論文 85-22-2-5

PLA 設計用 高速 論理最小化 알고리즘

(Fast Logic Minimization Algorithm for Programmable-Logic-Array Design)

崔 相 浩*, 林 寅 七*

(Sang Ho Choi and In Chil Lim)

要 約

本 論文은 PLA 面積最適化를 위한 論理最小化에 對한 새로운 알고리즘을 提案한다. 計算機 處理時間이 變數의 數에 直接的으로 依存하는 終來方式과는 달리, 本 方式은 論理函數에서 base minterm과 consensus가 되지않는 민텀들의 集合을 그 函數로부터 除去하여 줌으로써 計算機 處理時間은 base minterm의 consensus의 次數에 依存하여 變數의 數가 增加할수록 終來方式에 比하여 計算時間値의 差가 커진다. 實際 計算機 實驗을 通하여 終來方式과의 計算機 遂行時間 比較値의 例를 提示한다.

Abstract

This paper proposes an algorithm to simplify Boolean functions into near minimal sum-of-products for Programmable Logic Arrays. In contrast to the conventional procedures, where the execution time depends on the number of variables, the execution time by this procedure depends on the degree of consensus of base minterms. Thus as the number of variables is increased, the difference of CPU time becomes larger using this new procedure than using other procedures and consequently the executable range of input function increasing. The algorithm has been implemented on CYBER 170-740 and it's results were compared with those using Arevalo's algorithm.

I. 序 論

PLA는 一般的으로 AND-OR 行列의 配列로 VLSI 回路 및 그 시스템 構成에 있어서 重要한 빌딩블록이다.

2段 PLA의 경우 面積의 最小化는 論理式을 最小化하는 것으로 될 수 있다. 初期에 提案된 論理最小化 方法으로는 Quine과 McCluskey에 依한 表式法[8,9], Roth의 位相法[7]과 Tison의 컨센서스(consensus)法[3] 등이 있는데, 이들 古典的 接近方法은 主項生成 및 主項選擇의 過程을 遂行함으로써 計算過程이 複雜하다. 이를 改善한 方法으로는 Hong의 MINI[4]를 비롯하여 몇가지 방법이[2,5,6] 있지만 많은 積項을 生成하는 函數에 適用하기 어렵다는 缺點을 갖고있다.

最近에 Arevalo는 計算時間이 적게들고 PLA에 應用하기 쉬운 論理最小化 方法[1]을 提案하였다. 이 方法은 函數 f의 全集合에서 end minterm을 選擇하여 主項을 立證하는 方法이다. 그 結果 最終解는 必須解가 되고 많은 積項을 生成하는 函數에 適用하기 쉽기 때문에 PLA의 設計에 主로 利用된다.

本 論文에서는 論理函數의 base minterm과 consensus가 되지않는 集合을 定義하고, 이 集合을 그 函數로부터 除去하여 줌으로써 主項發生에 必要한 end minterm의 選擇範圍를 줄여준다. 따라서 計算機 處理

*正會員, 漢陽大學校 工科大學 電子工學科

(Dept. Electron. Eng., Han Yang Univ.)

接受日字：1984年 9月 11日

時間이 變數의 數와 直接的으로 依存하는 終來方式과는 달리, consensus의 次數에 計算機 處理時間이 依存하여, 變數의 數가 많은 函數일수록 終來方式에 比하여 計算時間이 短縮되고 一定한 容量의 計算機에서 處理할 수 있는 論理函數의 處理可能範圍가 擴張된다.

Ⅱ. 論理最小化의 基本方法

1. 基本定義 및 用語說明

PLA設計를 위한 論理函數의 積項數를 最小化하는데 있어서, 論理函數 f의 主項을 base minterm과 end minterm을 利用하여 求하는 方式을 생각한다. 이 base minterm 및 end minterm을 얻기위하여 各 민텀에 대하여 다음과 같은 4가지 라벨(r, m, p, d)값으로 이루어진 4 tuple이 주어진다.

a) r:주어진 민텀에 대한 consensus의 次數

b) m: 이미 만들어진 主項에 cover된 민텀 중 주어진 민텀과 隣接된 민텀의 數

c) p : 주어진 민텀이 cover되지 않으면 0, cover 되면 1이된다.

base minterm은 函數 f의 모든 민텀 중에서 主項에 cover되지 않는 민텀가운데 consensus의 次數가 가장 적은 민텀들을 選擇한 後, 그 민텀들 중에서 cover된 隣接민텀의 數가 가장 많은 민텀으로 定한다.

d) d : 위의 方法으로 구한 base minterm으로부터 주어진 민텀까지의 거리

[定義 1] 函數 f의 민텀을 on minterm, $f'(=1-f)$ 의 민텀을 off minterm이라 한다.

函數 f'의 민텀 중 base minterm으로부터 거리i $(1\leq i\leq n)$의 off minterm을 a'_{ij} $(1\leq j\leq K_i)$로 表示할 때 거리i의 off minterm 集合 (A'_i)은

$A'_i=\{a'_{i1}, a'_{i2}, \ldots a'_{iK_i}\}$ (K_i는 $d=_i$의 off minterm 의 갯수)

이 되고, 변수의 數를 n개라 假定할 경우 off minterm의 總集合 (A'_t)은

$$A'_t(=f')=\bigcup_{i=1}^{n} A'_i$$

가 된다.

[定義 2] base minterm으로부터 임의거리i $(1\leq i\leq n)$의 off minterm (a'_{ij})과 거리n의 민텀 (a_n)이 結合하여 만들어진 cube는 base minterm과 전혀 consensus가 없는 데, 이러한 cube를 no relation cube(nrc)라 定義한다.

[定義 3] 거리i까지 모든 off minterm 集合에 대하여 만들 수 있는 nrc들의 集合을 無關集合(no relation set)이라 하고, 이를 NRS_i로 表示한다. 또한 거리i의 off minterm집합에 대하여 만들 수 있는 nrc들의 集合을 nrs_i라 한다.

NRS_i와 NRS_t(總無關集合)는 各各 다음과 같은 式으로 표시 할 수 있다.

$$NRS_i=\bigcup_{i=1}^{i}(nrs_i)=\bigcup_{i=1}^{i}\left(\bigcup_{j=1}^{Ki} nrc_{ij}\right)$$

$$NRS_t=\bigcup_{i=1}^{i}(nrs_i)=\bigcup_{i=1}^{n}\left(\bigcup_{j=1}^{K_i} nrc_{ij}\right)$$

主項은 base minterm과 end minterm을 結合하여 만든다. end minterm은 base minterm에 consensus가 전혀 없는 集合을 除去하고 남은 集合 $f_B(=f-NRS)$ 중에서 base minterm과 結合하여 最大 cube를 만들 수 있는 민텀으로 選擇한다. 〈알고리즘 1의 스텝 6, 7 參照〉 하나의 base minterm에 대하여 NRS_t를 구할 수 있을 경우, end minterm은 자동적으로 f_B중 base minterm으로부터 가장 멀리 있는 민텀이 된다. 따라서 end minterm을 選擇하여 主項인지 아닌지 判別하는 過程이 必要없다. 그러나 變數의 數가 增加하고 函數 f의 off minterm의 갯수가 늘어남에 따라 NRS_t를 求하는 것은 사실 상 不可能하게 된다. 이러한 問題를 解決하기 위하여 off minterm을 探索하는 거리의 限界值$(l=d_{limit})$를 定하여 주어, 그 限界值 以內의 無關集合(NRS_l)만을 구할 수 있도록 하였다.

NRS內에는 다른 nrc에 依하여 cover되는 nrc가 存在하게 된다. 이들을 구하는 不必要한 過程을 除去하여야 한다.

[定義 4] NRS集合 内에 어떤 다른 nrc에 의하여도 cover되지 않는 nrc를 essential nrc라 하고, essential nrc를 만드는 off minterm을 essential off minterm, 이들의 集合을 essential off minterm set(EOS)이라한다.

다음과 같은 方法을 利用하여 이 essential nrc 를 效率的으로 選擇할 수 있다.

1) $i=1$일때(nrs_1) : 거리 1의 off minterm은 모두 essential off minterm이 되므로 EOS와 그 集合으로 이루어진 essential nrc를 쉽게 구할 수 있다.

$$f_{B1}=f-nrs_1$$
$$=f-\bigcup_{j=1}^{K_1} nrc_{1j} \langle \text{그림1-a} \rangle$$

2) $2\leq i\leq l$ $(nrs_2 \sim nrs_l)$: Rhyne이 提案한 RAD(Required Adjacency Direction) tree[2]를 利用하여 EOS를 求하고, 다음과 같은 演算을 통하여 f_B를 얻게 된다. 〈附録 1 參照〉

$$f_B=f_{B1}-\bigcup_{i=2}^{l} nrs_i$$
$$=f_{B1}-\bigcup_{i=2}^{l}\left(\bigcup_{j=1}^{K_i} nrc_{ij}\right) \langle \text{그림1-b. c} \rangle$$

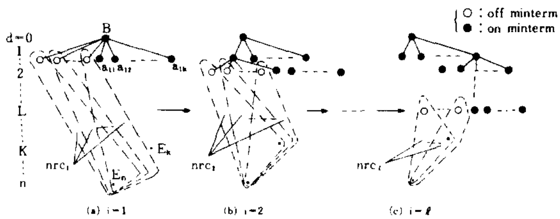


그림 1. Essential nrc 選擇方法
Fig. 1. The method of essential nrc selection.

2. 論理最小化의 基本方法

論理函數 f중 base minterm(B)을 選擇한 後, 定義 3에서 言及한 無關集合(NRS)을 구하여 그 集合을 函數 f에서 除去한다. 除去하고 남은 集合 $f_B(=f-NRS)$ 중에서 B에 대하여 條件[알고리즘 1의 스텝6, 7 參照]에 알맞은 end minterm(E)을 選擇한다. 이 B와 E로서 主項을 生成하고, 生成된 主項에 속한 모든 민텀이 함수 f_B에 包含되는 지를 點檢한다. 그 主項의 모든 민텀이 함수 f_B에 包含되지 않을 때는 條件에 알맞은 다른 E를 選擇하여 函數 f_B에 包含되는 主項을 生成할 때까지 되풀이 한다. 이와 같은 主項生成 過程을 函數 f의 全集合이 cover될 때까지 反復하여 f의 全集合이 cover될 때 알고리즘의 過程이 끝나게 된다. 그림 2는 이 過程을 順序圖로 나타내었다.

Ⅲ. 單一出力 函數의 簡單化

1. 主項의 選擇

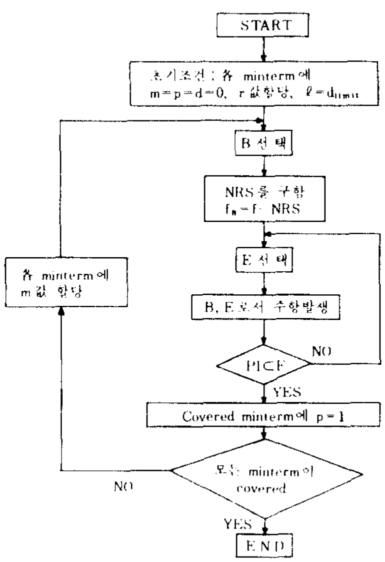


그림 2. 논리 최소화 알고리즘의 순서도
Fig. 2. Flow chart of logic minimization algorithm.

아래의 알고리즘에서 라벨의 演算式 $X_y \leq X(X_y \geq X)$란 考慮中인 민텀의 集合에서 라벨 X의 最小(最大)값을 갖는 민텀을 y민텀으로 選擇한다는 것을 意味한다.

[알고리즘 1]

스텝 1 : (初期値 設定) 各 민텀에 $m=p=d=0$, $l=d_{limit}$, r의 값을 割當한다.

스텝 2 : 다음 條件을 만족하는 base minterm(B)을 選擇한다.

a) $p_B=0$

b) $r_B \leq r(a)$를 만족하는 것 중에서)

c) $m_B \geq m(a)$, b)를 만족하는 것 중에서)

스텝 3 : 函數 f의 各 민텀에 대하여 B로부터의 거리를 割當한다.

스텝3′(選擇的) : $r<2$인 B는 스텝 5로 간다.(參考文獻[2]를 參照)

스텝 4 : $i=1$에서부터 $i=l$까지의 essential off minterm을 얻어 NRS 및 f_B를 구한다.

$$NRS=\bigcup_{i=1}^{l} nrs_i, \quad f_B=f-NRS$$

스텝 5 : $R=r_B$

스텝 6 : f_B중에서 다음 條件을 만족하는 end minterm(E)을 選擇한다.

a) $d_E=R$

b) $p_E=0$

c) $r_E \leq r(a)$, b)를 만족하는 것 중)

d) $m_E \leq m(a)$, b), c)를 만족하는 것 중)

인 민텀이 存在하지 않으면 스텝 7에, 그렇지 않으면 스텝 8로 간다.

스텝 7 : f_B중에서 다음 條件을 만족하는 E를 選擇한다.

a) $d_E=R$

b) $p_E=1$

c) $r_E \geq r(a)$, b)를 만족하는 것 중)

d) $m_E \leq m(a)$, b), c)를 만족하는 것 중)

인 민텀이 存在하지 않으면, $R=R-1$로 놓고 스텝 6으로, 그렇지 않으면 스텝 8로 간다.

스텝 8 : 알고리즘 2로 간다.

스텝 9 : 主項에 의하여 cover되는 민텀들을 $p=1$로 한다. 函數 f의 모든 민텀이 $p=1$이면, 알고리즘이 끝난다. 그렇지 않으면, 스텝10으로 간다.

스텝10 : 各 민텀에 m값을 割當한다.

2. 主項의 立證

[알고리즘 2](스텝 8)

스텝11 : $B=(b_1, b_2, \cdots, b_n)$, $E=(e_1, e_2 \cdots e_n)$는 各各 base minterm과 end minterm이다.

스텝12 : B와 E의 서로 다른 位値를 發見한다. 例를 들어 j=1, 2, …, i≤n인 位値에서 $b_j \oplus e_j = 1$이라 假定하자.

스텝13 : $I = (\top\ \top\ \top \cdots \top,\ b_{i+1}, \cdots, b_n)$의 形을 갖는 2^i個의 민텀이 f_B에 存在한다면, 다음과 같은 主項을 生成한 後, 스텝 9 로 간다. $(i+1 \le j \le n)$

$$X^*_{i+1} X^*_{i+2} \cdots X^*_n \begin{cases} X^*_j = X_j, & \text{if } b_j = 1 \\ X^*_j = X'_j, & \text{if } b_j = 0 \end{cases}$$

그렇지 않으면 스텝 6 으로 간다.

例題) 函數 $f(W, X, Y, Z) = \sum(0, 2, 3, 4, 5, 7, 8, 9, 10, 11, 12, 13, 15)$를 簡單히 하라. 단, $l=1$이라 假定한다. 여기서는 간략히 하기 위하여 알고리즘2의 스텝들을 省略한다.

스텝 1 : 初期 4-tuple을 發生한다. 그림 3 을 參照 (스텝2, 3, 4도 함께) 하라.

스텝 2 : B는 0

스텝 3 : 各 minterm에 base부터 거리를 割當한다.

스텝 4 : a'_{11} (essential off minterm) 1 이되어 NRS $= Z$, $f_B = f - NRS = \sum(0, 2, 4, 8, 10, 12)$

스텝 5 : R= 3

스텝6~7 : d= 3 의 end minterm이 存在하지 않으므로 R=3−1로 놓고, 다시 스텝 6 으로 간다.

스텝 6 : E는 10

스텝 8 : PI(주항) $= X'Z'$

스텝 9 : minterm 0, 2, 8, 10에 대하여 p=1

스텝10 : 모든 minterm에 m값을 決定하여 넣는다. (그림 4 參照)

스텝 2 : B는 3

스텝 3 : (그림 4 參照)

스텝 4 : $a'_{11} = 1$이 되어 NRS=Y', $f_B = \sum(2, 3, 7, 10,$ 11, 15)

(r. m. p. d)

YZ \ WX	00	01	11	10
00	0 3.0.0.0	4 3.0.0.1	12 3.0.0.2	8 4.0.0.1
01 (NRS)	/	5 3.0.0.2	13 4.0.0.3	9 3.0.0.2
11 (NRS)	3 3.0.0.2	7 3.0.0.3	15 3.0.0.4	11 4.0.0.3
10	2 3.0.0.1	6	14	10 3.0.0.2

그림 3. 예제에 대한 map
Fig. 3. The map of example.

YZ \ WX	00	01	11	10
00 (NRS)	0 / 3.2.1.2	4 3.1.0.3	12 3.1.0.4	8 4.2.1.3
01 (NRS)	/	5 3.0.0.2	13 4.0.0.3	9 3.1.0.2
11	3 3.1.0.0	7 3.0.0.1	15 3.0.0.2	11 4.1.0.1
10	2 3.2.1.1	6	14	10 3.2.1.2

그림 4. 예제에 대한 map
Fig. 4. The map of example.

스텝 5 : R= 3

스텝6~7 : d=3의 end minterm이 존재하지 않으므로 R=3−1로 놓고, 스텝 6 으로 간다.

스텝 6 : E는 15

스텝 8 : PI=YZ

이러한 過程을 되풀이하여 다음 主項은 PI=WY' (B=9, NRS=W, E=12)이고, 그 다음 主項은 PI= XY' (B=5, NRS=X', E=12)이다.

스텝 9 : 모든 minterm이 p=1이 되어 알고리즘의 遂行이 끝난다. 最終解는 $f(W, X, Y, Z) = X'Z' + YZ + WY' + XY'$가 된다.

3. Don't Care Minterm을 갖는 函數

Don't care민텀은 알고리즘을 遂行하기 前에 만들어진 假想主項에 依하여 cover된 민텀으로 생각할 수 있다. 따라서 다음과 같은 修正을 알고리즘 1 에 可한다.

1) 다음 스텝을 스텝 1 뒤에 넣는다.

스텝1' : 주어진 函數의 모든 Don't care민텀에 대하여 p=1로 놓은 後 스텝10으로 가라.

2) 나머지는 알고리즘1, 2와 同一

Ⅳ. 多出力 函數의 簡單化

M出力 函數 簡單化 問題에서 모든 민텀이 완전히 커버(totally covered)될 때 알고리즘이 끝난다.

[定義 5] 函數 $f_1, f_2, \cdots, f_M$의 곱함수 $f_1 \cdot f_2 \cdots f_M$에 대한 레벨 L은 包含되는 函數의 個數로 定義한다.

[定義 6] 레벨 L의 곱함수의 全集合 가운데서 exclusive minterm은 그 레벨의 모든 可能한 곱함수 중 단지 하나의 함수에만 存在하는 민텀이다.

[알고리즘 3]

1) L＝1로 놓는다.

2) 레벨 L의 곱함수와 exclusive minterm을 얻는다.

3) 레벨 L의 各 곱함수에 알고리즘 1, 2를 適用한다. 이때 다음과 같은 알고리즘1, 2의 修正이 必要하다.

알고리즘 1의 스텝 2 : 완전히 커버되지 않은 exclusive minterm을 B로 選擇한다.

알고리즘 2의 스텝13 : 곱함수에 의하여 얻어진 主項이 函數 f_i에서 B와 E 둘 모두 p＝1이 되지 않을 때 f_i의 最終 sum of products에 包含될 수 있다. 여기서 레벨 1의 곱함수 各各에 대하여 알고리즘 1을 適用하기 前에 r＝m＝0으로 놓는다.

4) 레벨 L의 모든 곱함수에 대하여 스텝 3을 反復한다. 만일 모든 레벨 L의 곱함수가 考慮되었다면 L＝L+1로 놓고 스텝 2로 간다.

여기서 Don't care민텀을 갖는 多出力 函數의 簡單化는 省略한다.

V. 結果 및 檢討

表 1에서는 單一出力에서 d_{limit}가 1일 때 Arevalo 方法과 提案된 方法을 積項數(PT's)와 컴퓨터(CYBER 170－740, 주기억용량 : 92KB) 遂行時間(T_i(s))으로서 간단히 比較하여 보았다. n變數函數일 때 最大 민텀수(2^n)의 70～100%를 갖는 函數가 2^{2^n}個 函數集合중에서 가장 많은 主項을 갖기 때문에 主로 이들을 例로 選擇하였다.[10]

比較한 結果, 그림 5에 보여준 바와 같이 컴퓨터 遂行時間이 變數의 數에 依存하는 Arevalo方法과는 달리, base minterm의 consensus의 次數에 依存함으로써 變數의 數가 增加할수록 컴퓨터 遂行時間의 差異가 더욱더 커짐을 보여준다. ⟨附錄 2 參照⟩

表 1. 單一出力函數에서 Arevalo의 프로그램과 比較

Table 1. Time Comparison to Arevalo's program in single output function (Computer : CYBER170－740, Main memory capacity : 92KB).

Number	#MIN	# PI's	Arevalo's PT's/Ti(s)	提案된 方法의 PT's/Ti(s)	Comments
			a) 5入力變數		
1	31	5	5 /0.156	5 / 0.136	0을 제외한 모든 minterm
2	30	20	5 /0.148	5 / 0.126	0, 31을 제외한 모든 minterm
3	29		6 /0.150	6 / 0.139	
4	28	20	6 /0.151	6 / 0.123	
5	27		8 /0.155	8 / 0.120	
6	25		7 /0.126	7 /0 .111	
			b) 6入力變數		
1	60		6 /0.629	6 / 0.508	
2	59	20	9 /0.747	9 / 0.568	
3	57		12/0.962	12/ 0.590	
4	56		12/0.923	12/ 0.577	
5	55		13/0.863	13/ 0.552	
6	52		11/0.771	11/ 0.499	
			c) 7入力變數		
1	124		12/3.795	12/ 2.549	
2	116		18/5.291	18/ 2.648	
3	106		20/4.692	20/ 2.168	
4	103		22/4.826	22/ 2.131	
5	99		24/5.310	24/ 1.941	
6	92		27/4.747	27/ 1.808	
			d) 8入力變數		
1	248		22/20.361	22/12.803	
2	237		29/27.844	29/12.442	
3	227		32/31.619	32/10.628	
4	219		36/32.435	36/10.050	
5	208		26/20.953	26/ 7.632	
6	195		39/31.560	39/ 8.480	
			e) 9入力變數		
1	447		43/167.453	43/40.940	
2	429		45/159.635	45/39.921	
3	392		48/127.400	48/33.867	

Don't Care例

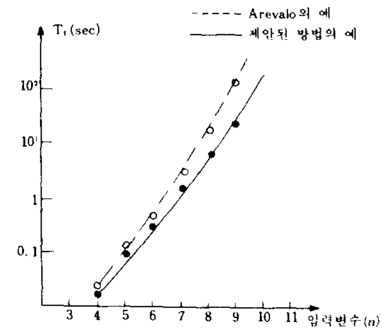
Number	#MIN	#Dont'Care	Areralo's PT's/Ti (s)	提案된方法의 PT's/Ti (s)	Comments
1	9	5	5/0.080	5/0.081	變數는 4個
2	8	6	4/0.009	4/0.009	變數는 4個

*提案된 方法의 거리의 限界値 d_{limit}는 1이다

VI. 結　　論

本 論文에서는 PLA 面積最適化를 위한 새로운 論理最小化 方法을 提案하였다. 이 方法은 base minterm과 end minterm으로 主項을 生成하고, 生成된 主項에 根據하여 다음 主項을 選擇하였기 때문에 最終解는 必須解(irredundant solution)가 된다.

CYBER 170－740을 使用하여 여기서 提案된 알고리즘을 實現하여 終來方式과 比較, 檢討한 結果 CPU time이 減少됨을 確認하였으며, 積項의 數와 變數의 數가 增加할수록 그 差異가 커짐을 例를 들어 보였다. 즉 Arevalo方法과의 差異는 5變數에서 0.01～0.03초, 6變數에서 0.10～0.35초, 7變數에서 1～3초 그리고 8變數에서 8～23초가 됨을 보여주었다. 또한 base minterm의 consensus의 次數에 計算機 處理時間이 依存하므로 本 알고리즘은 主項數가 많은 論理函數를 處理하기 有用하다. 특히 入力函數의 特性(민텀數 및 變數의 數에 依存함)에 따라 거리의 限界値(d_{limit})를 可變시켜 除去시킬 無關集合을 增減시켜줌으로써 入力函數의 컴퓨터 處理可能 範圍를 擴大시킨다.



*총민텀수(2^n)의 70~95%로 된 함수를 샘플(20개 이상)로 정하였다.

그림 5. CPU 시간에 결과치에 대한 그래프
Fig. 5. The graph for CPU time results.

〈附錄 1〉

거리 2에서부터 거리 k까지 essential off minterm 集合(EOS)을 求하는 過程 : 단, d=1의 on minterm數가 x個라 假定한다.

㉠ B와 d=1의 on minterm사이에 Required Adjacency Direction(RAD)를 얻는다.

㉡ x個의 RAD를 反復適用하여 EOS를 구한다.

한 RAD를 適用하여 얻어진 項들 중에서 거리가 가장적은 位置에 ($d \leq d_1$) 있는 off minterm들을 EOS에 넣고, 그 거리 以上의 項들은 除去한다. 그리고 演算過程 중에 얻는 d=k의 on minterm들은 除去한다.

例題) $f(W, X, Y, Z) = \sum(0, 2, 3, 4, 5, 8, 9, 11, 12, 13, 15)$에서 B는 11이고, $l=2$라 할 때, d=2에서의 EOS는?

1) d=1의 off minterm은 10, NRS_1은 Z'이다.

2) $f_{B1} = \sum(3, 5, 9, 11, 13, 15)$이고 d=1의 on minterm은 3, 9, 15이다.

㉠ RAD={−8, 4, −2}

㉡ 11[0]

−8↓

11[0] , 3[1]

4↓

11[0], 3[1], 15[1], 7[$\cancel{2}$] EOS={7}

−2↓

11[0], 3[1], 15[1], 9[1], 1[$\cancel{2}$], 13[$\cancel{2}$] EOS={7, 1}

* ×는 그항이 제거됨을 의미함

〈附錄 2〉

變數가 n個, B의 consensus가 r, $l=1$, 結果로 만들어진 主項이 m cube이고, d=1의 off minterm數(=nrc_1의 數)가 p個라 하자. ⟨$n > r \geq m$, $n = r + p$⟩ 이때 Arevalo方法과 提案된 方法에서 主項을 發生하는 횟수(N)는? 단, 여기서 S_x는 거리 r에서 거리 x까지 函數 $f'(=1-f)$에 包含되는 minterm의 갯수라 하고, nrc 發生過程과 主項 發生過程을 同一하다 假定한다.

i) Arevalo의 方法

(最大) $N_{max} = nC_r + nC_{r-1} \cdots nC_m - S_m$

(最小) $N_{min} = nC_r \cdots nC_{m+1} + 1 - S_{m+1}$

ii) 提案된 方法($d_{limit} = 1$)

(最大) $N_{max} = rCr + rC_{r-1} \cdots rC_m + p - S_m$

(最小) $N_{min} = rCr + rC_{r-1} \cdots rC_{m+1} + p + 1 - S_{m+1}$

參 考 文 献

[1] Z. Arevalo, "A method to simplify a boolean function into a near minimal sum-of-products for programmable logic arrays," *IEEE Trans. Comput.*, vol. C-27, pp. 1028-1039, Nov. 1978.

[2] V. Thomas Rhyne et al, "A new technique for the fast minimization of switching functions," *IEEE Trans. Comput.*, vol. C-26, pp. 757-763, Aug. 1977.

[3] P. Tison, "Generalization of consensus theory and application to the minimization of boolean functions," *IEEE Trans. Comput.*, vol. EC-16, pp. 446-456, Aug. 1967.

[4] S.J. Hong et al, "MINI: A heuristic approach for logic minimization," *IBM J. Res. Dev.*, vol. 18, pp. 443-457, Sept. 1974.

[5] J.R. Slagle et al, A new algorithm for generating prime implicants," *IEEE Trans. Comput.*, C-19, 304 (1970).

[6] B.L. Hulme and R.B. Worrell, "A prime implicant algorithm with factoring," *IEEE Trans. Comput.*, vol. C-24, pp. 1129-1131, Nov. 1975.

[7] J.P. Roth, "Algebraic topological methods for the synthesis of switching systems I," *Trans. Amer. Math. Soc.*, vol. 88, pp. 301-326, July., 1958.

[8] W.V. Quine, "A way to simplify truth functions," *Am. Math. Monthly* 62, 627 (1955).

[9] E.J. McCluskey, "Minimization of boolean functions," *Bell Syst. Tech. J.* 35, 1417 (1956).

[10] F.Mileto et al, "Average values of quantities appearing in boolean functions minimization," *IEEE Trans. Comput.*, vol. EC-13, pp. 87-92, Apr. 1964. *