2013 IEEE 43rd International Symposium on Multiple-Valued Logic

An Application of Autocorrelation Functions to Find Linear Decompositions for Incompletely Specified Index Generation Functions

Tsutomu Sasao
Dept. of Computer Science
Meiji University
Kawasaki, Kanagawa 214-8571, Japan
sasao@ieee.org

***Abstract*—This paper shows that autocorrelation functions are useful to find the number of variables to represent incompletely specified index generation functions. It also shows a strategy to reduce the number of variables to represent incompletely specified index generation functions in the autocorrelation domain.**

***Keywords*-logic minimization; linear transformation; incompletely specified logic function; autocorrelation; decomposition**

I. Introduction

Various methods exist to decompose logic functions into linear and non-linear parts. They are used to simplify AND-OR logic circuits [9], [7], [3], [5], [20], or to simplify binary decision diagrams [4], [6], [8].

To find a linear part of a decomposition, Walsh spectrum and autocorrelation functions are used. Since autocorrelations are invariant under a linear transformation of the input variables [10], they are useful for linear decompositions.

In this paper, we use linear decompositions to realize functions for which only k combinations of inputs are defined. For this class of functions, the circuit sizes can be reduced drastically with linear decompositions. To implement the linear part, we use a special programmable circuit that consists of registers, multiplexers, and EXOR gates. To implement the general part, we use a look-up table (LUT). Thus, Fig. 1.1 can be a reconfigurable circuit.

Given an incompletely specified function, the problem of the linear decomposition is to obtain a linear transformation that minimizes the number of variables p for the general part. When a given function is defined for only k input combinations, the number of variables can be reduced to $2\lceil \log_2 k \rceil - 3$, in many cases, and by this, the size of the LUT is reduced drastically.

In this paper, we show that the minimization of variables for incompletely specified index generation functions can be done in the autocorrelation function domain. The rest of the paper is organized as follows: Section 2 defines index generation functions. Section 3 shows the number of variables to represent incompletely specified index generation functions. Section 4 shows a method to reduce the number of variables by linear transformations. Section 5 introduces autocorrelation functions. Section 6 shows an algorithm to compute autocorrelation function. Section 7 shows a method to find good linear transformations, and Section 8 summarizes the paper.

II. Index Generation Function

Definition 2.1: [13], [17] Consider a set of k different vectors of n bits. These vectors are **registered vectors**. For each registered vector, assign a unique integer from 1 to k. A **registered vector table** shows an **index** for each registered vector. An **incompletely specified index generation function** produces a corresponding index when the input vector matches a registered vector. Otherwise, the value of the function is undefined (d, *don't care*). The incompletely specified index generation function represents a mapping $M \rightarrow \{1, 2, \ldots, k\}$, where $M \subset B^n$ denotes the set of registered vectors. k is the **weight** of the function.

Example 2.1: Consider the registered vectors shown in Table 2.1. These vectors show an index generation function with weight $k = 7$. ■

Index generation functions are useful for address tables for the Internet, terminal access controllers of local area networks, databases, memory patch circuits, text compressions, password tables, code converters [13], [16], [17], and computer virus scanning engines. In many cases, functions must be updated frequently. Thus, a reconfigurable architecture such as Fig. 1.1 is desirable for the implementation.

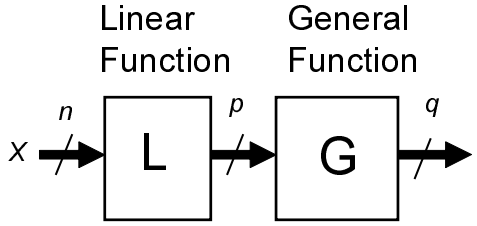

Figure 1.1. Linear Decomposition.

0195-623X/13 $26.00 © 2013 IEEE
DOI 10.1109/ISMVL.2013.12

Table 2.1
REGISTERED VECTOR TABLE

Vector							Index
x_1	x_2	x_3	x_4	x_5	x_6	x_7	
1	0	0	0	0	0	0	1
0	1	0	0	0	0	0	2
0	0	1	0	0	0	0	3
0	0	0	1	0	0	0	4
0	0	0	0	1	0	0	5
0	0	0	0	0	1	0	6
0	0	0	0	0	0	1	7

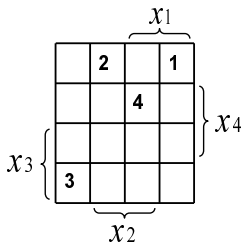

Figure 3.1. Index Generation Function of 4 variables.

III. NUMBER OF VARIABLES TO REPRESENT INCOMPLETELY SPECIFIED FUNCTIONS

In an incompletely specified function f, *don't care* values can be chosen either as 0 or 1, to minimize the number of variables to represent f. This property is useful to realize a function using a smaller look-up table (LUT).

Theorem 3.1: Suppose that an incompletely specified function f is represented by a decomposition chart [11]. If each column has at most one *care* element, then the function can be represented by using only the column variables.

(Proof) In each column, let the values of *don't cares* elements be set to the value of the *care* element in the column, then the function depends only the column variables. □

Example 3.1: Consider the decomposition chart shown in Fig. 3.1, where x_1 and x_2 specify the columns, and x_3 and x_4 specify the rows, and blank elements denote *don't cares*. Note that in Fig. 3.1, each column has at most one *care* element. Thus, this function can be represented by only the column variables x_1 and x_2:

$$F = 1 \cdot x_1\bar{x}_2 \vee 2 \cdot \bar{x}_1x_2 \vee 3 \cdot \bar{x}_1\bar{x}_2 \vee 4 \cdot x_1x_2.$$

■

Algorithms to minimize the number of variables in incompletely specified functions have been developed [1], [2], [12], [14]. As for incompletely specified index generation functions, we have the following [16]:

Conjecture 3.1: When the number of the input variables is sufficiently large, most incompletely specified index generation functions with weight k (≥ 8) can be represented by $p = 2\lceil \log_2 k \rceil - 3$ variables.

There exist exceptions. An example of the *exceptions* is:

Example 3.2: Consider the registered vectors shown in Table 2.1. It shows an index generation function with weight $k = 7$. To distinguish these seven vectors, 6 variables are necessary. ■

However, for such functions, as shown in the next section, by a linear transformation, the number of variables can be reduced. Thus, when we use linear transformations, almost all functions can be realized with the number of variables given by Conjecture 3.1.

As for the lower bound on the number of variables, we have the following:

Theorem 3.2: [18] To represent any incompletely specified index generation function f with weight k, at least $q = \lceil \log_2 k \rceil$ variables are necessary.

Thus, when the weight k of an n-variable index generation function is greater than 2^{n-1}, we cannot reduce the number of variables.

IV. REDUCTION OF THE NUMBER OF VARIABLES BY LINEAR TRANSFORMATIONS

In the previous section, we showed a method to reduce the number of variables for incompletely specified functions. Unfortunately, the applicability of such method is limited. In this section, we show that more variables can be reduced by using linear transformations.

Example 4.1: For the function in Table 2.1, the numbers of 1's in the registered vectors are all one. Thus, the number of variables for the function can be reduced to six: Any one variable can be removed. If we remove x_7, then we have:

$$\begin{aligned} F = {} & 1 \cdot x_1\bar{x}_2\bar{x}_3\bar{x}_4\bar{x}_5\bar{x}_6 \vee 2 \cdot \bar{x}_1x_2\bar{x}_3\bar{x}_4\bar{x}_5\bar{x}_6 \\ & \vee 3 \cdot \bar{x}_1\bar{x}_2x_3\bar{x}_4\bar{x}_5\bar{x}_6 \vee 4 \cdot \bar{x}_1\bar{x}_2\bar{x}_3x_4\bar{x}_5\bar{x}_6 \\ & \vee 5 \cdot \bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4x_5\bar{x}_6 \vee 6 \cdot \bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4\bar{x}_5x_6 \\ & \vee 7 \cdot \bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4\bar{x}_5\bar{x}_6. \end{aligned}$$

However, we cannot remove two or more variables simultaneously. ■

Definition 4.1: A **linear transformation** is defined as

$$\begin{aligned} y_1 &= c_{11}x_1 \oplus c_{12}x_2 \oplus c_{13}x_3 \oplus \ldots \oplus c_{1n}x_n, \\ y_2 &= c_{21}x_1 \oplus c_{22}x_2 \oplus c_{23}x_3 \oplus \ldots \oplus c_{2n}x_n, \\ y_3 &= c_{31}x_1 \oplus c_{32}x_2 \oplus c_{33}x_3 \oplus \ldots \oplus c_{3n}x_n, \\ &\vdots \\ y_p &= c_{p1}x_1 \oplus c_{p2}x_2 \oplus c_{p3}x_3 \oplus \ldots \oplus c_{pn}x_n, \end{aligned}$$

where $c_{ij} \in \{0, 1\}$.

Definition 4.2: Given an incompletely specified index generation function, a linear transformation that minimizes the number of variables is **optimum**.

By Theorem 3.2, if the linear transformation reduces the number of variables to $q = \lceil \log_2 k \rceil$ variables, then it is optimum.

Table 4.1
REGISTERED VECTORS AFTER LINEAR TRANSFORMATION

Vector			Index
y_4	y_2	y_1	
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Example 4.2: For the function in Table 2.1, consider the linear transformation:

$$\begin{aligned} y_1 &= x_1 \oplus x_3 \oplus x_5 \oplus x_7, \\ y_2 &= x_2 \oplus x_3 \oplus x_6 \oplus x_7, \\ y_4 &= x_4 \oplus x_5 \oplus x_6 \oplus x_7. \end{aligned}$$

The transformed registered vectors are shown in Table 4.1. In this case, three variables (y_4, y_2, y_1) distinguish 7 vectors. Note that this is an optimum transformation. ∎

Definition 4.3: Consider an incompletely specified index generation function of n variables:

$$F : M \to \{1, 2, \ldots, k\}, M \subset \{0,1\}^n.$$

The corresponding **characteristic logic function** is

$$f : \{0,1\}^n \to \{0,1\},$$

where

$$f(\vec{x}) = \begin{cases} 1 & (\vec{x} \in M) \\ 0 & (\text{Otherwise}). \end{cases}$$

In other words, the characteristic logic function is a characteristic function of M.

Example 4.3: Fig. 4.1 shows the characteristic logic function for the incompletely specified index generation function shown in Fig. 3.1. ∎

Note that two index generation functions whose indices are permutated have the same characteristic logic function. Also, they require the same number of variables in linear decompositions. From this, we have the following:

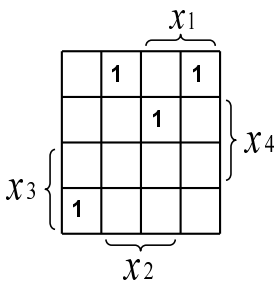

Figure 4.1. Characteristic Logic Function for Fig. 3.1.

Lemma 4.1: Incompletely specified index generation functions sharing the same characteristic logic functions require the same number of variables in their linear decompositions.

V. AUTOCORRELATION FUNCTION

In this part, we show that an autocorrelation function is useful to find the number of variables to represent an incompletely specified index generation function.

Definition 5.1: Let f be an n-variable logic function, and $\vec{\tau} \in \{0,1\}^n$. Then, the the **autocorrelation function** is

$$B_f(\vec{\tau}) = \sum_{\vec{v} \in \{0,1\}^n} f(\vec{v}) \cdot f(\vec{v} \oplus \vec{\tau}).$$

Definition 5.2: Let $\vec{e}_i = (0, \ldots, 0, 1, 0, \ldots, 0)$ be the **unit vector** whose i-th element is 1.

First, we consider the condition that a single variable can be removed.

Lemma 5.1: Let f be the characteristic logic function of an incompletely specified index generation function F. Then, F can be represented without x_i iff $B_f(\vec{e}_i) = 0$.

(Proof) Consider the decomposition chart, where the row variable is x_i, and the column variables are the remaining variables. Suppose that F can be represented without using x_i. In this case, each column has at most one non-zero element. This means that $f(\vec{\tau}) \cdot f(\vec{\tau} \oplus \vec{e}_i) = 0$ for each $\vec{\tau}$. In other words, $B_f(\vec{e}_i) = 0$. Next, assume that $B_f(\vec{e}_i) = 0$. In this case, we have $f(\vec{\tau}) \cdot f(\vec{\tau} \oplus \vec{e}_i) = 0$ for each $\vec{\tau}$. This means that F can be represented without using x_i. □

The condition that two variables can be removed simultaneously is:

Lemma 5.2: Let f be the characteristic logic function of an incompletely specified index generation function F. Then, F can be represented without x_i and x_j iff $B_f(\vec{e}_i) = 0$, $B_f(\vec{e}_j) = 0$, and $B_f(\vec{e}_i \vee \vec{e}_j) = 0$.

Example 5.1: First, consider the incompletely specified index generation function shown in Fig. 5.1. The non-zero coefficients of the autocorrelation function is shown in Table 5.1. The last column of the table denotes the weight of $\vec{\tau}$, the number of 1's in the vector. Since $B_f(\vec{e}_i) = 0$ for $i = 1, 2, 3, 4$, any single variable can be removed. However, since $B_f(\vec{e}_i \vee \vec{e}_j) \neq 0$ for $(1 \leq i < j \leq 4)$, two variables cannot be removed simultaneously. Thus, we need at least three variables to represent F.

Second, consider the transformation:

$$\begin{aligned} x_1 &\Leftarrow x_1 \oplus x_4, \\ x_2 &\Leftarrow x_2 \oplus x_4. \end{aligned}$$

The map for the transformed function is shown in Fig. 3.1. Table 5.2 shows the non-zero coefficients of the autocorrelation function. In this case,

$$B_f(\vec{e}_3) = B_f(\vec{e}_4) = B_f(\vec{e}_3 \vee \vec{e}_4) = 0.$$

Thus, we can remove two variables x_3 and x_4 simultaneously, and the function can be represented with only x_1 and x_2.

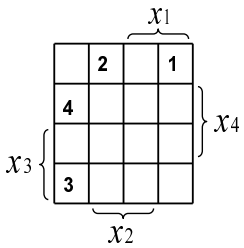

Figure 5.1. Index Generation Function of 4 variables.

Table 5.1
AUTOCORRELATION COEFFICIENTS FOR FIG. 5.1

$\vec{\tau}$				$B_f(\vec{\tau})$	$\lvert\vec{\tau}\rvert$
x_1	x_2	x_3	x_4		
0	0	0	0	4	0
0	0	1	1	2	2
0	1	0	1	2	2
0	1	1	0	2	2
1	0	0	1	2	2
1	0	1	0	2	2
1	1	0	0	2	2

Third, consider the transformation in Fig. 3.1:

$$x_2 \quad \Leftarrow \quad x_2 \oplus x_3.$$

We have the function shown in Fig. 5.2. The autocorrelation function is shown in Table 5.3. In this case, $B_f(\vec{e}_3) = 2$. This means that to represent f, we need x_3. However, we can remove any one of the other variables. ∎

In general, the condition to remove s variables simultaneously is given by

Theorem 5.1: Let f be the characteristic logic function of an incompletely specified index generation function F. Then, F can be represented without x_{t_1}, $x_{t_2}, \ldots$, and x_{t_s}, iff $B_f(\vec{t}) = 0$, where $\vec{t} = a_1\vec{e}_{t_1} \vee a_2\vec{e}_{t_2} \vee \ldots \vee a_s\vec{e}_{t_s}$ for

Table 5.2
AUTOCORRELATION COEFFICIENTS FOR FIG. 3.1

$\vec{\tau}$				$B_f(\vec{\tau})$	$\lvert\vec{\tau}\rvert$
x_1	x_2	x_3	x_4		
0	0	0	0	4	0
1	1	1	1	2	4
1	0	0	1	2	2
0	1	1	0	2	2
0	1	0	1	2	2
1	0	1	0	2	2
1	1	0	0	2	2

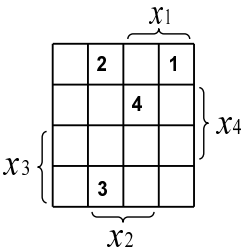

Figure 5.2. Index Generation Function of 4 variables.

Table 5.3
AUTOCORRELATION COEFFICIENTS FOR FIG. 5.2

$\vec{\tau}$				$B_f(\vec{\tau})$	$\lvert\vec{\tau}\rvert$
x_1	x_2	x_3	x_4		
0	0	0	0	4	0
0	1	0	1	2	2
0	0	1	0	2	1
1	0	0	1	2	2
1	1	1	0	2	3
1	1	0	0	2	2
1	0	1	1	2	3

Table 5.4
AUTOCORRELATION COEFFICIENTS FOR THE FUNCTION IN TABLE 2.1

$\vec{\tau}$							$B_f(\vec{\tau})$	$\lvert\vec{\tau}\rvert$
x_1	x_2	x_3	x_4	x_5	x_6	x_7		
0	0	0	0	0	0	0	7	0
0	0	0	0	0	1	1	2	2
0	0	0	0	1	0	1	2	2
0	0	0	0	1	1	0	2	2
0	0	0	1	0	0	1	2	2
0	0	0	1	0	1	0	2	2
0	0	0	1	1	0	0	2	2
0	0	1	0	0	0	1	2	2
0	0	1	0	0	1	0	2	2
0	0	1	0	1	0	0	2	2
0	0	1	1	0	0	0	2	2
0	1	0	0	0	0	1	2	2
0	1	0	0	0	1	0	2	2
0	1	0	0	1	0	0	2	2
0	1	0	1	0	0	0	2	2
0	1	1	0	0	0	0	2	2
1	0	0	0	0	0	1	2	2
1	0	0	0	0	1	0	2	2
1	0	0	0	1	0	0	2	2
1	0	0	1	0	0	0	2	2
1	0	1	0	0	0	0	2	2
1	1	0	0	0	0	0	2	2

$a_i \in \{0, 1\}$, $\vec{t} \neq \vec{0}$.

Example 5.2: Consider the incompletely specified index generation function shown in Table 2.1. The non-zero coefficients of the autocorrelation function are shown in Table 5.4. Since $B_f(\vec{e}_i) = 0$, for $i = 1, 2, \ldots, 7$, any single variable can be removed. However, since $B_f(\vec{e}_i \vee \vec{e}_j) \neq 0$ for $(1 \leq i < j \leq 7)$, two variables cannot be removed simultaneously.

Next, consider the transformation:

$$\begin{aligned} x_1 \quad &\Leftarrow \quad x_1 \oplus x_3 \oplus x_5 \oplus x_7, \\ x_2 \quad &\Leftarrow \quad x_2 \oplus x_3 \oplus x_6 \oplus x_7, \\ x_4 \quad &\Leftarrow \quad x_4 \oplus x_5 \oplus x_6 \oplus x_7. \end{aligned}$$

The non-zero coefficients of the autocorrelation function is shown in Table 5.5. In this case, $B_f(\vec{\tau}) = 0$, where

$$\vec{\tau} = a_1\vec{e}_3 \vee a_2\vec{e}_5 \vee a_3\vec{e}_6 \vee a_4\vec{e}_7,$$

$a_i \in \{0, 1\}$, and $\vec{\tau} \neq \vec{0}$.

Thus, four variables x_3, x_5, x_6, x_7 can be removed simultaneously, and the function can be represented with only x_1, x_2, and x_4. ∎

VI. ALGORITHM TO COMPUTE AUTOCORRELATION FUNCTIONS

Theorem 6.1: Let M be the set of binary vectors corresponding to the minterms of f. Let D_f be the set of vectors

Table 5.5
AUTOCORRELATION COEFFICIENTS FOR TRANSFORMED FUNCTION

$\vec{\tau}$							$B_f(\vec{\tau})$	$\lvert\vec{\tau}\rvert$
x_1	x_2	x_3	x_4	x_5	x_6	x_7		
0	0	0	0	0	0	0	7	0
1	0	0	0	0	1	1	2	3
0	1	0	0	1	0	1	2	3
1	1	0	0	1	1	0	2	4
1	1	0	0	0	0	1	2	3
0	1	0	0	0	1	0	2	2
1	0	0	0	1	0	0	2	2
0	0	1	1	0	0	1	2	3
1	0	1	1	0	1	0	2	4
0	1	1	1	1	0	0	2	4
1	1	1	1	0	0	0	2	4
1	0	0	1	0	0	1	2	3
0	0	0	1	0	1	0	2	2
1	1	0	1	1	0	0	2	4
0	1	0	1	0	0	0	2	2
1	0	1	0	0	0	0	2	2
0	1	0	1	0	0	1	2	3
1	1	0	1	0	1	0	2	4
0	0	0	1	1	0	0	2	2
1	0	0	1	0	0	0	2	2
0	1	1	0	0	0	0	2	2
1	1	0	0	0	0	0	2	2

$\vec{a} \oplus \vec{b}$, where $\vec{a}, \vec{b} \in M$, and $a \neq b$. Then,

$$B_f(\vec{\tau}) = \begin{cases} |M| & \text{if } \vec{\tau} = \vec{0} \\ \geq 2 & \text{if } \vec{\tau} \in D_f \\ 0 & \text{Otherwise} \end{cases}$$

(Proof) The minterm expansion of f is

$$f = \bigvee_{\vec{a} \in M} \vec{x}^{\vec{a}},$$

where $\vec{x}^{\vec{a}} = 1$ iff $\vec{x} = \vec{a}$. In this case, we have

$$\begin{aligned} B_f(\vec{\tau}) &= \sum_{\vec{v} \in \{0,1\}^n} [\bigvee_{\vec{a} \in M} \vec{v}^{\vec{a}}] \cdot [\bigvee_{\vec{b} \in M} (\vec{v} \oplus \vec{\tau})^{\vec{b}}] \\ &= \bigvee_{\vec{a} \in M} \bigvee_{\vec{b} \in M} B_{(\vec{a},\vec{b})}(\vec{\tau}), \end{aligned}$$

where $B_{(\vec{a},\vec{b})}(\vec{\tau}) = 1$ iff $\tau = \vec{a} \oplus \vec{b}$. Thus, we have $B_f(\vec{0}) = |M|$, and $B_f(\vec{a} \oplus \vec{b}) \geq 2$, where $\vec{a} \neq \vec{b}$ and $\vec{a}, \vec{b} \in M$. It is clear that $B_f(\tau) = 0$ for other vectors. □

Example 6.1: Consider the function shown in Fig. 5.2. The set of vectors corresponding to the minterms for f is $M = \{(1,0,0,0), (0,1,0,0), (0,1,1,0), (1,1,0,1)\}$. Thus, $D_f = \{(1,1,0,0), (1,1,1,0), (0,1,0,1), (0,0,1,0), (1,0,0,1), (1,0,1,1)\}$. Also, $B_f(\vec{0}) = |M| = 4$. Note that these correspond to Table 5.3. ■

Thus, when $|M| = k$, the number of non-zero coefficients of the autocorrelation function is at most $\binom{k}{2} + 1 = \frac{k(k-1)}{2} + 1$.

It is very interesting to note that D_f corresponds to the *difference matrix* used in [19].

VII. A METHOD TO FIND GOOD LINEAR TRANSFORMATIONS

A. Minimal Sets of Variables

From the results of the previous sections, we have the following:

Algorithm 7.1: (Expression Showing Minimal Sets of Variables for an Incompletely Specified Index Generation Function)

1) Let f be the characteristic logic function of the index generation function F.
2) Let D_f be the set of non-zero vectors defined in Theorem 6.1.
3) For each vector in D_f, make a product by replacing 1 with $\bar{x}_i$, where i denotes the index of the variable, and by replacing 0 with a missing variable.
4) Derive the sum-of-products expression (SOP) $\bar{R}$ consisting above products.
5) Obtain the SOP for R.
6) Each product shows a minimal set of variables to represent F.

Note that R is the **covering function** defined in [16].

Example 7.1:
1) Consider the function shown in Fig. 5.2.
2) Table 5.3 shows D_f.
3) The set of products to representing vectors in D_f is $\{\bar{x}_2\bar{x}_4, \bar{x}_3, \bar{x}_1\bar{x}_4, \bar{x}_1\bar{x}_2\bar{x}_3, \bar{x}_1\bar{x}_2, \bar{x}_1\bar{x}_3\bar{x}_4\}$.
4) The SOP for $\bar{R}$ is

$$R = \bar{x}_2\bar{x}_4 \vee \bar{x}_3 \vee \bar{x}_1\bar{x}_4 \vee \bar{x}_1\bar{x}_2\bar{x}_3 \vee \bar{x}_1\bar{x}_2 \vee \bar{x}_1\bar{x}_3\bar{x}_4.$$

5) The SOP for R is

$$R = x_2x_3x_4 \vee x_1x_3x_4 \vee x_1x_2x_3.$$

6) Minimum sets of variables to represent F are $\{x_2, x_3, x_4\}$, $\{x_1, x_3, x_4\}$, and $\{x_1, x_2, x_3\}$.

■

Example 7.2:
1) Consider the non-zero coefficients of the autocorrelation function shown in Table 5.5.
2) The SOP for $\bar{R}$ is

$$\begin{aligned} \bar{R} &= \bar{x}_1\bar{x}_6\bar{x}_7 \vee \bar{x}_2\bar{x}_5\bar{x}_7 \vee \bar{x}_1\bar{x}_2\bar{x}_5\bar{x}_6 \\ &\vee \bar{x}_1\bar{x}_2\bar{x}_7 \vee \bar{x}_2\bar{x}_6 \vee \bar{x}_1\bar{x}_5 \vee \bar{x}_3\bar{x}_4\bar{x}_7 \\ &\vee \bar{x}_1\bar{x}_3\bar{x}_4\bar{x}_6 \vee \bar{x}_2\bar{x}_3\bar{x}_4\bar{x}_5 \vee \bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4 \\ &\vee \bar{x}_1\bar{x}_4\bar{x}_7 \vee \bar{x}_4\bar{x}_6 \vee \bar{x}_1\bar{x}_2\bar{x}_4\bar{x}_5 \vee \bar{x}_2\bar{x}_4 \\ &\vee \bar{x}_1\bar{x}_3 \vee \bar{x}_2\bar{x}_4\bar{x}_7 \vee \bar{x}_1\bar{x}_2\bar{x}_4\bar{x}_6 \vee \bar{x}_4\bar{x}_5 \\ &\vee \bar{x}_1\bar{x}_4 \vee \bar{x}_2\bar{x}_3 \vee \bar{x}_1\bar{x}_2. \end{aligned}$$

3) The SOP for R is

$$\begin{aligned} R &= x_2x_3x_4x_5x_6 \vee x_1x_3x_4x_5x_6 \vee x_1x_2x_3x_5x_6 \\ &\vee x_2x_3x_4x_5x_7 \vee x_1x_3x_4x_6x_7 \vee x_1x_2x_5x_6x_7 \\ &\vee x_1x_2x_4. \end{aligned}$$

4) The minimum set of variables to represent F is $\{x_1, x_2, x_4\}$.

■

B. Linear Transformations in the Autocorrelation Domain

Let $|M| = k$. Then, the number of non-zero coefficients in the autocorrelation function is at most $\frac{k(k-1)}{2} + 1$. When $k << 2^n$, this value is not so large, and we can search for a good linear transformation in the autocorrelation domain instead of the original domain.

For example, Table 5.2 can be obtained from Table 5.1 by the linear transformation:

$$\begin{aligned} x_1 &\Leftarrow x_1 \oplus x_4, \\ x_2 &\Leftarrow x_2 \oplus x_4. \end{aligned}$$

Also, Table 5.3 can be obtained from Table 5.2 by the linear transformation:

$$x_2 \Leftarrow x_2 \oplus x_3.$$

C. Strategy to Find Good Linear Transformations in the Autocorrelation Domain

Since D_f contains all the necessary information, we can work on D_f. If $B_f(\vec{e}_s) \neq 0$, then we cannot remove the variable x_s. So, in the autocorrelation domain, we try to find the linear transformation $x_i \Leftarrow x_i \oplus x_j$ that modifies such vectors to increase their weights.

To remove x_i and x_j simultaneously, we have to remove the vectors $\vec{e}_i$, $\vec{e}_j$, and $\vec{e}_i \vee \vec{e}_j$ from D_f by increasing the weight of such vectors, if any of such vectors exist in D_f.

Thus, the strategy is to **increase the total number of 1's in the vectors of** D_f.

Example 7.3: Consider the function in Table 2.1. To represent this function, we need 6 variables. Let μ be the total number of 1's in the vectors. Then $\mu = 42$. By increasing the value of μ, we can reduce the number of variables to represent F. Consider the following operations:

$$\begin{aligned} x_1 &\Leftarrow x_1 \oplus x_3 \ (\mu = 46). \\ x_1 &\Leftarrow x_1 \oplus x_5 \ (\mu = 48). \\ x_1 &\Leftarrow x_1 \oplus x_7 \ (\mu = 48). \\ x_2 &\Leftarrow x_2 \oplus x_3 \ (\mu = 52). \\ x_2 &\Leftarrow x_2 \oplus x_6 \ (\mu = 54). \\ x_2 &\Leftarrow x_2 \oplus x_7 \ (\mu = 54). \\ x_4 &\Leftarrow x_4 \oplus x_5 \ (\mu = 58). \\ x_4 &\Leftarrow x_4 \oplus x_6 \ (\mu = 60). \\ x_4 &\Leftarrow x_4 \oplus x_7 \ (\mu = 60). \end{aligned}$$

Note that the values of μ after the operations are non-decreasing. These operations are equivalent to the transformation used in Example 5.2. Thus, after these transformations, F can be represented with only three variables. ■

VIII. Experimental Results

We developed a program to perform the algorithm described in the previous section. To improve the performance, we also incorporated the rule $x_i \Leftarrow x_i \oplus x_i \oplus x_k$. As for the benchmark function, we used the *1-out-of-n* code to index converter [18]. Table 2.1 shows the example of $n = 7$. It is an index generation functions with weight n. The i-th variable has 1 and other variables have 0 in the input if and only if the value of the function is i. The minimum number of variables to represent this function is $\lceil \log_2 n \rceil$. Table 8.1 shows the results. The first column shows n, the number of inputs; the second column (*MIN*) shows the minimum solution; the third column (*AUTO*) shows the number of variables obtained by the presented method; the fourth column (*Orig*) shows the number of 1's in the original difference matrix; the fifth column (*Aft*) shows the number of 1's after the transformation; and the last column shows the CPU time.

Up to $n = 14$, the program obtained minimum solutions. However, for $n = 15$, it obtained a solution that required two more variables than the minimum. Also, for $n \geq 22$, the algorithm failed to obtain the minimum. This implies that for these function, we need rules that incorporate more than three variables. However, the presented program is fast enough and requires much less memory than one in [17] for this class of functions. In the experiment, we used INTEL Core i5-3320M CPU @2.6 GHz, and Windows 7, 64-bit operating system. Currently, we are improving the algorithm. Table 8.2 compares the property of the present algorithm with existing method.

Table 8.1
Reduction of variables for 1-out-of-n code to index converters

	# of variables		μ		
n	MIN	$AUTO$	$Orig$	Aft	$CPU(ms)$
6	3	3	30	54	10.9
7	3	3	42	84	9.7
8	3	3	56	127	12.5
9	4	4	72	180	11.3
10	4	4	90	249	16.4
11	4	4	110	330	17.5
12	4	4	132	431	32.0
13	4	4	156	546	27.3
14	4	4	182	685	22.6
15	4	6	210	840	11.3
16	4	6	240	1023	18.3
17	4	5	272	1224	35.5
18	5	6	306	1457	16.7
19	5	6	306	1710	123.8
20	5	5	380	1999	230.4
21	5	5	420	2310	783.9
22	5	7	420	2661	249.2
23	5	7	420	3036	386.0
24	5	6	552	3455	3347.2

Table 8.2
Comparison with Existing Method

	Exhaustive Method ISMVL2011	This Method ISMVL2013
Required Memoroy	Large $O(k2^n)$	Small $O(nk^2)$
CPU time	Large	Small
Quality of Solutions	Exact Minimum	Near Minimum

IX. Summary

Major contributions of this paper are:

- Defined the characteristic logic function f of an incompletely specified index generation function F.
- Showed that the autocorrelation function of f are useful to reduce the number of variables to represent F.
- Presented an algorithm to derive minimal sets of variables to represent F.
- Presented an algorithm to derive the set of vectors that produce non-zero values in the autocorrelation function of f. The number of non-zero coefficients is at most $\frac{k(k-1)}{2}+1$, where k is the number of specified elements in the index generation function F.
- Showed a heuristic algorithm to find a good linear transformation in the autocorrelation domain. Also developed a program and presented some experimental results.
- Showed that the set of vectors that produce non-zero values in the autocorrelation function of f, is equivalent to the **covering function** defined in [16], and also to the *difference matrix* D_f used in [19].

Acknowledgments

This work is partially supported by the Japan Society for the Promotion of Science (JSPS), Grant in Aid for Scientific Research. Prof. Jon T. Butler's comments were useful to improve the presentation.

References

[1] C. Halatsis and N. Gaitanis, "Irredundant normal forms and minimal dependence sets of a Boolean functions," *IEEE Trans. on Computers*, vol. C-27, no. 11, Nov. 1978, pp. 1064-1068.

[2] Y. Kambayashi, "Logic design of programmable logic arrays," *IEEE Trans. on Computers*, vol. C-28, no. 9, Sept. l979, pp. 609-617.

[3] M. G. Karpovsky, *Finite Orthogonal Series in the Design of Digital Devices*, Wiley, 1976.

[4] M. G. Karpovsky, R. S. Stankovic, and J. T. Astola, "Reduction of sizes of decision diagrams by autocorrelation functions,"*IEEE Transactions on Computers*, vol. 52, no. 5, pp. 592-606, May, 2003.

[5] O. Keren and I. Levin, "Linearization of multi-output logic functions by ordering of the autocorrelation values,"*FACTA UNIVERSITATIS (NIS)*, vol. 20, no. 3, December 2007, pp. 479-498.

[6] O. Keren, I. Levin and R. S. Stankovic, "Determining the number of paths in decision diagrams by using autocorrelation coefficients," *IEEE Transactions on Computer-Aid Design of Integrated Circuits and Systems*, vol. 30, no. 1, Jan. 2011, pp. 31–44.

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

[8] C. Meinel, F. Somenzi, and T. Theobald, "Linear sifting of decision diagrams and its application in synthesis," *IEEE Trans. CAD*, vol. 19, no. 5, pp. 521-533, 2000.

[9] 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.

[10] J. Rice, "The autocorrelation transform and its application to the classification of Boolean functions," *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, (PacRim 2009), Aug. 23-26, 2009, pp. 94-99.

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

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

[13] T. Sasao, "Design methods for multiple-valued input address generators,"(invited paper) *International Symposium on Multiple-Valued Logic* (ISMVL-2006), Singapore, May 2006.

[14] 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.

[15] T. Sasao, T. Nakamura, and M. Matsuura, "Representation of incompletely specified index generation functions using minimal number of compound variables," *12th EUROMICRO Conference on Digital System Design, Architectures, Methods and Tools* (DSD 2009), Patras, Greece, Aug. 27-29, 2009, pp. 765-772.

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

[17] T. Sasao, "Index generation functions: Recent developments,"(invited paper) *International Symposium on Multiple-Valued Logic* (ISMVL-2011), Tuusula, Finland, May 23-25, 2011.

[18] T. Sasao, "Linear decomposition of index generation functions," *17th Asia and South Pacific Design Automation Conference* (ASPDAC-2012), Jan. 30- Feb. 2, 2012, Sydney, Australia, pp. 781-788.

[19] D. A. Simovici, M. Zimand, and D. Pletea, "Several remarks on index generation functions,"*International Symposium on Multiple-Valued Logic* (ISMVL-2012), May 2010, pp. 179-184.

[20] D. Varma and E. Trachtenberg, "Design automation tools for efficient implementation of logic functions by decomposition,"*IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 8, no.8, pp. 901-916, Aug. 1989.