On the Number of Variables To Represent Classification Functions Using Linear Decompositions

Tsutomu Sasao
Meiji University, Kawasaki 214-8571, Japan

Abstract— A classification function f is a mapping: $D \rightarrow M$, where $D \subset B^n$, $B = \{0,1\}$, and $M = \{1, 2, \ldots, m\}$. When $|D| \ll 2^n$, f can be represented with fewer variables than n, with a linear decomposition. We show a method to estimate the number of variables to represent the function. Experimental results using randomly generated functions are shown.

I. Introduction

Consider disjoint sets of binary vectors of n bits. Such sets denote a classification function. To reduce the cost of the circuit for the function, we can use a linear decomposition. An original function can be decomposed into two parts: a linear part, and a general part, as shown in Fig. I. We assume that the cost of the linear part is $O(np)$, while the cost of the general part is $O(q2^p)$, where n is the number of variables in the original function, and p is the number of variables for the general function, where $p < n$. Since the circuit cost mainly depends on the value of p, the estimation of p is vitally important. We derive an upper bound on p, the number of variables for the general part of the decomposition. Many randomly generated classification functions were decomposed, and compared with the analysis presented in this paper.

With the result of this paper, we can answer the following:

Problem 1 *We need to design a SPAM E-mail filter with the following specification on the architecture in Fig. I.*

- *The number of IP address in the white list: 10000.*
- *The number of IP address in the black list: 10000.*
- *The number of bits to represent IP address: 128.*
- *The number of outputs is one. It shows spam ($f = 1$) or ($f = 0$) not.*

Assume that the bit patterns of IP addresses are random. Also assume that only the E-mails with the IP address in the lists are applied. Estimate the value of p.

Conventional methods [1, 3, 9, 13] cannot find such decomposition.

The rest of the paper is organized as follows: Section II shows definitions and basic properties; Section III introduces linear decompositions; Section IV derives the number of variables to represent the general part; Section V shows experimental results; Section VI surveys related works; and Section VII concludes the paper.

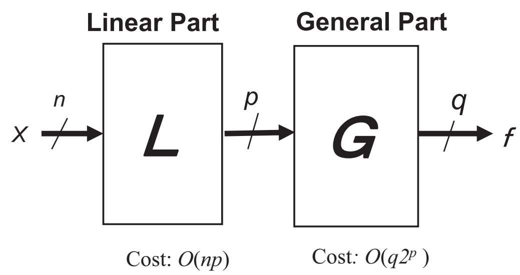

Fig. 1.1. Linear Decomposition

II. Definitions and Basic Properties

Definition 2.1 *A* **partially defined classification function** *f is a mapping $D \rightarrow M$, where $D \subset B^n$, $B = \{0,1\}$, and $M = \{1, 2, \ldots, m\}$. Assume that D contains k distinct vectors; k is called the* **weight** *of the function. These vectors are* **registered vectors***. For each registered vector, assign an integer between 1 and m, where $2 \leq m \leq k$. A* **registered vector table** *shows the* **function value** *for each registered vector. A partially defined classification function f produces the corresponding function value when the input vector matches a registered vector. Let F_i be the set of registered vectors which map to i. Then, $D = \bigcup_{i=1}^{m} F_i$. We assume that $F_i \neq \varnothing$ $(i = 1, 2, \ldots, m)$.*

Example 2.1 *The registered vector table in Table 2.1 shows a classification function with $n = 6$, $m = 2$ and $k = 8$.* ■

Definition 2.2 *[4] An m-tuple $(F_1, F_2, \ldots, F_m)$ denotes a partially defined classification function where $\bigcup_{i=1}^{m} F_i \subset B^n$ and $F_i \cap F_j = \varnothing$ for $(1 \leq i < j \leq m)$. For a partially defined function $(F_1, F_2, \ldots, F_m)$, the function $(E_1, E_2, \ldots, E_m)$ that satisfies*

$$F_i \subseteq E_i \subseteq B^n$$

TABLE 2.1
REGISTERED VECTOR TABLE.

x_1	x_2	x_3	x_4	x_5	x_6	f
1	1	0	0	1	1	1
0	1	1	0	1	1	1
0	1	0	1	0	0	1
0	0	0	0	1	0	1
1	1	0	1	1	1	2
1	0	1	1	1	1	2
1	0	0	0	1	1	2
0	0	1	0	1	0	2

TABLE 2.2
CLASSIFICATION FUNCTION WITH REDUCED VARIABLES.

x_1	x_2	x_3	x_4	f	TAG
1	1	0	0	1	1
0	1	1	0	1	2
0	1	0	1	1	3
0	0	0	0	1	4
1	1	0	1	2	5
1	0	1	1	2	6
1	0	0	0	2	7
0	0	1	0	2	8

is an **extension** *of* $(F_1, F_2, \ldots, F_m)$*, where* $E_i \cap E_j = \varnothing$ $(i \neq j)$*.*

Example 2.2 *Consider the case of* $n = 4$*. Let*

$$\begin{aligned} F_1 &= \{(0,0,0,0),(0,1,0,1),(0,1,1,0)\} \\ F_2 &= \{(0,0,1,0),(1,0,0,0),(1,0,1,1)\} \\ E_1 &= \{(0,0,0,0),(0,1,0,1),(0,1,1,0),(1,1,0,0)\} \\ E_2 &= \{(0,0,1,0),(1,0,0,0),(1,0,1,1),(1,1,0,1)\} \\ G_1 &= \{(0,0,0,0),(0,1,0,1),(0,1,1,0),(1,1,0,1)\} \\ G_2 &= \{(0,0,1,0),(1,0,0,0),(1,0,1,1),(1,1,0,1)\}. \end{aligned}$$

Then, (E_1, E_2) *is an extension of* (F_1, F_2)*, since* $F_1 \subset E_1$*,* $F_2 \subset E_2$ *and* $E_1 \cap E_2 = \varnothing$*. However,* (G_1, G_2) *is not an extension of* (F_1, F_2)*, since* $G_1 \cap G_2 \neq \varnothing$*.* ■

Definition 2.3 *For a subset* $U \subseteq B^n$ *and* $S \subseteq \{1, 2, \ldots, n\}$*, we denote by* $U|_S$ *the* **projection** *of* U *to* S*. In other words,* $U|_S = \{\vec{a}|_S\}$*, where* $\vec{a}|_S$ *is the point obtained from* $\vec{a} \in U$ *by considering only those components* a_j *with* $j \in S$*.*

Example 2.3 *Consider the case of* $n = 4$*. Let* $U = \{(1,0,0,1),(0,1,1,0),(0,0,1,0)\}$ *and* $S = \{2,3\}$*. Then, we have* $U|_S = \{(*,0,0,*),(*,1,1,*),(*,0,1,*)\}$*.* ■

Given a partially defined function, many extensions exist. In this paper, we seek the extension of f that depends on the fewest variables.

Definition 2.4 *Let* $F_i \subseteq B^n$ $(i = 1, 2, \ldots, m)$*. Given a partially defined function* $(F_1, F_2, \ldots, F_m)$*, and a subset* $S \subseteq \{1, 2, \ldots n\}$*, if* $F_i|_S \cap F_j|_S = \varnothing$*,* $(i \neq j)$ *holds, then* S *is a* **support set***. In such a case,* $(F_1|_S, F_2|_S, \ldots, F_m|_S)$ *is independent of the variable* x_j*,* $j \in \{1, 2, \ldots, n\} \setminus S$*, and the variable* x_j *is* **redundant***.*

Example 2.4 *Consider the function* (F_1, F_2) *shown in Table 2.1. In this case,* $S = \{1, 2, 3, 4\}$ *is a support set, since for*

$$\begin{aligned} F_1|_S &= \{(1,1,0,0,*,*),(0,1,1,0,*,*), \\ &\quad (0,1,0,1,*,*),(0,0,0,0,*,*)\}, \\ F_2|_S &= \{(1,1,0,1,*,*),(1,0,1,1,*,*), \\ &\quad (1,0,0,0,*,*),(0,0,1,0,*,*)\} \end{aligned}$$

$F_1|_S \cap F_2|_S = \varnothing$ *holds. Thus, this function can be represented by four variables as shown in Table 2.2.*

However, $T = \{1, 2, 3\}$ *is not a support set, since for*

$$\begin{aligned} F_1|_T &= \{(1,1,0,*,*,*),(0,1,1,*,*,*), \\ &\quad (0,1,0,*,*,*),(0,0,0,*,*,*)\}, \\ F_2|_T &= \{(1,1,0,*,*,*),(1,0,1,*,*,*), \\ &\quad (1,0,0,*,*,*),(0,0,1,*,*,*)\} \end{aligned}$$

$F_1|_T \cap F_2|_T = \{(1,1,0,*,*,*)\} \neq \varnothing$ *holds.* ■

III. LINEAR DECOMPOSITION

The number of variables of a partially defined classification function

$$f : D \to M,$$

where $D \subset B^n$ often can be reduced by a **linear decomposition** [6, 8, 14]. In the linear decomposition shown in Fig. I, L realizes a linear function, while G realizes a general function (in most cases, a non-linear function). We assume that the cost of the linear part is $O(np)$, while the cost of the general part is $O(q2^p)$.

Definition 3.1 *A* **compound variable** *has the form* $y = c_1x_1 \oplus c_2x_2 \oplus \cdots \oplus c_nx_n$*, where* $c_i \in \{0, 1\}$*. The* **compound degree** *of a variable* y *is* $\sum_{i=1}^{n} c_i$*, where* $\sum$ *denotes an ordinary integer addition, and* c_i *is treated as an integer. A* **primitive variable** *is a variable whose compound degree is one.*

Algorithms to represent a given function by using the minimum number of primitive variables have been developed [2, 5, 12].

When a partially defined function satisfies a certain condition, there exists a linear transformation that reduces the number of variables.

Definition 3.2 *In a partially defined function* $(F_1, F_2, \ldots, F_m)$*, let* $\vec{a} \in F_i$*, and* $\vec{b} \in F_j$*,* $(i \neq j)$*. The vector* $\vec{d} = \vec{a} \oplus \vec{b}$ *is a* **difference vector***. The set of distinct difference vectors is denoted by* D_f*.*

TABLE 3.1
Difference vectors for the function in Table 3.2.

x_1	x_2	x_3	x_4	Pairs of Vectors
0	0	0	1	$(1,5)$
0	1	1	1	$(1,6),(3,8)$
0	1	0	0	$(1,7),(2,8)$
1	1	1	0	$(1,8),(2,7),(3,6)$
1	0	1	1	$(2,5),(4,6)$
1	1	0	1	$(2,6),(3,7),(4,5)$
1	0	0	0	$(3,5),(4,7)$
0	0	1	0	$(4,8)$

Lemma 3.1 *[16] An n-variable partially defined function f can be represented with $n-1$ compound variables if and only if there exist a non-zero vector $\vec{u}$ such that*

$$\vec{u} \in B^n - D_f,$$

Theorem 3.1 *An n-variable classification function f can be represented with at most $n-1$ compound variables if and only if the set of difference vectors contains less than $2^n - 1$ distinct difference vectors.*

Algorithm 3.1 *(Reduction of Compound Variables [15])*

1. *Derive the set of distinct difference vectors D_f of an n-variable function.*
2. *If $|D_f| = 2^n - 1$, then stop, since reduction is impossible.*
3. *Obtain a non-zero vector $\vec{d} \in B^n - D_f$ with minimum weight.*
4. *Remove one variable from $\vec{d}$, and apply the linear transformation to the function.*
5. *Let $n \leftarrow n-1$, and go to step 1.*

Example 3.1 *Let us reduce the number of variables in the classification function shown in Table 2.2.*

1. *Table 3.1 shows the set of distinct difference vectors. The last column shows the pairs of registered vectors that produced the difference vectors. Note that 16 difference vectors were generated, but only 8 of them are distinct.*
2. $|D_f| = 8 < 2^4 - 1$.
3. *In this case, the set of difference vectors does not contain $\vec{d} = (1,1,0,0)$.*
4. *Replace x_1 and x_2 with $y_1 = x_1 \oplus x_2$, and we have a linear transformation that converts Table 2.2 into Table 3.2.*
5. *In this case, the function can be represented with three variables: $y_1 = x_1 \oplus x_2$, x_3, and x_4.*
6. *Next, we make a set of distinct difference vectors shown in Table 3.3. In this case, 16 difference vectors were generated, but only four of them are distinct. In this case, $|D_f| = 4 < 2^3 - 1$. The set of difference vectors does not contain $\vec{d} = (0,1,1)$.*
7. *Replace x_3 and x_4 with $y_2 = x_3 \oplus x_4$, and we have the linear transformation that converts Table 3.2 into Table 3.4.*
8. *In this case, the function can be represented with two variables: $y_1 = x_1 \oplus x_2$, $y_2 = x_3 \oplus x_4$.*
9. *In a similar way, the function can be represented with only one variable:*

$$y_3 = y_1 \oplus y_2 = x_1 \oplus x_2 \oplus x_3 \oplus x_4.$$

■

TABLE 3.2
Classification Function after the first linear transformation.

y_1	x_3	x_4	f	TAG
0	0	0	1	1
1	1	0	1	2
1	0	1	1	3
0	0	0	1	4
0	0	1	2	5
1	1	1	2	6
1	0	0	2	7
0	1	0	2	8

TABLE 3.3
Difference vectors for the function in Table 3.2.

y_1	x_3	x_4	Pair of Vectors
0	0	1	$(1,5),(2,6),(3,7),(4,5)$
1	1	1	$(1,6),(2,5),(3,8),(4,6)$
1	0	0	$(1,7),(2,8),(3,5),(4,7)$
0	1	0	$(1,8),(2,7),(3,6),(4,8)$

TABLE 3.4
Classification Function after the second linear transformation.

y_1	y_2	f	TAG
0	0	1	1
1	1	1	2
1	1	1	3
0	0	1	4
0	1	2	5
1	0	2	6
1	0	2	7
0	1	2	8

An efficient linear decomposition algorithm has been developed [15]. With this algorithm, functions with more than a thousand inputs have been successfully decomposed.

IV. Number of Variables to Represent General Part

This part considers a method to estimate p, the number of the variables for the general part.

Definition 4.1 *A classification function f is* **reducible** *if f can be represented with fewer variables than the original function when a linear decomposition is used. Otherwise, f is* **irreducible**.

Theorem 4.1 *An n-variable classification function f is reducible if and only if the set of difference vectors for f contains fewer than $2^n - 1$ elements.*

(Proof) This is a direct consequence of Lemma 3.1. □

Lemma 4.1 *Let $f = (F_1, F_2, \ldots, F_m)$ be a classification function. Let N be the number of distinct difference vectors. Then,*

$$N \leq \sum_{(i<j)} k_i k_j,$$

where $i, j \in \{1, 2, \ldots, m\}$, and $k_i = |F_i|$. .

Theorem 4.2 *Let N be the number of distinct difference vectors of a classification function f. Then, f can be represented with at most $p = \lfloor \log_2(N + 1) \rfloor$ compound variables.*

(Proof) By Theorem 4.1, if $N < 2^n - 1$, then we can reduce one variable. With repeated application of this theorem, the number of variables can be reduced to $p = \lfloor \log_2(N + 1) \rfloor$. When $N + 1 = 2^p$, the function can be represented with p variables. □

The above theorem gives an upper bound on p, the number of variables for the general part. Unfortunately, it is not tight. This is because 1) some of difference vectors are identical, and 2) the difference vectors are modified by the linear transformations, and some of them become identical. So, we use the following assumptions to estimate the number of distinct difference vectors.

Assumption 4.1 *Generated difference vectors are random, uniformly distributed, and independent.*

Assumption 4.2 *After the reduction of variables by linear transformation, resulting difference vectors are still random, uniformly distributed, and independent.*

Then, the number of distinct difference vectors is estimated by the **balls into bins model** [13] , where a ball corresponds to an n-bit difference vector, and a bin corresponds to a p-bit difference vector, where $p < n$. Note that there are $v = 2^p$ bins.

Lemma 4.2 *Assume that a set of N distinct n-bit vectors is reduced to that of p-bit vectors ($p < n$) by a linear transformation. Then, the estimated number of distinct p-bit vectors is*

$$2^p[1 - \exp(-\frac{N}{2^p})].$$

(Proof) Suppose that N balls are thrown independently and uniformly into $v = 2^p$ bins. Then, $\alpha = \frac{1}{v}$ is the probability of having a ball in a particular bin. Also, $\beta = 1 - \alpha$ is the probability of not having a ball in a particular bin. When we throw N balls,

1. The expected number of distinct p-bit vectors is the number of non-empty bins.
2. The probability that a particular bin has no ball is β^N.
3. The probability that a particular bin has at least one ball is $1 - \beta^N$.
4. The expected number of bins with at least one ball is

$$v[1-\beta^N] = v[1-(1-\alpha)^N] \simeq v[1-e^{-\alpha N}] = v[1-e^{-\frac{N}{v}}]$$

Here, we used the approximation $1 - \alpha \simeq e^{-\alpha}$. □

By applying Lemma 4.2 to N_1, we have

Conjecture 4.1 *Let N_1 be the number of distinct difference vectors. Then, the number of distinct p-bit difference vectors after reduction of variables is estimated by*

$$N_2 = 2^p[1 - \exp(-\frac{N_1}{2^p})].$$

Apply Theorem 4.2 to N_2 to obtain $p_2 = \lfloor \log_2(N_2+1) \rfloor$. Then, apply Lemma 4.2 again to obtain N_3:

Conjecture 4.2 *Let N_2 be the number of distinct difference vectors. Then, the number of distinct p_2-bit difference vectors after linear transformation is estimated by*

$$N_3 = 2^{p_2}[1 - \exp(-\frac{N_2}{2^{p_2}})].$$

V. Experimental Results

A. Random Functions

Classification functions with $n = 30$ were randomly generated for $m = 4$ and $m = 8$. Tables 5.1 and 5.2 show the numbers of distinct difference vectors, while Tables 5.3 and 5.4 show the numbers of variables to represent functions. k_i denotes the number of vectors such that $f(\vec{a}) = i$, where $i = 1, 2, 3, 4$, and $k_1 = k_2 = k_3 = k_4$.

N_1 denotes an upper bound on the number of distinct difference vectors, where

$$N_1 = \sum_{(i<j)} k_i k_j = 6k_i^2.$$

$\hat{N}_1$ denotes the number of distinct difference vectors (experimental result). It is equal to or nearly equal to N_1.

N_2 denotes the estimated number of distinct difference vectors specified by Conjecture 4.1.

N_3 denotes the estimated number of distinct difference vectors specified by Conjecture 4.2.

$\hat{N}_3$ denotes the number of distinct difference vectors (experimental result), after removing redundant variables.

UB_1 denotes the upper bound on the number of variables specified by $\lfloor \log_2(N_1+1) \rfloor$.

UB_2 denotes the upper bound on the number of variables specified by $\lfloor \log_2(N_2+1) \rfloor$.

$\hat{p}_1$ denotes the number of primitive variables to represent the function that is obtained by the heuristic method in [15].

UB_3 denotes the upper bound on the number of variables specified by $\lfloor \log_2(N_3+1) \rfloor$.

$\hat{p}_2$ denotes the number of compound variables to represent the function that is obtained by the Algorithm 3.1.

Note that $N_1 \geq N_2 \geq N_3$, $UB_1 \geq UB_2 \geq UB_3$, and $\hat{p}_1 \geq \hat{p}_2$.

The experimental results show that Lemma 4.2 can be used at least twice. However, it cannot be applied many times, since Assumption 4.2 becomes invalid.

Example 5.1 *Now, we can answer Problem 1.*

1. *The number of distinct elements in the difference vector set is* $N_1 \leq k^2 = 10^8$. *Since* $UB_1 = \lfloor \log_2 N_1 \rfloor = 26$, *the number of variables can be reduced to* $p_1 = 26$.

2. *Next, estimate* N_2, *the number of distinct elements in the 26-bit difference vector set. It is*

$$N_2 = 2^{p_1}[1-\exp(-\frac{N_1}{2^{p_1}})] = 51,982,368.$$

Since $UB_2 = \lfloor \log_2 N_2 \rfloor = 25$, *the number of variables can be reduce to 25.*

3. *Next, estimate* N_3, *the number of distinct elements in the 25-bit difference vector set. It is*

$$N_3 = 2^{p_2}[1-\exp(-\frac{N_2}{2^{p_2}})] = 26,426,838.$$

Since $UB_3 = \lfloor \log_2 N_3 \rfloor = 24$, *the number of variables can be reduce to 24.*

4. *A computer simulation using randomly generated vectors shows that* $\hat{p}_1 = 25$ *(without using linear part), and* $\hat{p}_2 = 22$ *(with a linear transformation).*

■

TABLE 5.1
NUMBER OF DISTINCT DIFFERENCE VECTORS ($n = 30$, $m = 4$).

k_i	N_1	$\hat{N}_1$	N_2	N_3	$\hat{N}_3$
10	600	600	354	192	125
20	2400	2400	1414	766	507
40	9600	9598	5654	3066	2021
80	38400	38360	22617	12264	8124
160	152600	152887	90468	49056	32470
320	614400	603310	361871	196222	129853

TABLE 5.2
NUMBERS OF DISTINCT DIFFERENCE VECTORS ($n = 30$, $m = 8$).

k_i	N_1	$\hat{N}_1$	N_2	N_3	$\hat{N}_3$
10	2800	2800	1526	793	967
20	11200	11198	6104	3173	2041
40	44800	44743	24418	12693	15345
80	179200	178224	97672	50771	97932
160	716800	701818	390688	203085	245376
320	2867200	2635368	1562750	812340	522099

TABLE 5.3
NUMBER OF VARIABLES TO REPRESENT FUNCTIONS ($n = 30$, $m = 4$).

k_i	UB_1	UB_2	$\hat{p}_1$	UB_3	$\hat{p}_2$
10	9	8	7	7	6
20	11	10	9	9	8
40	13	12	11	11	10
80	15	14	13	13	12
160	17	16	15	15	13
320	19	18	17	17	15

TABLE 5.4
NUMBER OF VARIABLES TO REPRESENT FUNCTIONS ($n = 30$, $m = 8$).

k_i	UB_1	UB_2	$\hat{p}_1$	UB_3	$\hat{p}_2$
10	11	10	10	9	9
20	13	12	11	11	10
40	15	14	14	13	12
80	17	16	17	15	14
160	19	18	18	17	15
320	21	20	20	19	17

B. Non Random Functions

MNIST MNIST data set [7] consists of handwritten digits with 6×10^4 images. Each image is represented by $28 \times 28 = 784$ pixels of 8-bit grayscale. We converted them into binary ones by setting the threshold to 96. In this way, we had an n-variable binary-input m-valued classification function, where $n = 28 \times 28 = 784$, and $m = 10$. During this process, we removed duplicated data, to obtain the set with $k = 59981$ images. The distribution of classes is $(k_0, k_1, \ldots, k_9) = (5923, 6726, 5958, 6131, 5842, 5421, 5918, 6265, 5851, 5949)$.

Telephone Numbers consists of 9-digit telephone numbers of 3700 Japanese companies [16]. The outputs

TABLE 5.5
Numbers of variables to represent MNIST and Telephone numbers.

Name	k	UB_1	UB_2	$\hat{p}_1$	UB_3	$\hat{p}_2$
MNIST	59981	30	29	37	28	25
Telephone	3700	22	21	21	20	18

show the names of the stock exchange. There are $m = 9$ different stock exchange. 1) Tokyo 1st; 2) Tokyo 2nd; 3) Tokyo Mothers; 4) Sapporo; 5) Nagoya; 6) Fukuoka; 7) JASDAQ; 8) REIT; and 9) Foreign. We converted them to binary numbers of 30-bits to obtain an $n = 30$ variable $m = 9$ valued classification function with weight $k = 3700$. The distribution of classes is $(k_1, k_2, \ldots, k_9) = (2018, 522, 239, 19, 75, 29, 739, 59, 0)$

Table 5.5 shows the numbers of variables to represent these functions. The table for N_i is omitted due to space limitation.

In the case of MNIST, the estimated values for N_i were far from the experimental values. The estimated number of input variables is $UB_2 = 29$, while the actual number of inputs is $\hat{p}_i = 37$. This shows that MNIST is far from a random function. However, the liner transformation converted the function into random. The experimental result is $\hat{p}_2 = 25$, while estimated value is $UB_3 = 28$.

In the case of telephone numbers, the estimated values are not so far from the experimental values. So, the function is near random.

VI. Related Works

The number of variables to represent partially defined functions has been studied for many years. [11] derived formulas for the number of variables to represent multi-valued input functions. [12] and [13] showed that most index generation functions with weight k can be represented with $\lceil 2\log_2(k+1)\rceil - 1$ primitive variables. In [14], an efficient algorithm to find a linear decomposition was developed, and after that decomposition of functions with more than a thousand variables become possible [15], [16].

VII. Conclusion

This paper shows a method to estimate the number of variables to represent a classification function after linear decompositions. Experimental results using randomly generated functions showed the usefulness of the approach.

Acknowledgments

This work was supported in part by a Grant in Aid for Scientific Research of the JSPS. The author thanks Prof. Jon T. Butler for discussion.

References

[1] R. L. Ashenhurst, "The decomposition of switching functions," *International Symposium on the Theory of Switching*, pp. 74-116, April 1957.

[2] F. M. Brown, *Boolean Reasoning: The Logic of Boolean Equations*, Kluwer Academic Publishers, Boston, 1990.

[3] H. A. Curtis, *A New Approach to the Design of Switching Circuits*, D. Van Nostrand Co., Princeton, NJ, 1962.

[4] T. Ibaraki,"Partially defined Boolean functions," Chapter 8 in: Y. Crama and P. L. Hammer, *Boolean Functions - Theory, Algorithms and Applications*, Cambridge University Press, New York, 2011.

[5] J. Kuntzmann, *Algèbre de Boole*, Dunod, Paris, 1965. English translation: *Fundamental Boolean Algebra*, Blackie and Son Limited, London and Glasgow, 1967.

[6] R. J. Lechner, "Harmonic analysis of switching functions,"in A. Mukhopadhyay (ed.), *Recent Developments in Switching Theory*, Academic Press, New York, 1971.

[7] http://yann.lecun.com/exdb/mnist/

[8] E. I. Nechiporuk, "On the synthesis of networks using linear transformations of variables,"*Dokl. AN SSSR*, vol. 123, no. 4, Dec. 1958, pp. 610-612 (in Russian).

[9] T. Sasao, "Totally undecomposable functions: applications to efficient multiple-valued decompositions," *ISMVL-1999*, Freiburg, Germany, May 20-23, 1999, pp. 59-65.

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

[11] T. Sasao, "On the number of dependent variables for incompletely specified multiple-valued functions," *30th International Symposium on Multiple-Valued Logic*, pp. 91-97, Portland, Oregon, U.S.A., May 23-25, 2000.

[12] T. Sasao, "On the number of variables to represent sparse logic functions," *ICCAD-2008*, San Jose, California, USA, Nov.10-13, 2008, pp. 45-51.

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

[14] T. Sasao, *Index Generation Functions*, Morgan & Claypool, Oct. 2019.

[15] T. Sasao, "Reduction methods of variables for large-scale classification functions," *International Workshop on Logic and Synthesis* (IWLS-2020), July 27-29, 2020, pp. 82-87.

[16] T. Sasao, "On the minimization of partially defined classification functions," *International Symposium on Multiple-Valued Logic*, (ISMVL-2020), Nov. 9-11, 2020, Miyazaki, Japan, pp. 117-123.