論 文
30～2～1

DA 表法에 依한 스위칭函數의 迅速 最小化

East Minimization of Switching Functions by DA-TABLE Method

黃 熙 隆*・朴 忠 圭**・金 民 煥***
(Hee-Yeung Hwang・Choong-Kyu Park・Min-Hwan Kim)

Abstract

This paper describes two methods which generate all the prime implicants (PI's) quickly by using the directions of adjacency table (DA-table) that gives the knowledge of adjacency relations among the given minterms. One is a graphical mehtod that enables us to generate PI's by hand, the other is a checking method that determines the existance of PI's quickly when it is programmed on a digital computer. And a fast minimization algorithm will be shown in this paper that can be implemented with reduced computational effort by selecting essential prime implicants (EPI's) first of all and using the concept of the integration of the PI identification and selection procedure.

The procedure, proposed in this paper, has all the advantages of Karnaugh mapping, so the procedure is simple and easy.

I. Introduction

The switching function minimization problem is very important in the design of switching circuits. Unfortunately there is no completely general criterion for the simplest expression of switching function because of the myriad possible forms of Boolean expressions. However it is possible to define a simplest form of the two level, or minimum delay time, circuit.

It becomes a difficult task to minimize a switching function, as the number of switching variables increases. But a wide variety of techniques [1]—[9] have been developed by many researchers because of its importance.

Most of them need several itearative procedures to generate the prime implicants of a Boolean expression. Especially to generate the higher-ordered PI from the lower-ordered implicants needs much effort because of comparing them one another.

Some advanced techniques were developed in [4] and [9]. Hwa[4] generated the PI's with reduced procedures but these procedures still contain complex comparisons of implicants in the multiple-PI case and treatment of binary numbers. In [9], the RAD-directed procedure allows multiple PI's covering the same minterm to be identified directly but it makes much effort to search for PI's by considering all the combinations of the required adjacency direction's (RAD's).

In our RAD-tree method by DA-table, only one-cubes are identified as RAD's and larger PI's are obtained easily by only tracing the given paths in DA-table.

And a new method (group checking method)

* 正會員：서울大 工大 電子計算機工學科 副敎授・工博
** 正會員：崇田大 工大 電氣工學科 敎授(當學會監事)
*** 正會員：서울大 工大 電子計算機工學科 大學院
接受日字：1980年 11月 14日

provides with a useful checking procedure that determines the existance of PI's quickly without checking all the given minterms repeatedly.

These two methods are suitable for identifying all the PI's that cover a selected minterm.

The minimization procedure can be devided into two parts: 1) the identification of PI's; and 2) the selection of PI's for a minimal cover. In the past, many works [1]—[6] treated the first part as a main part. Nowadays, a lot of attentions have been paid to the second part in[7]—[9]. Biswas [7] tried to select all the EPI's in each step of PI identification procedure and used an approximate method to select PI's for a minimal cover. In [8], the concept of integration of the PI identification and selection procedure was suggested. The above concept was extended in [9].

In our paper, this concept of integration is also used, but the EPI's are selected first of all, there by reducing the time required by the searching procedure remarkably.

II. DA-Table

We define that any minterm $m(I)$ isADJACENT to another $m(J)$, if the binary representation of them differs in only one bit.

〈Theorem〉

Let the decimal integers assigned to the minterms B and T be B_d and T_d respectively and difference $R=T_d-B_d$ be equal to 2^i $(i=0,1,\cdots,n-1)$. Iff the integer part of B_d/R even number, two minterms T_d and B_d satisfy the relation of adjacency eliminating i-th bit. Then $R(=2^i)$ is a RAD of B_d and $-R(=-2^i)$ is a RAD of T_d.

The proof was given in Hwang's paper[6].

〈Lemma〉

If a minterm $m(I)$ has a RAD R_i, where I is the decimal value of minterm $m(I)$, then there must exist a corresponding minterm $m(I+R_i)$ in the given switching function.

Proof. A RAD R_i of a minterm $m(I)$ means that $R_i=X-I$ (when $R_i>0$) or $-R_i=I-X$(when $R_i<0$) by theorem, where X is the decimal value of a minterm $m(X)$ that is given in the switching function. Therefore there exists a minterm $m(X)=m(I+R_i)$ in the given switching function.

(i) Constrution of DA-Table

A new method to construct the DA-table without the Simple Table defined in [6] is represented in following steps.

Step 1. Choose a lowest decimal-valued minterm $m(l)$ among the given minterms that are not chosen yet. If all the given minterms have been chosen, this process terminates. If not, make $i=0$ and go to step 2.

Step 2. If the square (2^i) of the chosen minterm in this table is empty, check if the integer part of $m(l)/2^i$ is even number and go to step 3. If not, go to step 5.

Step 3. If the integer part of $m(l)/2^i$ is even number, add 2^i to the chosen minterm $m(l)$ and go to step 4. If not, go to step 5.

Step 4. Compare the sum obtained in step 3 with only higher decimal-valued minterms. If there is a minterm that is equal to the sum, fill the related square(2^i) of the chosen minterm with the symbol 0 and the related square(2^i) of the correaponding minterm with the symbol 1. If not, go to step 5.

Step 5. Increase i by one. If $i>n-1$, where n is the number of variables, go to step 1. If not, skip to the next square(2^i) of the chosen minterm $m(l)$ and go to step 2.

(ii) Properties of DA-Table

The DA-table obtained from above steps has the following properties.

Property 1

The symbol 0 (symbol 1) of a minterm in DA-table means that this minterm has a larger(smaller) minterm as a couple˙

Property 2

Each couple of the symbols 0 and 1 means that two related minterms have the same valued bits except only one bit and the union of two minterms can be reduced to one after eliminating that bit.

Property 3

Since all the reducible relations are represented completely in DA-table, the larger PI's can be generated easily.

Property 4

The symbol 0's and 1's of a minterm in DA-table are equal to the original bits of that minterm when it is expressed in binary numbers.

III. Expansion of Minterms

When minimizing the switching functions, we must generate all the PI's that are related to a chosen minterm in order to select a PI that will be listed on the cover of the given switching function. This process is designated as the expansion of minterms.

Minterms are categorized into three subsets, as

1) a true form (*TF*) if $m(i)$ ia a true minterm;

2) a false form (*FF*) if $m(i)$ is a false minterm;

3) a redundancy (*XF*) if $m(i)$ is unspecified.

We will give two expansion procedures; One is the RAD-tree method for a manual method, the other is the group-checking method for an automated method.

(i) RAD-Tree Method

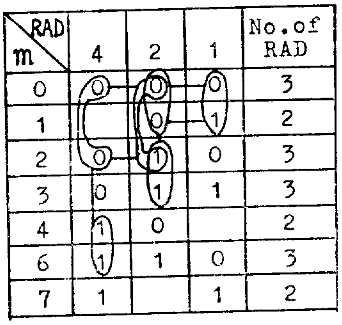
This method is a graphical method and the procedure is simillar to that of Karnaugh mapping, so PI's are identified easily. The concept of the RAD-tree defined in [6] is used in the following process.

A *TF* minterm is selected as a starting point and referred to as the origin *TF*, or TF_0. Then we can obtain a pair of *TF*'s (or a *TF-XF* pair) that generates an 1-ordered(or 1-cube) implicant containing TF_0 by finding a corresponding minterm to TF_0. The set of these two minterms is designated as the set of minterms of 1-ordered implicant, or SMI(1). See if there is the same RAD in both minterms. If so, we can obtain the four minterms that generate a 2-ordered implicant by finding two minterms related to SMI(1). The set of these four minterms is designated as SMI(2).

See if there is the same RAD in the four minterms in order to obtain the eight minterms that generate a 3-ordered implicant. If so, the set of these eight minterms is designated as SMI(3) and we repeat this process to obtain the higher-ordered PI covering TF_0. We can obtain SMI(4), SMI(5), and so on. If a SMI(i) is not identified, we store the SMI($i-1$) as a PI covering TF_0 and choose another path to obtain another PI covering TF_0 and repeat above precess. We should try to obtain the undominated PI's by considering all the paths related to TF_0 in the DA-table.

In an actual application, this procedure is very easy because all the paths to the large PI's are represented in the DA-table.

An example of a RAD-tree in DA-table is shown in Fig. 1. The origin *TF* is $m(0)$ that has three RAD's;$+1(=2^0)$, $+2(=2^1)$, and $+4(2^2)$. And we assume that there is not $m(5)$ in the given switching



RAD / m	4	2	1	No. of RAD
0	0	0	0	3
1		0	1	2
2	0	1	0	3
3	0	1	1	3
4	1	0		2
6	1	1	0	3
7	1		1	2

Fig. 1. RAD-tree in DA-table

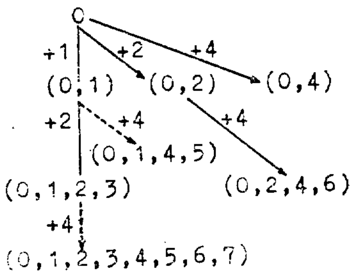


Fig. 2. RAD-tree in [9]

ching function. First of all, we obtain $m(1)$ related to $m(0)$. These two minterms have the same RAD +2, so we obtain the SMI(2) $\{m(0), m(1), m(2), m(3)\}$. But these four minterms don't have the same RAD, so the SMI(2) is a 2-ordered PI that covers $m(0)$. Next, another pair $\{m(0), m(2)\}$ can be obtained by adding the RAD +2 of $m(0)$. These two minterms have the same RAD +4, so the SMI(2) $\{m(0), m(2), m(4), m(6)\}$ can be obtained. But these four minterms don't have the same RAD, so the SMI(2) is a 2-ordered PI. Another pair $\{m(0), m(4)\}$ is dominated by the PI$\{m(0), m(2), m(4), m(6)\}$. Therefore the PI's covering $m(0)$ are as follows:

$PI_1=\{m(0), m(1), m(2), m(3)\}$

$PI_2=\{m(0), m(2), m(4), m(6)\}$

A RAD-tree in [9] that is changed a little is shown in Fig. 2., where real lines represent success paths and dotted lines represent failure paths. We do not have to identify (0,1,4,5) and (0,1,2,3,4,5,6,7) in this RAD-tree method because of their failure paths.

(ii) Group-Checking Method

It takes much time to identify the high-ordered PI's in a digital computer, so a new technique (group-checking method) is developed, which determines the existance of PI quickly by checking the RAD's of a few selected minterms.

First of all, it is tested whether the higher-ordered PI is identified successfully. If it is identified successfully, no attempt is made to identfy the dominated lower-ordered implicants. We try to identify the only undominated PI's by considering the RAD's that consist of the PI identified already. If the higher-ordered PI is not identified successfully, it is checked quickly by group-checking and then we skip to the other PI's that are not checked yet. This process enables us to save much efforts.

(a) Determination of 2-ordered PI

We assume that a *TF* minterm $m(I)$ has two RAD's (R_i, R_j). If a *TF* minterm $m(I+R_i+R_j)$ has two RAD's $(-R_i, -R_j)$, then these two RAD's generate 2-ordered PI. This property can be justified as follows.

By the lemma, there must be two minterms $\{m(I+R_i), m(I+R_j)\}$ related to $m(I)$ in the given switching function. These four minterms $\{m(I), m(I+R_i), m(I+R_j), m(I+R_i+R_j)\}$ consist of 2-ordered PI.

(b) Determination of 3-ordered PI

We assume that a *TF* minterm $m(I)$ has three RAD's (R_i, R_j, R_k). If a *TF* minterm $m(I+R_i+R_j+R_k)$ has three RAD's $(-R_i, -R_j, -R_k)$, then these three RAD's generate 3-ordered *PI*. This property can be justified as follows.

By the lemma, there must be three minterms $\{m(I+R_i), m(I+R_j), m(I+R_k)\}$ related to three RAD's (R_i, R_j, R_k) of $m(I)$ respectively and three minterms $\{m(I+R_i+R_j), m(I+R_k+R_i), m(I+R_j+R_k)\}$ related to three RAD's $(-R_k, -R_j, -R_i)$ of $m(I+R_i+R_j+R_k)$ respectively in the given switching function. These eight minterms $\{m(I), m(I+R_i), m(I+R_j), m(I+R_k), m(I+R_i+R_j), m(I+R_j+R_k), m(I+R_k+R_i), m(I+R_i+R_j+R_k)\}$ consist of 3-ordered PI.

(c) Determination of higher than three-ordered PI

We assume that a *TF* minterm $m(I)$ has n number of RAD's$(R_1, R_2, R_3, \ldots R_{n-1}, R_n)$. If there are $\sum_{i=1}^{n-3}\binom{n-3}{i}=2^{n-3}$ *TF* minterms $\{m(I), m(I+R_4), m(I+R_5), \ldots\ldots\ldots, m(I+R_n), m(I+R_4+R_5), m(I+R_4+R_6), \ldots, m(I+R_{n-2}+R_n), m(I+R_{n-1}+R_n), m(I+R_4+R_5+R_6), \ldots, m(I+R_4+R_5+\ldots+R_n)\}\triangleq Gm(I)$ each member of which has three RAD's (R_1, R_2, R_3) and 2^{n-3} *TF* minterms $Gm(I+R_1+R_2+R_3)$ each member of which has three RAD's $(-R_1, -R_2, -R_3)$ in the given switching function, then n-ordered *PI* can be identified successfully. This property can be justified as follows.

By the property of (b), each of 2^{n-3} couples $[m(I), m(I+R_1+R_2+R_3)]$, $[m(I+R_4), m(I+R_1+R_2+R_3+R_4)]$, ..., $[m(I+R_4+R_5+\ldots+R_n), m(I+R_1+R_2+R_3+\ldots+R_n)]$ has six minterms. Therefore 2^n $(=2^{n-3}\times8)$ minterms that consist of n-ordered PI exist in the given switching function and n-ordered PI is identified successfully.

Example 1

We assume that a *TF* minterm $m(5)$ has six RAD's $(-1, 2, -4, 8, 16, 32)$ and sixteen *TF* minterms

$m(5)$: $-1, 2, -4, 8, 16, 32$
$m(13=5+8)$: $-1, 2, -4, -8, X, X$
$m(21=5+16)$: $-1, 2, -4, X, -16, X$
$m(37=5+32)$: $-1, 2, -4, X, X, -32$
$m(29=5+8+16)$: $-1, 2, -4, -8, -16, X$
$m(53=5+16+32)$: $-1, 2, -4, X, -16, -32$
$m(45=5+32+8)$: $-1, 2, -4, -8, X, -32$
$m(61=5+8+16+32)$: $-1, 2, -4, -8, -16, -32$
$m(2=5+\Delta)$: $1, -2, 4, 8, 16, 32$
$m(10=5+\Delta+8)$: $1, -2, 4, -8, X, X$
$m(18=5+\Delta+16)$: $1, -2, 4, X, -16, X$
$m(34=5+\Delta+32)$: $1, -2, 4, X, X, -32$
$m(26=5+\Delta+8+16)$: $1, -2, 4, -8, -16, X$
$m(50=5+\Delta+16+32)$: $1, -2, 4, X, -16, -32$
$m(42=5+\Delta+32+8)$: $1, -2, 4, -8, X, -32$
$m(58=5+\Delta+8+16+32)$: $1, -2, 4, -8, -16, -32$

(X means the unspecified RAD and $\Delta=(-1)+2+(-4)=-3$)

exist in the given switching function. Then, by the property of (b), we know that a couple $[m(5), m(2)]$ has six minterms $\{m(4), m(7), m(1), m(3), m(0), m(6)\}$ and a couple$[m(13), m(10)]$ has six minterms $\{m(12), m(15), m(9), m(11), m(8), m(14)\}$, and so on. Therefore there exist $2^6(=64)$ minterms in the given switching function and six-ordered PI is identified successfully.

Example 2.

Let's generate all the PI's related to a minterm $m(0)$ in the following switching function.

$F(A, B, C)=\sum m(0, 1, 2, 3, 4, 6, 7)$

In Fig. 1. we know that $m(0)$ has three RAD's $(+1, +2, +4)$. First of all, the existance of 3-ordered PI is checked. Since $m(7=0+1+2+4)$ doesn't have a RAD (-2), the 3-ordered PI is not identified. Next, we check that any 2-ordered PI's are identified successfully. The *TF* minterm $m(3=0+1+2)$ has two RAD's $(-1, -2)$, so a 2-ordered PI is identified successfully. This PI is $PI_1=\{m(0), m(1), m(2), m(3)\}$. And another PI is obtained from the two RAD's $(-2, -4)$ of the *TF* minterm $m(6=0+2+4)$. That PI is $PI_2=\{m(0), m(2), m(4), m(6)\}$. As a next step, we check that any 1-ordered PI's are identified successfully. They are not identified because each RAD's are dominated by the RAD's generating 2-ordered PI's. Therefore, only two PI's are identified.

IV. DA-Search Algorithm (Fast Minimization)

This algorithm will generate only those PI's which are essential or necessary for a minimal sum of products of a given switching function by considering the properties of the don't care minterms.

At first, we try to get the EPI's. The properties of the EPI's are given as follows;

1) A minterm that has less than two RAD's generates an EPI because it has only one chance to cover itself.

2) If a *TF* minterm that has n number of RAD's generates a n-ordered PI, then this PI is an EPI because this *TF* minterm generates only it. (This observation was first noted by Biswas and Sureschander used it as a key part of his procedure).

The EPI's obtained from above are listed on the cover of switching function and all the *TF*'s contained within them are changed to *XF* status.

Select a *TF* minterm (TF_0) that has the fewest number of RAD's and expand it. It generates more than one PI's because all the minterms that generate only one PI was covered already as the EPI's. This set of PI's that cover TF_0 is designated as $\{PI\}_0$.

All the members of $\{PI\}_0$ are compared, pairwise, to see if any row dominance is present; if so, the dominated PI's are deleted from $\{PI\}_0$. Further, all but one of a group of PI's that are equal in *TF*-coverage and cost can be deleted. This may reduce $\{PI\}_0$ to a single pseudoessential PI(PEPI), which is then added to the cover of switching function, and all *TF*'s within it are changed to *XF* status.

If dominance cannot reduce $\{PI\}_0$ to a single member, then the search for PI's is continued, using another *TF* that has the fewest number of RAD's among the unexpanded *TF* minterms in the list, as a next origin *TF* for expansion. That *TF* can be designated as TF_1, and the PI's that cover it as the set $\{PI\}_1$. If, following the removal of dominated or equaled *PI's*, it reduces to a single PEPI, then that PI is added to the cover of switching function and all *TF*'s contained within it are changed to *XF* status. Any newly dominated PI's in the $\{PI\}_0$ are deleted. If this reduces $\{PI\}_0$ to a PEPI, then that PI is added to the cover of switching function, and its *TF*'s are changed to *XF* status.

If, however, either $\{PI\}_0$ or $\{PI\}_1$ (or both) still contain two or more PI's, then another *TF* that has the fewest number of RAD's among the unexpanded minterms in the list is chosen and another expansion is made. This can lead to $\{PI\}_2$, $\{PI\}_3$, and so on. Each time PEPI is identified, it is added to the cover of switching function and all *TF*'s within it are changed to *XF* status. The effects of this change upon all of the remaining sets of PI's are noted, and new PEPI's are selected, where identified. This iterative expansion process continues until all of the *TF*'s in the list have been covered or expanded.

If all the remaining set(s) of PI's have two or more members, then a cyclic condition must be resolved by appropriate method to get the best cover of a switching function.

V. Examples

To illustrate the DA-search algorithm, four examples are given as follows.

Example 1

$F(A,B,C,D,E,F,G,H,I)=\sum m(6,7,22,23,38,70,102,134,262)$

Its DA-table is shown in Fig.3. Minterms $m(134)$ and $m(262)$ have one RAD. Thus $PI_1=\{m(134), m(6)\}$ and $PI_2=\{m(262), m(6)\}$ are selected as the EPI's and added to the cover of *F*. Four *TF* minterms contained within them are changed to *XF* status. Minterms $m(7)$ and $m(38)$ that have two RAD's generate 2-ordered PI's. Thus $PI_3=\{m(7),m(6),m(22),m(23)\}$ and $PI_4=\{m(38),m(6),m(70),m(102)\}$ and selected as the EPI's and added to the cover of *F*. All the *TF*'s contained within them are changed to *XF* status, then all the given minterms become *XF*'s. Thus the algorithm terminates. The solution is $F=PI_1+PI_2+PI_3+PI_4$.

Example 2

$F(A,B,C,D,E)=\sum m(0,1,5,7,10,15,16,26,30)+d(14)$

This example is given to illustrate the treatment of don't care minterms. Its DA-table is shown in Fig. 4. Two EPI's $PI_1=\{m(16),m(0)\}$ and $PI_2=\{m(10),m(14),m(26),m(30)\}$ are selected and all the *TF*'s contained within them are changed to *XF* status. Next, $m(1)$ is chosen and expanded, because it is one of minterms that have the fewest number of RAD's among the remaining *TF* minterms. Two PI's can be identified; $PI_3=\{m(1),m(0)\}$ and $PI_4=\{m(1),m(5)\}$. PI_3 is dominated by PI_4, because $m(0)$ is a *XF*. Thus PI_4 is selected as a PEPI and two minterms $m(1)$ and $m(5)$ are changed to *XF* status. Next, a *TF* minterm $m(7)$ is chosen and expanded, then two PI's are identified; $PI_5=\{m(7),m(5)\}$ and $PI_6=\{m(7),m(15)\}$. PI_5 is dominated by PI_6, so PI_6 is selected and added to the cover of *F*. Two minterms $m(7)$ and $m(15)$ are changed to *XF* status. Then all the given minterms are changed to *XF* status, so we terminate the algorithm. In this example, no attempt is made to cover $m(14)$ but it is used to extend the PI_2. The solution is $F=PI_1+PI_2+PI_4+PI_6$.

Example 3

$F(A,B,C,D,E)=\sum m(1,4,5,7,8,9,11,13,14,15,18,19,20,21,23,24,25,26,27,28,29,30)$

Its DA-table is shown in Fig. 5. Four EPI's are identified; $PI_1=\{m(1),m(5),m(9),m(13)\}$, $PI_2=\{m(4),m(5),m(20),m(21)\}$, $PI_3=\{m(8),m(9),m(24),m(25)\}$, and $PI_4=\{m(18),m(19),m(26),m(27)\}$. All the *TF*'s contained within them are changed to *XF* status. The *TF* minterm $m(14)$ generates two PI's; $PI_5=\{m(14),m(30)\}$ and $PI_6=\{m(14),m(15)\}$. These two PI's are independant

$F(A,B,C,D,E,F,G,H,I)=\sum m(6,7,22,23,38,70,102,134,262)$

DA TABLE										No. of RAD's	EPI's			
m \ RAD	256	128	64	32	16	8	4	2	1		$\bar A\bar C\bar D\bar E\bar F GH\bar I$	$\bar B\bar C\bar D\bar E\bar F GH\bar I$	$\bar A\bar B\bar C\bar D\bar F GH$	$\bar A\bar B\bar E\bar F GH\bar I$
6	0	0	0	0	0				0	6	V	V	V	V
7					0				1	2			Ⓥ	
22					1				0	2			V	
23					1				1	2			V	
38			0	1						2				Ⓥ
70			1	0						2				V
102			1	1						2				V
134		1								1	Ⓥ			
262	1									1	V	Ⓥ		

$F(A,B,C,D,E,F,G,H,I)=\bar A\bar C\bar D\bar E\bar F GH\bar I+\bar B\bar C\bar D\bar E\bar F GH\bar I+\bar A\bar B\bar C\bar D\bar F GH+\bar A\bar B\bar E\bar F GH\bar I$

Fig. 3. An example of a function with 9 variables

$F(A,B,C,D,E)=\sum m(0,1,5,7,10,15,16,26,30\} +d(14)$

DA TABLE						No. of RAD's	EPI's		PEPI's			
m \ RAD	16	8	4	2	1		$\bar B\bar C\bar D\bar E$	$BD\bar E$	$\bar A\bar B CE$		$\bar A CDE$	
0	0				0	2	V		*			
1			0		1	2			*	Ⓥ		
5			1	0		2				V	*	
7		0		1		2					*	Ⓥ
10	0		0			2		Ⓥ				
-14	0		1		0	3		V				
15		1			1	2						V
16	1					1	Ⓥ					
26	1		0			2		V				
30	1		1			2		V				

$F(A,B,C,D,E)=\bar B\bar C\bar D\bar E+BD\bar E+\bar A\bar B CE+\bar A CDE$

Fig. 4. An example of Rhyne's with don't care minterm

$$F(A,B,C,D,E)=\sum m(1,4,5,7,8,9,11,13,14,15,18,19,20,21,23,24,25,26,27,28,29,30)$$
$$=\bar{A}\bar{D}E+\bar{B}C\bar{D}+B\bar{C}\bar{D}+A\bar{C}D+BCD\bar{E}+\bar{B}CE+\bar{A}BE+AB\bar{D}$$

DA TABLE						No. of RAD's	EPI's				PEPI's			
m \ RAD	16	8	4	2	1		$\bar{A}\bar{D}E$	$\bar{B}C\bar{D}$	$B\bar{C}\bar{D}$	$A\bar{C}D$	$BCD\bar{E}$	$\bar{B}CE$	$\bar{A}BE$	$AB\bar{D}$
1		0	0			2	Ⓥ							
4	0				0	2		Ⓥ						
5	0	0	1	0	1	5	V	V				* V		
7	0	0		1		3						* Ⓥ		
8	0				0	2			Ⓥ					
9	0	1	0	0	1	5	V		V				V *	
11	0		0	1		3							Ⓥ *	
13	0	1	1	0		4	V					*	V	
14	0				0	2					Ⓥ *			
15		1	1	1	1	4					*	*	V	
18		0			0	2				Ⓥ				
19		0	0		1	3				V				
20	1	0			0	3		V						*
21	1	0		0	1	4		V				V		*
23	1		1	1		3						V		
24	1		0	0	0	4			V					V *
25	1		0	0	1	4			V				*	V
26		1	0	1	0	4				V				*
27	1	1		1	1	4				V			*	
28		1	1	0	0	4								* Ⓥ *
29	1	1	1		1	4								* V
30	1		1	1		3					V			*

Fig. 5. An example of Biswas's

each other, so we store them. The *TF* minterm $m(7)$ generates two PI's that are independant each other; $PI_7=\{m(7),m(5),m(13),m(15)\}$ and $PI_8=\{m(7),m(5),m(21),m(23)\}$. The *TF* minterm $m(11)$ generates two PI's; $PI_9=\{m(11),m(9),m(13),m(15)\}$ and $PI_{10}=\{m(11),m(9),m(25),m(27)\}$. PI_{10} is dominated by PI_9, so PI_9 is selected as a PEPI and added to the cover of *F*. All the *TF*'s contained within PI_9 are changed to *XF* status, then PI_5 and PI_8 dominate PI_6 and PI_7 respectively. Thus PI_5 and PI_8 are added to the cover of *F* as the PEPI's and all the *TF*'s contained within them are changed to *XF* status. There remain only two *TF* minterms $m(28)$ and $m(29)$ in

$F(A,B,C,D)=\sum m(0,1,3,4,6,7,9,10,11,13,14,15)$

DA TABLE					No. of RAD's	EPI's		CYCLIC CHART					
m \ RAD	8	4	2	1		AC	AD		$\bar{A}C\bar{D}$		BC	$\bar{B}D$	
0		0		0	2			✓	✓				
1	0		0	→1	3			✓				✓	
3	0	0	→1		3							✓	✓
4		→1	0		2				✓	✓			
6	0		→1	0	3					✓	✓		
7	0	→1		→1	3						✓		✓
9	→1	0	0		3		✓					✓	
10		0		0	2	Ⓥ							
11	→1	0	→1	→1	4	✓	✓					✓	✓
13		→1	0		2		Ⓥ						
14	→1	→1		0	3	✓					✓		
15	→1	→1	→1	→1	4	✓	✓				✓		✓

$F(A,B,C,D)=AC+AD+\bar{A}C\bar{D}+\bar{B}D+BC$

Fig. 6. An example of Slagle's with cyclic condition

F. The *TF* minterm $m(28)$ is chosen and expanded, then three PI's are identified; $PI_{11}=\{m(28)$ $m(20), m(21), m(29)\}$, $PI_{12}=\{m(28), m(24), m(25), m(29)\}$, and $PI_{13}=\{m(28), m(24), m(26), m(30)\}$. PI_{13} is dominated by PI_{11} and PI_{12}, so PI_{13} is deleted. PI_{11} and PI_{12} have the same cost and they are equal in *TF*-coverage, so any one can be selected arbitrarily. This is an option case. The solution is $F=PI_1+PI_2+PI_3+PI_4+PI_5+PI_8+PI_{11}$ (or PI_{12}).

Example 4

$F(A,B,C,D)=\sum m\{0,1,3,4,6,7,9,10,11,13,14,15)$

This example is given to illustrate the cyclic condition. Its DA-table is shown in Fig. 6. Two EPI's are identified; $PI_1=\{m(10), m(11), m(14), m(15)\}$ and $PI_2=\{m(13), m(9), m(11), m(15)\}$. They are listed on the cover of *F* and all the *TF*'s contained within them are changed to *XF* status. Next, we generate the PI's related to the remaining *TF* minterms, but the PEPI's are not found in the each set of PI's. Thus we must resolve this cyclic condition by appropriate method, such as Petrick method. The solution of cyclic condition resolved by Petrick method is given as follows.

$PI_3=\{m(0), m(4)\}$

$PI_4=\{m(6), m(7), m(14), m(15)\}$

$PI_5=\{m(1), m(3), m(9), m(11)\}$

The solution of *F* is $F=PI_1+PI_2+PI_3+PI_4+PI_5$..

VI. Conclusions

In the DA-table, the concept of adjacency used in Karnaugh mapping is extended to large switching functions. The graphical relation of minterms in the DA-table is similiar to the relation of adjacent squares in the Karnaugh map. Therefore, in the DA-table, the large PI's can be identified easily by inspection and the largest PI that can cover the selected *TF* minterm can be identified quickly without having to identify all the smaller implicants contained within it. But the DA-table method has no problem in dealing with the switching function having more than six variables as against the Karnaugh mapping.

In the RAD-tree method, we need not try to obtain the PI's by complex comparisons because all the paths to the larger PI's are given in the DA-table. All the PI's that cover a selected *TF* minterm can be obtained easily by only tracing the given paths in the DA-table.

In the group-checking method, eight minterms are checked simultaneously and we try to obtain the larger PI's without checking all the smaller implicants contained within it. This method can be implemented with reduced time because there are only a few comparisons in the PI identification procedure.

Both PI identification methcd can be applied to other works [1]~[7] if a special algorithm that can keep us from generating the PI's identified already is designed.

In the DA-search algorithm, we try to get the EPI's before everything and change *TF*'s contained within them to *XF* status. The effects of this change on determining the PEPI's are notable, reducing the search procedure. The expansion of minterms is made from a *TF* minterm that has the fewest number of RAD's because its expansion is easy and it generates a few PI's.

The treatment of data in a manual method is easy because all the data are treated in decimal numbers.

The treatment of input sources in a digital computer is very simple because the number of them is proportional to that of true and redundant minterms.

This algorithm is suitable for obtaining the minimization concept (especially the concept of EPI's and row dominance) because the procedure is visible and simple.

In an actual case, this algorithm is well suited to minimization of most functions of more than six variables.

As a comparison, we have run a set of example minimization problems,using FortranIV language. The results of these comparisons are summarized in Table I. These are only loose comparisons. Table I shows that the DA method is faster. The memory capacity required for a computer program is smaller than the others, because it is proportional to the number of minterms given as the input sources.

Table I. Results of comparison between the DA method and another algorithm in [5]

Example	Number of Variables	Execution time(sec) Rhyne, IBM 360	Execution time(sec) DA method, IBM 360	Ratio of Rhyne/DA method
1	4	2. 0	1. 0	2. 0
2	4	2. 0	1. 0	2. 0
3	5	3. 0	2. 0	1. 5
4	5	2. 0	1. 0	2. 0
5	6	3. 0	2. 0	1. 5
6	6	4. 0	2. 0	2. 0
7	7	8. 0	4. 0	2. 0
8	8	5. 0	2. 0	2. 5
9	9	6. 0	2. 0	3. 0

References

1. F.J.Hill and G.R. Peterson; Introduction to Switching Theory & Logical Design, John Willy & Sons, New York, 1974, pp. 97~174.
2. G. Karnaugh; "The Map Method for Synthesis of Combinational Logic Circuit," AIEE Trans. Commun. Electron., pt. 1, Vol. 72, pp. 593~599, Nov. 1953.
3. J.R. Slagle, C.L. Chang, and R.C.T. Lee; "A New Algorithm for Generating Prime Implicants," IEEE Trans. Comput., Vol. C-19, pp. 304~310, Apr. 1970.
4. H.R. Hwa; "A Method for Generating Prime Implicants of a Boolean Expression," IEEE Trans. Comput., Vol. C-23, pp. 637~644, June 1974.
5. V.T.Rhyne; Fundamentals of Digital Systems Design, Englewood Cliffs, N.J., 1973, pp. 163~165.
6. H.Y. Hwang; "A New Approach to the Minimization of Switching Functions by the Simple Table Method," The Journal of the Korean Institute of Electrical Engineers, pp. 451~467, June 1979.
7. N.N. Biswas; "Minimization of Boolean Fluncttions," IEEE Trans. Comput., Vol. C-20, pp. 925~929, Aug. 1971.
8. Sureshchander; "Minimization of Switching Functions-A Fast Technique," IEEE Trans. Comput., Vol. C-24, pp. 753~756, July 1975.
9. V.T. Rhyne; R.S. Noe, M.H. McKinney, and U.W.Pooch, "A New Technique for the Fast Minimization of Switching Functions," IEEE Trans. Comput., Vol. C-26, pp. 757~764, Aug. 1977.