An Improved Method for Minimization Programming of Boolean Functions

Hidekatşu MIYAKOSHI

Abstract

This paper describes an improved method for minimization programming of Boolean functions. There are three main improved methods in the contents.

When minimization calculation is executed, many overlapped redundant prime implicants (RPIs) are derived. It is very useless, and long time, large memory, and heavy work are wasted for deriving them.

The method introduced here is one of improved ones which proposes a lemma.

This lemma shows a way not to derive duplicated RPIs, and makes it possible to save much time and work.

This paper describes an improved method for minimizing Boolean functions. The principle of this theory "A Literal Code Method For Minimizing Boolean Functions" is explained and discussed in detail in Reference [1].

In the actual calculation it is necessary for minimization programming to contrive many sorts of calculation techniques. As the numbers of literals and minterms of the given function become greater, the calculation time and used memory become longer and more. As a result, time and memory are too large to calculate even by using computer. Thus, how to make them less is the most important matter in the minimization programming.

There are many reasons why functions with multi literals take long time. Of course, it cannot be helped that it takes some time to a certain extent in the calculation of multi literal functions. But, it cannot be said that there is no room to save time. As one of the great causes to take long time there is a matter of overlapping.

The matter of overlapping is one feature in the solution course of Boolean functions. It is the same to a literal code method. The scale of overlapping becomes greater in proportion to literals and minterms of functions and it takes longer useless time. It is not rare that one fourth of execution time is the time of overlapped execution. As an extreme instance, There is a case that the overlapped calculation time is more than a half of execution

time.

The greatest execution time in the literal code process is the time to derive RPIs (Redundant Prime Implicants). Because, many overlapped RPIs appear in the time when deriving RPIs.

The reason why many overlapped RPIs appear is explained in the following :

It is assumed that the derivation of EPIs (Essential Prime Impricants) is not considered in the following.

When RPIs are derived from the minterms in the given function, one minterm cannot be an RPI by itself alone. Thus, an RPI consists of two minterms or more. Also, a minterm derives two RPIs or more. Suppose that one minterm derives one RPI, and this RPI is subsumed by three other minterms except this minterm. Then these three minterms derive the same RPIs respectively in their turns of derivation. Thus, it means that the same four RPIs derive.

This is the cause that overlapped RPIs derive by the nature of RPI.

It is quite unnecessary to calculate the same RPIs except one. It is very useless calculation time. If there is a way not to derive overlapped RPIs except one, it is possible to save much useless time and memory and to make time of calculation less as possible.

Thus, the following lemma has been studied for a long time and hereby is proposed for this purpose as an improved way. After describing and proving this lemma, the wonderful improved effects of time and calculation will be demonstrated in the following :

Lemma : When the minterms in the given function derive RPIS in turn, a minterm which has been already used to derive RPIs two times or more can be marked as been considered and laid aside.

Proof : It is natural that a new minterm which has never been used derives two or more new RPIs. A minterm which has been used only once derives an RPI which has been already derived by another minterm and two or more other new RPIs. Thus, these two sorts of minterms must not be omitted to derive RPIs.

Now, it is assumed that a minterm m_i has been already used to derive RPIs two times or more before its turn of derivation, and m_i derives a new RPI R_i which has not been derived yet.

If R_i contains a new minterm m_j which will appear in its subsequent turn or be used less than two times, it is unnecessary to derive R_i in m_i's turn since m_j will derive R_i in m_j's turn.

If all the minterms subsuming R_i have been used two times or more and they are all omitted to derive RPIs, then R_i will not be derived. It means the reduction of derivation

of RPIs, but it may not be minded since they will be dominated by other RPIs if R_i is prooved not to be a SEC EPI (Secondary EPI).

If R_i is a SEC EPI, it means that a SEC EPI has disappeared and some essential minterms subsuming it will not be included in the minimal sum. But, the minterms subsuming R_i have all dispersed in other RPIs since they have been used two times or more, so that these minterms are all included in the minimal sum. Therefore, R_i cannot be a SEC EPI. Thus, the minterms which have been already used two or more times can be omitted to derive RPIs.

Some examples of RPI derivation applying this lemma are demonstrated in the following :

Example 1 : F = (1,4,5,7,8,9,11,13,14,15,18,19,20,21,23,24,25,26,27,28,29,30)

F derives four EPIs, and fourteen minterms subsume the EPIs, so that eight unused minterms are stored in set G, that is, G = (7,11,14,15,23,28,29,30).

Table. A will be explained according to the derivation of RPIs by 8 minterms of G. The upper column shows minterms in G which derive RPIs, used times of each minterm, and derived RPIs.

Minterm 7 derives $\bar{V}XZ$ and $\bar{W}XZ$. They are subsumed by minterms of G, 15 and 23, so that the column of used times of 15 and 23 plus 1 respectively.

(In this case minterm 7 is not considered since it has already been used, and the minterms subsuming EPIs are also out of consideration.)

Table. A.

Minterms in G	used times	Derived RPIs
7	0	$\bar{V}XZ$ (5, 7, 13, 15), $\bar{W}XZ$ (5, 7, 21, 23)
11	0	$\bar{V}WZ$ (9, 11, 13, 15), $W\bar{X}Z$ (9, 11, 25, 27)
14	0	$\bar{V}WXY$ (14, 15), $WXY\bar{Z}$ (14, 30)
15	3	
23	1	~~$\bar{W}XZ$~~ $V\bar{W}XYZ$ (19, 23)
28	0	$VW\bar{Y}$ (24, 25, 28, 29), $VX\bar{Y}$ (20, 21, 28, 29), $VW\bar{Z}$ (24, 26, 28, 30)
29	2	
30	2	

Note. Overlapping RPI is lined through.
Underlined figures are minterms in G and others in D.
SEC EPIs are underlined in red.

Next, minterm 11 derives $\bar{V}WZ$ and $W\bar{X}Z$, so that the used times of minterm 15 becoms 2. Then minterm 14 will start to derive RPIs, so on.

Minterm 15 dose not derive RPIs since its used times are 3. Minterm 23 is used once, so that it derives duplicated $\bar{W}XZ$ and new $V\bar{W}YZ$.

As a result, minterm 15, 29, and 30 do not derive RPIs, minterms which derived RPIs reduced to 5 minterms from 8, and duplicated RPIs reduced to 1.

Example 2 : F = (0,1,2,4,5,6,7,8,9,11,13,14,15,16,18,19,20,21,23,24,25,26,27,28,29,30)

This is the famous example of McCluskey's paper. All the minterms of F are the un-

Table. B.

Minterms in G	used times	Derived RPIs
0	0	$\bar{X}\bar{Y}\bar{Z}$ (0, 8, 16, 24), $\bar{W}\bar{Y}\bar{Z}$ (0, 4, 16, 20), $\bar{W}\bar{X}\bar{Z}$ (0, 2, 16, 18), $\bar{V}\bar{W}\bar{Z}$ (0, 2, 4, 6), $\bar{V}\bar{X}\bar{Y}$ (0, 1, 8, 9), $\bar{V}\bar{W}\bar{Y}$ (0, 1, 4, 5)
1	2	
2	2	
4	3	
5	1	~~$\bar{V}\bar{W}\bar{Y}$~~ $\bar{V}\bar{Y}\bar{Z}$ (1, 5, 9, 13), $\bar{V}\bar{W}X$ (4, 5, 6, 7), $\bar{W}X\bar{Y}$ (4, 5, 20, 21), $\bar{V}XZ$ (5, 7, 13, 15), $\bar{W}XZ$ (5, 7, 21, 23), $X\bar{Y}Z$ (5, 13, 21, 29)
6	2	
7	3	
8	2	
9	2	
11	0	$\bar{V}WZ$ (9, 11, 13, 15), $W\bar{X}Z$ (9, 11, 25, 27)
13	4	
14	0	$WX\bar{Y}\bar{Z}$ (14, 30), $\bar{V}XY$ (6, 7, 14, 15)
15	3	
16	3	
18	1	~~$\bar{W}\bar{X}\bar{Z}$~~ $V\bar{X}\bar{Z}$ (16, 18, 24, 26), $V\bar{X}Y$ (18, 19, 26, 27)
19	1	~~$\bar{V}\bar{X}\bar{Y}$~~ $V\bar{W}YZ$ (19, 23)
20	2	
21	3	
23	2	
24	2	
25	1	~~$\bar{W}\bar{X}Z$~~ $W\bar{X}\bar{Y}$ (8, 9, 24, 25), $W\bar{Y}Z$ (9, 13, 25, 29), $VW\bar{X}$ (24, 25, 26, 27), $VW\bar{Y}$ (24, 25, 28, 29)
26	3	
27	3	
28	1	~~$\bar{V}\bar{W}\bar{Y}$~~ $VW\bar{Z}$ (24, 26, 28, 30), $VX\bar{Y}$ (20, 21, 28, 29), $V\bar{Y}\bar{Z}$ (16, 20, 24, 28)
29	4	
30	2	

used minterms since there is no EPI. Table. B. illustrates the derivation of RPIs. Only 8 minterms derive 26 RPIs instead of the derivation of RPIs by all the 26 minterms of F, and only 5 RPIs are duplicated.

This effect of saving time and work becomes greater and more remakable as the numbers of literals and minterms increase.

Table. C. demonstrates this remarkable effect to shorten calculation time.

(Personal Computer PC 9801, NEC and Large Computer 260H, HITACHI were used to complete this table, and used language was FORTRAN 77.)

Table. C. C. P. U. Times With 8 Literals By Personal Computer And Large Sized Computer

No	S %	KN	C. P. U. Time (PC 9801)	C. P. U. Time (HC 260H)	KM	C. P. U. Time (PC 9801)	C. P. U. Time (HC 260H)
1	7.8	20	4"	3"			
2	15.6	40	7"	3"			
3	19.5	50	8"	4"			
4	23.4	60	11"	5"			
5	28.5	73	15"	5"			
6	35.2	90	21"	5"			
7	59.4	152	44"	8"	104	21"	4"
8	77.3	198	1'02"	11"	58	9"	3"
9	90.2	231	1'17"	23"	25	5"	1"
10	94.5	242	46"	8"	14	3"	1"

KN=The Number of Minterms of the Given Function F.
KM=The Number orf Minterms of the Complementary Function F when S>50 S=KN / 256 *100%

S is the percentage of the number of the given minterms compared to all the minterms. As the calculation by the complementary function $\bar{F}$ when $S > 50$ is far faster than the calculation by the function F, its data by $\bar{F}$ are shown in the right half side. Here, KN + KM $= 2^8 = 256$.

When the number of minterms of the given function with 8 literals is not so many, the time differences of both tables are almost same, however, when the number of minterms becomes bigger, their time differences are so greater.

Because the derivation of RPIs taks the greater part of calculation time and to make time lag of this part shorter is very important in the whole process of minimization and it can be completed by this lemma.

Table. D. demonstrates the number of minterms to derive RPIs and used times of each minterm of example No. 10 in table. C. "G = 79" is the number of minterms and the value of each minterm is shown. "IG" is used times of each minterm.

Table. D.

G=79																								
	3	8	9	16	17	18	19	24	25	33	35	41	49	51	57	64	65	66	67	72	73	80	81	82
	83	88	89	97	99	113	115	121	128	129	130	131	134	135	136	137	142	143	144	145	146	147	151	152
	158	159	161	163	165	167	169	173	179	181	185	189	192	193	194	198	199	200	206	208	209	210	211	216
	217	225	227	229	233	241	249																	
IG=	0	0	3	0	1	2	4	6	3	0	7	4	3	5	3	0	3	3	3	4	2	7	4	
	5	4	7	3	5	4	4	4	1	0	3	3	4	2	3	6	2	4	4	5	2	2	2	1
	5	1	8	5	5	4	3	3	4	1	0	1	6	7	3	3	2	1	5	3	8	4	3	2
	8	2	4	1	4	1	2	4																

It means that the actual used minterms to derive RPIs are zero and one time used ones and they are 16 minterms in 79 minterms. Thus, it became possible to save time and work.

The above report is my work which I have studied for this one year.

If this lemma is certified, it will bring the most remakable effect in time and work for the minimizing of Boolean functions.

Acknowledgment

The author wishes to express his gratitude to Mr H. Allen Curtis of Williamsburg, Virginia, U. S. A., for his kind guidance and constant suggestions during the development and writing of this paper.

References :

1. H. Miyakoshi. A Literal Code Method for Minimizing Boolean Functions. B. C. S. The Computer Journal.

2. E. J. McCluskey, Jr. Minimization of Boolean Functions. B. S. T. J.

3. H.Allen Curtis. Adjacency Table Method of Deriving Minimal Sums. I. E. E. E. Trans. on Comp.

4. T. Rhyne et al. A New Technique for The Fast Minimization of Switching Functions. I. E. E. E. Trans. on Comp.