2012 IEEE 42nd International Symposium on Multiple-Valued Logic

Multiple-Valued Input Index Generation Functions: Optimization by Linear Transformation

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

***Abstract*—In an incompletely specified function f, don't care values can be chosen to minimize the number of variables to represent f. We consider incompletely specified multiple-valued input index generation functions $f : D \rightarrow \{1, 2, \ldots, k\}$, where $D \subseteq P^n$ and $P = \{0, 1, 2, \ldots, p-1\}$. We show that most functions can be represented with $2\lceil \log_p(k+1) \rceil$ or fewer variables, where k denotes the number of elements in D. Also, we introduce linear transformations to further reduce the number of variables. Experimental results support these observations.**

I. Introduction

In an incompletely specified function f, don't care values can be chosen to minimize the number of variables to represent f. This property is useful to represent the function compactly. In this paper, we consider the minimization of number of variables for incompletely specified index generation functions. We show that most p-valued input index generation functions of n variables with weight k can be represented by $2\lceil \log_p(k+1) \rceil$ or fewer variables, when k is sufficiently smaller than p^n. *i.e.*, the functions are highly unspecified.

Index generation functions have applications in pattern matching in internet [10]. The problem is also related to data mining and perfect hashing. The rest of the paper is organized as follows: Section 2 defines words; Section 3 derives the number of variables to represent an incompletely specified index generation functions with k registered vectors; Section 4 shows statistical results for uniformly distributed functions; Section 5 shows reduction of the number of variables by linear transformations; Section 6 shows experimental results using list of English words; and Section 7 concludes the paper.

II. Definitions and Basic Properties

Definition 2.1: Consider a set of k different vectors with n components. These vectors are **registered vectors**. For each registered vector, assign a unique integer from 1 to k. A **registered vector table** shows the **index** of each registered vector.

Definition 2.2: A **incompletely specified index generation function** f is a mapping $D \rightarrow \{1, 2, \ldots, k\}$, where D denotes the set of registered vectors, $D \subseteq P^n$, $P = \{0, 1, \ldots, p-1\}$, $|D| = k$, and $|D|$ denotes the number of elements in D. A **completely specified index generation function** produces the corresponding index if the input matches a registered vector, and produces 0 otherwise. k is the **weight** of the index generation function.

TABLE 2.1
Registered vector table.

x_1	x_2	x_3	x_4	x_5	f
0	0	1	0	0	1
0	1	0	0	1	2
0	1	1	1	0	3
1	0	0	1	1	4
1	0	1	0	1	5
1	1	0	1	0	6

Example 2.1: Table 2.1 shows a registered vector table consisting of 6 vectors. It shows an incompletely specified index generation function with weight 6.

Definition 2.3: f **depends on** x_i if there exists a pair of vectors

$$\begin{aligned} \vec{a} &= (a_1, a_2, \ldots, a_i, \ldots, a_n) \text{ and} \\ \vec{b} &= (a_1, a_2, \ldots, b_i, \ldots, a_n), \end{aligned}$$

such that both $f(\vec{a})$ and $f(\vec{b})$ are specified, $a_i \neq b_i$, and $f(\vec{a}) \neq f(\vec{b})$.
If f depends on x_i, then x_i is **essential** in f, and x_i must appear in every expression for f.

Definition 2.4: Two functions f and g are **compatible** when the following condition holds for any $\vec{a} \in P^n$: If both $f(\vec{a})$ and $g(\vec{a})$ are specified, then $f(\vec{a}) = g(\vec{a})$.

Lemma 2.1: Let $f_i = f(|x = i)$ for $i = 0, 1, \ldots, p-1$. Then, x is **non-essential** in f iff f_i and f_j are compatible for all the pair (i, j).

If x is non-essential in f, then f can be represented by an expression without x. Essential variables must appear in every expression for f, while non-essential variables may appear in some expressions and not in others. Algorithms to represent a given function by using the minimum number of variables have been considered [1], [3], [4].

III. Number of Variables to Represent Index Generation Functions

In this part, we derive the number of variables to represent an incompletely specified index generation function with k registered vectors. We assume that k is much smaller than p^n, the total number of input combinations. The basic idea is given by

Lemma 3.1: Suppose that an incompletely specified function $f(X_1, X_2)$ is represented by a decomposition chart, where

0195-623X/12 $26.00 © 2012 IEEE
DOI 10.1109/ISMVL.2012.21

IEEE computer society

TABLE 3.1
DECOMPOSITION CHART FOR $f(X_1, X_2)$.

		0	0	0	0	1	1	1	1	x_1
		0	0	1	1	0	0	1	1	x_2
		0	1	0	1	0	1	0	1	x_3
0	0		1							
0	1			2			5			
1	0				3			6		
1	1					4				
x_4	x_5									

X_1 labels the columns and X_2 labels the rows. If each column has at most one *care* (non-zero) element, then the function can be represented by using only variables in X_1.
(Proof) In each column, let the values of *don't cares* elements be set to the value of the non-zero element in the column, then the function depends only on the column variables. □

Example 3.1: Consider the decomposition chart shown in Table 3.1. In Table 3.1, x_1, x_2, and x_3 specify the columns, while x_4 and x_5 specify the rows. Blank elements denote *don't cares*. Note that in Table 3.1, each column has at most one *care* element. Thus, the function can be represented by only the column variables: x_1, x_2, and x_3. $f = 1 \cdot \bar{x}_1\bar{x}_2x_3 \vee 2 \cdot \bar{x}_1x_2\bar{x}_3 \vee 3 \cdot \bar{x}_1x_2x_3 \vee 4 \cdot x_1\bar{x}_2\bar{x}_3 \vee 5 \cdot x_1\bar{x}_2x_3 \vee 6 \cdot x_1x_2\bar{x}_3$. ■

Theorem 3.1: To represent any incompletely specified p-valued input index generation function with weight k, at least $\lceil \log_p k \rceil$ variables are necessary.
(Proof) Let $q = \lceil \log_p k \rceil$. The number of different vectors specified with $q-1$ variables is at most $p^{q-1} < k$. Thus, to distinguish k outputs, at least q variables are necessary. □

From here, we derive numbers of variables to represent functions.

Theorem 3.2: Consider a set of uniformly distributed p-valued input n-variable incompletely specified index generation functions $f(x_1, x_2, \ldots, x_n)$ with weight k, where $p \leq k < p^{n-2}$. Let $\eta(p, n, t, k)$ be the probability that f can be represented with $x_1, x_2, \ldots, x_{t-1}$ and x_t, where $t < n$. Then,

$$\eta(p, n, t, k) = \frac{{}_{p^t}P_k \cdot p^{(n-t)k}}{{}_{p^n}P_k}. \tag{3.1}$$

(Proof) From Theorem 3.1, we have $k \leq p^t$. The probability is given as $\eta(p, n, t, k) = \frac{A}{B}$, where A denotes the number of incompletely specified index generation functions with weight k that can be represented with $x_1, x_2, \ldots, x_{t-1}$ and x_t, and B denotes the total number of incompletely specified index generation functions with weight k.

1) Derive A, the number of incompletely specified index generation functions with weight k such that each column has at most one care element. First, enumerate the numbers of ways to specify the non-zero columns. It is equal to the number of ways to distribute k distinct elements into p^t distinct bins: ${}_{p^t}P_k$. Second, enumerate the number of ways to specify the rows for all these elements. The number of ways to select a row is p^{n-t} for each element. Since there are k elements, the total number of ways to select the rows is $(p^{n-t})^k = p^{(n-t)k}$. Thus, we have $A = {}_{p^t}P_k \cdot p^{(n-t)k}$.
2) Derive B, the total number of n-variable incompletely specified index generation functions with weight k. This is equal to the number of ways to distribute k distinct elements into p^n distinct bins. It is

$${}_{p^n}P_k = p^n \cdot (p^n - 1) \cdot (p^n - 2) \cdots (p^n - (k-1)).$$

Hence, we have the theorem. □

The above theorem shows the case when the column variables are $(x_1, x_2, \ldots, x_t)$. In practice, we can select the set of column variables so that the number of variables is minimized.

Theorem 3.3: Consider a set of uniformly distributed incompletely specified index generation functions $f(x_1, x_2, \ldots, x_n)$ with weight k, where $p \leq k < p^{n-2}$. Let PR be the probability that f can be represented with t variables, then

$$PR = 1 - (1 - \eta(p, n, t, k))^{\binom{n}{t}}, \tag{3.2}$$

where $\eta(p, n, t, k)$ is the probability that f can be represented with $x_1, x_2, \ldots,$ and x_t.
(Proof) The probability that a function cannot be represented by using $x_1, x_2, \ldots, x_{t-1}$ and x_t is $\sigma = 1 - \eta(p, n, t, k)$. Since there are $\binom{n}{t}$ ways to choose t variables out of n variables, the probability that a function cannot be represented by using any combinations of t variables is $\sigma^{\binom{n}{t}}$. The probability that a function can be presented by using at least one combination of t variables is $1 - \sigma^{\binom{n}{t}}$. □

Since $\eta(p, n, t, k)$ is not easy to treat, we use the following approximation to simplify it.

Lemma 3.2: If $0 < \alpha << 1$, then $1 - \alpha$ can be approximated by $e^{-\alpha}$, where e denotes the base of natural logarithm.

Lemma 3.3: When $\frac{k}{p^t}$ is small enough, $\eta(p, n, t, k)$ in Theorem 3.1 can be approximated by $\tilde{\eta}(p, t, k) = \exp(-\frac{k^2}{2p^t})$.
(Proof) $\eta(p, n, t, k) = \frac{{}_{p^t}P_k \cdot p^{(n-t)k}}{{}_{p^n}P_k}$
$= \frac{p^t(p^t-1)(p^t-2)\cdots(p^t-(k-1))}{p^n(p^n-1)(p^n-2)\cdots(p^n-(k-1))} p^{k(n-t)}$
$= \frac{p^n}{p^n} \cdot \frac{p^n-1 \cdot p^{n-t}}{p^n-1} \cdot \frac{p^n-2 \cdot p^{n-t}}{p^n-2} \cdot \frac{p^n-3 \cdot p^{n-t}}{p^n-3} \cdots \frac{p^n-(k-1) \cdot p^{n-t}}{p^n-(k-1)}$
Assume that k is sufficiently smaller than p^n, and assume that $p^n - i$ is approximated by p^n. We have
$\tilde{\eta}(p, t, k)$
$= \frac{p^n}{p^n} \cdot \frac{p^n-1 \cdot p^{n-t}}{p^n} \cdot \frac{p^n-2 \cdot p^{n-t}}{p^n} \cdot \frac{p^n-3 \cdot p^{n-t}}{p^n} \cdots \frac{p^n-(k-1) \cdot p^{n-t}}{p^n}$
$= (1 - 1\alpha) \cdot (1 - 2\alpha) \cdot (1 - 3\alpha) \cdots (1 - (k-1)\alpha)$,
where $\alpha = p^{-t}$.

When $i\alpha$ is small, by Lemma 3.2, $1 - i\alpha$ is approximated by $\exp(-i\alpha)$. Thus, $\tilde{\eta}(p, t, k)$ is approximated by

$$\begin{aligned} \tilde{\eta}(p, t, k) &\simeq \prod_{i=1}^{k-1} \exp(-i\alpha) = \exp(-\sum_{i=1}^{k-1} i\alpha) \\ &\simeq \exp(-\frac{k(k-1)\alpha}{2}) \simeq \exp(-\frac{k^2\alpha}{2}) \end{aligned}$$

□

From this, we have the following:

Conjecture 3.1: Consider a set of uniformly distributed incompletely specified p-valued input n-variable index generation functions with weight k, where $p^3 \leq k \leq p^{n-2}$ and $n \geq 10$. If $t \leq n-3$ satisfies the following conditions, then more than 95% of the functions can be represented with t variables.

$$t \geq \lceil 2\log_p k - \log_p 5.485 \rceil$$

(Explanation supporting the Conjecture) $1-\sigma^{\binom{n}{t}}$ approaches 1.0, as n increases, since $\sigma = 1-\eta(p,n,t,k) < 1.0$. When $t \leq n-3$, $\binom{n}{t} \geq n(n-1)(n-2)/6$. Assume that $n \geq 10$. In this case, we have $\binom{n}{t} \geq 120$. The condition that $\sigma^{\binom{n}{t}} \leq 0.05$ derives $\sigma < 0.9753$. Thus, if $\eta(p,n,t,k) \geq 0.02465$, then at least 95% of the functions can be represented with t variables. Thus, we have $exp(-\frac{k^2}{2p^t}) \geq 0.02465$. When $t \geq \lceil 2\log_p k - \log_p 5.485 \rceil$, we have $\eta > 0.02465$. (End of explanation)

Note that there exist functions that require all the variables as shown below. However, we conjecture that the fraction of such functions approaches to zero as n increase.

Example 3.2: Consider the n-variable incompletely specified index generation function f with weight $k = n+1$ and $p = 2$:

$$\begin{array}{rcl}
f(1,0,0,\ldots,0,0) & = & 1 \\
f(0,1,0,\ldots,0,0) & = & 2 \\
f(0,0,1,\ldots,0,0) & = & 3 \\
\vdots & & \vdots \\
f(0,0,0,\ldots,1,0) & = & n-1 \\
f(0,0,0,\ldots,0,1) & = & n \\
f(0,0,0,\ldots,0,0) & = & n+1 \\
f(a_1,a_2,a_3,\ldots,a_{n-1},a_n) & = & d \quad \text{(for other combinations).}
\end{array}$$

In this function, all the variables are essential, and no variable can be removed. ■

IV. Statistical Results for Uniformly Distributed Functions

We generated uniformly distributed index generation functions, and obtained statistical data. Table 4.1 shows the average numbers of variables to represent p-valued input n-variables index generation functions with weight k. The columns headed with *Exp* show that the average numbers of variables to represent the functions. For each parameter, we generated 100 functions. The columns headed with *Conj* show the number of variables to represent incompletely specified index generation functions with weight k given by Conjecture 3.1. For example, when $k = 1023$ and $p = 2$, to represent a uniformly distributed function, experimental results show that, on the average, 16.32 variables are necessary to represent the functions. On the other hand, Conjecture 3.1 shows that 18 variables are sufficient. Experimental results show that only 13 functions out of 5400 functions exceeded the bound given by Conjecture 3.1.

Table 4.2 shows the probability that index generation functions with weight k can be represented with t variables. These values are derived from Theorems 3.2 and 3.3, and Lemma 3.3. For example, when $p = 2$, $n = 20$, and $k = 1023$, the probability that the function can be represented with $t = 15$ variables is 0.00185. However, when $t = 16$ the probability is 0.79731, and when $t = 17$ the probability is 1.0000. This is consistent with the experimental results: Out of 100 functions, 1 function required 15 variables; 66 functions required 16 variables; and 33 functions required 17 variables. We performed additional experiments and confirmed Conjecture 3.1

TABLE 5.1
Original List of English Words.

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	f
a	c	c	o	m	m	o	d	a	t	i	o	n	s	1
a	d	m	i	n	i	s	t	r	a	t	i	o	n	2
c	h	a	r	a	c	t	e	r	i	s	t	i	c	3
c	o	n	g	r	a	t	u	l	a	t	i	o	n	4
c	o	n	s	t	i	t	u	t	i	o	n	a	l	5
d	i	s	a	p	p	o	i	n	t	m	e	n	t	6
d	i	s	c	r	i	m	i	n	a	t	i	o	n	7
g	e	n	e	r	a	l	i	z	a	t	i	o	n	8
i	d	e	n	t	i	f	i	c	a	t	i	o	n	9
i	n	t	e	r	p	r	e	t	a	t	i	o	n	10
r	e	c	o	m	m	e	n	d	a	t	i	o	n	11
r	e	p	r	e	s	e	n	t	a	t	i	o	n	12
r	e	p	r	e	s	e	n	t	a	t	i	v	e	13
r	e	s	p	o	n	s	i	b	i	l	i	t	y	14
t	r	a	n	s	p	o	r	t	a	t	i	o	n	15
39	41	33	27	33	39	35	45	39	113	105	125	89	87	ω

V. Reduction of the Number of Variables By Linear Transformations

This section shows a method to reduce the number of variables to represent a given incompletely specified index generation function f by using linear transformations.

Definition 5.1: A **compound variable** has a form $y = c_1x_1 \oplus c_2x_2 \oplus \cdots \oplus c_nx_n$ where $c_i \in \{0,1\}$ and $\oplus$ denotes the mod p sum operation. The **compound degree** of y is $\sum_{i=1}^{n} c_i$, where c_i is viewed as an integer and $\sum$ denotes an ordinary integer addition. A **primitive variable** is one with compound degree one.

It is also possible to consider the case where $c_i \in P$. However, in this case, we need multipliers in addition to adder. So, in this paper, we consider only the case of $c_i \in \{0,1\}$.

Definition 5.2: Given an incompletely specified index generation function, a linear transformation that minimizes the number of variables is **optimum**.

By Theorem 3.1, if the linear transformation reduce the number of variables to $q = \lceil \log_p k \rceil$ variables, then it is an optimum. A brute force way to find an optimum transformation is first to construct the compound variables whose degrees are t or less than t. The number of such variables is $\sum_{i=1}^{t} \binom{n}{i}$. Then, apply the method shown in [10]. However, such method takes too much computation time, and is impractical.

Example 5.1: Table 5.1 shows a list of 15 English words consisting of 14 characters. Each variable can take one of 27 values i.e., 26 alphabets and a - (hyphen). To distinguish

TABLE 4.1
AVERAGE NUMBER OF VARIABLES TO REPRESENT INCOMPLETELY SPECIFIED INDEX GENERATION FUNCTION.

p: Number of values. n: Number of original variables. k: Weight of the function.

	$p=2$, $n=20$		$p=3$, $n=13$		$p=4$, $n=10$		$p=5$, $n=10$		$p=10$, $n=10$		$p=27$, $n=10$	
k	Exp	Conj	Exp	Conj	Exp	Conj	Exp	Conj	Exp	Conj	Exp	Conj
15	4.93	6	3.24	4	3.00	3	2.84	3	2.00	2	1.96	2
31	6.07	8	4.52	5	3.98	4	3.13	4	2.81	3	2.00	2
63	8.00	10	5.85	7	4.97	5	4.00	5	3.00	3	2.04	2
127	10.00	12	6.99	8	6.01	6	5.00	5	3.98	4	3.00	3
255	11.98	14	8.00	9	6.93	7	6.00	6	4.00	5	3.00	3
511	14.02	16	9.59	10	7.95	8	6.97	7	5.00	5	3.89	4
1023	16.32	18	10.97	12	9.02	9	7.85	8	5.27	6	4.00	4
2047	18.66	20	12.43	13	9.95	10	8.78	9	6.00	6	4.02	5
4095	19.97	22	13.00	14	10.00	11	9.69	10	6.94	7	5.00	5

TABLE 4.2
PROBABILITY THAT INDEX GENERATION FUNCTIONS WITH WEIGHT $\boldsymbol{k}$ CAN BE REPRESENTED WITH $\boldsymbol{t}$ VARIABLES.

p: Number of values. n: Number of original variables. k: Weight of the function.

$p=2$, $n=20$, $k=1023$		$p=3$, $n=13$, $k=511$		$p=4$, $n=10$, $k=511$		$p=5$, $n=10$, $k=511$		$p=10$, $n=10$, $k=255$		$p=27$, $n=10$, $k=255$	
t	PR	t	PR	t	PR	t	PR	t	PR	t	PR
15	0.00185	8	0.00000	6	0.00000	5	0.00000	3	0.00000	2	0.00000
16	0.79731	9	0.59347	7	0.03805	6	0.04471	4	0.99972	3	1.00000
17	1.00000	10	1.00000	8	0.99863	7	1.00000	5	1.00000		

TABLE 5.2
REDUCED LIST OF ENGLISH WORDS.

x_3	x_{13}	x_6	x_8	y_1	y_2	z_1	f
c	n	m	d	p	p	r	1
m	o	i	t	!	a	a	2
a	i	c	e	i	g	s	3
n	o	a	u	a	u	g	4
n	a	i	u	n	b	b	5
s	n	p	i	e	x	x	6
s	o	i	i	f	q	h	7
n	o	a	i	a	i	k	8
e	o	i	i	s	q	o	9
t	o	p	e	g	t	m	10
c	o	m	n	q	z	q	11
p	o	s	n	c	e	i	12
p	v	s	n	j	e	p	13
s	t	n	i	k	v	e	14
a	o	p	r	o	f	y	15
33	89	39	45	17	19	15	ω

these 15 words, three variables (characters) are necessary and sufficient. For example, it can be represented by (x_3, x_6, x_{13}). However, by using the linear transformation:

$$\begin{aligned} y_1 &= x_3 \oplus x_{13}, \\ y_2 &= x_6 \oplus x_8 \end{aligned}$$

we have the registered vectors shown in Table 5.2. In this case, two variables (y_1, y_2) distinguish 15 vectors. Note that $a, b, c, \ldots, y$, and z have values $0, 1, 2, \ldots, 24$, and 25, respectively. Also, the character - has the value 26. ■

Example 5.2: In Table 5.1, consider the linear transformation:

$$z_1 = x_1 \oplus x_5 \oplus x_{10} \oplus x_{13}$$

As shown in Table 5.2, only one variable z_1 can distinguish 15 vectors. ■

As shown in this example, by the linear transformation, we can often reduce the number of variables to represent the function. Since the linear transformation makes a more balanced decision tree, it reduces the number of variables. To obtain the linear transformation that produces a more balanced decision tree, we define a measure showing the distribution of values in the registered vector table.

Definition 5.3: In the the registered vector table, let $\nu(x_i, j)$ be the number of vectors with $x_i = j$, where $j \in P$. The **imbalance measure** of x_i is defined as

$$\omega(x_i) = \sum_{j=0}^{p-1} \nu(x_i, j)^2.$$

In the registered vector table, when the numbers of occurrences of j's in the column x_i are the same, $\omega(x_i)$ takes its minimum. The larger the difference of the frequency of values, the larger the imbalance measure. Let k be the number of registered vectors. Then, $\sum_{j=0}^{p-1} \nu(x_i, j) = k$.

Example 5.3: In Table 5.1, consider the variable x_1. Note that a, d and i appear twice, c appear three times, r appears four times, g and t appear only once. Thus,

$$\begin{aligned} \omega(x_1) &= \sum_{j=0}^{26} \nu(x_1, j) \\ &= 3 \times 2^2 + 1 \times 3^2 + 1 \times 4^2 + 2 \times 1^2 = 39. \end{aligned}$$

The last row of Table 5.1 show the imbalance measure for the variables x_i.

In Table 5.2, consider the variable y_1. Note that only a appears twice, but other 13 characters appear only once. Thus,

TABLE 5.3
INDEX GENERATION FUNCTION.

x_1	x_2	x_3	x_4	x_5	f
0	0	0	0	0	1
0	1	0	1	0	2
0	1	1	1	0	3
1	1	1	0	0	4
1	0	0	1	1	5
1	0	1	1	1	6
1	1	1	0	1	7

we have

$$\omega(y_1) = 1 \times 2^2 + 13 \times 1^2 = 17.$$

Also, consider the variables y_2. Note that only e and q appear twice, but other 11 characters appear only once. Thus, we have

$$\omega(y_2) = 2 \times 2^2 + 11 \times 1^2 = 19.$$

The last row of Table 5.2 show the imbalance measure for the variables y_i.

In other words, the linear transformation in Example 5.1 reduces the imbalance measure, and improves the balance of the decision tree. ■

When the imbalance measure is large, the reduction of variables tends to be difficult. However, if a linear transformation reduces the imbalance measure, then we may reduce more variables.

Definition 5.4: [12] Let $f(x_1, x_2, \ldots, x_n)$ be an incompletely specified index generation function with weight $|f|$. Let $\vec{x} = (x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(t)})$ be a vector consisting of a subset of the variables $\{x_1, x_2, \ldots, x_n\}$, where π denotes a permutation of $\{1, 2, \ldots, n\}$. Let $N(f, \vec{x}, \vec{a})$ be the number of registered vectors of f that takes non-zero values, when the values of $\vec{x}$ are set to $\vec{a} = (a_1, a_2, \ldots, a_t)$, $a_i \in P$. The **ambiguity** of f with respect to $\vec{x}$ is defined as

$$AMB(\vec{x}) = -|f| + \sum_{\vec{a} \in P^t} N(f, \vec{x}, \vec{a})^2.$$

Example 5.4: Consider the index generation function shown in Table 5.3. Assume that the values of (x_1, x_2, x_3) are changed as $(0,0,0)$, $(0,0,1)$, $(0,1,0)$, $(0,1,1)$, $(1,0,0)$, $(1,0,1)$, $(1,1,0)$, $(1,1,1)$, in this order. Then, the values of f change as follows:

$$[1], [d], [2], [3], [5], [6], [d], [4, 7],$$

where [d] denotes undefined or *don't care*. In this case, the ambiguity with respect to (x_1, x_2, x_3) is

$$\begin{aligned} & AMB(x_1, x_2, x_3) \\ = \quad & -7 + (1^2 + 0^2 + 1^2 + 1^2 + 1^2 + 1^2 + 0^2 + 2^2) = 2 \end{aligned}$$

When $(x_1, x_2, x_3) = (0, 0, 1)$, the value of f is **undefined**, while when $(x_1, x_2, x_3) = (1, 1, 1)$, the value of f is **ambiguous**, since f can be either 4 or 7.

Next, let the variable set be (x_1, x_3, x_5). Similarly, the values of f change as follows:

$$[1, 2], [d], [3], [d], [d], [5], [4], [6, 7].$$

In this case, the ambiguity with respect to (x_1, x_3, x_5) is

$$\begin{aligned} & AMB(x_1, x_3, x_5) \\ = \quad & -7 + (2^2 + 0^2 + 1^2 + 0^2 + 0^2 + 1^2 + 1^2 + 2^2) = 4 \end{aligned}$$

When $(x_1, x_3, x_5) = (0, 0, 0)$ and $(1, 1, 1)$, the values of f are ambiguous.

Finally, let the variable set be (x_3, x_4, x_5). Similarly, the values of f change as follows:

$$[1], [d], [2], [5], [4], [7], [3], [6].$$

In this case, the ambiguity with respect to (x_3, x_4, x_5) is

$$\begin{aligned} & AMB(x_3, x_4, x_5) \\ = \quad & -7 + (1^2 + 0^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2) = 0. \end{aligned}$$

Note that f can be represented with only (x_3, x_4, x_5). ■

Theorem 5.1: $AMB(\vec{x}) = 0$ iff $\vec{x}$ can represent f.

(Proof) Let D be the set of registered vectors for f and let $\tilde{D}$ be the set of vectors consisting of variables for $\vec{x}$.

($\Rightarrow$) We prove this by contradiction. Assume that $\vec{x}$ cannot represent f. Two cases are possible.

1) f is undefined for some $\vec{a} \in \tilde{D}$. In this case, $N(f, \vec{x}, \vec{a}) = 0$. Since $\vec{a}$ is a registered vector where some variables are omitted, this cannot happen.
2) f is ambiguous for some $\vec{a} \in \tilde{D}$. In this case, $N(f, \vec{x}, \vec{a}) \geq 2$. Since $\sum N(f, \vec{x}, \vec{a})^2 > |f|$, we have $AMB(\vec{a}) > 0$.

From these, for each $\vec{a} \in \tilde{D}$, the value of f is uniquely defined. Thus, f can be represented with $\vec{x}$.

($\Leftarrow$) Assume that f is represented with $\vec{x}$. In this case, the value of f is uniquely defined or undefined for all possible cases. This implies that $N(f, \vec{x}, \vec{a}) = 1$ for all $\vec{a} \in \tilde{D}$. From this, we have $AMB(\vec{x}) = -|f| + \sum_{\vec{a} \in \tilde{D}} 1^2 = 0$, since, $|f| = |\tilde{D}|$. □

By using these two measures, we have a heuristic algorithm to reduce the number of variables. In this algorithm, the imbalance measure is used to guide the linear transformation. The compound variable is chosen to minimize the imbalance measure in a greedy manner. Then the ambiguity measure (AMB) is tested. If the $AMB > 0$, more compound variables are required to distinguish the registered vectors. This process stops when $AMB = 0$.

Algorithm 5.1: (Heuristic Method to Find a Linear Transformation that Reduces the Number of Variables)

1) Let the input variables be $x_1, x_2, \ldots, x_n$. Let $t \geq 2$ be the maximal compound degree.
2) Generate the compound variables y_i whose compound degrees are t or less than t. The number of such compound variables is $\sum_{i=1}^{t} \binom{n}{i}$. Let T be the set of compound variables.
3) Let y_1 be the variable with the smallest imbalance measure. Let $\vec{Y} \leftarrow (y_1)$, $T \leftarrow T - y_1$.
4) While $AMB(\vec{Y}) > 0$, find the variable y_j in T that minimizes the value of $AMB(\vec{Y}, y_j)$. Let $\vec{Y} \leftarrow (\vec{Y}, y_j)$, $T \leftarrow T - y_j$.

TABLE 6.1
LIST OF ENGLISH WORDS ($p = 27$).

		Compound Degree: t					
n	k	1	2	3	4	5	6
8	548	8	5	4	4	3	3
9	380	6	4	4	3	3	3
10	272	6	4	3	3	3	3
11	143	5	3	3	3	3	3
12	75	4	**2**	**2**	**2**	**2**	**2**
13	38	3	**2**	**2**	**2**	**2**	**2**
14	15	3	2	2	**1**	**1**	**1**

5) Stop.

Experimental results show that this algorithm obtains a fairly good solutions in a short time.

VI. EXPERIMENTAL RESULTS

From a list of 5000 frequently used English words, we made seven sub-lists of words, each consisting of 8, 9, 10, 11, 12, 13 and 14 characters. For each list, we minimized the number of variables (i.e., the characters) using Algorithm 5.1. Table 6.1 shows the numbers of variables to represent the sub-lists. In the table, n denotes the number of characters; k denotes the number of words in the sub-list; and t denotes the compound degree. These sub-lists correspond to 27-valued input index generation functions with weight k.

The sub-list of English words for $n = 14$ is shown in Table 5.1. In Table 6.1, the bold letters show exact minimum. The experimental results in Table 6.1 are consistent with Tables 4.1 and 4.2.

For example, in the case of $n = 14$. To distinguish 15 words,

1) When $t = 1$, three variables $\{x_3, x_8, x_{13}\}$ are sufficient.
2) When $t = 2$, two variables $\{y_1 = x_3 \oplus x_{13}, y_2 = x_6 \oplus x_8\}$ are sufficient.
3) When $t = 4$, one variable $\{z_1 = x_1 \oplus x_5 \oplus x_{10} \oplus x_{13}\}$ is sufficient.

Up to $t = 5$, the number of variables are reduced by increasing the value of t. However, for $t = 6$ the number of variables could not be reduced any more.

It is known that the numbers of characters appearing English words are not uniform: e appears the most frequently, while z appears the least frequently. This means that the decision tree according to the original alphabets is not balanced. By using compound variables, the decision tree can be made more balanced.

It is also possible to represent a characters with five two-valued variables [12]. In this case, the total number of variables would be five times, and the computation time would be very large, although more variables can be reduced.

VII. CONCLUDING REMARKS

In this paper, we derived the number of variables to represent incompletely specified p-valued input index generation functions with weight k. Most functions can be represented by $2\lceil \log_p(k+1)\rceil$ or fewer variables, when k is sufficiently smaller than p^n.

Also, in this paper, we considered linear transformations of index generation functions. To find good linear transformations, we introduced two measures: the imbalance measure and the ambiguity measure. We showed a heuristic method to find linear transformation that reduces the number of variables to represent the functions. When the imbalance measures are large, the reduction of primitive variables is difficult. However, with a linear transformation that reduces imbalance measures, we can reduce more variables.

ACKNOWLEDGMENTS

This work was supported in part by a Grant in Aid for Scientific Research of the JSPS, and Knowledge Cluster Project of MEXT. The author thanks Prof. Jon T. Butler for discussion and Mr. M. Matsuura for experiment.

REFERENCES

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

[2] R. K. Brayton, G. D. Hachtel, C. T. McMullen, and A. L. Sangiovanni-Vincentelli, *Logic Minimization Algorithms for VLSI Synthesis*, Boston, MA. Kluwer Academic Publishers, 1984.

[3] 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.

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

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

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

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

[8] 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.

[9] T. Sasao, "On the numbers of variables to represent multi-valued incompletely specified functions," *13th EUROMICRO Conference on Digital System Design* (DSD-2010), Lille, France, Sept. 1-3, 2010, pp. 420-423.

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

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

[12] 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.

[13] T. Sasao, "Row-shift decompositions for index generation functions," *Design, Automation & Test in Europe* (DATE-2012), March 12-16, 2012, Dresden, Germany.

[14] D. A. Simovici, D. Pletea, and R. Vetro, "Information-theoretical mining of determining sets for partially defined functions,"*ISMVL-2010*, May 2010, pp. 294-299.