Bounds on the Average Number of Products in the Minimum Sum-of-Products Expressions for Multiple-Valued Input Two-Valued Output Functions

Tsutomu Sasao

Abstract—**A lower bound $L_p(n,u)$ and an upper bound $U_p(n,u)$ on $S_p(n,u)$ are derived, where $S_p(n,u)$ is the average number of products in minimum sum-of-products expression for p-valued input two-valued output functions, n is the number of the inputs, and u is the number of minterms. The values of $S_p(n,u)$ are obtained by minimizing randomly generated functions, and they are compared to the calculated values of $U_p(n,u)$ and $L_p(n,u)$. The upper bound is based on the minimization results of the functions with fewer variables. The lower bound is based on an assumption, so it is incorrect until the assumption is proven. These bounds are useful for estimating the size of programming logic arrays.**

Index Terms—**Complexity, logic minimization, multiple-valued logic, prime implicants, programmable logic array, sum-of-products expression.**

I. Introduction

A p-valued input two-valued output function f is a mapping $f\colon\ p^n \to B$, where $P = \{0, 1, \cdots, p-1\}$ and $B = \{0, 1\}$. It is a generalization of an ordinary switching function $f\colon\ B^n \to B$. A programmable logic array (PLA) with r-bit decoders shown in Fig. 1 directly realizes a sum-of-products expression (SOPE) of a 2^r-valued input two-valued output function [1]. In Fig. 1, the AND array generates product terms, and the OR array realizes the desired functions with sum-of-products forms. The minimization of the PLA's can be done by multiple-valued logic minimizers such as MINI, MINI II, ESPRESSO-MV, QM, and ESPRESSO-EXACT. Computer experiments show that PLA's with r-bit decoders ($r \geq 2$) often require smaller arrays than standard PLA's [1].

This paper deals with the average number of products in the minimum SOPE's for p-valued input two-valued output functions. Especially for $p = 2$, the problem is to obtain the average number of products in the minimum SOPE's for switching functions, which is equal to the average number of AND gates in minimum AND–OR two-level logic circuits. Therefore, many researchers have spent considerable effort [2]. Cobham, Fridshal, and North [3] obtained the average number of products in minimum SOPE's for $p = 2$ of up to nine variables by computer simulation. Mileto and Putzolu [4] derived formulas for the average number of prime implicants and essential prime implicants for $p = 2$: their formulas give upper and lower bounds on the average number of products in minimum SOPE's. Glagolev [5] obtained a lower bound on the number of products in minimum SOPE's for almost all functions for $p = 2$. Cook and Flynn [6] investigated the average minimum cost of SOPE's and attempted to relate it to the entropy function. This author derived the formula for the average number of prime implicants for the p-valued case [7], and also obtained the average number of products in minimum SOPE's for $p = 2$ and $p = 4$ by computer simulation [8], [9]. Recently, Bender and Butler [10] improved the upper and lower bounds.

In this paper, we derive an upper and a lower bound on the average number of products in the minimum SOPE's. They are tighter than any other bounds reported to date. The upper bound is obtained by using the minimization results of functions with fewer variables. The lower bound is based on an assumption, so it is incorrect until this assumption is proven.

Manuscript received May 11, 1988; revised April 10, 1989.

The author is with the Department of Computer Science and Electronics, Kyushu Institute of Technology, Iizuka 820, Japan.

IEEE Log Number 9040671.

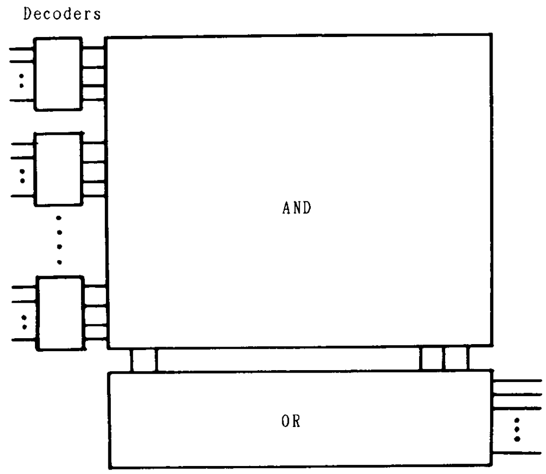

Fig. 1. PLA with t-bit decoders.

II. Definitions and Basic Properties

Definition 2.1: Let $P = \{0, 1, \cdots, p-1\}$ be a set of truth values, and X be a variable which takes a value in P. Let S be a subset of P, then X^S denotes a two-valued function $P \to B$ such that

$$X^S = \begin{cases} 0 & (\text{when } X \notin S) \\ 1 & (\text{when } X \in S) \end{cases}$$

where $B = \{0, 1\}$. The symbol X^S is called a *literal.*

Definition 2.2: A product of literals is called a *product term* (or product), and a sum of products is called a *sum-of-products expression (SOPE).*

Lemma 2.1: An arbitrary p-valued input two-valued output function $f\colon\ P^n \to B$ can be represented by the following SOPE:

$$f(X_1, X_2, \cdots, X_n) = \vee_{(S_1, S_2, \cdots, S_n)} X_1^{S_1} \cdot X_2^{S_2} \cdot \cdots \cdot X_n^{S_n}, \quad \text{where } S_i \subseteq P, \quad (i = 1, 2, \cdots, n).$$

A p-valued input two-valued output function is often simply called a *function.*

Definition 2.3: An SOPE which represents f is said to be *minimum* if the SOPE has the minimum number of products. The number of products in the minimum SOPE for f is denoted by $t(f)$.

Definition 2.4: The set of inputs which is mapped into 1 by a function f is denoted by $f^{-1}(1)$. The number of elements in $f^{-1}(1)$ (i.e., the number of the minterms of f) is called a *weight* of f and denoted by $|f|$. The average number of products in minimum SOPE's for n-variable p-valued input two-valued output functions with weight u is denoted by $S_p(n,u)$.

Lemma 2.2: $S_p(n,u) \leq \text{Min}\{u, p^{n-1}\}$.

Proof: a) An arbitrary function with weight u can be represented by an SOPE:

$$f(X_1, X_2, \cdots, X_n) = \vee_a X_1^{a_1} X_2^{a_2} \cdot \cdots \cdot X_n^{a_n},$$

where the logical sum is taken for all input combinations $\boldsymbol{a} = (a_1, a_2, \cdots, a_n)$ such that $f(\boldsymbol{a}) = 1$. There are u such combinations and so, we have $t(f) \leq u$.

0018-9340/91/0500-0645$01.00

b) An arbitrary function can be represented by an SOPE:

$$f(X_1, X_2, \cdots, X_n) = \vee_b g_b(X_1) X_2^{b_2} X_3^{b_3} \cdot \cdots \cdot X_n^{b_n} \qquad (2.1)$$

where the logical sum is taken for all the input combinations of $b = (b_2, b_3, \cdots, b_n)$ in p^{n-1}. Therefore, we have $t(f) \leq p^{n-1}$. From a) and b), we have the lemma. (Q.E.D.)

Definition 2.5: A map of an n-variable p-valued input two-valued output function consists of p^n *cells*. Cells that contain 1's are called *1-cells* while cells that contain 0's are called *0-cells*.

Example 2.1: Fig. 2 shows a map of a four-valued input two-valued output function. The SOPE for this function having the form (2.1) is

$$f(X_1, X_2) = X_1^{\{0,1,2,3\}} \cdot X_2^0 \vee X_1^{\{0,2,3\}} \cdot X_2^1 \vee X_1^{\{0,1,3\}} \cdot X_2^2 \vee X_1^{\{1\}} \cdot X_2^3. \qquad \text{(End of example).}$$

X 2 \ X 1	0	1	2	3
0	1	1	1	1
1	1		1	1
2	1	1		1
3		1		

Fig. 2. Four-valued input two-valued output function.

(X3, X4) \ (X1, X2)	00	01	11	10
00	1	1	1	1
01	1		1	1
11	1	1		1
10		1		

Fig. 3. Four-variable switching function.

Mileto and Putzolu [4] derived formulas for $G_2'(n,u)$, the average number of prime implicants of switching functions with weight u, and $G_2''(n,u)$, the average number of essential prime implicants of switching functions with weight u. Because

$$G_2''(n,u) \leq S_2(n,u) \leq G_2'(n,u),$$

$G_2'(n,u)$ and $G_2''(n,u)$ are upper and lower bounds on $S_2(n,u)$, respectively. Unfortunately, when $u \geq 2^{n-1}$ and $n \geq 10$, $G_2'(n,u)$ is greater than 2^{n-1} and $G_2''(n,u)$ is very small compared to $S_2(n,u)$. (See Fig. 5 for the result of numerical computation). Therefore, in such cases, these bounds give little information about the value of $S_2(n,u)$.

III. Upper Bound on the Average Number of Products in Minimum SOPE's

In this section, we derive $U_p(n,u)$, the upper bound on the average number of products in minimum SOPE's for n-variable p-valued input two-valued output functions with weight u.

There are $F^{(u)} = \binom{w}{u}$ different functions with weight u, where $w = p^n$. Let $f_i^{(u)}$ be the ith function with weight u $(i = 1, 2, \cdots, F^{(u)})$. By the definition of $S_p(n,u)$, we have

$$S_p(n,u) = \frac{1}{F^{(u)}} \sum_{i=1}^{F^{(u)}} t\left(f_i^{(u)}\right). \qquad (3.1)$$

Lemma 3.1: An arbitrary n-variable function f can be represented by an expression

$$f(X_1, X_2, \cdots, X_n) = \vee_{a \in P^{n-k}} E(a),$$

$$\text{where } E(a) = g_a(X_1, X_2, \cdots, X_k) \cdot X_{k+1}^{a_{k+1}} \cdot X_{k+2}^{a_{k+2}} \cdot \cdots \cdot X_n^{a_n},$$

$$g_a(X_1, X_2, \cdots, X_k) = f(X_1, X_2, \cdots, X_k, a_{k+1}, \cdots, a_n),$$

$$a = (a_{k+1}, a_{k+2}, \cdots, a_n),$$

$$\text{and } a_j \in P \ (j = k+1, \cdots, n).$$

$E(a)$ in Lemma 3.1 is called an *E-term.*

Example 3.1: The function shown in Fig. 3 can be represented by an expression: $f(X_1, X_2, X_3, X_4) = E(0,0) \vee E(0,1) \vee E(1,0) \vee E(1,1)$, where

$$E(0,0) = (\,1\,)\,\overline{x}_3\,\overline{x}_4,$$

$$E(0,1) = (x_1 \vee \overline{x}_2)\,\overline{x}_3\, x_4,$$

$$E(1,0) = (\overline{x}_1\, x_2)\, x_3\, \overline{x}_4,$$

$$\text{and } E(1,1) = (\overline{x}_1 \vee \overline{x}_2)\, x_3\, x_4. \quad \text{(End of Example).}$$

By Lemma 3.1, $f_i^{(u)}$ in (3.1) can be represented as $f_i^{(u)} = \vee_{a \in P^{n-k}} E_i(a)$. Therefore, we have

$$t\left(f_i^{(u)}\right) \leq \sum_{a \in P^{n-k}} t(E_i(a)). \qquad (3.2)$$

Example 3.2: In the expression of Example 3.1, $t(E(0,0)) = 1$, $t(E(0,1)) = 2$, $t(E(1,0)) = 1$, and $t(E(1,1)) = 2$. Therefore, the right-hand side of (3.2) is equal to $\sum_{a \in P^2} t(E(a)) = 1 + 2 + 1 + 2 = 6$. On the other hand, the left-hand side of (3.2) is $t(f) = 4$ as shown in Fig. 4. (End of example).

By (3.1) and (3.2), we have $S_p(n,u) \leq U_p(n,u)$, where

$$U_p(n,u) = \frac{1}{F^{(u)}} \sum_{i=1}^{F^{(u)}} \sum_{a \in P^{n-k}} t(E_i(a)).$$

By changing the order of the summation, we obtain

$$U_p(n,u) = \frac{1}{F^{(u)}} \sum_{a \in P^{n-k}} \sum_{i=1}^{F^{(u)}} t(E_i(a)). \qquad (3.3)$$

Note that

$$\frac{1}{F^{(u)}} \sum_{i=1}^{F^{(u)}} t(E_i(a)) \qquad (3.4)$$

denotes the average number of products in minimum SOPE's for the functions with weight u having $E_i(a) = g_{i\,a}(X_1, X_2, \cdots, X_k) \cdot X_{k+1}^{a_{k+1}} \cdot X_{k+2}^{a_{k+2}} \cdot \cdots \cdot X_n^{a_n}$ as E-terms. $g_{i\,a}(X_1, X_2, \cdots, X_k)$, $(i = 1, 2, \cdots, F^{(u)})$ are k-variable functions, and have 2^{p^k} different patterns. Note that the value of (3.4) does not depend on a. Let $[F^k]$ be the set of all the k-variable functions, then (3.3) can be represented by the sum with respect to the functions $g_i \in [F^k]$ and (3.3) is represented as

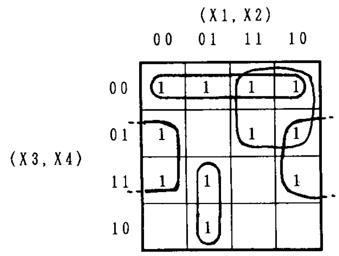

Fig. 4. Minimum SOPE for Fig. 3.

$$U_p(n,u) = \frac{p^{n-k}}{F^{(u)}} \cdot \sum_{g_j \in [F^k]} t(E(g_j)) \cdot L_p(n, g_j, u),$$

where $E(g_j) = g_j(X_1, X_2, \cdots, X_k) X_{k+1}^{a_{k+1}} X_{k+2}^{a_{k+2}} \cdot \cdots \cdot X_n^{a_n}$, $L_p(n, g_j, u)$ denotes the number of n-variable functions with weight u having $E(g_j)$ as an E-term.

Let c_0 be an E-term, and the weight of c_0 be $|g_j|$. $L_p(n, g_j, u)$ is equal to the number of different functions with weight u, where $|g_j|$ 1-cells and $(p^k - |g_j|)$ 0-cells are fixed. Therefore, $L_p(n, g_j, u) = \binom{w-p^k}{u-|g_j|}$. Because $t(E(g_j)) = t(g_j)$, we have

$$U_p(n,u) = \frac{p^{n-k}}{F^{(u)}} \cdot \sum_{g_j \in [F^k]} t(g_j) \cdot \binom{w - p^k}{u - |g_j|}. \tag{3.5}$$

In order to compute the value of (3.5), we have to obtain the minimum SOPE's of all the k-variable functions. This work can be greatly reduced by considering the representative functions of the equivalence classes.

Definition 3.1: The relation $\sim$ satisfying the following conditions is called *VP-equivalence relation.*

1) $f \sim f$.
2) If $f_1 = f(\cdots, X_i, \cdots, X_j, \cdots)$ and $f_2 = f(\cdots, X_j, \cdots, X_i, \cdots)$, then $f_1 \sim f_2$. (Permutation of input variables)
3) Let $\sigma : P \to P$ be an arbitrary one-to-one mapping. If $f_1 = f(\cdots, X_i, \cdots)$ and $f_2 = f(\cdots, \sigma(X_j), \cdots)$, then $f_1 \sim f_2$. (Permutation of values in a variable)

Especially, when $p = 2$, VP-equivalence is called NP-equivalence [11], [12].

Theorem 3.1: Let $U_p(n, u)$ be an upper bound on the average number of products in minimum SOPE's for n-variable p-valued input two-valued output functions. Then

$$U_p(n,u) = \frac{p^{n-k}}{F^{(u)}} \sum_{j=1}^{p^k} c(j) \cdot \binom{w - p^k}{u - j},$$

where

$$c(j) = \sum_{|g_i| = j} \mu(g_j) \cdot t(g_i),$$

and $g_1, g_2, \cdots, g_\lambda$ are representative functions of VP-equivalence classes, $t(g_i)$ is the number of products in a minimum SOPE for g_i, and $\mu(g_i)$ is the number of functions which are VP-equivalent to g_i.

Proof: By using VP-equivanence relation, we can partition $[F^k]$ into the VP-equivalence classes. Let $g_1, g_2, \cdots, g_\lambda$ be the representative functions of the VP-equivalence classes. Then (3.5) can be rewritten as

$$U_p(n,u) = \frac{p^{n-k}}{F^{(u)}} \cdot \sum_{j=1}^{\lambda} t(g_j) \cdot \mu(g_j) \cdot \binom{w - p^k}{u - |g_j|}$$

where $\mu(g_i)$ is the number of functions which are VP-equivalent to g_i. Next, by classifying λ different equivalence classes by the weight of the functions, we have

$$\sum_{j=1}^{\lambda} t(g_j) \cdot \mu(g_j) \binom{w - p^k}{u - |g_j|} = \sum_{j=1}^{p^k} c(j) \binom{w^k - p}{u - j},$$

$$\text{where } c(j) = \sum_{|g_i| = j} \mu(g_j) \cdot t(g_i)$$

and $g_1, g_2, \cdots, g_\lambda$ are representative functions of VP-equivalence classes. Hence, we have the theorem. (Q.E.D.)

Example 3.3: Suppose that $p = 2$ and $k = 3$. Table I shows the representative functions of VP (= NP)-equivalence classes of three variables. There are 22 classes. By minimizing all the representative functions, we have the coefficients shown in Table II.

(End of example).

Example 3.4: Suppose that $p = 2$ and $k = 4$. There are 402 different VP-equivalence classes of 4-variable functions. In a similar way, we have the coefficients $c(j)$ shown in Table III.

(End of example).

Example 3.5: Suppose that $p = 4$ and $k = 2$. There are 192 different representative functions of four-valued input two-valued output functions. Table III also shows the coefficients $c(j)$.

(End of example).

Logic minimization in Examples 3.3–3.5 were done by QM [13], a modified Quine–McCluskey algorithm for p-valued input two-valued output functions. The total computation time was within one hour by using a personal computer. In general, the larger k, the tighter the upper bound. For $p = 2$ and $k = 5$, there are 1 228 158 VP equivalence classes [12], which may not be impossible to compute the coefficients. However, for $p = 2$ and $k = 6$, the number of equivalence classes is about 4×10^{14} [12], which is too many to minimize all the representative functions.

IV. Lower Bound on the Average Number of Products in Minimum SOPE's

In this section, we derive $L_p(n, u)$, a lower bound on the average number of products in minimum SOPE's for n-variable p-valued input two-valued output functions with weight u.

Definition 4.1: A product $p_1 = X_1^{S_1} X_2^{S_2} \cdot \cdots \cdot X_n^{S_n}$ is an *implicant* of f if $p_1 \le f$. A product p_1 is said to be a *prime implicant* of f if there is no product p_2 such that $p_2 \le f$, $p_1 \le p_2$, and $p_1 \ne p_2$. Let there be k_j S_i's such that $|S_i| = j$ for $j = 1, 2, \cdots, p$. Then, this product is said to be k-cube, where $\boldsymbol{k} = (k_1, k_2, \cdots, k_p)$.

Example 4.1: Let $p = 4$ and $n = 4$. $X_1^{\{0,1,3\}} \cdot X_2^{\{0,1\}} \cdot X_3^{\{1,2\}} \cdot X_4^{\{0\}}$ is a $(1, 2, 1, 0)$-cube, while $X_1^{\{0,1,2,3\}} \cdot X_2^{\{1,2\}} \cdot X_3^{\{2,3\}} \cdot X_4^{\{0,1,2\}}$ is a $(0, 2, 1, 1)$-cube.

Definition 4.2: $G_p'(n, \boldsymbol{k}, u)$ denotes the average number of prime k-cubes of n-variable p-valued input two-valued output functions with weight u.

Theorem 4.1:

$$G_p'(n, \boldsymbol{k}, u) = \frac{C^{(\boldsymbol{k})}}{F^{(u)}} \cdot \sum_{t=0}^{\eta(\boldsymbol{k})} (-1)^t \cdot \sum_{\boldsymbol{t}} \lambda(\boldsymbol{k}, \boldsymbol{t}) \cdot \binom{w - w(\boldsymbol{k}, \boldsymbol{t})}{u - w(\boldsymbol{k}, \boldsymbol{t})},$$

TABLE I
REPRESENTATIVE FUNCTIONS OF THREE VARIABLES

j	Representative Function	Weight	Number of Functions in the class	Number of Products in Minimum SOPE
	g_j	$\lvert g_j \rvert$	$\mu(g_j)$	$t(g_j)$
1	0	0	1	0
2	$\bar a \bar b \bar c$	1	8	1
3	$\bar a \bar b$	2	12	1
4	$\bar a(\bar b c \vee b \bar c)$	2	12	2
5	$\bar a(\bar b \vee \bar c)$	3	24	2
6	$\bar a$	4	6	1
7	$\bar a bc \vee \bar a b \bar c \vee a \bar b \bar c$	3	8	3
8	$\bar a \bar b \vee \bar b \bar c \vee \bar c \bar a$	4	8	3
9	$a \bar b \bar c \vee \bar a bc$	2	4	2
10	$\bar a bc \vee \bar b \bar c$	3	24	2
11	$\bar a c \vee \bar b \bar c$	4	24	2
12	$\bar a b \vee \bar a c \vee a \bar b \bar c$	4	24	3
13	$\bar a \vee \bar b \bar c$	5	24	2
14	$\bar a b \vee a \bar b$	4	6	2
15	$\bar a b \vee a \bar b \vee \bar a \bar c$	5	24	3
16	$\bar a \vee \bar b$	6	12	2
17	$a \oplus b \oplus c \oplus 1$	4	2	4
18	$\bar a c \vee \bar a \bar b \vee \bar b c \vee ab \bar c$	5	8	4
19	$\bar a \vee \bar b c \vee b \bar c$	6	12	3
20	$a \bar b \vee b \bar c \vee \bar a c$	6	4	3
21	$\bar a \vee \bar b \vee \bar c$	7	8	3
22	1	8	1	1

TABLE II
$C(j)$ FOR $p = 2$ AND $k = 2$

j	c(j)
1	8
2	44
3	120
4	170
5	152
6	72
7	24
8	1

where

$$F^{(u)} = \binom{w}{u}, \qquad w = p^n, \qquad C^{(\boldsymbol{k})} = (n!) \cdot \prod_{i=1}^{p} \left[\frac{1}{k_i!} \binom{p}{i}^{k_i} \right],$$

$$w(\boldsymbol{k}, \boldsymbol{t}) = \xi(\boldsymbol{k}) \cdot \left\{ 1 + \sum_{i=1}^{p-1} \frac{t_i}{i} \right\}, \qquad \xi(\boldsymbol{k}) = \prod_{i=1}^{p} (i)^{k_i},$$

$$\eta(\boldsymbol{k}) = \sum_{i=1}^{p-1} a_i, \qquad a_i = k_i(p - i),$$

TABLE III
$C(j)$

j	c(j)	
	p=2 k=4	p=4 k=2
1	16	16
2	208	192
3	1328	1184
4	5288	4508
5	14720	12336
6	29872	24248
7	46368	36992
8	54992	42720
9	50992	37072
10	36336	24632
11	19856	13056
12	8056	5120
13	2352	1360
14	448	240
15	64	32
16	1	1

$\boldsymbol{t} = (t_1, t_2, \cdots, t_{p-1})$ is a partition of t, $\lambda(\boldsymbol{k}, \boldsymbol{t}) = \prod_{i=1}^{p-1} \binom{a_i}{t_i}$, and $t_i \le a_i (i = 1, 2, \cdots, p-1)$. (Proof is available from the author.)

Lemma 4.1: Let $V_p(n, u)$ be the average volume of prime cubes. Then, $V_p(n, u) = A/B$, where $A = \sum_{\boldsymbol{k}} w(\boldsymbol{k}) \cdot G'_p(n, \boldsymbol{k}, u)$, and $B = \sum_{\boldsymbol{k}} G'_p(n, \boldsymbol{k}, u)$.

Proof: It is easy to see that A denotes the average sum of the volume of the prime cubes, and that B denotes the average number of the prime cubes. Hence, A/B denotes the average volume of prime cubes. (Q.E.D.)

Now we will make the following:

Assumption 4.1: The average volume of prime cubes in a minimum SOPE for f is equal to the average volume of all the prime cubes of f. (Note that, in general, there are many minimum SOPE's for a function f.)

By Assumption 4.1, we have the following:

Conjecture 4.1: $S_p(n, u) \ge L_p(n, u)$, where $L_p(n, u) = (u \cdot B)/A$,

$$A = \sum_{\boldsymbol{k}} w(\boldsymbol{k}) \cdot G'_p(n, \boldsymbol{k}, u), \qquad \text{and } B = \sum_{\boldsymbol{k}} G'_p(n, \boldsymbol{k}, u).$$

(Explanation supporting the conjecture) There are u minterms in f. Because the average volume of each cube is $V_p(n, u)$, any SOPE for f requires at least $u/V_p(n, u)$ cubes to cover all the minterms of f. (End of the explanation).

Example 4.2: Consider the function shown in Fig. 3 where $n = 4$, $p = 2$, and $u = 11$. The number of prime cubes is 7, and the sum of volumes of all prime cubes is 22. The average volume of prime cubes is $22/7 = 3.14$. If Assumption 4.1 holds, then the lower bound on the number of products in minimum SOPE for f is $11/3.14 = 3.5$. Fig. 4 shows a minimum SOPE for f. Note that only two prime implicants are essential. Therefore, Assumption 4.1 gives a tighter lower bound than the bound given by the number of essential prime implicants. (End of example).

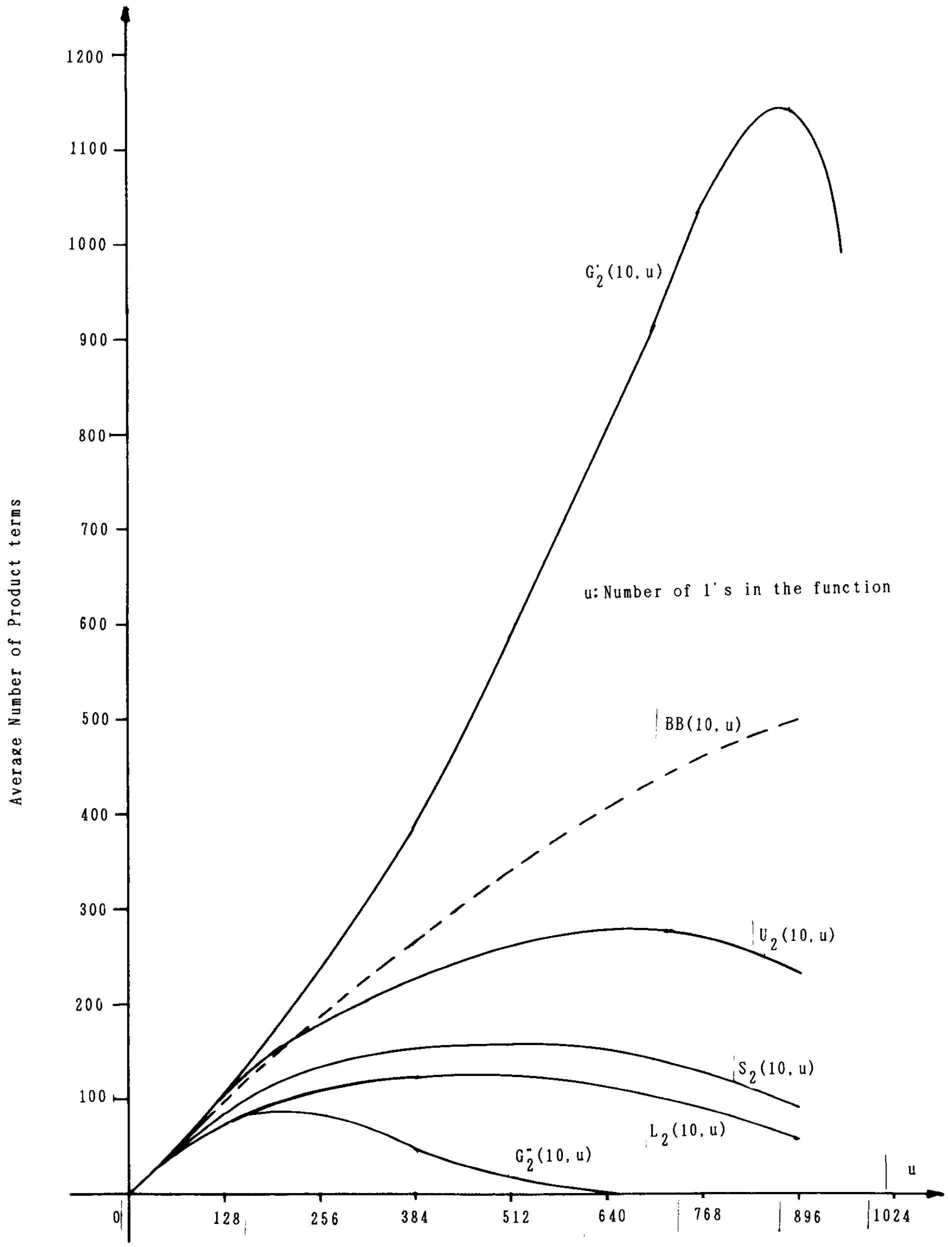

Fig. 5. Upper and lower bounds on the average number of products in the minimum SOPE's of functions with $p = 2$ and $n = 10$ versus the number of the minterms.

V. Experimental Results

In order to obtain $S_p(n, u)$, we generated functions by a pseudorandom number generation algorithm. For $u = 128$, 256, 384, 512, 640, 768, and 896, we generated ten sample functions, and minimized each function by QM. Fig. 5 compares the values of $S_2(n, u)$, $G_2'(n, u)$, $G_2''(n, u)$, $U_2(n, u)$, $L_2(n, u)$, and $\mathrm{BB}(n, u)$ for $n = 10$. $\mathrm{BB}(n, u)$ is the upper bound derived by Bender and Butler [10], where

$$\mathrm{BB}(n, u) = u - \frac{2^{n-1}}{\binom{2^n}{u}} \sum_{j=1}^{n} \binom{2^n}{u-2}.$$

This bound was derived by covering all 1's with implicants consisting of pairs of 1's and single 1's. The highest curve in Fig. 5 is $G_2(10, u)$, the average number of prime implicants of ten-variable functions.

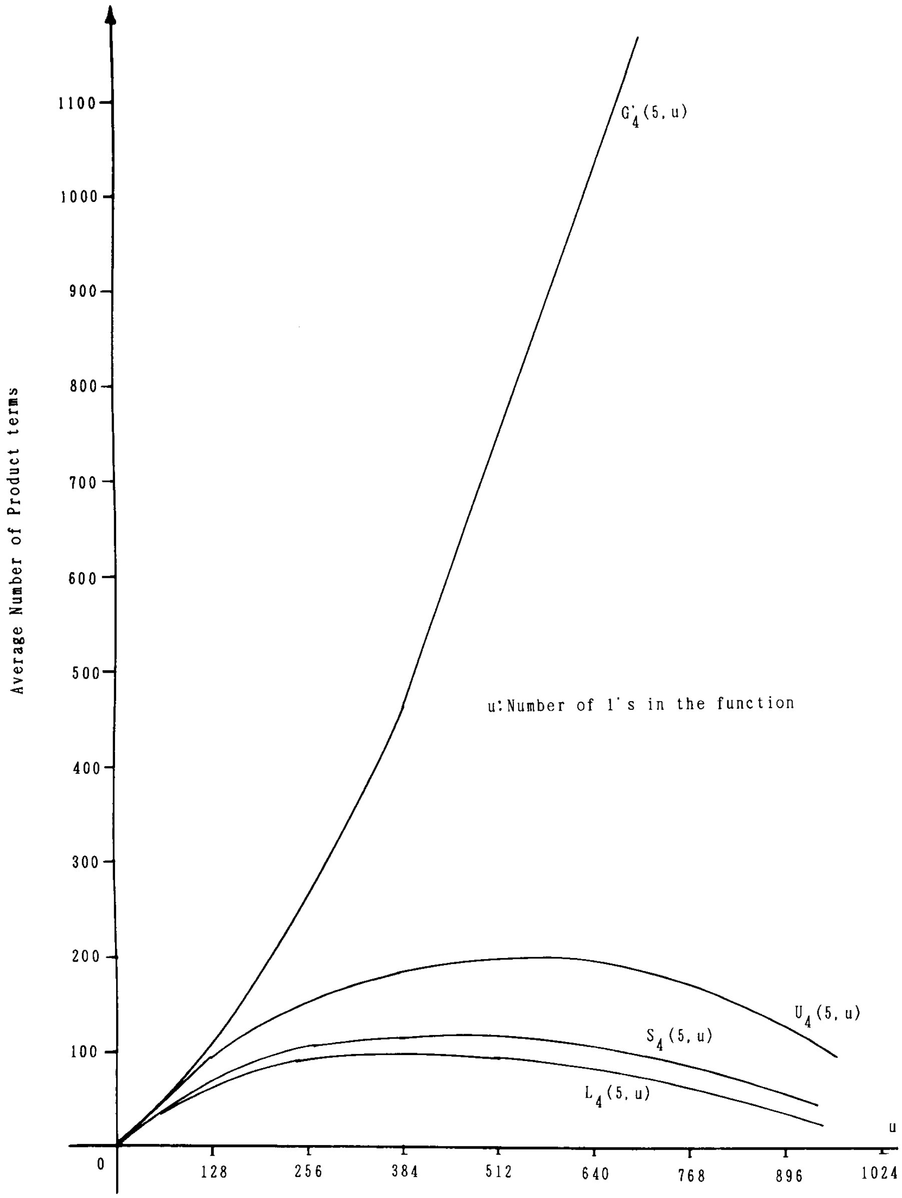

Fig. 6. Upper and lower bounds on the average number of products in the minimum SOPE's of functions with $p = 4$ and $n = 5$ versus the number of the minterms.

Note that $G_2(10, u)$ is greater than u when $u \geq 384$. The dashed curve lying mostly below $G_2(10, u)$ is $\mathrm{BB}(10, u)$. The solid curve below $\mathrm{BB}(10, u)$ is $U_2(10, u)$, the upper bound obtained in this paper. $U_2(10, u)$ is better than $\mathrm{BB}(10, u)$ when $u \geq 240$. However, for $u \leq 240$, $\mathrm{BB}(10, u)$ is better than $U_2(10, u)$.

As for lower bounds, $L_2(10, u)$ is much better than $G_2''(10, u)$. Especially, when $u \geq 512$, $G_2(10, u)$ is very small compared to $L_2(10, u)$. Bender and Butler also obtained a lower bound on $S_2(n, u)$ by considering all the essential prime implicants plus certain added implicants. However, there is very little difference between their bound and $G_2''(n, u)$. So it is omitted from Fig. 5.

Fig. 6 shows the case for $p = 4$ and $n = 5$. In this case, however, $G_4''(n, u)$ is omitted because no formula is known for it. We obtained values of $U_p(n, u)$ by the coefficients shown in Table III.

VI. Conclusion

In this paper, we derived an upper bound $U_p(n, u)$ and a lower bound $L_p(n, u)$ on $S_p(n, u)$, the average number of prime implicants in minimum SOPE's for n-variable p-valued input two-valued output functions, where u is the number of minterms. $U_p(n, u)$ was derived by using the statistical data of minimum SOPE's for all k-variable functions ($k \leq n$). On the other hand, $L_p(n, u)$ was derived by using the concept of the average volume of the prime cubes. These bounds are tighter than any other bounds reported to date, and applicable to any value of p. Especially for $p = 2$ and $p = 4$, we have obtained the coefficients for the upper bounds. So, we can easily estimate the average size of standard PLA's with two-bit decoders by using a personal computer.

References

[1] T. Sasao, "Multiple-valued logic and optimization of programmable logic arrays," *IEEE Trans. Comput.*, vol. 21, no. 4, pp. 71–80, Apr. 1988.

[2] C. A. Papachristou, "Characteristic measures of switching functions," *Inform. Sci.*, vol. 13, pp. 51–75, 1977.

[3] A. Cobham, R. Fridshal, and J. H. North, "A statistical study of the minimization of Boolean functions using integer linear programming," IBM Res. Rep. RC-756, June 1962.

[4] F. Mileto and G. Putzolu, "Average values of quantities appearing Boolean function minimization," *IEEE Trans. Electron. Comput.*, vol. EC-13, pp. 87–92, Apr. 1964.

[5] V. V. Glagolev, "Some bounds for disjunctive normal forms of the algebra of logic," *Problemi Kibernetiki*, vol. 19, pp. 74–93, 1970. (English translation, System Theory Research, Consultants Bureau, New York).

[6] R. W. Cook, and M. J. Flynn, "Logical network cost and entropy," *IEEE Trans. Comput.*, vol. C-22, no. 9, pp. 823–826, Sept. 1973.

[7] T. Sasao and H. Terada, "Multi-valued logic and the design of programmable logic arrays with decoders," in *Proc. ISMVL-79*, Bath, England, May 1979, pp. 27–37.

[8] ——, "On the complexity of shallow logic functions and the estimation of programmable logic array size," in *Proc. ISMVL-80*, Evanston, May 1980, pp. 65–73.

[9] T. Sasao, "Multiple-valued decomposition of generalized Boolean functions and the complexity of programmable logic arrays," *IEEE Trans. Comput.*, vol. C-30, no. 9, pp. 635–643, Sept. 1981.

[10] E. A. Bender and J. T. Butler, "On the size of PLA's required to realize binary and multiple-valued functions," *IEEE Trans. Comput.*, vol. 38, no. 1, pp. 82–98, Jan. 1989.

[11] M. A. Harrison, *Introduction to Switching and Automata Theory.* New York: McGraw-Hill, 1965.

[12] S. Muroga, *Logic Design and Switching Theory.* New York: Wiley, 1979.

[13] T. Sasao, "Input variable assignment and output phase optimization of PLA's," *IEEE Trans. Comput.*, vol. C-33, no. 10, pp. 879–984, Oct. 1984.

Efficient Replicated Remote File Comparison

John J. Metzner

***Abstract*— This paper improves on some previous work involving communication to find the location of disagreements between two or more large remote data files. Some new procedures are described which permit a significant reduction in the number of back-and-forth interchanges and in one version also significantly reduce the total amount of data transfer required. Suppose the file is divided into 2^P pages. The previous scheme could locate a disagreeing page with an extremely high degree of confidence after $P + 1$ interchanges. The new procedures require at most *one* back-and-forth interchange for each disagreeing page, often considerably less, up to a maximum of $P + 1$ interchanges for any number of disagreeing pages.**

***Index Terms*—Communication efficiency, error location, fault location, fault recovery, reliable distributed systems, replicated files, tree parity structure.**

I. Introduction

A feature which is becoming increasingly more common in distributed processing systems is remote replication of large data files. These are maintained for the joint purposes of convenient local availability and improved fault recovery. Since the files are supposed to be identical, it often is important to verify this fact and to resolve differences. These differences may arise for various reasons. An important case is where differences arise as a result of revisions and updatings. Although procedures have been devised for systematically updating and maintaining consistency in multicopy databases [1]–[5], the problem is difficult and procedures are not always foolproof [4]. Also, faults, system failures, or operator errors may create differences.

When differences are found in large remote file copies, it might be very costly in communication resources to send the entire file or a substantial part of it from one location to another in order to resolve the discrepancy. Often the disagreement is slight, and great savings in communication are possible by a disagreement location scheme.

A prior paper [6] described a tree parity structure as a mechanism for locating discrepancies with a relatively small amount of communication between remotely-located supposedly identical large data files. The files are assumed to be constructed as identical on a bit-by-bit basis. The procedure may not be applicable without additional processing if the information, although identical, is stored in a different way in the two locations. Each file is assumed to be divided into units, referred to as "pages." These units could be of arbitrary size and need not all be the same size. To allow the comparison technique to be practical, an insertion or deletion within a page should not directly affect the contents of the other pages; otherwise, if division was made into fixed size blocks, insertions or deletions could shift data from one block to another and cause almost all blocks to be different.

The procedure in [6] works rather efficiently in terms of amount of bits communicated to locate regions of discrepancy, but tends to have a large number of information exchanges, which is undesirable due to the possibly large turnaround overhead. If the disagreement is limited to a single page or a cluster of pages, more efficient procedures are possible. Reference [7] shows how to encode so as to locate one page or a cluster of pages in one exchange using a moderate number of signature unit transmissions and less parity storage, and [8] shows how to locate a single disagreeing page in one exchange by sending only two signature-size units. However, we are seeking here a more robust procedure which is able to locate any pattern of disagreeing pages.

This paper will show methods for further significant reductions both in the number of interchanges and in the total amount of data transferred.

II. The Basic Parity Tree Structure

A binary parity sequence or "signature" is derived from each page of each file. The page size is normally many times larger than the size of a signature. Let there be 2^P pages. The tree then will have $P + 1$ levels including the root level, which will be designated level 0. Fig. 1 illustrates the notation being used for the tree structure. The page signatures will be at level P at the ends of the leaves. Page i data will have a C-bit signature $S[P, i]$. Level k will have 2^k nodes, numbered from $(k, 0)$ to $(k, 2^k - 1)$.

Manuscript received June 1, 1988. This work was supported by the National Science Foundation under Grant 8796229.

The author is with the Department of Electrical Engineering, Pennsylvania State University, University Park, PA 16802.

IEEE Log Number 9040674.