On the Minimization of SOPs for Bi-Decomposable Functions

Tsutomu Sasao

Center for Microelectronic Systems
and Department of Computer Science and Electronics
Kyushu Institute of Technology
Iizuka, Fukuoka, 820-8205 JAPAN
Tel: +81-948-29-2675
Fax: +81-948-29-2603
e-mail: sasao@cse.kyutech.ac.jp

Jon T. Butler

Department of Electrical
and Computer Eng.
Naval Postgraduate School
Monterey, CA 93943-5121 U.S.A.
Tel: 831-656-3299
Fax: 831-656-2760
e-mail butler@cs.nps.navy.mil

Abstract— A function f is AND bi-decomposable if it can be written as $f(X_1, X_2) = h_1(X_1)h_2(X_2)$. In this case, a sum-of-products expression (SOP) for f is obtained from minimum SOPs (MSOP) for h_1 and h_2 by applying the law of distributivity. If the result is an MSOP, then the complexity of minimization is reduced. However, the application of the law of distributivity to MSOPs for h_1 and h_2 does not always produce an MSOP for f. We show an incompletely specified function of $n(n-1)$ variables that requires at most n products in an MSOP, while 2^{n-1} products are required by minimizing the component functions separately.

We introduce a new class of logic functions, called orthodox functions, where the application of the law of distributivity to MSOPs for component functions of f always produces an MSOP for f. We show that orthodox functions include all functions with three or fewer variables, all symmetric functions, all unate functions, many benchmark functions, and few random functions with many variables.

I. Introduction

A logic function f is said to have a bi-decomposition if f is written as $f(X,Y) = g(h_1(X), h_2(Y))$, where $X \cap Y = \emptyset$. If g is the AND (OR) function, then f has an AND (OR) bi-decomposition. Many practical functions have bi-decomposition [6], and minimum sum-of-products expressions (MSOPs) for them are often easy to obtain.

For example, if f has an OR bi-decomposition: $f = h_1(X) \vee h_2(Y)$, then the MSOP for f is obtained as the OR of MSOPs for h_1 and h_2. This is a desirable property, since the time to optimize an n-variable function given as an SOP of t products is at least $O(nt^2)$.

Assume that f is known to have an AND bi-decomposition, $f = h_1(X)h_2(Y)$. If an MSOP for f is obtained from MSOPs for h_1 and h_2 followed by the application of the law of distributivity, then this computation would be much faster. Let the number of products in MSOPs for h_1 and h_2 be t_1 and t_2, respectively. Then, an SOP for f contains t_1t_2 products. A function with this many products normally takes at least $O(n(t_1t_2)^2)$ time to minimize. However, since the law of distributivity can be used to form the MSOP, one can minimize two smaller component functions, requiring only $O(nt_1^2 + nt_2^2 + t_1t_2)$ time, which is usually much less.

Unfortunately, in the case of AND bi-decompositon, we cannot always achieve an MSOP by optimizing the component functions independently. In this paper, we consider a special case where the independent optimization of component MSOPs indeed produces an MSOP for f.

II. Notation

Definition 2.1 *x and $\bar{x}$ are* **literals** *of a variable x. The AND of literals is a* **product** *or* **implicant**. *The OR of products is a* **sum-of-products expression** *(***SOP***).*

Definition 2.2 *A* **prime implicant** *(***PI***) of a function f is an implicant that implies f, such that the deletion of any literal results in a new implicant that does not imply f.*

Definition 2.3 *An* **irredundant sum-of-products expression** *(***ISOP***) is the OR of PIs, such that no PI can be deleted without changing the function represented by the expression.*

Definition 2.4 *Among the ISOPs for f, one with the fewest PIs is a* **minimum SOP** *or* **MSOP**.

Definition 2.5 *$\tau(MSOP : f)$ denotes the number of PIs in an MSOP for f.*

In the discussion to follow, we will use symmetric functions.

Definition 2.6 *S^n_A, a (***totally***) symmetric function, is 1 if m of its n variables are 1, where $m \in A$ and is 0 otherwise.*

Example 2.1 *The AND and OR functions of n variables are symmetric and represented by $S^n_{\{n\}}$ and $S^n_{\{1,2,\ldots n\}}$, respectively. (End of Example)*

Definition 2.7 *Given an n-variable function $f(X)$, f^p denotes the np-variable function*

$$f^p(X_1 \cup X_2 \cup \ldots \cup X_p) = f(X_1)f(X_2)\ldots f(X_p),$$

where $X_1, X_2, \ldots$, and X_p are pairwise disjoint sets of variables.

III. Problems in the Minimization of SOPs of Bi-Decomposable Functions

A. AND and OR Bi-Decompositions

Lemma 3.1 *Let p_1 and p_2 be implicants on X_1 and X_2, respectively, such that $X_1 \cap X_2 = \emptyset$. p_1 and p_2 are PIs of $h_1(X_1)$ and $h_2(X_2)$, respectively iff*

1. *p_1 and p_2 are PIs of $h_1(X_1) \vee h_2(X_2)$, and*
2. *p_1p_2 is a PI of $h_1(X_1)h_2(X_2)$.*

We prove Part 1 of this lemma; Part 2 is done in a similar manner. The "if" part is true because if p is a PI of either h_1 or h_2, it is trivially an implicant of $h_1 \vee h_2$. Because the variable sets X_1 and X_2 do not overlap, p is also a PI of $h_1 \vee h_2$. The "only if" part is true as follows. Let p be a PI of $h_1 \vee h_2$, and let it be expressed as $p = p_1 p_2$, where p_1 consists of literals from X_1 only and p_2 consists of literals from X_2 only. Since p is a PI of $h_1 \vee h_2$, then an assignment of values to the variables associated with $p_1 p_2$ causes either h_1 or h_2 or both to be 1. Suppose h_1 is 1; the case where h_2 is 1 is similar. Since h_1 is 1, p_1 is an implicant of h_1. But, $p_1 p_2$ cannot be a PI unless $p_2 = 1$. Further, p_1 must be a PI of h_1. On the contrary, if not, it implies a PI, p_1' of h_1. Thus, $p_1' p_2$ must be an product that implies $h_1 \vee h_2$, that is implied by $p_1 p_2$. But, this results in a contradiction, since $p_1 p_2$ is a PI. It must be that p_1 is a PI of h_1.

The OR of MSOPs for $h_1(X_1)$ and $h_2(X_2)$ is an SOP that represents $h_1(X_1) \vee h_2(X_2)$. Similarly, the AND of MSOPs for $h_1(X_1)$ and $h_2(X_2)$ is an SOP that represents $h_1(X_1)h_2(X_2)$. Thus, it follows that

Lemma 3.2 *Let $h_1(X_1)$ and $h_2(X_2)$ be functions each not identically 1, such that $X_1 \cap X_2 = \emptyset$. Then,*

$$\tau(MSOP : h_1 \vee h_2) \leq \tau(MSOP : h_1) + \tau(MSOP : h_2) \text{ and}$$
$$\tau(MSOP : h_1 h_2) \leq \tau(MSOP : h_1)\tau(MSOP : h_2).$$

It is tempting to believe that Lemma 3.2 is true when the two $\leq$ relations are replaced by $=$. Consider these two statements separately.

Proposition 3.1 *Let $h_1(X_1)$ and $h_2(X_2)$ be functions each not identically 1, such that $X_1 \cap X_2 = \emptyset$. Then,*

$$\tau(MSOP : h_1 \vee h_2) = \tau(MSOP : h_1) + \tau(MSOP : h_2).$$

Proposition 3.1 follows directly from 1. Lemma 3.1 and 2. the observation that no PI of h_1 is a PI of h_2; i.e. they depend on different variables.

Proposition 3.2 *Let $h_1(X_1)$ and $h_2(X_2)$ be functions not identically 1, such that $X_1 \cap X_2 = \emptyset$. Then,*

$$\tau(MSOP : h_1 h_2) = \tau(MSOP : h_1)\tau(MSOP : h_2).$$

That is, since the two variable sets, X_1 and X_2, are disjoint, it seems reasonable that finding an MSOP for $h_1(X_1)$ and $h_2(X_2)$ separately and forming an SOP by applying the law of distributivity to $h_1(X_1)h_2(X_2)$ yields an MSOP.

B. A Counterexample

However, this is *not* true. Voight-Wegener [7] show a 5-variable function for which Proposition 3.2 does *not* hold. This is related to a result by Odlyzko [3] who shows that the covering number of the product of two graphs is less than the product of the covering numbers of the component graphs. In this paper, we show a 4-variable counterexample, $f(x_1, x_2, x_3, x_4)$ that is simpler than that of Voight-Wegener [7]. As will be discussed, there is no simpler function with this property. Fig. 3.1 shows its Karnaugh Map.

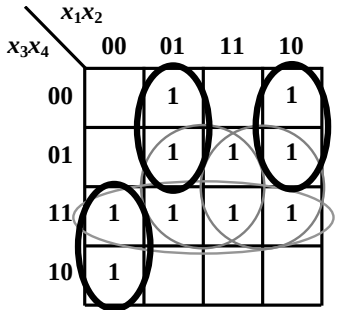

Fig. 3.1. Example of a four variable function showing Proposition 3.2 does not hold .

This figure shows all six PIs of this function. Three, shown by dark ellipses, are essential. Of the remaining three, only two are needed to completely cover the function. Therefore, $\tau(MSOP, f) = 5$. Note that the three essential PIs cover three minterms covered by the non-essential PIs, and, since all three essential PIs are required in an MSOP, the essential PIs, in effect, create three don't care minterms covered by non-essential PIs. There are three MSOPs for f, one of which is

$$f(x_1, x_2, x_3, x_4) = [\bar{x}_1 \bar{x}_2 x_3 \vee \bar{x}_1 x_2 \bar{x}_3 \vee x_1 \bar{x}_2 \bar{x}_3] \vee [x_2 x_4 \vee x_1 x_4]. \quad (1)$$

Here, the first three terms in brackets represent the essential PIs, while the last two terms in brackets represent the non-essential PIs.

Consider the 8-variable function $f^2 = f(X)f(Y)$. Using the expression for f in (1), we can derive an expression for f^2 as follows.

$$\begin{aligned} f^2 &= f(X)f(Y) = f(x_1, x_2, x_3, x_4) f(y_1, y_2, y_3, y_4) \\ &= A(X,Y) \vee B(X,Y) \vee C_1(X,Y), \end{aligned}$$

where

$$\begin{aligned} A(X,Y) = {} & \bar{x}_1 \bar{x}_2 x_3 \bar{y}_1 \bar{y}_2 y_3 \vee \bar{x}_1 \bar{x}_2 x_3 \bar{y}_1 y_2 \bar{y}_3 \vee \bar{x}_1 \bar{x}_2 x_3 y_1 \bar{y}_2 \bar{y}_3 \\ & \vee \bar{x}_1 x_2 \bar{x}_3 \bar{y}_1 \bar{y}_2 y_3 \vee \bar{x}_1 x_2 \bar{x}_3 \bar{y}_1 y_2 \bar{y}_3 \vee \bar{x}_1 x_2 \bar{x}_3 y_1 \bar{y}_2 \bar{y}_3 \\ & \vee x_1 \bar{x}_2 \bar{x}_3 \bar{y}_1 \bar{y}_2 y_3 \vee x_1 \bar{x}_2 \bar{x}_3 \bar{y}_1 y_2 \bar{y}_3 \vee x_1 \bar{x}_2 \bar{x}_3 y_1 \bar{y}_2 \bar{y}_3 \\ B(X,Y) = {} & \bar{x}_1 \bar{x}_2 x_3 y_2 y_4 \vee \bar{x}_1 \bar{x}_2 x_3 y_1 y_4 \vee \bar{x}_1 x_2 \bar{x}_3 y_2 y_4 \\ & \vee \bar{x}_1 x_2 \bar{x}_3 y_1 y_4 \vee x_1 \bar{x}_2 \bar{x}_3 y_2 y_4 \vee x_1 \bar{x}_2 \bar{x}_3 y_1 y_4 \\ & \vee x_2 x_4 \bar{y}_1 \bar{y}_2 y_3 \vee x_2 x_4 \bar{y}_1 y_2 \bar{y}_3 \vee x_2 x_4 y_1 \bar{y}_2 \bar{y}_3 \\ & \vee x_1 x_4 \bar{y}_1 \bar{y}_2 y_3 \vee x_1 x_4 \bar{y}_1 y_2 \bar{y}_3 \vee x_1 x_4 y_1 \bar{y}_2 \bar{y}_3 \\ C_1(X,Y) = {} & x_2 x_4 y_2 y_4 \vee x_2 x_4 y_1 y_4 \vee x_1 x_4 y_2 y_4 \vee x_1 x_4 y_1 y_4 \end{aligned}$$

Here, $A(X,Y)$ is the product of PIs that are essential in both $f(X)$ and $f(Y)$, $B(X,Y)$ is the product of one essential and one non-essential PI, while $C_1(X,Y)$ is the product of PIs that are non-essential in both f's. There are a total of 25 PIs. However, f^2 can be represented using only 24 PIs, as follows.

$$f^2 = A(X,Y) \vee B(X,Y) \vee C_2(X,Y), \quad (2)$$

where

$$C_2(X,Y) = x_3 x_4 y_3 y_4 \vee x_2 x_4 y_2 y_4 \vee x_1 x_4 y_1 y_4.$$

This SOP is the same as the one above, except that $C_2(X,Y)$ replaces $C_1(X,Y)$ achieving a reduction of one PI. This is a counterexample to Proposition 3.2. We have shown, by a computer program, that (2) is an MSOP for f^2, and so $\tau(MSOP: f^2) = 24$. This example shows that decomposing a function into subfunctions on disjoint sets of variables, minimizing the two SOPs separately, followed by applying the law of distributivity does *not* always yield an MSOP.

In this example, only a small penalty is paid for using functional decomposition. This leads to the question of whether a large reduction is possible. That is, are there any AND bi-decomposable functions in which the application of the law of distributivity to the MSOPs of component functions produces an SOP with many more PIs than in the MSOP?

C. Incompletely Specified Functions

Definition 3.1 *Let f_S be an incompletely specified symmetric function on n-variables given as*

$$\begin{aligned} f_S(x_1,x_2,\ldots,x_n) &= 0 \text{ if all variables are } 0 \\ &= 1 \text{ if one or zero variables are } 0 \\ &= - \text{ (don't care) otherwise}, \end{aligned}$$

where $n > 2$.

Example 3.1 *An example of this function for $n = 3$ is shown in Fig. 3.2.*

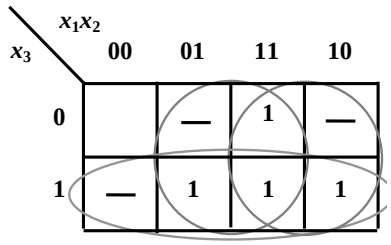

Fig. 3.2. Example of a three variable incompletely specified function.

Definition 3.2 *A completely specified function g is said to cover an incompletely specified function f if g is 0 and 1 for all assignments of values to variables for which f is 0 and 1, respectively.*

A cover for $f_S(x_1,x_2,\ldots,x_n)$ is

$$f_S(x_1,x_2,\ldots,x_n) = x_i \vee x_j, \qquad i \neq j,$$

and an SOP for f_S^{n-1} is

$$\underbrace{[x_{i_1} \vee x_{j_1}][y_{i_2} \vee y_{j_2}] \cdots [z_{i_{n-1}} \vee z_{j_{n-1}}]}_{n-1 \text{ sum (OR) terms}},$$

which has 2^{n-1} PIs, when expanded into an SOP using the law of distributivity. However, only n PIs are needed. That is,

$$\underbrace{x_1\, y_1 \ldots z_1 \;\vee\; x_2\, y_2 \ldots z_2 \;\vee\; \ldots \;\vee\; x_n\, y_n \ldots z_n}_{n \text{ product (AND) terms}} \qquad (3)$$

is also an SOP of f_S^{n-1}, where x_1, x_2, ... , and x_n are the variables of the first f, y_1, y_2, ... , y_n are the variables of the second f, ... , and z_1, z_2, ... , z_n are the variables of the $n-1$th f. f_S^{n-1} is 1 when one or none of $x_1, x_2, \ldots,$ and x_n are 0, one or none of $y_1, y_2, \ldots,$ and y_n are 0, $\cdots$, and one or none of $z_1, z_2, \ldots,$ and z_n are 0. At most $n-1$ variables are 0; the remaining are 1. For such an assignment of values, (3) is also 1, since there are n product terms on distinct variables, and not enough 0's for all product terms to be 0. Further, (3) is 0 when all variables in a group (i.e., $x_1x_2\ldots x_n$, $y_1y_2\ldots y_n$, ... , and $z_1z_2\ldots z_n$) are 0, as is f^{n-1}. It follows that (3) is an SOP for f_S^{n-1}. This proves

Theorem 3.1 *An ISOP for f_S^{n-1} formed by applying the law of distributivity to the product of MSOPs of f_S has 2^{n-1} PIs, while an MSOP of f_S^{n-1} has no more than n PIs.*

We form a completely specified function g_S from f_S by adding one more variable. In general,

Definition 3.3 *Let $g_S(x_1,x_2,\ldots,x_n,x_{n+1})$ be the $(n+1)$-variable function*

$$\begin{aligned} g_S(x_1,x_2,\ldots,x_n,x_{n+1}) = & S^n_{\{1,2,\ldots,n-2\}}(x_1,x_2,\ldots,x_n) \\ & \vee x_{n+1} S^n_{\{n-1,n\}}(x_1,x_2,\ldots,x_n), \end{aligned}$$

where $n > 2$.

Example 3.2 *The completely specified function shown in Fig. 3.1 is $g_S(x_1,x_2,\ldots,x_n,x_{n+1})$ for $n = 3$. Removing the essential PIs in that function yields an incompletely specified function of the form shown in Fig. 3.2.*

Each minterm in $g_S|_{x_{n+1}=0}$ corresponds to values of x_1, x_2, ... , and x_n for which f_S is a don't care. Further, a covering of $g_S|_{x_{n+1}=0} = S^n_{\{1,2,\ldots,n-2\}}$ also covers corresponding minterms in $g_S|_{x_{n+1}=1}$, that are don't care values in the underlying incompletely specified function. These PIs are essential because they are the only PIs that cover the $\binom{n}{2}$ minterms in $g_S|_{x_{n+1}=0}$ with exactly $n-2$ 1's and two 0's. There are only two non-essential PIs, e.g. x_1 and x_2. Specifically, x_1 covers all minterms in $S_{\{n-1,n\}}$ except $x_1x_2\ldots x_n = 011\ldots1$ and x_2 covers all minterms in $S_{\{n-1,n\}}$ except $x_1x_2\ldots x_n = 101\ldots1$. Thus, OR'd together, they cover all minterms in $S_{\{n-1,n\}}$. They also cover other minterms covered by essential PIs. Since $n > 2$, the number of PIs needed is $\binom{n}{n-2} = \binom{n}{2}$ [7]. Thus, g_S has $\binom{n}{2}$ essential and two non-essential PIs. From this, we can state,

Theorem 3.2 *An ISOP for g^{n-1} formed by applying the law of distributivity to the product of MSOPs of g has*

$$\tau(ISOP: g)^{n-1} = \left(\binom{n}{2} + 2\right)^{n-1},$$

PIs, while an MSOP of g^{n-1} has

$$\tau(SOP: g^{n-1}) = \left(\binom{n}{2} + 2\right)^{n-1} - 2^{n-1} + n.$$

PIs.

Theorem 3.2 shows that the reduction in PIs for the corresponding completely specified function is not as significant as in the case of the underlying incompletely specified function.

IV. Orthodox functions

As discussed in the previous section, minimization is easier when a function f has an AND bi-decomposition, and an MSOP is formed by applying the law of distributivity to the component functions. We characterize a subclass of functions with this property, orthodox functions.

A. Independent Sets of Minterms

Definition 4.1 *Given a function $f(X)$, let $M(f)$ be the set of true minterms for f. Then, $MI(f) \subseteq M(f)$ is an* **independent set of minterms** *of f iff no PI of f covers more than one minterm in $MI(f)$.*

Definition 4.2 *Given a function $f(X)$, $\eta(f)$ is the number of elements in a maximum independent set of minterms of f.*

The concept of a maximal independent set of minterms was proposed 30 years ago by Michalski and Kulpa [2]. It is used in ESPRESSO [1] [5] to obtain a lower bound on the number of products in an MSOP, which is useful in reducing computation time.

Example 4.1
Symmetric function $S^3_{\{1,2\}}(x_1,x_2,x_3)$ has two maximal independent sets of minterms $\{001,010,100\}$ and $\{110,101,011\}$. Thus, $\eta(S^4_{\{1,2\}}(x_1,x_2,x_3)) = 3$. *(End of Example)*

Definition 4.3 *Given a function $f(X)$, let $M(f)$ be the set of true minterms for f. Then, $MD(f) \subseteq M(f)$ is a set of* **distinguished minterms** *if exactly one PI of f covers each minterm in $MD(f)$.*

Example 4.2 *Symmetric function $S^3_{\{1,2\}}(x_1,x_2,x_3)$ has no distinguished minterms, because every minterm is covered by two PIs. However, $S^3_{\{1,2,3\}}(x_1,x_2,x_3) = x_1 \vee x_2 \vee x_3$ has three distinguished minterms,* 100, 010, *and* 001. *(End of Example)*

Note that a function covered only by essential PIs has a maximal independent set that corresponds to set of distinguished minterms.

B. Relationship Between MSOPs and Independent Sets of Minterms

Lemma 4.1

$$\tau(MSOP : f) \geq \eta(f).$$

Lemma 4.2 *Let $MI(g)$ be an independent set of minterms of $g(X)$ and $MI(h)$ be an independent set of minterms of $h(Y)$. Then, $MI(g) \times MI(h)$ is an independent set of minterms for $g(X)h(Y)$, where $MI(g) \times MI(h)$ is the AND of all minterms in $MI(g)$ with all minterms in $MI(h)$.*

Definition 4.4 *A function $f(X)$ is* **orthodox** *iff*

$$\tau(MSOP : f(X)) = \eta(f).$$

Otherwise, $f(X)$ is **non-orthodox**.

Example 4.3 *Symmetric function $f = S^3_{\{1,2\}}(x_1,x_2,x_3) = x_1\bar{x_2} \vee x_2\bar{x_3} \vee x_3\bar{x_1}$ is orthodox, since*

$$\tau(MSOP : f) = \eta(f) = 3.$$

(End of Example)

Theorem 4.1 *Let $f(X)$ and $g(Y)$ be orthodox. If X and Y are disjoint sets of variables, then $f(X)g(Y)$ is orthodox.*

It is possible to make a number of observations. First,

Theorem 4.2 *Let $f(X)$ and $g(X)$ be orthodox, where $X \cap Y = \emptyset$. Then,*

$$\tau(MSOP : f(X)g(Y)) = \tau(MSOP : f(X))\tau(MSOP : g(Y)).$$

Definition 4.5 *Function $f(X)$ is NP-equivalent to $g(X)$ if, given $g(X)$, a complementation and/or permutation of variables in X yields $f(X)$.*

Theorem 4.3 *If f and g are NP-equivalent, then f is orthodox iff g is orthodox.*

This follows from the observation that an MSOP of $f(X)$ can be formed from $g(X)$, which is NP-equivalent to $f(X)$ by a suitable complementation and permutation of variables in X. Therefore, if the MSOP for $g(X)$ has α independent minterms, so also does $f(X)$.

Definition 4.6 *Function $f(X)$ is NPN-equivalent to $g(X)$ if, given $g(X)$, a complementation and/or permutation of variables in X and/or complementation of the function yields $f(X)$.*

Theorem 4.3 cannot be extended to NPN-equivalent functions. For example, the function shown in Fig. 3.1, which is non-orthodox, is NPN-equivalent to its complement function, which is orthodox.

Lemma 4.3 *If an MSOP of $f(X)$ consists of essential PIs only, then f is orthodox.*

This follows from the observation that an essential PI covers a distinguished minterm that is covered by only that PI. All such minterms form an independent set. It is interesting that the converse does not hold. That is, a set of independent minterms is not necessarily a set of distinguished minterms. This is discussed in the next section.

C. Classes of Functions That Are Orthodox

Lemma 4.4 *Unate functions are orthodox.*

Lemma 4.5 *Parity functions are orthodox.*

These two lemmas follow from the observation that the MSOPs for a unate function and a parity function consist of essential PIs only.

Consider symmetric functions. In general, such functions may or may not have essential PIs. The parity function is symmetric, has essential PIs only, and is thus orthodox. However, the 3-variable symmetric function $S^3_{\{1,2\}}$ has no essential PIs, but it is orthodox. Thus, it has a non-empty set of independent minterms, and an empty set of distinguished minterms.

Theorem 4.4 *Symmetric functions are orthodox.*

D. Functions With Three or Fewer Variables

Theorem 4.5 *All switching functions of three or fewer variables are orthodox.*

It is impossible to extend the statement of Theorem 4.5 to four variables because the function in Fig. 3.1 is a four variable function that is non-orthodox.

V. Experimental Results

A. Non-Orthodox Functions With Four Variables

There are 65,536 functions of 4 variables. They are divided into 402 NP-equivalence classes. By a computer program, it has been verified that only four equivalence classes are non-orthodox. A representative from each class occurs as an assignment of values to the don't care values in Fig. 5.1.

x_3x_4 \ x_1x_2	00	01	11	10
00	—	1		1
01		1	1	1
11	1	1	1	1
10	1		—	

Fig. 5.1. The four-variable non-orthodox functions.

Each representative function is NP-equivalent to 63 other functions. So, there are $64 \times 4 = 256$ non-orthodox functions, representing 0.4% of the 65,536 4-variable functions. The program has verified that each of these functions has the property that $\tau(MSOP : f^2) < \tau(MSOP : f)^2$. Note that the running example of Fig. 3.1 corresponds to choosing the two dashed entries in Fig. 5.1 as 0's.

B. Functions With Five or More Variables

Extending this observation to functions on more variables is difficult because of the large number of functions. For example, the number of NP-equivalence classes of 5 variable is more than 1,200,000, so it is impractical to do exhaustive analysis.

For the functions with more than 4 variables, we generated functions by using a pseudo-random number generator, and did a computer simulation. For each n, we generated 100 functions with 2^{n-1} true minterms and determined which were non-orthodox. Table 5.1 show the results. It can be seen that when the number of variables is 10, the percentage of functions that are non-orthodox is 100%. Thus, it is interesting to compare this with functions on three or fewer variables, 0% of which are non-orthodox.

Fig. 5.2 shows a plot of the percentage of 4-, 5-, ... , and 13-variable functions that are non-orthodox verses the percentage of true minterms. When the percentage of true minterms is small, so also are the percentage of non-orthodox functions. In this case, minterms tend to be isolated and most PIs are essential. As the percentage of true minterms increases, so also does the percentage of functions that are non-orthodox, until most of the minterms are true, in which case the function is covered by a few large PIs. In the limit, when 100% of the minterms are true, there is a single PI that covers a single function, and 0% of the functions are non-orthodox. The data associated with 4-variable functions is exact, since we know the exact number of non-orthodox functions. For functions with more than 4 variables, the data is approximate. The program that computes $\tau(MSOP : f)$ and $\eta(f)$ applies a heuristic for finding an MSOP and a maximal independent set.

TABLE 5.1
Percentage of n-Variable Functions That Are Non-Orthodox

Number of Variables	Percentage that are Non-orthodox
5	1
6	4
7	13
8	34
9	81
10	100

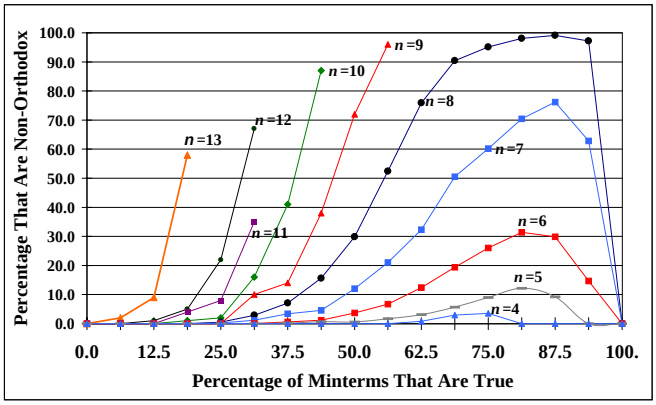

Fig. 5.2. Percentage of 4-, 5-, ... , and 13- variable functions that are non-orthodox.

C. Benchmark Functions

Most logic functions used in practical logic design are not random, but have special properties. Benchmark functions, used to compare SOP minimization algorithms, are thus appropriate subjects for experiments to determine the orthodox property. We have analyzed the MCNC91 benchmark functions with respect to this property. Table 5.2 shows the result. For multiple-output functions with up to 30 input variables, we minimized each output independently; orthodox functions are only defined for single-output functions. For functions with

TABLE 5.2
BENCHMARK FUNCTIONS THAT ARE ORTHODOX.

Circuit Name	No. In.	No. Out.	No. Or'dx	No.Non -Or'dx	No. Unate
5xp1	7	10	10	0	2
9sym	9	1	1	0	0
apex4	9	19	18	1	1
b12	15	9	9	0	2
cps	24	109	107	2	50
duke2	22	29	29	0	8
ex1010*	10	10	9	1	*
inc	7	9	9	0	0
misex1	8	7	7	0	0
misex2	25	18	18	0	12
misex3	14	14	13	1	0
misex3c	14	14	13	1	0
pdc*	16	40	40	0	*
rd53	5	3	3	0	1
rd73	7	3	3	0	1
rd84	8	4	4	0	1
sao2	10	4	4	0	0
spla	16	46	46	0	12
t481	16	1	1	0	0
vg2	25	8	8	0	0

*This function is incompletely specified.

more input variables, the program failed to finish because of memory overflow. Of the 20 functions analyzed, and of a total of 358 outputs, only 6 outputs or 1.7% are non-orthodox. This, we feel, is interesting, given our experiments on random functions, which show that, random functions with 10 or more variables are predominantly non-orthodox. It suggests to us that benchmark functions tend to be simpler than random functions.

VI. CONCLUDING REMARKS

From the results of this paper, we can state the following minimization algorithm.

Algorithm 6.1

1) If f has an OR bi-decomposition ($f = g_1(X_1) \vee g_2(X_2)$), then minimize SOPs for g_1 and g_2, independently. The OR of two MSOPs is an MSOP for f.

2) If f has an AND bi-decomposition ($f = g_1(X_1)g_2(X_2)$), determine if g_1 and g_2 are orthodox. If both are orthodox, minimize the SOPs for g_1 and g_2, independently. Use the law of distributivity to derive the MSOP for f.

3) Otherwise, use a conventional approach to minimize the SOP for f.

In this paper, we have shown that the AND of MSOPs of certain functions followed by the application of the law of distributivity does *not* always produce an MSOP for the product function. We show an incompletely specified $n(n-1)$-variable function where the an MSOP requires n PIs while an SOP derived from independent optimization followed by the application of the law of distributivity requires 2^{n-1} PIs.

In [6], many benchmark functions were analyzed to determine if they had functional decompositions. In this experiment, each output was decomposed separately in the case of multiple-output functions. The experimental results show that 3027 functions out of 4388 functions or 69% have bi-decompositions, most of which are AND or OR type bi-decompositions. This shows the importance of orthodox functions, suggesting that minimization of practical functions can be done by a divide-and-conquer algorithm. That is, a bi-decomposition allows one to identify two component functions (each simpler than the original function) and then to minimize those separately, and later combine them using the law of distributivity. We have proven that the MSOP will be obtained in the case of orthodox functions.

It is important to note that that the majority of switching functions do not have bi-decompositions and are not orthodox. The benefit of our proposed approached is demonstrated by the fact that a large number of switching functions used in practice (e.g. unate, symmetric, and benchmark functions) have these properties, and thus our techniques can be used to advantage.

VII. ACKNOWLEDGEMENTS

The authors acknowledge the research support of the Ministry of Education, Science, Sports and Culture of Japan.

REFERENCES

[1] R. K. Brayton, G. D. Hactel, C. T. McMullen, and A. Sangiovanni-Vincentelli, *Logic Minimization Algorithms for VLSI Synthesis*, Norwall, MA, Kluwer Academic Publishers, April 1984.

[2] R. S. Michalski and Z. Kulpa, "A system of programs for the synthesis of switching circuits using the method of disjoint stars," *Proceedings of IFIP Congress*, pp. 61-65, April 1971.

[3] A. Odlyzko, "On covering a product of sets with products of subsets," *Discrete Mathematics*, pp. 373-380, 1973.

[4] W. J. Paul, "Realizing Boolean functions on disjoint set of variables," *Theoretical Computer Science 2*, pp. 383-396, 1976.

[5] R. L. Rudell and A. Sangiovanni-Vincentelli, "Multiple-valued minimization for PLA optimization," *IEEE Trans. Computer-Aided Design*, vol. CAD-6, No. 5 pp. 727-749, September 1987.

[6] T. Sasao and M. Matsuura, "DECOMPOS: An integrated system for functional decomposition," *1998 International Workshop on Logic Synthesis*, Lake Tahoe, pp. 471-477, June 1998.

[7] B. Voight and I. Wegener, "A remark on minimal polynomials of Boolean functions," *CSL'88, 2nd Workshop on Computer Science Logic Proceedings*, pp. 372-383, 1989.