 2023.

[19] S. Nagayama, T. Sasao, and J. T. Butler, "Functional decomposition of symmetric multiple-valued functions and their compact representation in decision diagrams," *IEICE Trans. on Information and Systems*, Vol. E107-D, No. 8, pp. 922–929, Aug. 2024.

[20] J. Pieprzyk and C. X. Qu, "Fast hashing and rotation symmetric functions," *Journal of Universal Computer Science*, Vol. 5, No. 1, pp. 20–31, 1999.

[21] O. S. Rothaus, "On "bent" functions," *Journal of Combinatorial Theory, Series A*, Vol. 20, Issue 3, pp. 300-305, 1976.

[22] P. Sarkar and S. Maitra, "Balancedness and correlation immunity of symmetric Boolean functions," *R. C. Bose Centenary Symp.*, Vol. 15, pp. 176–181, 2003.

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

[24] T. Sasao, *Memory-Based Logic Synthesis*, Springer, 2011.

[25] T. Sasao, "Index generation functions: recent developments (invited paper)," *41st International Symposium on Multiple-Valued Logic*, pp. 1–9, May 2011.

[26] T. Sasao, *Classification Functions for Machine Learning and Data Mining*, Springer, 2023.

[27] P. Savicky, "On the bent Boolean functions that are symmetric," *European J. Combinatorics*, Vol. 15, Issue 4, pp. 407-410, 1994.

[28] R. S. Stanković, M. Stanković, J. T. Astola, and C. Moraga, "Remarks on similarities among ternary bent functions," *49th International Symposium on Multiple-Valued Logic*, pp.79-84, 2019.

[29] P. Stănică and S. Maitra, "Rotation symmetric Boolean functions – count and cryptographic properties," *Discrete Applied Mathematics*, Vol. 156, No. 10, pp. 1567-1580, 2008.

[30] P. Stănică, T. Sasao, and J. T. Butler, "Distance duality on some classes of Boolean functions," *Journal of Combinatorial Mathematics and Combinatorial Computing*, Vol. 107, pp. 181-198, 2018.

[31] I. Stojmenovic, "On Sheffer symmetric functions in three-valued logic," *Discrete Applied Mathematics*, Vol. 22, Issue 3, pp. 267-274, 1988.

[32] N. Tokareva, *Bent functions: Results and Applications to Cryptography*, Academic Press, 2015.

[33] B. Xue, S. Nagayama, M. Inagi, and S. Wakabayashi, "A programmable architecture based on vectorized EVBDDs for network intrusion detection using random forests," *International Symposium on Nonlinear Theory and Its Applications*, pp. 132–135, 2017.

[34] S. S. Yau and Y. S. Tang, "Transformation of an arbitrary switching function to a totally symmetric function," *IEEE Trans. Comput.*, Vol. C-20, No. 12, pp. 1606–1609, 1971.