Journal of The Korea Society of Computer and Information
Vol. 19, No. 3, March 2014
www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2014.19.3.109

회로 최소화를 위한 개선된 Quine-McCluskey 알고리즘

이 상 운*

An Improved Quine-McCluskey Algorithm for Circuit Minimization

Sang-Un Lee *

요 약

본 논문은 회로 최소화 문제에 대한 Quine-McCluskey 법을 개선한 알고리즘을 제안하였다. Quine-McCluskey 법은 주 내포 항을 반복적인 방법으로 찾고, 회로 최소화 방법으로 시행착오법, 분기한정법 또는 Petrick 법을 적용한다. 반면에 제안된 알고리즘은 사전에 항표를 생성하여 주 내포 항을 간단히 찾는 방법을 제안하였으며, 집합피복을 결정하는 방법을 적용하여 1차와 2차 필수 주 내포 항을 간단히 찾는 방법을 제안하였다. 3-변수와 4-변수 실험 데이터에 적용한 결과 제안된 알고리즘이 Quine-McCluskey 법에 비해 보다 간단하면서도 정확히 해를 구할 수 있었다.

▸ Keywords : 주 내포 항, 필수 주 내포 항, 빈도수, 원소 개수, 집합피복

Abstract

This paper revises the Quine-McCluskey Algorithm to circuit minimization problems. Quine-McCluskey method repeatedly finds the prime implicant and employs additional procedures such as trial-and-error, branch-and-bound, and Petrick's method as a means of circuit minimization. The proposed algorithm, on the contrary, produces an implicant chart beforehand to simplify the search for the prime implicant. In addition, it determines a set cover to streamline the search for 1st and 2nd essential prime implicants. When applied to 3-variable and 4-variable experimental data, the proposed algorithm has indeed proved to obtain the optimal solutions much more simply and accurately than the Quine-McCluskey method.

▸ Keywords : Prime Implicant, Essential Prime Implicant, Frequency, Cardinality, Set Cover

•제1저자 : 이상운
•투고일 : 2013. 12. 24, 심사일 : 2014. 01. 21, 게재확정일 : 2014. 02. 10.
* 강릉원주대학교 멀티미디어공학과 (Dept. of Multimedia Eng., Gangneung-Wonju National University)

I. 서 론

유어떠한 기능을 수행하는 논리회로 (logic circuit)를 설계하였을 때, 동일한 기능을 수행하면서도 가능한 부품 수와 입력단자 수를 최소화시키는 문제는 물리적 공간 제약성, 설계의 복잡성 뿐 아니라 생산 비용과 시간을 절약할 수 있는 경제적 측면에서 매우 요구되는 과정이다. 이를 회로 최소화 (circuit minimization, CM) 문제라 한다.

부울대수 (Boolean algebra)에서, 논리회로는 일반적으로 부울 함수 (Boolean function)로 표현된다. 일반적인 회로 최소화 문제는 다루기 힘든 문제로 알려져 왔다[1]. 그러나 1950년대에 수기식 방법인 카르노 맵 (Karnaugh maps)과 전산화가 가능한 효율적인 휴리스틱 방법인 Quine-McCluskey 알고리즘이 제안된 이후 지금까지 보다 효율적이고 간단한 방법에 대한 연구는 거의 없는 실정이다. 따라서, 디지털공학 분야에서 카르노 맵이나 Quine-McCluskey 알고리즘을 일반적으로 활용하고 있다[2,3].

카르노 맵은 주어진 부울 함수를 수작업으로 최소화시키는 전통적인 방법으로 많은 시간과 노력이 소요되며, 지루하고 오류를 유발시킬 수 있는 방법이다. 이 방법은 변수가 6개 이상이 되면 적합하지가 않으며, 실제적으로는 단지 4 변수까지만 적용되고 있다. 또한, 이 방법은 컴퓨터 프로그램으로 자동화시킬 수 없다.

n개의 변수는 2^n개의 최소 항 (minterms)을 가질 수 있다. n개의 변수에 대한 주어진 부울 함수에서 나타난 최소 항의 개수를 k라 하면, $k<2^n$이 된다. 이 부울 함수로부터 $2^i(i=1,2,\cdots,n-1)$개의 그룹을 만들 수 있으며, 이를 내포 항 (implicant)이라 한다.

Quine-McCluskey 알고리즘[3]은 주어진 부울 함수에 대한 내포 항들을 구하고, 다른 내포 항에 완전히 포함되지 않는 하나의 내포 항인 주 내포 항 (prime implicants, PI)과 다른 PI에는 없는 최소 항을 포함하는 주 내포 항인 필수 주 내포 항 (essential PI, EPI)을 구하여, 회로를 단순화시킨다.

따라서, Quine-McCluskey 알고리즘은 주 내포 항 방법 (the method of prime implicants) 또는 도표 방법 (tabulation method)이라고도 부르며, 1956년에 카르노 맵의 대안으로 제안되었다. 이 방법은 카르노 맵과 기능적으로 동일하지만, 도표를 활용하기 때문에 컴퓨터 알고리즘으로 사용하기에 보다 효율적이며, 부울 함수의 최소 형태를 검증하는데 있어서 결정론적 방법을 제공하는 장점을 갖고 있다. Quine-McCluskey 알고리즘은 4 변수 이상을 다루는데 있어 카르노 맵에 비해 보다 효율적이지만, 변수의 개수 n이 증가하면 주 내포 항의 개수는 $3^n/n$으로 증가하기 때문에 알고리즘 수행시간과 메모리는 기하급수적으로 증가하여 효율적이지 못한 단점을 갖고 있다.

보다 근본적으로 별개의 접근법으로, Brayton[4]은 일명 에스프레소 알고리즘인 Espresso heuristic logic minimizer를 제안하였으며, 회로 최소화 문제에서 사실상 표준 방법으로 사용되고 있다. 이 알고리즘은 일반적으로 10개의 출력을 가진 10 변수까지 다룰 수 있으며, 주어진 부울 함수를 입력하면 최소화된 진리표를 출력하도록 프로그램이 되어 있다.

이 밖에도 Vinodchandran[5], Siggh et al.[6], Morrison과 Ranganathan[7]의 연구결과가 있었지만 Quine-McCluskey 알고리즘보다 간단한 결과는 얻지 못하였다.

본 논문에서는 프로그램화된 도구인 Espresso heuristic logic minimizer를 활용하지 않고, Quine-McCluskey 알고리즘을 효율적으로 개선하여 적용할 수 있는 방법을 제안한다. 2장에서는 Quine-McCluskey 알고리즘을 고찰한다. 3장에서는 Quine-McCluskey 알고리즘을 효율적으로 개선한 알고리즘을 제안한다. 4장에서는 제안된 알고리즘을 다양한 사례들에 적용하여 본다.

II. Quine-McCluskey 알고리즘

Quine-McCluskey 알고리즘[3]은 다음과 같이 2단계를 수행한다.

Step 1. 주어진 부울 함수의 모든 주 내포 항 (PI)을 찾는다.
- 반복적 방법을 적용하여 $2^0,2^1,2^2,\cdots,2^{n-1}$항들 중에서 PI를 찾는다.

Step 2. Step 1에서 찾은 PI를 행으로, 부울 함수의 최소항을 열로 하는 주 내포 항 차트 (prime implicant chart, PIC)를 사용하여 주어진 부울 함수의 최소항들을 모두 커버하는 필수 주 내포 항 (EPI) 뿐 아니라 다른 PI들을 찾는다.
- 시행착오법 (trial-and-error)으로 1차 EPI와 2차 EPI를 찾는 방법
- Petrick's Method [8]
- 분기한정법 (Branch-and-Bound Method)

Quine-McCluskey 알고리즘[3]을 수행하는 과정을 보다

자세히 파악하기 위해 식 (1)의 부울 함수를 고찰해 보자.

$$F_1(A,B,C,D) = \Sigma m(4,8,9,10,11,12,14,15) \qquad (1)$$

식 (1)은 4 변수 A,B,C,D 입력에 대한 회로도이다. 여기서 m은 최소 항을 의미하며, 숫자는 최소 항의 인덱스이다. 식 (1)을 곱의 합 (sum of product, SoP)로 다시 표현하면 식 (2)와 같다.

$$f_1(A,B,C,D) = A'BC'D' + AB'C'D + AB'C'D + \quad (2)$$
$$AB'CD' + AB'CD + ABC'D' +$$
$$ABCD' + ABCD$$

Step 1은 표 1과 같이 수행된다. 주어진 부울 함수에 대해 $k=8<2^4$으로 $2^0,2^1,2^2,2^3$항으로 그룹으로 묶을 수 있는 내포 항을 결정한다. 2^i항은 2^{i-1}항의 조합으로 만들어진다. 만약, 2^i 항의 모든 인덱스 값이 2^{i+1}의 하나의 항에 포함되지 않으면 '*'를 표시한다. 이 '*'가 부여된 항을 주 내포 항 (PI)이라 하며, Step 2에 활용된다. Step 1을 수행하는데 있어 다음과 같은 2가지 문제점을 갖고 있다.

(1) 주어진 부울 함수에 대해 어떤 최소 항들의 그룹이 주 내포 항 (PI)이 되는지를 알 수 없기 때문에 $2^1,\cdots 2^{n-1}$의 가능한 항들 중에서 주 내포 항을 결정하는데 어려움이 있다.

(2) $2^0,2^1,\cdots 2^{n-1}$의 순서로 항을 발췌하고, 역으로 보다 작은 크기의 항이 보다 큰 항에 포함되는지 여부를 결정하여야 한다.

표 1. 주 내포 항 결정
Table 1. Determination of Prime Implicants

1의 개수	2^0 항	2^1 항	2^2 항	2^3 항
1	m(4) (0100)	**m(4,12) -100***	-	-
	m(8) (1000)	m(8,9) 100- m(8,9) 100- m(8,12) 1-00	**m(8,9,10,11) 10--*** **m(8,10,12,14) 1--0***	-
2	m(9) (1001)	m(9,11) 10-1	-	-
	m(10) (1010)	m(10,11) 101- m(10,14) 1-10	**m(10,11,14,15) 1-1-***	-
	m(12) (1100)	m(12,14) 11-0	-	-
3	m(11) (1011)	m(11,15) 1-11	-	-
	m(14) (1110)	m(14,15) 111-	-	-
4	m(15) (1111)	-	-	-

Step 2에서는 첫 번째로, Step 1에서 얻은 '*' 항들을 행으로, 주어진 부울 함수를 열로 하는 표 2의 주 내포 항 차트를 작성한다. 여기서 '*'는 필수 주 내포 항 (EPI)을 의미한다.

주어진 부울 함수를 최소화시키기 위해서는 EPI를 모두 선택하고, 추가적으로 필요시 나머지 PI들 중 2차 EPI를 선택해야 한다. 이 과정을 수행하기 위해서는 EPI가 어떤 것인지 결정해야 하는 문제가 발생한다. Step 2를 수행하는 방법에는 시행착오법과 보다 체계적인 Petrick 방법[8]이나 분기한정법이 있다. 여기서는 Petrick법을 고찰한다.

표 2. 주 내포 항 차트
Table 2. Prime Implicant Chart

주 내포 항	부울 함수								선택
	m4	m8	m9	m10	m11	m12	m14	m15	
m(4,12)*	X					X			O
m(8,9,10,11)		X	X	X	X				O
m(8,10,12,14)		X		X		X	X		
m(10,11,14,15)*				X	X		X	X	O

$$F_1(A,B,C,D) = BC'D' + AB' + AC$$

Petrick 방법[8]은 $X+XY=X, XX=X, X+X=X$로 단순화 시키는 방법으로 다음과 같이 수행된다.

(1) PIC에서 EPI 행과 이와 연관된 열을 삭제하여 PIC를 축소시킨다.

(2) 축소된 PIC의 행에 라벨을 $P_1,P_2,\cdots$로 부여한다.

(3) 모든 열을 커버하는 논리함수 P를 형성한다. P는 각 합 항 (sum term) $P_{i0}+P_{i1}+\cdots+P_{in}$의 PoS (product of sums)로 구성되며, P_{ij}는 열 j를 커버하는 행을 표현한다.

(4) 각 열에 대해 X가 부여된 해당 행의 주 내포 항을 더하고, 각 열을 곱한 PoS를 생성하고, $X+XY=X, XX=X, X+X=X$를 적용하여 최소 SoP인 P로 축소시킨다.

(5) 결과로 얻은 각 항은 해를 의미한다. 최소 해를 얻었는지 결정하기 위해 그들 항이 최소의 PI를 포함하고 있는지를 찾는다.

(6) 다음으로, 각 PI에 있는 문자의 수를 계산하고, 총 문자수를 계산한다.

(7) 총 문자수가 최소인 항을 해로 결정한다.

F_1 부울 함수 문제에 대해 Petrick 방법을 적용하면 다음과 같이 수행된다.

$$w = m(4,12) = AB'C', x = m(8,9,10,11) = AB',$$
$$y = m(8,10,12,14) = AD', z = m(10,11,14,15) = AC$$

$$m4 = w, m8 = (x+y), m9 = x, m10 = (x+y+z),$$
$$m11 = (x+z), m12 = (w+y), m14 = (y+z),$$
$$m15 = z$$

$$= w(x+y)x(x+y+z)(x+z)(w+y)(y+z)z$$

$$(x+y)(x+z) = xx+xy+xz+yz$$
$$= x+x(y+z)+yz$$
$$= x+x+yz$$
$$= x+yz$$

$$(y+w)(y+z) = yy+yz+yw+wz$$
$$= y+y(z+w)+wz$$
$$= y+y+wz$$
$$= y+wz$$

$$= w(x+yz)x(x+y+z)(y+wz)z$$

$$(x+yz)(x+y+z) = xx+xy+xz+xyz+yyz+yzz$$
$$= x+x(y+z)+xyz+yz+yz$$
$$= x+xyz+yz$$
$$= x+yz$$

$$= w(x+yz)x(y+wz)z$$

$$(x+yz)(y+wz) = xy+wxz+yyz+wyzz$$
$$= xy+wxz+yz+wyz$$
$$= xy+wxz+yz(1+w)$$
$$= xy+wxz+yz$$

$$= w(xy+wxz+yz)xz$$
$$= wxzxy+wxzwxz+wxzwz$$
$$= wxzyz+wxz+wxz$$
$$= wxz(yz+1+1)$$
$$= wxz$$

문자 개수가 가장 작은 항 선택 : wxz

$$F_1(A,B,C,D) = wxz = BC'D' + AB' + AC$$

Petrick 방법은 회로 단순화 방법을 비록 시행착오법에 비해 보다 체계적인 방법으로 전환시켰지만, 너무 복잡하여 실수를 유발할 가능성이 높은 단점을 갖고 있다.

따라서, 3장에서는 Quine-McCluskey 알고리즘의 Step 1과 Step 2의 문제점을 개선할 수 있는 알고리즘을 제안한다.

III. 개선된 Quine-McCluskey 알고리즘

본 장에서는 주어진 부울 함수의 회로를 최소화하기 위해 Quine-McCluskey 알고리즘의 2단계를 수행하지만 Step 1에서 보다 체계적으로 내포 항을 정확히 추출하는 항표(Implicant table)를 이용하는 방법을 적용하고, Step 2에서 Petrick 방법에 비해 보다 간단히 회로를 최소화 시키는 방법으로 집합피복 (Set Cover)을 찾는 방법을 제안한다.

Step 1에서 항표를 이용하기 위해, 사전에 부울 함수의 변수 개수 $n(n=2,3,\cdots,10)$에 대한 카르노 맵으로부터 2^{n-1}, $2^{n-2},\cdots,2^1$의 항을 구한 항표 (Implicant table)를 얻어 이를 활용하는 방법을 적용한다. 단, 필요시 n은 증가시킬 수 있다.

[사전 준비]

주어진 부울 함수의 변수 개수 $n(n=2,3,\cdots,10)$에 대한 카르노 맵을 그려 $2^{n-1},2^{n-2},\cdots,2^1$개씩 그룹으로 묶은 항을 구하여 항표를 저장한다. 단, 2^0인 최소 항은 구하지 않는다.

Step 1. 부울 함수의 주 내포 항 선택

(1) 항표의 해당 원소들이 주어진 부울 함수 F의 진부분집합인 것만 선택 ("O")한다. 즉, 항표의 특정 항에 대해 해당 항의 어떠한 하나의 원소라도 부울 함수에 존재하지 않으면 선택하지 않는다.

(2) 보다 작은 크기의 하위 항 2^i의 모든 원소들이 보다 큰 크기의 상위 항 $2^{n-1},2^{n-2},\cdots,2^{i+1}$의 어느 하나의 항에 존재 ("O")하 면 선택하지 않으며, 그렇지 않으면 ("x") 선택한다. 단, 최상위 항 은 모두 선택 ("x") 한다.

Step 2. 회로 최소화 (EPI와 PI 선택)

주 내포 항을 행으로, 부울 함수의 최소 항을 열로 하는 주 내포 항 차트를 작성한다.

SelectProcess()

(1) 부울 함수 F의 각 원소 x에 대해 Step 1에서 선택된 주 내포 항 집합에서의 발생빈도 f_x를 구한다.

(1-1) 만약, $f_x=0$인 원소 x가 존재하면, x를 C에 포함시키고, F에서 x를 삭제한다.

(1-2) 만약, $f_x=1$인 원소 x가 존재하면, $x \in P_i$인 주 내포 항을 선택하여 C에 포함시키고, F와 다른 주 내포 항에 대해 선택된 주 내포 항의 원소들을 삭제한다.

(2) 만약, $f_x=1$인 원소 x가 존재하지 않으면 **IncludingProcess()**를 수행한다.

IncludingProcess()

(1) 원소개수가 다른 주 내포 항들로 구성된 경우

(1-1) 원소개수 (cardinality)가 최소인 주 내포 항 $_{\min}c$이 가능한 최대 원소개수를 가진 주 내포 항 $_{\max}c$의 진부분집합이면 $_{\min}c$를 삭제한다. 이 과정은 더 이상 삭제할 $_{\min}c$가 없을 때까지 혹은 $f_x=1$이 발생할 때까지 반복 수행한다.

(1-2) 만약, $f_x=1$이 존재하면 $f_x=1$의 원소를 갖고 있는 주항을 C에 포함시킨다. 만약, $f_x=1$이 존재하지 않고, 원소개수가 1인 집합이 다수 존재하면 해당 집합들 중 어느 하나를 C에 포함시킨다.

(1-3) 만약, $F=\phi$이면 알고리즘을 종료한다.

(2) 동일한 원소개수를 가진 주 내포 항들만 존재하는 경우

(2-1) 임의의 주 내포 항을 선택하여 C에 포함시키고, F와 다른 주 내포 항에 대해 선택된 주 내포 항의 원소들을 삭제한다.

(2-2) 남은 주 내포 항들에 대해 $f_x=1$이 존재하면 $x \in P_i$인 주 내포 항을 선택하여 C에 포함시키고, f와 다른 주 내포항에 대해 선택된 주 내포 항의 원소들을 삭제한다. 만약, $f_x=1$이 존재하지 않으면 최대 원소 개수를 가진 주 내포 항을 선택하여 C에 포함시키고, F와 다른 주 내포 항에 대해 선택된 주 내포 항의 원소들을 삭제한다. 이 과정을 $F=\phi$이 될 때까지 반복 수행한다.

식 (1)의 F_1 부울 함수에 대해 제안된 알고리즘을 적용한 과정은 표 3에 제시되어 있다.

표 3. 개선된 Quine-McCluskey 알고리즘
Table 3. Improved Quine-McCluskey Algorithm
(a) Step 1 (항표에서 내포 항 선택)

$F_1(A,B,C,D) = \Sigma m(4,8,9,10,11,12,14,15)$

구분	항	F의 진부분집합	항	F의 진부분집합
2^3 항 (8가지)	m(0,1,2,3,4,5,6,7)=A′	x	m(1,3,5,7,9,11,13,15)=D	x
	m(0,1,4,5,8,9,12,13)=C′	x	m(2,3,6,7,10,11,14,15)=C	x
	m(0,2,4,6,8,10,12,14)=D′	x	m(4,5,6,7,12,13,14,15)=B	x
	m(0,1,2,3,8,9,10,11)=B′	x	m(8,9,10,11,12,13,14,15)=A	x
2^2 항 (24가지)	m(0,1,2,3) = A′B′	x	m(3,7,11,15) = CD	x
	m(0,4,8,12) = C′D′	x	m(4,5,6,7) = A′B	x
	m(0,1,4,5) = A′C′	x	m(4,5,12,13) = BC′	x
	m(0,2,4,6) = A′D′	x	m(4,6,12,14) = BD′	x
	m(0,1,8,9) = B′C′	x	m(5,7,13,15) = BD	x
	m(0,2,8,10) = B′D′	x	m(6,7,14,15) = BC	x
	m(1,3,5,7) = A′D	x	m(8,9,10,11) = AB′	O
	m(1,3,9,11) = B′D	x	m(8,9,12,13) = AC′	x
	m(1,5,9,13) = C′D	x	m(8,10,12,14) = AD′	O
	m(2,3,10,11) = B′C	x	m(9,11,13,15) = AD	x
	m(2,6,10,14) = CD′	x	m(10,11,14,15) = AC	O
	m(2,3,6,7) = A′C	x	m(12,13,14,15) = AB	x
2^1 항 (32가지)	m(0,1) = A′B′C′	x	m(5,13) = BC′D	x
	m(0,2) = A′B′D′	x	m(6,7) = A′BC	x
	m(0,4) = A′C′D′	x	m(6,14) = BCD′	x
	m(0,8) = B′C′D′	x	m(7,15) = BCD	x
	m(1,3) = A′B′D	x	m(8,9) = AB′C′	O
	m(1,5) = A′C′D	x	m(8,10) = AB′D′	O
	m(1,9) = B′C′D	x	m(8,12) = AC′D′	O
	m(2,3) = A′B′C	x	m(9,11) = AB′D	O
	m(2,6) = A′CD′	x	m(9,13) = AC′D	x
	m(2,10) = B′CD′	x	m(10,11) = AB′C	O
	m(3,7) = A′CD	x	m(10,14) =ACD′	O
	m(3,11) = B′CD	x	m(11,15) = ACD	O
	m(4,5) = A′BC′	x	m(12,13) = ABC′	x
	m(4,6) = A′BD′	x	m(12,14) = ABD′	O
	m(4,12) = BC′D′	O	m(13,15) = ABD	x
	m(5,7) = A′BD	x	m(14,15) = ABC	O

구분	항	상위 항의 진부분집합	항	상위 항의 진부분집합
2^2 항	**(8,9,10,11)**	x	**(10,11,14,15)**	x
	(8,10,12,14)	x		
2^1 항	**(4,12)**	x	(10,11)	O
	(8,9)	O	(10,14)	O
	(8,10)	O	(11,15)	O
	(8,12)	O	(12,14)	O
	(9,11)	O	(14,15)	O

(b) Step 2 (회로 최소화)

내포 항	부울 함수								선택
	m4	m8	m9	m10	m11	m12	m14	m15	
m(4,12)	**X**					X			O
m(8,9,10,11)		X	**X**	X	X				O
m(8,10,12,14)		X		X		X	X		
m(10,11,14,15)				X	X		X	**X**	O
빈도수	**1**	2	**1**	3	2	2	2	**1**	

내포 항	부울 함수								선택
									$f=\phi$
빈도수									

$F_1(A,B,C,D) = BC'D' + AB' + AC$

IV. 실험 및 결과 분석

본 장에서는 먼저, 식 (3)의 부울 함수에 대해 Quine-McCluskey 알고리즘과 제안된 알고리즘을 비교하여 제안된 알고리즘이 Quine-McCluskey 알고리즘에 비해 얼마나 간단한지를 고찰해 본다. 다음으로, 식 (4) ~ 식 (7)에 대해 제안된 알고리즘만을 적용하여 본다.

$$F_2(A,B,C) = \Sigma m(0,1,2,5,6,7) \quad (3)$$

$$F_3(A,B,C) = \Sigma m(1,3,4,5,6) = A'C + AB' + AC' \quad (4)$$

$$F_4(A,B,C,D) = \Sigma m(0,2,5,6,7,8,10,12,13,14,15) = B'D' + BD + CD' + AD' \quad (5)$$

$$F_5(A,B,C,D) = \Sigma m(0,2,3,4,5,6,7,8,9,10,11,12,13) = A'D' + A'C + BC' + AB' \text{ or } A'D' + B'C + A'B + AC' \quad (6)$$

$$F_6(A,B,C,D) = \Sigma m(1,5,6,7,11,12,13,15) = ABC' + ACD + A'BC + A'C'D \quad (7)$$

$F_2(A,B,C)$에 대한 Quine-McCluskey 알고리즘의 PIC는 다음과 같다.

1의 개수	2^0 항	2^1 항	2^2 항
0	m0 (000)	m(0,1) 00-* m(0,2) 0-0*	-
1	m1 (001) m2 (010)	m(1,5) -01* m(2,6) -10*	- -
2	m5 (101) m6 (110)	m(5,7) 1-1* m(6,7) 11-*	- -
3	m7 (111)	-	-

내포 항	부울 함수					
	m0	m1	m2	m5	m6	m7
K m(0,1)	**X**	**X**	-	-	-	-
L m(0,2)	**X**	-	**X**	-	-	-
M m(1,5)	-	**X**	-	**X**	-	-
N m(2,6)	-	-	**X**	-	**X**	-
P m(5,7)	-	-	-	**X**	-	**X**
Q m(6,7)	-	-	-	-	**X**	**X**

위 PIC에 대해 Petrick 방법을 적용하면 다음과 같이 수행된다.

$$= (K+L)(K+M)(L+N)(M+P)(N+Q)(P+Q)$$

$$(K+L)(K+M) = KK + KM + KL + LM = K + K(M+L) + LM = K + K + LM = K + LM$$

$$(L+N)(N+Q) = NN + NL + NQ + LQ = N + N(L+Q) + LQ = N + N + LQ = N + LQ$$

$$(M+P)(P+Q) = PP + PM + PQ + MQ = P + P(M+Q) + MQ = P + P + MQ = P + MQ$$

$$= (K+LM)(N+LQ)(P+MQ)$$

$$(K+LM)(N+LQ) = KN + KLQ + LMN + LLMQ = KN + KLQ + LMN + LMQ$$

$$= (KN + KLQ + LMN + LMQ)(P+MQ)$$

$$= KNP + KMNQ + KLPQ + KLMQ + LMNP + LMMNQ + LMPQ + LMMQQ$$

$= KNP + KMNQ + KLPQ + KLMQ + LMNP + LMNQ + LMPQ + LMQ$
$KLMQ + LMNQ + LMPQ + LMQ$
$= LMQ(K + N + P + 1)$
$= LMQ$
$= KNP + KMNQ + KLPQ + LMQ + LMNP$
(3) (4) (4) (3) (4)
$F_2 = KNP$ or LMQ
$F_2 = KNP = A'B' + BC' + AC$ or
$LMQ = A'C' + B'C + AB$

$F_2(A,B,C)$에 대해 제안된 알고리즘은 다음과 같이 수행된다.

Step 1.

$F_2(A,B,C) = \Sigma m(0,1,2,5,6,7)$

구분	항	F의 진부분집합	항	F의 진부분집합
2^2 항 (6가지)	m(0,1,2,3)	x	m(1,3,5,7)	x
	m(0,1,4,5)	x	m(2,3,6,7)	x
	m(0,2,4,6)	x	m(4,5,6,7)	x
2^1 항 (12가지)	m(0,1)	O	m(2,6)	O
	m(0,2)	O	m(3,7)	x
	m(0,4)	x	m(4,5)	x
	m(1,3)	x	m(4,6)	x
	m(1,5)	O	m(5,7)	O
	m(2,3)	x	m(6,7)	O

Step 2.

내포 항	부울 함수						선택
	m0	m1	m2	m5	m6	m7	
m(0,1)	1	1	-	-	-	-	O
m(0,2)	1	-	1	-	-	-	-
m(1,5)	-	1	-	1	-	-	-
m(2,6)	-	-	1	-	1	-	-
m(5,7)	-	-	-	1	-	1	-
m(6,7)	-	-	-	-	1	1	-
빈도수	2	2	2	2	2	2	

내포 항	부울 함수						원소 개수	선택
			m2	m5	m6	m7		
m(0,2)			1	-	-	-	1	-
m(1,5)			-	1	-	-	1	-
m(2,6)			1	-	1	-	2	O
m(5,7)			-	1	-	1	2	-
m(6,7)			-	-	1	1	2	-
빈도수			3	2	3	2		

내포 항	부울 함수						원소 개수	선택
				m5		m7		
m(0,2)				-		-	-	-
m(0,6)				-		-	-	-
m(1,2)				-		-	-	-
m(1,5)				1		-	1	-
m(5,7)				1		1	2	-
m(6,7)				-		1	1	O
빈도수				2		2		

내포 항	부울 함수						원소 개수	선택
빈도수								$f = \phi$

$F_2 = m(0,1) + m(2,6) + m(5,7) = A'B' + BC' + AC$

$F_3(A,B,C)$에 대해 제안된 알고리즘은 다음과 같이 수행된다.

Step 1.

$F_3(A,B,C) = \Sigma m(1,3,4,5,6)$

구분	항	F의 진부분집합	항	F의 진부분집합
2^2 항 (6가지)	m(0,1,2,3)	x	m(1,3,5,7)	x
	m(0,1,4,5)	x	m(2,3,6,7)	x
	m(0,2,4,6)	x	m(4,5,6,7)	x
2^1 항 (12가지)	m(0,1)	x	m(2,6)	x
	m(0,2)	x	m(3,7)	x
	m(0,4)	x	m(4,5)	O
	m(1,3)	O	m(4,6)	O
	m(1,5)	O	m(5,7)	x
	m(2,3)	x	m(6,7)	x

Step 2.

내포 항	부울 함수					선택
	m1	m3	m4	m5	m6	
m(1,3) = A'C	1	1				O
m(1,5) = B'C	1			1		
m(4,5) = AB'			1	1		
m(4,6) = AC'			1		1	O
빈도수	2	1	2	2	1	

내포 항	부울 함수					선택
				m5		
m(1,5) = B'C				1		O
m(4,5) = AB'				1		
빈도수				2		

$F_3(A,B,C) = A'C + B'C + AC'$ or $A'C + AB' + AC'$

$F_4(A,B,C,D)$에 대해 제안된 알고리즘은 다음과 같이 수행된다.

$F_4(A,B,C,D) = \Sigma m(0,2,5,6,7,8,10,12,13,14,15)$

구분	항	F의 진부분집합	항	F의 진부분집합
2^3 항 (8가지)	m(0,1,2,3,4,5,6,7)=A'	x	m(1,3,5,7,9,11,13,15)=D	x
	m(0,1,4,5,8,9,12,13)=C'	x	m(2,3,6,7,10,11,14,15)=C	x
	m(0,2,4,6,8,10,12,14)=D'	x	m(4,5,6,7,12,13,14,15)=B	x
	m(0,1,2,3,8,9,10,11)=B'	x	m(8,9,10,11,12,13,14,15)=A	x
2^2 항 (24가지)	m(0,1,2,3) = A'B'	x	m(3,7,11,15) = CD	x
	m(0,4,8,12) = C'D'	x	m(4,5,6,7) = A'B	x
	m(0,1,4,5) = A'C'	x	m(4,5,12,13) = BC '	x
	m(0,2,4,6) = A'D'	x	m(4,6,12,14) = BD'	x
	m(0,1,8,9) = B'C'	x	m(5,7,13,15) = BD '	O
	m(0,2,8,10) = B'D'	O	m(6,7,14,15) = BC	O
	m(1,3,5,7) = A'D	x	m(8,9,10,11) = AB'	x
	m(1,3,9,11) = B'D	x	m(8,9,12,13) = AC'	x
	m(1,5,9,13) = C'D	x	m(8,10,12,14) = AD'	O
	m(2,3,10,11) = B'C	x	m(9,11,13,15) = AD	x
	m(2,6,10,14) = CD'	O	m(10,11,14,15) = AC	x
	m(2,3,6,7) = A'C	x	m(12,13,14,15) = AB	O

내포 항	부울 함수												선택
	m0	m2	m5	m6	m7	m8	m10	m12	m13	m14	m15		
m(0,2,8,10) = B'D'	1	1				1	1						O
m(2,6,10,14) = CD'		1		1			1			1			
m(5,7,13,15) = BD '			1		1				1		1		O
m(6,7,14,15) = BC				1	1					1	1		
m(8,10,12,14) = AD'						1	1	1		1			
m(12,13,14,15) = AB								1	1	1	1		
빈도수	1	2	1	2	2	2	3	2	2	4	3		

내포 항	부울 함수															선택
				m6				m12		m14						
m(2,6,10,14) = CD′				1						1						O
m(6,7,14,15) = BC				1						1						
m(8,10,12,14) = AD′								1		1						
m(12,13,14,15) = AB								1		1						
빈도수				2				2		4						

내포 항	부울 함수															선택
								m1								
m(6,7,14,15) = BC																
m(8,10,12,14) = AD′								1								O
m(12,13,14,15) = AB								1								
빈도수								2								

$$F_4(A,B,C,D) = B'D' + CD' + BD' + AD'$$

$F_5(A,B,C,D)$에 대해 제안된 알고리즘은 다음과 같이 수행된다.

$$F_5(A,B,C,D) = \Sigma m(0,2,3,4,5,6,7,8,9,10,11,12,13)$$

구분	항	F의 진부분집합	항	F의 진부분집합
2^3 항 (8가지)	m(0,1,2,3,4,5,6,7)=A′	x	m(1,3,5,7,9,11,13,15)=D	x
	m(0,1,4,5,8,9,12,13)=C′	x	m(2,3,6,7,10,11,14,15)=C	x
	m(0,2,4,6,8,10,12,14)=D′	x	m(4,5,6,7,12,13,14,15)=B	x
	m(0,1,2,3,8,9,10,11)=B′	x	m(8,9,10,11,12,13,14,15)=A	x
2^2 항 (24가지)	m(0,1,2,3) = A′B′	x	m(3,7,11,15) = CD	x
	m(0,4,8,12) = C′D′	O	m(4,5,6,7) = A′B	O
	m(0,1,4,5) = A′C′	x	m(4,5,12,13) = BC ‘	O
	m(0,2,4,6) = A′D′	O	m(4,6,12,14) = BD′	x
	m(0,1,8,9) = B′C′	x	m(5,7,13,15) = BD ‘	x
	m(0,2,8,10) = B′D′	O	m(6,7,14,15) = BC	x
	m(1,3,5,7) = A′D	x	m(8,9,10,11) = AB′	O
	m(1,3,9,11) = B′D	x	m(8,9,12,13) = AC′	O
	m(1,5,9,13) = C′D	x	m(8,10,12,14) = AD′	x
	m(2,3,10,11) = B′C	O	m(9,11,13,15) = AD	x
	m(2,6,10,14) = CD′	x	m(10,11,14,15) = AC	x
	m(2,3,6,7) = A′C	O	m(12,13,14,15) = AB	x

내포 항	부울 함수													선택
	m0	m2	m3	m4	m5	m6	m7	m8	m9	m10	m11	m12	m13	
m(0,4,8,12) = C′D′	1			1				1				1		O
m(0,2,4,6) = A′D	1	1		1		1								
m(0,2,8,10) = B′D	1	1						1		1				
m(2,3,10,11) = B′C		1	1							1	1			
m(2,3,6,7) = A′C		1	1			1	1							
m(4,5,6,7) = A′B				1	1	1	1							
m(4,5,12,13) = BC ‘				1	1							1	1	
m(8,9,10,11) = AB′								1	1	1	1			
m(8,9,12,13) = AC′								1	1			1	1	
빈도수	3	4	2	4	2	3	2	4	2	3	2	3	2	

내포 항	부울 함수												선택
	m2	m3		m5	m6	m7		m9	m10	m11		m13	
m(0,2,4,6) = A′D	1				1								
m(0,2,8,10) = B′D	1								1				
m(2,3,10,11) = B′C	1	1							1	1			O
m(2,3,6,7) = A′C	1	1			1	1							
m(4,5,6,7) = A′B				1	1	1							
m(4,5,12,13) = BC ‘				1								1	
m(8,9,10,11) = AB′								1	1	1			
m(8,9,12,13) = AC′								1				1	
빈도수	4	2		2	3	2		2	3	2		2	

내포 항	부울 함수													선택
				m5	m6	m7		m9				m13		
m(0,2,4,6) = A′D					1									
m(2,3,6,7) = A′C					1	1								
m(4,5,6,7) = A′B				1	1	1								O
m(4,5,12,13) = BC ‘				1								1		
m(8,9,10,11) = AB′								1						
m(8,9,12,13) = AC′								1				1		
빈도수				2	3	2		2				2		

내포 항	부울 함수													선택
								m					m13	
m(4,5,12,13) = BC ‘													1	
m(8,9,10,11) = AB′								1						
m(8,9,12,13) = AC′								1					1	O
빈도수								2					2	

$$F_5(A,B,C,D) = C'D' + B'C + A'B + AC'$$

$F_6(A,B,C,D)$에 대해 제안된 알고리즘은 다음과 같이 수행된다.

$$F_6(A,B,C,D) = \Sigma m(1,5,6,7,11,12,13,15)$$

구분	항	F의 진부분집합	항	F의 진부분집합
2^3 항 (8가지)	m(0,1,2,3,4,5,6,7)=A′	x	m(1,3,5,7,9,11,13,15)=D	x
	m(0,1,4,5,8,9,12,13)=C′	x	m(2,3,6,7,10,11,14,15)=C	x
	m(0,2,4,6,8,10,12,14)=D′	x	m(4,5,6,7,12,13,14,15)=B	x
	m(0,1,2,3,8,9,10,11)=B′	x	m(8,9,10,11,12,13,14,15)=A	x
2^2 항 (24가지)	m(0,1,2,3) = A′B′	x	m(3,7,11,15) = CD	x
	m(0,4,8,12) = C′D′	x	m(4,5,6,7) = A′B	x
	m(0,1,4,5) = A′C′	x	m(4,5,12,13) = BC ‘	x
	m(0,2,4,6) = A′D′	x	m(4,6,12,14) = BD′	x
	m(0,1,8,9) = B′C′	x	m(5,7,13,15) = BD ‘	O
	m(0,2,8,10) = B′D′	x	m(6,7,14,15) = BC	x
	m(1,3,5,7) = A′D	x	m(8,9,10,11) = AB′	x
	m(1,3,9,11) = B′D	x	m(8,9,12,13) = AC′	x

$$F_6(A,B,C,D) = \Sigma m(1,5,6,7,11,12,13,15)$$

구분	항	F의 진부분집합	항	F의 진부분집합
2^2 항 (24가지)	m(1,5,9,13) = C′D	x	m(8,10,12,14) = AD′	x
	m(2,3,10,11) = B′C	x	m(9,11,13,15) = AD	x
	m(2,6,10,14) = CD′	x	m(10,11,14,15) = AC	x
	m(2,3,6,7) = A′C	x	m(12,13,14,15) = AB	x
2^1 항 (32가지)	m(0,1) = A′B′C′	x	m(5,13) = BC′D	O
	m(0,2) = A′B′D′	x	m(6,7) = A′BC	O
	m(0,4) = A′C′D′	x	m(6,14) = BCD′	x
	m(0,8) = B′C′D′	x	m(7,15) = BCD	O
	m(1,3) = A′B′D	x	m(8,9) = AB C′	x
	m(1,5) = A′C′D	O	m(8,10) = AB D′	x
	m(1,9) = B′C′D	x	m(8,12) = AC′D′	x
	m(2,3) = A′B′C	x	m(9,11) = AB D	x
	m(2,6) = A′CD′	x	m(9,13) = AC′D	x
	m(2,10) = B′CD′	x	m(10,11) = AB′C	x
	m(3,7) = A′CD	x	m(10,14) =ACD′	x
	m(3,11) = B′CD	x	m(11,15) = ACD	O
	m(4,5) = A′BC′	x	m(12,13) = ABC ′	O
	m(4,6) = A′BD′	x	m(12,14) = ABD′	x
	m(4,12) = BC′D′	x	m(13,15) = ABD	O
	m(5,7) = A′BD	O	m(14,15) = ABC	x

구분	항	상위 항의 진부분집합	항	상위 항의 진부분집합
2^2 항	m(5,7,13,15) = BD ‘			
2^1 항	m(1,5) = A′C′D	x	m(7,15) = BCD	O
	m(5,7) = A′BD	O	m(11,15) = ACD	x
	m(5,13) = BC′D	O	m(12,13) = ABC′	x
	m(6,7) = A′BC	x	m(13,15) = ABD	O

내포 항	부울 함수								선택
	m1	m5	m6	m7	m11	m12	m13	m15	
m(5,7,13,15) = BD		1		1			1	1	
m(1,5) = A'C'D	1	1							O
m(6,7) = A'BC			1	1					O
m(11,15) = ACD					1			1	O
m(12,13) = ABC						1	1		O
빈도수	1	2	1	2	1	1	2	2	

주 내포 항	부울 함수								선택
빈도수									$f=\phi$

$$F_6(A,B,C,D) = ABC' + ACD + A'BC + A'C'D$$

본 논문에서 거론된 실험데이터에 대해 Quine-McCluskey 법과 제안된 빈도수 기반 알고리즘의 수행횟수 비교는 표 4에 요약되어 있다.

표 4. 알고리즘 성능 비교
Table 4. Compare with Algorithm Performance

문제	주 내포항 수	수행횟수	
		Quine-McCluskey 알고리즘	빈도수 기반 제안된 알고리즘
F_1	4	24	3
F_2	6	24	3
F_3	4	-	3
F_4	6	-	4
F_5	9	-	4
F_6	5	-	4

F_1과 F_2에서 알 수 있듯이 주 내포항 개수가 4개나 6개임에도 불구하고, Quine- McCluskey 법은 24회의 계산을 수행해야 함에도 불구하고, 제안된 빈도수 기반 알고리즘은 단지 3회의 선택만을 수행하여 동일한 결과를 얻을 수 있는 장점을 갖고 있다. 제안된 빈도수 기반 알고리즘은 주 내포항의 개수가 4개에서 9개 범위에 있는 경우 단지 3회나 4회의 선택만으로 해를 찾을 수 있음을 알 수 있다.

V. 결론

디지털공학 분야에서 회로 최소화 문제는 설계된 복잡한 회로도에 대해 동일한 기능을 수행하는 보다 단순화 회로로 변환시켜 구성품의 개수도 감소시키고, 보다 소형화 시켜, 제품의 가격 경쟁력을 얻을 수 있는 방법으로 설계시 반드시 수행해야만 하는 과정이다.

대표적인 회로 최소화 방법에는 수기식으로 수행하는 카르노 맵과 전산화 할 수 있는 Quine-McCluskey 법이 있다. 본 논문은 Quine-McCluskey법의 주 내포 항을 찾고, 1차 필수 주 내포 항과 2차 필수 주 내포 항을 찾는 Petrick법을 보다 정확하고, 간단히 수행하는 방법을 제안하였다. 즉, 주 내포 항을 찾는 방법은 기존의 반복적인 방법 대신 항표를 이용하는 방법으로 대체시켜 보다 간단히 찾을 수 있었다. 또한, 1차와 2차 필수 주 내포 항을 찾는 방법은 집합피복을 찾는 방법을 적용하였다.

3-변수와 4-변수 실험 데이터들에 대해 제안된 방법을 Quine-McCluskey 방법과 비교시 제안된 방법이 보다 정확하고, 간단히 주 내포 항을 결정하고, 회로를 최소화하는 1차와 2차 필수 주 내포 항을 결정할 수 있었다.

참고문헌

[1] V. Kabanets and J. Y. Cai, "Circuit Minimization Problem," Proceedings of 32nd Symposium on Theory of Computing, Portland, Oregon, USA, pp. 73-79, Jun. 2000.

[2] M. Karnaugh, "The Map Method for Synthesis of Combinational Logic Circuits," Transactions of the American Institute of Electrical Engineers, part I, Vol. 72, No. 9, pp. 593-599, Nov. 1953.

[3] N. Sarkar, K. Petrus, and H. Hossain, "Software Implementation of the Quine-McCluskey Algorithm for Logic Gate Minimisation," Proceedings of the NACCQ, pp. 375-378, Napier, New Zealand, Jul. 2001.

[4] R. K. Brayton, G. D. Hachtel, C. McMullen, and A. L. Sangiovanni-Vincentelli, "Logic Minimization Algorithms for VLSI Synthesis," Springer, 1984.

[5] N. V. Vinodchandran, "Nondeterministic Circuit Minimization Problem and Derandomizing Arthur-Merlin Games," International Journal of Foundations of Computer Science, Vol. 16, No. 6, Dec. 2005.

[6] R. Siggh, A. Arora, G. Singh, and J. Malhotra, "Circuit Minimization in VLSI Using PSO & GA Algorithms," International Journal of Engineering Trends and Technology, Vol. 3, No. 1, pp. 43-46, Feb. 2012.

[7] M. Morrison and N. Ranganathan, " A Novel Optimization Method for Reversible Logic

Circuit Minimization," IEEE Computer Society Annual Symposium on VLSI (ISVLSI), pp. 182-187, Aug. 2013.

[8] S. R. Petrick, "A Direct Termination of the Irredundant Forms of a Boolean Function from the Set of Prime Implicants," Technical Report AFCRC-TR-56-110, Air Force Cambridge Res. Center, Cambridge, MA, USA, 1956.

저 자 소 개

이 상 운(Sang-Un, Lee)

1983년~1987년 :
한국항공대학교 항공전자공학과 (학사)
1995년~1997년 :
경상대학교 컴퓨터과학과 (석사)
1998년~2001년 :
경상대학교 컴퓨터과학과 (박사)
2003.3~현 재 :
강릉원주대학교 멀티미디어공학과 부교수
관심분야 : 소프트웨어 프로젝트 관리,
소프트웨어 개발 방법론,
소프트웨어 신뢰성,
그래프 알고리즘
e-mail : sulee@gwnu.ac.kr