Journal of The Korea Society of Computer and Information
Vol. 20, No. 4, April 2015

www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2015.20.4.095

빈도수 기반 주 내포 항 선택과 삭제 알고리즘을 적용한 회로 최소화

이 상 운*

A Selection-Deletion of Prime Implicants Algorithm Based on Frequency for Circuit Minimization

Sang-Un Lee *

요 약

본 논문은 회로 최소화 문제를 간단하게 풀 수 있는 알고리즘을 제안하였다. 회로 최소화 문제는 수기식 방법인 카르노 맵과 전산화가 가능한 휴리스틱 방법인 Quine-McCluskey 알고리즘이 있다. 그러나 Quine- McCluskey 알고리즘은 변수 개수 n이 증가하면 $3^n/n$의 메모리와 수행횟수가 요구되는 단점을 갖고 있다. 제안된 방법은 빈도수에 기반하여 내포 항 표를 이용하여 주어진 부울 함수의 최소 항을 포함하는 주 내포 항을 빠르게 추출하는 방법을 적용하였다. 추출된 주 내포 항들 중에서 중복 선택된 여분의 주 내포 항을 빈도수를 적용하여 제거하는 방법을 제안하였다. 제안된 알고리즘은 비록 변수 개수 n이 증가하여도 다항시간으로 회로를 최소화시킬 수 있는 해를 구할 수 있는 장점을 갖고 있다. 제안된 알고리즘을 3-변수와 4-변수의 다양한 사례들에 적용한 결과 해를 빠르고 정확하게 구할 수 있었다.

▸ Keywords : 발생빈도, 최소 항, 내포 항, 주 내포 항, 필수 주 내포 항

Abstract

This paper proposes a simple algorithm for circuit minimization. There are currently two effective heuristics for circuit minimization, namely manual Karnaugh maps and computable Quine-McCluskey algorithm. The latter, however, has a major defect: the runtime and memory required grow $3^n/n$ times for every increase in the number of variables n. The proposed algorithm, however, extracts the prime implicants (PI) that cover minterms of a given Boolean function by deriving an implicants table based on frequency. From a set of the extracted prime implicants, the algorithm then eliminates redundant PIs again based on frequency. The proposed algorithm is therefore capable of minimizing circuits polynomial time when faced with an increase in n. When applied to various 3-variable and 4-variable cases, it has proved

•제1저자 : 이상운
•투고일 : 2015. 01. 21, 심사일 : 2015. 02. 16, 게재확정일 : 2015. 03. 30.
* 강릉원주대학교 멀티미디어공학과 (Dept. of Multimedia Eng., Gangneung-Wonju National University)

to swiftly and accurately obtain the optimal solutions.

▸ Keywords : Frequency, Minterm, Implicant, Prime implicant, Essential prime implicant

I. 서 론

어떠한 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 생산의 비용과 시간을 절약할 수 있는 경제적 측면에서 매우 요구되는 과정이다. 이를 회로 최소화 문제 (circuit minimization problem, CMP)라 한다.

부울 대수 (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 휴리스틱 방법인 Quine-McCluskey 알고리즘이 제안되었으며, 이후에는 보다 효율적이면서 간단한 알고리즘에 대한 연구가 거의 수행되지 않고 있다. 따라서, 디지털공학 분야에서는 카르노 맵이나 Quine-McCluskey 알고리즘을 일반적으로 활용하고 있는 실정이다[2,3].

본 논문에서는 Quine-McCluskey 알고리즘과 근본적으로 다른 접근법으로 n 개 변수에 대한 부울 함수의 가능한 내포항인 내포 항 표 (implicants table, IT)을 이용하여 주어진 부울 함수의 모든 원소를 커버할 수 있는 내포 항들 중에서 주 내포 항을 추출 (선택)하고, 추출된 주 내포 항들 중에서 중복된 주 내포 항을 빈도수를 적용하여 삭제하는 방법을 제안하였다. 2장에서는 Quine-McCluskey 알고리즘을 고찰한다. 3장에서는 제안된 빈도수 기반 선택과 삭제 알고리즘을 제안한다. 4장에서는 제안된 알고리즘을 다양한 사례들에 적용하여 본다.

II. 관련 연구와 문제점

본 논문에서는 회로 최소화 문제의 해를 얻을 수 있는 알고리즘을 대상으로 한다.

카르노 맵은 주어진 부울 함수를 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 없는 단점을 갖고 있다.

n개의 변수는 2^n개의 최소항 (minterms)을 가질 수 있다. n개의 변수에 대한 주어진 부울 함수에서 나타난 최소항의 개수를 k라 하면, $k<2^n$이 된다. 이 부울 함수로부터 $2^i(i=1,2,\cdots,n-1)$ 개로 그룹을 만들어 변수들을 단순화 시킬 수 있으며, 이를 내포 항 (implicant)이라 한다.

Quine-McCluskey 알고리즘[3]은 주어진 부울 함수에 대한 내포 항을 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 $3^n/n$까지 발생할 수 있어 알고리즘 수행시간가 메모리는 기하급수적으로 증가하여 효율적이지 못한 단점을 갖고 있다.

보다 근본적으로 다른 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 자동적으로 출력하도록 프로그램이 되어 있다. 이 외에도 Vinodchandran[5], Siggh et al.[6], Morrison과 Ranganathan[7]의 연구결과가 있었지만 Quine-McCluskey 알고리즘보다 간단한 결과는 얻지 못하였다.

따라서, 본 장에서는 Quine-McCluskey 알고리즘을 상세히 살펴보고, 3장에서는 이와 다른 접근법을 제안한다.

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

Step 1. 주어진 부울 함수의 모든 주 내포 항 (PI)을 찾는다.
Step 2. Step 1에서 찾은 PI를 행으로, 부울 함수의 최소 항을 열로 하는 주 내포 항 차트 (prime implicant chart, PIC)를 사용하여 주어진 부울 함수의 최소 항들을 모두 커버하는 필수 주 내포 항 (EPI) 뿐 아니라 다른 PI들을 찾는다.

Quine-McCluskey 알고리즘[3]을 수행하는 과정을 보다 자세히 파악하기 위해 식 (1)의 부울 함수를 고찰해 보자.

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

식 (1)은 4 변수 A,B,C,D 입력에 대한 회로도이다. 여기서 m 은 최소항을 의미하며, 숫자는 최소항의 인덱스이다.

Quine-McCluskey 알고리즘의 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 항을 결정한다. 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. 그러나 이 방법은 주어진 부울 함수에 대해 어떤 최소항들의 그룹이 항들로 되는지를 알 수 없기 때문에 $2^1, \cdots 2^{n-1}$ 항을 결정하는데 어려움이 있다. 또한, $2^0, 2^1, \cdots 2^{n-1}$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.

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

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

여기서 '*'는 필수 주 내포 항 (EPI)를 의미한다. 주어진 부울 함수를 최소화시키기 위해서는 EPI를 모두 선택하고, 추가적으로 필요시 나머지 PI를 선택해야 한다. 이 과정을 수행하기 위해서는 EPI가 어떤 것인지 결정해야 하는 문제가 발생한다. Step 2를 수행하는 방법에는 시행착오법 (trial and error)과 보다 체계적인 Petrick 방법[8]이 있다.

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 합항 (sum term) $P_{i0} + P_{i1} + \cdots + P_{in}$의 PoS (product of sums)로 구성되며, P_{ij}는 열 j를 커버하는 행이다.
(4) 곱하고, $X + XY = X, XX = X, X + X = X$ 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.

$$\begin{aligned}
&w = m(4,12) = AB'C', x = m(8,9,10,11) = AB', \\
&y = m(8,10,12,14) = AD', z = m(10,11,14,15) = AC \\
&m4 = w, m8 = (x+y), m9 = x, m10 = (x+y+z), \\
&m11 = (x+z), m12 = (w+y), m14 = (y+z), \\
&m15 = z \\
&= w(x+y)x(x+y+z)(x+z)(w+y)(y+z)z \\
&\quad (x+y)(x+z) = xx + xy + xz + yz \\
&\qquad\qquad = x + x(y+z) + yz \\
&\qquad\qquad = x + x + yz \\
&\qquad\qquad = x + yz \\
&\quad (y+w)(y+z) = yy + yz + yw + wz \\
&\qquad\qquad = y + y(z+w) + wz \\
&\qquad\qquad = y + y + wz \\
&\qquad\qquad = y + wz \\
&= w(x+yz)x(x+y+z)(y+wz)z \\
&\quad (x+yz)(x+y+z) = xx + xy + xz + xyz + yyz + yzz \\
&\qquad\qquad = x + x(y+z) + xyz + yz + yz \\
&\qquad\qquad = x + xyz + yz \\
&\qquad\qquad = x + yz
\end{aligned}$$

$$= w(x+yz)x(y+wz)z$$
$$(x+yz)(y+wz) = xy+wxz+yyz+wyzz$$
$$= xy+wxz+yz+wyz$$
$$= xy+wxz+yz(1+w)$$
$$= xy+wxz+yz$$
$$= w(xy+wxz+yz)xz$$
$$= wxzxy+wxzwxz+wxzwz$$
$$= wxzyz+wxz+wxz$$
$$= wxz(yz+1+1)$$
$$= wxz$$

문자 개수가 가장 작은 항 선택 : wxz

$$F_1 = wxz = BC'D' + AB' + AC$$

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

Lee[10]는 Step 1에서 Quine-McCluskey 방법에 비해 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 제안하였다.

III. 빈도수 기반 선택-삭제 알고리즘

본 장에서는 주어진 부울 함수의 회로를 최소화하기 위해 빈도수에 기반하여 주 내포항을 선택하고, 중복된 여분의 주 내포항을 삭제하는 알고리즘을 제안한다.

제안된 알고리즘은 주 내포 항을 빠르게 추출하기 위해 사전에 부울 함수의 내포 항 표 (IT)를 얻어 이를 활용하는 방법을 적용한다.

변수 개수가 n인 부울 함수 $F(v_1, v_2, \cdots, v_n)$의 가능한 최소항의 개수는 2^n개 이다. 주어진 부울 함수의 최소항의 개수 k는 항상 $k < 2^n$으로 주어진다. 왜냐하면 $k = 2^n$개이면 이를 단순화하면 입력이 전혀 없는 회로가 되어 의미가 없기 때문이다.

[사전 준비]

부울 함수의 변수 개수 $n(n = 3,4\cdots)$에 대한 r-변수 항 $(r = 1,2,\cdots,n-1)$에 대한 가능한 조합 수 ${}_nC_r \times 2^r$개 각각에 대한 최소항의 묶음을 구한 내포 항 표를 작성한다. 단, 2^0인 내포 항은 제외한다. 여기서는 3-변수와 4-변수에 대한 내포 항 표를 제시하였다.

3-변수 $F(A,B,C)$

r-변수	항		
1-변수 2^1 항 묶음 (6가지)	m(0,1,2,3) = A′ m(0,1,4,5) = B′	m(0,2,4,6) = C′ m(1,3,5,7) = C	m(2,3,6,7) = B m(4,5,6,7) = A
2-변수 2^2 항 묶음 (12가지)	m(0,1) = A′B′ m(0,2) = A′C′ m(0,4) = B′C′ m(1,3) = A′C	m(1,5) = B′C m(2,3) = A′B m(2,6) = BC′ m(3,7) = BC	m(4,5) = AB′ m(4,6) = AC′ m(5,7) = AC m(6,7) = AB

4-변수 $F(A,B,C,D)$

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

빈도수에 기반한 회로 최소화 알고리즘은 다음과 같이 수행된다. 여기서, Step 2는 부울 함수 F의 모든 최소 항을 포함하고 있는 부분집합인 내포 항들을 선택하는 집합피복 문제 (Set Cover Problem) 알고리즘을 적용하였다. 집합피복 문제 [6]는 정확한 해를 다항시간으로 구하는 알고리즘이 제안되지 않고 있어 NP-완전 (NP-complete) 문제로 알려져 있다.

Step 1. $k\,(k < 2^n)$개의 최소 항으로 구성된 주어진 부울 함수 $F(v_1, v_2, \cdots, v_n)$에 대해 $2^i \le k < 2^{i+1}$를 계산하여, 내포 항 표에서 $r = n - i$인 r-변수 항을 결정하고, ${}_nC_r \times 2^r$개의 항 각각에 대해 해당 항의 모든 최소 항이 부울 함수의 진부분집합인 내포 항들을 추출한다.

(1) 만약, r-변수 항에서 U의 최소 항들에 대한 진부분집합 내포 항이 존재하지 않으면 $r+1$-변수 항에서 찾는다.

(2) 추출된 내포 항들에 대해 주어진 부울 함수의 각 최소 항 x에 대한 빈도수 f_x를 계산하여 $f_x = 0$인 최소 항 x가 존재하면, $r+1$-변수 항에서 $f_x = 0$인 최소 항 x의 조합을 모두 포함하거나 일부는 x를 포함하고, 나머지는 부울 함수의 최소 항인 내포 항을 선택한다. 최종적으로 선택된 내포 항들이 주 내포 항이 된다.

Step 2. 선택된 주 내포 항들에 대해 부울 함수의 각 최소 항 x에 대한 빈도수를 계산하고, $F = \phi$이 될 때까지 다음 과정을 반복 수행한다.

(1) $f_x = 1$을 포함하고 있는 주 내포 항들을 선택한다. 만약, $\forall_x, f_x \ge 2$이면 최대 원소 개수를 가진 주 내포 항들 중 임의의 주 내포 항을 선택한다.

(2) 선택된 주 내포 항들의 최소 항을 부울 함수에서 삭제한다.

(3) 부울 함수의 남은 최소 항들을 최대한 포함하고 있는 주 내포 항들을 선택한다.

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

표 3. 빈도수 기반 회로 최소화
Table 3. Circuit Minimization Based on Frequency

$F_1(A,B,C,D) = \Sigma m(4,8,9,10,11,12,14,15)$
$n=4, k=8, r=1$ (1-변수 항 : 없음)

구분	항	F의 진부분집합	항	F의 진부분집합
2-변수 항	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

1-변수 내포 항	부울 함수							
	m4	m8	m9	m10	m11	m12	m14	m15
m(8,9,10,11)		1	**1**	1	1			
m(8,10,12,14)		1		1		**1**	1	
m(10,11,14,15)				1	1		1	**1**
빈도수	**0**	2	**1**	3	2	**1**	2	**1**

3-변수 항	부울 함수							
	m4	m8	m9	m10	m11	m12	m14	m15
m(4,12)	**1**					1		

주 내포 항	부울 함수								선택
	m4	m8	m9	m10	m11	m12	m14	m15	
m(8,9,10,11) = AB'		1	**1**	1	1				O
m(8,10,12,14) = AD'		1		1		**1**	1		
m(10,11,14,15) = AC				1	1		1	**1**	**O**
m(4,12) = BC'D'	**1**					**1**			**O**
빈도수	**1**	2	**1**	3	2	2	2	**1**	

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

$$F_1(A,B,C,D) = m(4,12) + m(8,9,10,11) + m(10,11,14,15)$$
$$= BC'D' + AB' + AC$$

$F_1(A,B,C,D) = \Sigma m(4,8,9,10,11,12,14,15)$ 을 진부분 집합으로 갖는 1-변수 항 ([0,7] 또는 [8,15])은 존재하지 않으므로, 2-변수 항에서 구하면 m(8,9,10,11), m(8,10,12,14), m(10,11 ,14,15)로 m9, m12와 m15는 빈도수가 1로 3개의 2-변수 항이 골고루 갖고 있다. 또한, m4의 빈도수는 0으로 이는 3-변수 항에서 m(4,12)를 찾을 수 있다. m(4,12) 추가로 인해 m12의 빈도수는 2로 변경된다. 따라서, 빈도수는 1인 변수항만을 선택하면 m(8,10,12,14)이 삭제되고 m(8,9,10,11), m(10,11,14,15)과 m(4,12)만이 남게 되어 해를 구할 수 있었다. 여기서 채색된 항들은 빈도수가 "0" 또는 "1" 인 항을 표기하고 있다.

IV. 실험 및 결과 분석

본 장에서는 다음 부울 함수에 대해 제안된 알고리즘만을 적용하여 본다. 실험 방법은 각 부울함수에 대해 3장에서 제안된 알고리즘을 수행하는 방식을 표로 나타내면서, $f=\phi$이 될 때까지 f의 항을 포함하는 해당 주 내포항을 선택하는 방식으로 진행하였다.

$F_2(A,B,C) = \Sigma m(0,1,2,5,6,7)$
$F_3(A,B,C) = \Sigma m(1,3,4,5,6)$
$F_4(A,B,C,D) = \Sigma m(0,2,5,6,7,8,10,12,13,14,15)$
$F_5(A,B,C,D) = \Sigma m(0,2,3,4,5,6,7,8,9,10,11,12,13)$
$F_6(A,B,C,D) = \Sigma m(1,2,3,7,9,10,11,13,15)$
$F_7(A,B,C,D) = \Sigma m(1,3,4,5,10,11,12,13,14,15)$
$F_8(A,B,C,D) = \Sigma m(1,5,6,7,11,12,13,15)$

$F_2(A,B,C)$에 대해 제안된 알고리즘은 다음과 같이 수행된다. 2-변수 항에서 f의 진부분집합들을 선택한 결과 m(0,1), m(0,2), m(1,5), m(2,6), m(5,7), m(6,7)을 얻었으며, 모든 부울함수의 항 m0, m1, m2, m5, m6 ,m7의 빈도수가 2로 동일하므로, 임의의 2-변수 항 m(0,1)을 선택하고, m2, m5, m6, m7에 대해 다시 m(2,6)을 선택한 후 m5와 m7을 커버하는 m(5,7)을 선택하였다.

$F_2(A,B,C) = \Sigma m(0,1,2,5,6,7)$
$n=3, k=6, r=1$ (1-변수 항 : 없음)

구분	항	F의 진부분집합	항	F의 진부분집합
2-변수 항	**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

주 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**		**O**
m(5,7)				**1**		**1**	**O**
m(6,7)					**1**	**1**	
빈도수	**2**	**2**	**2**	**2**	**2**	**2**	

주 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. 2-변수 항에서 f의 진부분집합들을 선택한 결과 m(1,3), m(1,5), m(4,5)와 m(4,6)을 얻었으며, 이들로부터 빈도수는 1인 m3과 m6을 포함한 m(1,3)과 m(4,6)이 선택되었다. 이 두 항이 커버하지 못하는 m5에 대해 m(1,5)와 m(4m5)중 임의

로 m(1,5)가 선택되었다.

$F_3(A,B,C) = \Sigma m(1,3,4,5,6)$
$n=3, k=5, r=1$ (1-변수 항 : 없음)

구분	항	f의 진부분집합	항	f의 진부분집합
2-변수 항	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

주 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	

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

$F_3 = A'C + B'C + AC'$ or $A'C + AB' + AC'$

$F_4(A,B,C,D)$ 에 대해 제안된 알고리즘은 다음과 같이 수행된다. 2-변수 항에서 B′D′와 BD′가 선택되고, 다음으로 m6, m12와 m14중에서 m6과 m14를 갖는 CD′를 선택한 후 마지막으로 m12를 포함하고 있는 AD′가 선택되어 $F_4 = B'D' + CD' + BD' + AD'$ 를 얻었다.

$F_4(A,B,C,D) = \Sigma m(0,2,5,6,7,8,10,12,13,14,15)$
$n=4, k=11, r=1$ (1-변수 항 : 없음)

구분	항	F의 진부분집합	항	F의 진부분집합
2-변수 항	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

주 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		

주 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		

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

$F_4 = B'D' + CD' + BD' + AD'$

$F_5(A,B,C,D)$ 에 대해 제안된 알고리즘은 다음과 같이 수행된다. 2-변수 항으로부터 C′D′, B′C와 A′B를 선택한 결과 남은 m9와 m13을 갖는 AC′가 마지막으로 선택되어 $F_5 = C'D' + B'C + A'B + AC'$ 를 얻었다.

$F_5(A,B,C,D) = \Sigma m(0,2,3,4,5,6,7,8,9,10,11,12,13)$
$n=4, k=13, r=1$ (1-변수 항 : 없음)

구분	항	F의 진부분집합	항	F의 진부분집합
2-변수 항	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

주 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			O
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	

주 내포 항	부울 함수													선택
					m5	m6	m7		m9				m13	
m(0,2,4,6) = A′D						1								
m(0,2,8,10) = B′D														
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	

주 내포 항	부울 함수													선택
									m9				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	

$F_5 = C'D' + B'C + A'B + AC'$

$F_6(A,B,C,D)$ 에 대해 제안된 알고리즘은 다음과 같이 수행된다. 2-변수 항으로부터 B′D, B′C, CD와 AD를 선택한 결과 $F = \phi$로 $F_6(A,B,C,D) = B'D + B'C + CD + AD$ 를 얻었다.

$F_6(A,B,C,D) = \Sigma m(1,2,3,7,9,10,11,13,15)$
$n=4, k=9, r=1$ (1-변수 항 : 없음)

구분	항	F의 진부분집합	항	F의 진부분집합
2-변수 항	m(0,1,2,3) = A′B′	x	**m(3,7,11,15) = CD**	O
	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′	x
	m(1,3,9,11) = B′D	O	m(8,9,12,13) = AC′	x
	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**	O
	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

주 내포 항	부울 함수									선택
	m1	m2	m3	m7	m9	m10	m11	m13	m15	
m(1,3,9,11) = B′D	1		1		1		1			O
m(2,3,10,11) = B′C		1	1			1	1			O
m(3,7,11,15) = CD			1	1			1		1	O
m(9,11,13,15) = AD					1		1	1	1	O
빈도수	1	1	3	1	2	1	4	1	2	

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

$$F_6(A,B,C,D) = B'D + B'C + CD + AD$$

$F_7(A,B,C,D)$ 에 대해 제안된 알고리즘은 다음과 같이 수행된다. 3-변수 항에서 A′B′D를 선택한 후, 2-변수 항에서 BC′와 AC를 선택하여 $F_7(A,B,C,D) = AC + BC' + A'B'D$ 를 얻었다.

$F_7(A,B,C,D) = \Sigma m(1,3,4,5,10,11,12,13,14,15)$
$n=4, k=10, r=1$ (1-변수 항 : 없음)

구분	항	F의 진부분집합	항	F의 진부분집합
2-변수 항	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′**	O
	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′	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′	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**	O
	m(2,3,6,7) = A′C	x	**m(12,13,14,15) = AB**	O

주 내포 항	부울 함수										선택
	m1	m3	m4	m5	m10	m11	m12	m13	m14	m15	
m(4,5,12,13) = BC′			1	1			1	1			
m(10,11,14,15) = AC					1	1			1	1	
m(12,13,14,15) = AB							1	1	1	1	
빈도수	0	0	1	1	1	1	2	2	2	2	

$f_x = 0, x = m1, m3$, $m(1,3)$이 3-변수 항에 존재

구분	항	F의 진부분집합	항	F의 진부분집합
3-변수 항	**m(1,3) = A′B′D**	O		

주 내포 항	부울 함수										선택
	m1	m3	m4	m5	m10	m11	m12	m13	m14	m15	
m(4,5,12,13) = BC′			1	1			1	1			O
m(10,11,14,15) = AC					1	1			1	1	O
m(12,13,14,15) = AB							1	1	1	1	
m(1,3) = A′B′D	1	1									O
빈도수	1	1	1	1	1	1	2	2	2	2	

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

$$F_7(A,B,C,D) = AC + BC' + A'B'D$$

$F_8(A,B,C,D)$에 대해 제안된 알고리즘은 다음과 같이 수행된다.

2-변수 항으로부터 BD′를, 3-변수 항으로부터 A′C′D, A′BC, ACD와 ABC′를 선택하여 $F_8(A,B,C,D) = ABC' + ACD + A'BC + A'C'D + BD'$를 얻었다.

제안된 빈도수 기반 주 내포 항 선택과 여분의 주 내포 항 삭제 알고리즘은 Quine-McCluskey 알고리즘[3]의 주 내포 항 결정 방법에 비해 주 내포 항을 빠르게 추출할 수 있는 장점을 갖고 있다. 또한, Petrick 방법[5]에 의한 여분의 중복된 주 내포 항을 삭제하는 방법에 비해 빈도수를 적용하여 빠르고 쉽게 수행할 수 있는 장점을 갖고 있다.

$F_8(A,B,C,D) = \Sigma m(1,5,6,7,11,12,13,15)$
$n=4, k=8, r=1$ (1-변수 항 : 없음)

구분	항	F의 진부분집합	항	F의 진부분집합
2-변수 항	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
	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

주 내포 항	부울 함수								선택
	m1	m5	m6	m7	m11	m12	m13	m15	
m(5,7,13,15) = BD′		1		1			1	1	
빈도수	0	1	0	1	0	0	1	1	

$f_x = 0, x = m1, m6, m11, m12$
$m(1,6), m(1,11), m(1,12), m(6,11), m(6,12), m(11,12)$이 존재
1,6,11,12를 포함하고, 나머지 최소 항이 F의 진부분집합인 경우

구분	항	F의 진부분집합	항	F의 진부분집합
3-변수 항	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**	O
	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′	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	x
	m(5,7) = A′BD	x	m(14,15) = ABC	x

주 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	

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

$$F_8(A,B,C,D) = ABC' + ACD + A'BC + A'C'D + BD'$$

V. 결론 및 향후 연구과제

본 논문은 회로 최소화 문제의 해를 간단히 얻을 수 있는 알고리즘을 제안하였다.

회로 최소화 문제에 대해 1950년대에 수기식 방법인 카르노 맵과 전산화가 가능한 휴리스틱 방법인 Quine-McCluskey 알고리즘[3]이 제안된 이후 지금까지 보다 단순하며, 효율적이고, 빠른 방법이 제안되지 않고 있었다. Quine-McCluskey 알고리즘[3]의 경우 주 내포 항을 추출하는 과정이 복잡할 뿐 아니라 선택된 주 내포 항들 중에서 종복 선택된 여분의 주 내포 항이 있는지 여부는 체계적인 Petrick 방법[5]을 적용하였다. 그러나 Petrick 방법[5] 또한 매우 지루하고 복잡한 과정을 거쳐 오류를 유발시킬 수 있는 단점을 갖고 있었다.

본 논문은 단순히 사전에 작성된 내포 항 표로부터 빈도수에 기반하여 주 내포 항들을 추출 (선택)하는 간단한 방법을 제안하였으며, 선택된 주 내포항 들 중에서 여분의 중복된 주 내포 항이 존재하는지 여부를 빈도수를 이용하여 검증하고, 여분의 주 내포 항을 간단히 삭제할 수 있었다.

본 논문에서 제안된 알고리즘은 k-변수의 가능한 모든 항을 대상으로 F의 진부분집합을 가진 항들을 추출하고, 각 단일 항의 빈도수를 계산하는 방법을 적용하였다. 이로 인해, 변수 개수 n이 증가하면 k-변수 항들은 기하급수적으로 증가하는 단점을 갖고 있다. 추후에는 n이 증가하더라도 F의 진부분집합을 가진 k-변수 항을 쉽게 추출하는 방법을 연구하여 알고리즘을 보다 단순화 시킬 예정이다.

REFERENCES

[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, pp.1297-1308, 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.

[9] T. H. Cormen, C. E. Leiserson, R. L. Rivest, L. Ronald, and C. Stein, "Introduction to Algorithms," McGraw-Hill, pp. 1033-1038, 2001.

[10] S. U. Lee, "An Improved Quine-McCluskey Algorithm for Circuit Minimization," Journal of KSCI, Vol. 19, No. 3, pp. 109-117, Mar. 2014.

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 ~ 2015.3 :
강릉원주대학교 멀티미디어공학과 부교수
2015.4 ~ 현 재 :
강릉원주대학교 멀티미디어공학과 정교수
관심분야 : 소프트웨어 프로젝트 관리, 소프트웨어 개발 방법론, 소프트웨어 신뢰성, 그래프 알고리즘
e-mail : sulee@gwnu.ac.kr