Journal of the Korea Academia-Industrial cooperation Society
Vol. 12, No. 8 pp. 3715-3722, 2011

http://dx.doi.org/10.5762/KAIS.2011.12.8.3715

2-큐브 몫 행렬을 이용한 공통식 산출

권오형[1*]
[1]한서대학교 전자컴퓨터통신학부

Common Expression Extraction Using Two-cube Quotient Matrices

Oh-Hyeong Kwon[1*]
[1]**Division of Electroinc, Computer, and Communication, Hanseo University**

요 약 본 논문에서는 논리합성을 위한 부울 공통식 추출 방법을 제안한다. 제안하는 방법은 주어진 각 논리식들에서 제수/2-큐브 몫들과 2-큐브 몫 쌍들을 산출하고, 이들을 이용해서 2-큐브 몫 행렬을 만든다. 2-큐브 몫들과 행렬로부터 후보식들을 찾고, 다음 이 후보식들의 교집합에 의해 여러 논리식에서 사용되는 공통식을 산출한다. 실험 결과 제안하는 방법은 기존의 방법들보다 전체 논리식의 리터럴 개수를 줄일 수 있었다.

Abstract This paper presents a new Boolean extraction technique for logic synthesis. This method first calculates divisor/2-cube quotients, 2-cube quotient pairs, and 2-cube quotient matrices. Then we find candidates, which can be common sub-expressions, from 2-cube quotients and matrices. Next, candidate intersection provides the common sub-expressions for several logic expressions. Experimental results show the improvements in literal counts over the previous methods.

Key Words : 2-cube quotient, 2-cube quotient matrix, Boolean extraction

1. 서론

여러 논리식들에서 존재할 수 있는 공통식을 찾아 전체 논리식을 간략화하는 공통식 추출(extraction) 방법을 제안한다. 공통식 추출 방법은 일반적으로 여러 논리식을 동시에 나누는 제수(divisor)를 찾고, 이 제수를 포함하는 논리식의 일부를 새로운 변수로 치환하여 전체 논리식들을 간략화 한다. 이러한 공통식 산출에 대한 연구는 크게 대수 나눗셈을 적용한 방법과 부울 나눗셈을 적용한 방법으로 구분할 수 있다.

Brayton 등은 커널(kernel) 개념을 도입하여 다항 큐브 제수를 찾는 방법과 커널 교집합을 적용하여 공통식을 찾는 방법을 제안하였다[1-3]. Rajski와 Vasudevamurthy는 단항 큐브와 이 단항 큐브의 보수(complement)를 함께 고려하여 공통식을 찾는 방법[4]을 제안하였다. Singh와 Diwan는 변수 치환을 이용한 공통식 산출 방법[5]을 제안하였다. 이상의 방법들은 대수적 나눗셈에 기반을 둔 방법으로 부울 공리인 등멱법칙 ($a\ a = a$)과 보수법칙 ($a\ a' = 0$)을 적용하지 않는다. 따라서, 이들의 방법은 다음에 소개하는 부울 방법들과 비교해서 상대적으로 수행 속도는 빠르나 때때로 최적화된 결과를 산출하지 않는다. Hsu와 Shen은 부울공리를 활용한 coalgebraic division이라는 나눗셈 방법[6]을 고안하였으며, Yang[7]과 Wu[8-9]는 이진 결정 그래프 (Binary Decision Diagram, BDD)를 이용한 논리식 간략화 방법을 제안하였다. 최근에 Kwon은 Brayton의 방법을 확장하여 커널 쌍과 커널 교집합을 이용하여 부울 공통식을 산출하는 방법[10]을 제시하였다. 이러한 방법들 모두 선험 방식(heuristic method)에 의한 부울 나눗셈 기법으로 실험 결과를 보면 항상 대수적인 나눗셈[1-3] 보다 좋은 결과를 산출하지 못했다. 이러한 관점에서 본 논문은 [10,11]의 방법을 확장한 새로운 방법을 제안한 것이다. 제안한 방

*교신저자 : 권오형(ohkwon@hanseo.ac.kr)
접수일 11년 07월 14일 수정일 (1차 11년 07월 27일, 2차 11년 08월 08일) 게재확정일 11년 08월 11일

법은 주어진 논리식들에서 2개씩의 큐브를 선택하여 제수와 2-큐브 몫을 산출하고, 다시 몫들 사이에 등멱법칙과 보수법칙을 적용하여 몫 행렬을 만들어 2개 이상의 큐브로 구성된 공통 후보식(candidates)들을 찾도록 고안한 것이다. 그리고, 찾은 후보식들의 교집합을 구하여 부울 공통식을 산출하도록 하였다.

논문의 구성은 다음과 같다. 2절에서는 본 논문 서술에 필요한 용어들을 소개하고, 3절에서 본 논문의 핵심인 공통 논리식 산출 방법인 2-큐브 몫들과 행렬들을 이용한 방법을 소개한다. 4절에서 제시한 방법으로 벤치마크 회로들을 대상으로 실험한 결과를 보이고, 5절에서 결론을 제시한다.

2. 용어

본 논문의 공통식 산출 방법을 서술하는데 필요한 용어들에 대하여 서술한다.

정의 1: 변수(variable)는 부울 공간(Boolean space)에서 한 좌표를 나타내는 문자다. 리터럴(literal)은 변수 그 자체 또는 그의 보수(complement)다. 큐브(cube)는 리터럴들의 집합으로 만일 리터럴 a가 존재하면, 그의 보수 리터럴 a'을 포함하지 않는다. 단순식(expression 또는 sum-of-products(SOP) form)은 큐브들의 집합이다.

예 1: 문자 a는 변수이며, a와 a'은 리터럴이다. 리터럴 집합 $\{a,b\}$는 큐브이나 집합 $\{a,a'\}$은 큐브가 아니다. $\{\{a,b'\},\{b,c\}\}$는 단순식이다.

본 논문에서는 큐브와 단순식을 표현하는 경우 집합 표기와 보편적으로 사용되는 논리식 표기를 모두 사용한다. 따라서 큐브 $\{a,b\}$는 ab와 동일한 표현이며, $\{\{a,b'\},\{b,c\}\}$는 $ab'+bc$와 동일한 표현이다.

정의 2: 논리식 F의 서포트(support)는 논리식 F를 구성하는 변수들 집합으로 $sup(F)$로 표현한다. 논리식을 구성하는 모든 큐브들 간에 공통으로 사용되는 리터럴을 갖지 않은 경우 그 논리식은 큐브면제(cube-free) 되었다고 한다. 논리식이 어떤 큐브로부터 나누어졌을 때, 몫이 큐브면제라면 그 몫을 커널이라 한다. 이 때 커널을 산출한 큐브를 코커널(co-kernel)이라 한다.

예 2: 논리식 $F=a+bc'$의 경우, $sup(F)=\{a, b, c\}$. 논리식 $ab+c$는 큐브 면제된 경우이나, 논리식 $ab+ac$ 및 abc는 큐브 면제된 것이 아니다. $F=bc'd'e+ab'c+ab'e+ac'd'$은 다시 $F=bc'd'e+a(b'c+b'e+c'd')$으로 표현될 수 있으며, 이 때 $b'c+b'e+c'd'$은 코커널 a에 대한 커널이 된다.

정의 3: 논리회로는 비순환 방향 그래프(directed acyclic graph)로 표현되며, 각 노드 i는 변수 y_i를 나타내고, 논리식 F_{y_i}를 표현한다. 2개의 논리식 F와 G의 곱 FG는 $\{C_i \cup D_j | C_i \in F$와 $D_j \in G\}$를 의미한다. F와 G가 서로 독립 서포트(disjoint support)인 경우 FG는 대수 곱이고, 그 외의 경우 FG는 부울 곱이다. F/G는 가장 큰 큐브 집합인 몫 Q를 산출하여, 논리식 F를 $F=QG+R$로 표현 할 수 있다. 여기서, R은 나머지를 나타낸다. QG가 대수 곱인 경우, F/G는 대수 몫을, 그 외의 경우 F/G는 부울 몫이 된다. $F/G=Q\neq\varnothing$ 이고 Q가 대수 나눗셈에 의해 산출된 경우, G는 대수 제수(algebraic divisor)라 하고, 그 외의 경우 부울 제수(Boolean divisor)라 한다.

예 3: $(a+b)(c+d)=ac+ad+bc+bd$는 대수 곱이며, $(a+b)(a+c)=a+ab+ac+bc$는 부울 곱이다. $F=ad+abc+bcd$ 이고 $G=a+bc$ 가 주어진 경우를 보자. F/G의 결과로는 대수 몫 d와 나머지 abc가 산출된다. 이 때, $a+bc$는 대수 제수가 된다. $F=abg+acg+adf+aef+afg+bd+be+cd+ce$와 $G=ag+d+e$가 주어진 경우, $F=(af+b+c)(ag+d+e)$로 표현될 수 있다. 이 때, $af+b+c$는 부울 몫이며, $ag+d+e$는 부울 제수가 된다.

3. 공통 논리식 산출

대수 나눗셈에 의한 커널 기반 공통식 산출 방법[1-3]을 소개하고 다음 본 논문에서 제시하는 2-큐브 몫 행렬을 이용한 부울 공통식 산출 방법에 대하여 설명한다.

3.1 커널 기반 공통식 산출

Brayton과 McMullen은 각 논리식 별로 커널들을 찾고, 이들의 교집합을 찾아 대수 공통식을 찾는 방법을 제

시하였다.

예 4: 다음과 같이 논리식 F_0와 F_1이 주어졌다고 하자. $F_0 = ace + bce + de + g$, $F_1 = ad + bd + cde + ge$. 첫 번째 단계에서, F_0의 커널 집합 $K(F_0)$은 $K(F_0) = \{(a+b),\ (ac+bc+d),\ (ace+bce+de+g)\}$이며, F_1의 커널 집합은 $K(F_1) = \{(a+b+ce),\ (cd+g),\ (ad+bd+cde+ge)\}$. 두 번째 단계에서, F_0와 F_1의 커널로부터 공통식을 찾는다. 즉, $(a+b) \in K(F_0)$와 $(a+b+ce) \in K(F_1)$의 교집합 $a+b$를 얻게 된다. 그러면, 간략화된 다음의 논리식들이 산출된다.

$$F_0 = Gce + de + g$$
$$F_1 = Gd + cde + ge$$
$$G = a + b$$

3.2 2-큐브 몫 행렬을 이용한 공통식 산출

본 절에서 기술하는 부분이 논문에서 제안하는 핵심 부분이다. 각 논리식 별로 제수/2-큐브 몫, 2-큐브 몫 쌍, 그리고 2-큐브 몫 행렬을 만들어 공통식이 될 수 있는 후보식들을 산출한다. 다음, 후보식들의 교집합을 구해서 여러 논리식에서 공통으로 사용되는 논리식, 즉 공통식을 산출하여 여러 논리식에서 반복 사용된 부분을 줄이는 방법이다.

3.2.1 2-큐브 몫 쌍 산출

주어진 논리식에서 2개의 큐브를 선택하고 이 2개의 큐브들에서 공통 인수, 즉 공통 큐브를 찾는다. 이 때, 공통 큐브가 제수가 되며 이 제수로 2개의 큐브를 나눈 것이 몫이 된다. C를 제수 집합, Q를 2개의 큐브로 구성된 몫 집합이라 하자. 표기상 제수/몫 쌍을 괄호를 이용하여 표현하고, $c_i \in C$, $c_j \in C$, $q_i \in Q$, $q_i \in Q$ 이고 $i \neq j$라 하자. 그러면, (c_i, q_i), (c_j, q_j)는 대수 나눗셈에 의한 제수/몫 쌍을 표현한 것이다. 만일 $c_i \in q_j$, $c_j \in q_i$ 이고 $q_i q_j$가 주어진 논리식에 포함되면, (q_i, q_j)는 주어진 논리식에 포함되는 2-큐브 몫 쌍이라 한다.

예 5: 논리식 $F_0 = abcd + abfg + a'efh + cdh + fgh$로부터 표 1과 같이 각 큐브에 양의 값을 갖는 인덱스를 부여한다. 여기서 표 1은 2-큐브 몫 행렬 산출에 필요한 부분으로, 예 5에서는 사용되는 부분이 아니다. 다음 2개의 큐브들 사이의 공통인수를 추출하여 산출한 대수 나눗셈 결과를 표 2에 나열한다. 표 2에서는 제수와 몫을 열로 구분해서 표시한다. 표 2의 제 1열은 설명을 쉽게 하기 위해 부여한 행 번호다.

[표 1] 큐브 인덱스
[Table 1] Indexes of cubes

큐브 C_i	$abcd$	$abfg$	$a'efh$	cdh	fgh
인덱스 $index(C_i)$	1	2	3	4	5

[표 2] 제수와 2-큐브 몫
[Table 2] Divisors and 2-cube quotients

행	선택된 큐브	2-큐브 나눗셈	
		제수	2-큐브 몫
1	$C_1 \cap C_2$	ab	$cd + fg$
2	$C_1 \cap C_4$	cd	$ab + h$
3	$C_2 \cap C_3$	f	$abg + a'eh$
4	$C_2 \cap C_5$	fg	$ab + h$
5	$C_3 \cap C_4$	h	$a'ef + cd$
6	$C_3 \cap C_5$	fh	$a'e + g$
7	$C_4 \cap C_5$	h	$cd + fg$

표 2에서 행1와 행4의 제수와 몫을 보면 행1의 제수 ab가 행4의 몫 $ab+h$에 포함되고, 다시 행4의 제수 fg가 행1의 몫 $cd+fg$에 포함된다. 또한 $(cd+fg)(ab+h) \subset F_0$이 된다. 같은 방법으로 행2와 행5로부터 2-큐브 몫 쌍을 산출할 수 있다. 이를 정리하면 표 3과 같다.

[표 3] 2-큐브 몫 쌍
[Table 3] 2-cube quotient pairs

선택된 2개 행	2-큐브 몫 쌍
행1, 행4	$(cd+fg)(ab+h)$
행2, 행5	$(ab+h)(a'ef+cd)$

3.2.2 2-큐브 몫 행렬과 후보식 산출

2-큐브 몫 행렬은 제수/2-큐브 몫과 2-큐브 몫 쌍들을 이용해서 만든다. 행렬의 행은 제수에 해당하는 큐브, 그리고 2-큐브 몫 쌍에 포함되는 큐브에 대응된다. 행렬의 열은 2-큐브 몫을 구성하는 각 큐브 당 하나의 열이 배당

된다. 2-큐브 몫 행렬을 M으로 표기하고, CR을 행에 대응되는 큐브들의 집합, CQ를 열에 해당하는 큐브들의 집합으로 표기 하자. 그리고 α_i, β_j를 각각 행렬 M의 i 번째 행, j번째 열을 나타낸다고 하면 $\alpha_i \in CR$, $\beta_j \in CQ$ 이다. 이 때, 행렬 M의 원소 $M(\alpha_i, \beta_j)$는 다음과 같은 식(1)에 따라 원소 값이 결정된다.

$$M(\alpha_i,\beta_j)=\begin{cases} index(\alpha_i\beta_j) & : \alpha_i \in CR, \\ & \quad \beta_j는 \alpha_i로 나눈 몫에 속하는 큐브 \\ index(c) & : \alpha_i와 \beta_j는 2-큐브 몫 쌍에 \\ & \quad 포함되는 큐브, c=\alpha_i\beta_j \neq 0 \\ * & : \alpha_i\beta_j = 0, \\ & \quad \alpha_i와 \ \beta_j는 \ 2-큐브 몫 \ 쌍의 큐브 \\ 0 & : 그 \ 외의 \ 경우 \end{cases} \quad (1)$$

예 6: 예 5의 $F_0 = abcd + abfg + a'efh + cdh + fgh$에 대한 표 1, 표 2와 표 3을 이용해서 2-큐브 몫 행렬 M을 만든다. 행렬의 행들은 표 2의 제수에 해당하는 큐브들과 표 3의 2-큐브 몫 쌍에 포함되는 큐브들에 대응되도록 한다. 행렬의 열은 표 2의 2-큐브 몫에 포함되는 큐브들에 대응되도록 한다. 그러면, 행렬의 행은 집합 {ab, cd, f, fg, h, fh, $a'ef$}의 각 원소에 대응된다. 열은 집합 {cd, fg, ab, h, abg, $a'eh$, $a'ef$, $a'e$, g}의 각 원소에 대응된다. 산출된 행렬은 표 4와 같다. 표 4의 행렬에서 첫 번째 행과 첫 번째 열에 해당하는 $M(1,1) = M(ab, cd)$의 원소 값은 표 1에서 $index(abcd) = 1$이 된다. 또한 첫 번째 행과 일곱 번째 열에 해당하는 $M(1,7)$=$M(ab, a'ef)$의 원소 값은 $(ab)(a'ef) = 0$이기 때문에 $M(1,7)$=* 가 된다. 나머지 원소들에 대해서도 같은 방법으로 식 (1)을 적용하여 행렬의 값이 결정된다. 행렬의 빈칸은 0이 입력된 부분을 나타낸다.

[표 4] F_0의 2-큐브 몫 행렬
[Table 4] 2-cube quotient matrix for F_0

행 \ 열	cd	fg	ab	h	abg	$a'eh$	$a'ef$	$a'e$	g
ab	1	2					*		
cd			1	4					
f					2	3			
fg			2	5					
h	4	5					3		
fh								3	5
$a'ef$			*	3					

후보식은 주어진 식으로부터 산출된 제수와 2-큐브 몫의 쌍들과 행렬의 프라임 사각형에 해당하는 논리식들이다. 사각형과 프라임 사각형에 대한 정의를 다음에 서술한다. 프라임 사각형을 찾는 알고리즘은 본 논문의 선행결과인 [11]에 소개되어 있으며, 지면 관계상 본 논문에서는 생략한다.

정의 4: 2-큐브 몫 행렬 M에서 사각형은 행과 열의 부분 집합 (R, C)이며 다음 조건을 갖는다. $\alpha, \gamma \in R$ 및 $\beta, \delta \in C$에 대하여 $M(\alpha, \beta) \neq 0$이며 $\alpha \neq \gamma$이고 $\beta \neq \delta$인 경우 $M(\alpha, \beta) > 0$ 이고 $M(\gamma, \delta) > 0$ 이면 $M(\alpha,\beta) \neq M(\gamma,\delta)$. 프라임 사각형은 다른 사각형에 포함되지 않는 사각형이다. 사각형이나 프라임 사각형에 포함되는 행들이나 열들은 서로 인접할 필요는 없다.

예 7: 예 5의 논리식 F_0에 대하여 프라임 사각형을 산출해 보자. 표 4로부터 프라임 사각형 집합은 { ({ab, h}, {cd, fg, $a'ef$}) }이 되며, 편의상 회색으로 표시하였다. 프라임 사각형을 나타내는 논리식은 $(ab+h)(cd+fg+a'ef)$가 된다. 따라서, 논리식 F_0에서 후보식들은 다음과 같다.

$(ab+h)(cd+fg+a'ef)$
$(cd)(ab+h)$
$(f)(abg+a'eh)$
$(fg)(ab+h)$
$(h)(a'ef+cd)$
$(fh)(a'e+g)$
$(h)(cd+fg)$

예 8: 다른 예로 논리식 F_1= abc + $abfg$ + $a'efh$ + ch + fgh로부터 2-큐브 몫 행렬은 표 5와 같다. 후보식들을 구하면 다음과 같다. 다음 식들에서 첫 번째의 것은 표 5의 회색으로 표시된 프라임 사각형으로부터 산출된 것이고, 나머지는 2-큐브 나눗셈에 의해 산출된 후보식들이다.

$(ab+h)(a'ef+c+fg)$
$(ab)(c+fg)$
$(c)(ab+h)$
$(f)(abg+a'ef)$
$(fg)(ab+h)$
$(h)(a'ef+c)$

$(fh)(a'e+g)$
$(h)(c+fg)$

[표 5] F_1의 2-큐브 몫 행렬
[Table 5] 2-cube quotient matrix for F_1

행 \ 열	c	fg	ab	h	abg	$a'ef$	$a'e$	g
ab	1	2						
c			1	4				
f					2	3		
fg			2	5				
h	4					3		
fh							3	5
$a'ef$			*	3				

3.2.3 여러 논리식에서 공통식 추출

각 논리식 별로 모든 후보식들이 산출되었으면, 후보식들의 교집합을 이용해서 공통식을 찾는다. 만일 공통식이 발견되면 새로운 변수를 도입하고 공통식을 포함하는 부분은 새로운 변수로 대치하여 전체 논리식들을 간략화 한다. 만일 공통식이 여러 개 발견되면 전체 논리식의 리터럴 개수를 가장 많이 줄이는 공통식을 선택한다. 공통식을 찾는 방법을 정리하면 다음과 같다. d_i, $(i=1,\cdots,n)$를 후보식이라 하고, $D=\{d_1, d_2, \cdots, d_n\}$는 후보식들의 집합이라 하자. 그러면, 후보식 교집합 $I(D)$는 다음과 같은 방법으로 산출된다. 후보식 집합 D에 대응하는 새로운 식 $IF(D)$를 만들고, $IF(D)$에서 제수들을 찾으면 바로 이것이 후보식들의 교집합이다. $IF(D)$를 만들 때, 후보식들을 구성하는 큐브들을 새로운 변수로 치환하고, 후보식을 새로운 변수들의 곱으로 나타낸다. 예로, 산출된 후보식 중에 하나가 $d_1=a'ef+fg+cd$, $d_2=a'ef+fg+h$라 하자. 그러면, $t_1=a'ef$, $t_2=fg$, $t_3=cd$, $t_4=h$로 각 큐브를 치환하면, 후보식 d_1은 $t_1t_2t_3$, 후보식 d_2는 $t_1t_2t_4$가 되어 $IF(D)= t_1t_2t_3+t_1t_2t_4$로 표현한다. 이때, $IF(D)$에서 제수들인 $Divisor(IF(D))$를 찾으면, 이 제수들이 바로 후보식 교집합 $I(D)$이다. 후보식 교집합을 찾는 구체적인 예를 다음에 소개한다.

예 9: 예 5와 예 8에서 사용된 2개의 논리식 F_0와 F_1에 대하여 후보식 교집합을 산출하자. 편의상 논리식들을 다음에 다시 기술한다.

$F_0 = abcd+abfg+a'efh+cdh+fgh$
$F_1 = abfg+abc+a'efh+fgh+ch$

본 예에서는 지면 관계상 2개의 논리식에서 산출된 후보식 일부 D에 대하여 후보식 교집합 $I(D)$를 산출하자.

$D=\{d_1, d_2, d_3\}$
$d_1 = ab+h$
$d_2 = cd+fg+a'ef$
$d_3 = c+fg+a'ef$

그러면, 위 식을 구성하는 각 큐브에 새로운 변수를 배당한다.

$t_1 = ab$
$t_2 = h$
$t_3 = cd$
$t_4 = fg$
$t_5 = a'ef$
$t_6 = c$

그러면, $IF(D)$는 새로운 변수들을 이용해서 $IF(D) = t_1t_2 + t_3t_4t_5 + t_4t_5t_6$ 가 된다. 이 $IF(D)$에서 제수들의 집합 $Divisor(IF(D))$를 산출하면 $Divisor(IF(D)) = \{t_4t_5\}$가 된다. 그러면, 후보식 교집합 $I(D)$는 $IF(D)$의 제수로 해석할 수 있다. 그래서 $I(D) = \{t_4+t_5\} = \{fg+a'ef\}$가 된다.

3.2.4 가중치 계산

후보식 교집합에 의해 다수 개의 공통식이 발견된 경우 주어진 여러 논리식들에서 가장 많은 리터럴들을 줄일 수 있는 공통식을 선택하는 것이 중요하다. 본 논문에서는 공통식 선택을 위해 가중치 부여 방법을 사용하였다. 공통식 q_i에 대한 가중치를 $weight(q_i)$로 표기하고, 식(2)와 같이 가중치를 배당한다. 식(2)는 SIS[3]에서 제시한 것과 동일하다.

$$weight(q_i) = (NF(q_i)-1)(L(q_i)-1)-1 \quad (2)$$

여기서, $NF(q_i)$는 공통식 q_i를 포함하는 논리식들의 개수이고, $L(q_i)$는 공통식 q_i 자체의 리터럴 개수다.

3.2.5 알고리즘

알고리즘은 후보식들 산출과 후보식 교집합 수행 과정들과 가중치가 가장 큰 공통식을 선택하여 새로운 변수로 주어진 논리식을 간략화 하는 과정으로 구성되어 있다. 전체 알고리즘은 다음과 같다.

알고리즘 : 부울 공통식 추출
입력 : 단순식 형태의 다변수 출력 논리식
출력 : 공통식이 추출된 다변수 출력 논리식
방법 :

단계 1: 각 출력별로 제수/2-큐브 몫인 후보식 집합 *D*을 산출;
단계 2: 2-큐브 몫 쌍 집합 *K* 산출;
단계 3: *D*와 *K*를 이용해서 2-큐브 몫 행렬을 만들고, 프라임 사각형을 찾아 프라임 사각형에 해당하는 논리식들을 후보식 집합 *D*에 추가;
단계 4: 후보식 집합 *D*로부터 공통식 집합 *I(D)*를 산출;
단계 5: 가장 가중치가 큰 2개의 공통식 q_i와 q_j를 선택하고 각각에 새로운 변수 Q_i와 Q_j를 부여;
단계 6: 각 출력에서 q_i와 q_j를 포함하는 부분을 새로운 변수 Q_i와 Q_j로 대치;
단계 7: 나머지 부분은 단일항 공통식 추출에 의해 간략화;

예 10: 예 9의 논리식 F_0와 F_1에 대하여 위 알고리즘을 적용한다. 예 5는 알고리즘의 단계 1-2의 결과를 보인 것이다. 예 6-8은 단계 3의 결과를 보인 것이다. 단계 4는 예 9과 같이 후보식 교집합을 산출한다. 그러면, $q_1 = fg + a'ef$, $q_2 = ab + h$를 얻는다. 즉, q_1은 2개의 식 F_0와 F_1에 포함되고, q_1은 5개의 리터럴로 되어 있기 때문에 $weight(q_1) = (2-1)*(5-1)-1 = 3$ 또한 $weight(q_2) = 1$이 산출되어, q_1과 q_2를 선택하고, $Q_1 = fg + a'ef$, $Q_2 = ab + h$로 새로운 변수 Q_1, Q_2를 도입한다. 단계 6에서 F_0의 $fg + a'ef$에 해당 하는 부분은 Q_1으로, $ab + h$에 해당 하는 부분은 Q_2로 대치하여 $F_0 = (Q_1 + cd)Q_2$로, 같은 방법으로 $F_1 = (Q_1 + c)Q_2$로 표현한다. 그러면 단계 6까지의 결과는 다음과 같게 된다.

$$F_0 = (Q_1 + cd)Q_2$$
$$F_1 = (Q_1 + c)Q_2$$

[표 6] 실험 결과
[Table 6] Experimental results

회로	입력 수	출력 수	SIS[3]		커널-커널[10]		제안방법	
			리터럴 수	시간 (초)	리터럴 수	시간 (초)	리터럴 수	시간 (초)
b12	15	9	124	0.1	118	0.1	117	0.2
rd53	5	3	77	0.2	71	0.2	71	0.3
rd73	7	3	176	0.4	172	0.5	165	0.7
rd84	8	4	243	0.2	234	0.3	232	0.3
con1	7	2	23	0.1	23	0.1	23	0.2
z4ml	7	4	70	0.2	63	0.2	61	0.5
cmb	16	4	70	0.1	70	0.1	72	0.2
vg2	25	8	107	0.1	105	0.2	107	0.4
decod	5	16	64	0.1	54	0.2	51	0.4
misex1	8	7	80	0.1	80	0.2	80	0.4
alu4	14	8	1755	2.0	1554	2.3	1534	3.1
sao2	10	4	203	0.3	192	0.5	190	0.7
e64	65	65	254	0.1	254	0.2	254	0.4
apex6	135	99	904	0.1	901	0.3	902	0.4
C880	60	26	702	0.1	640	0.3	624	0.5
C1355	41	32	1032	0.1	990	0.3	988	0.6
C1908	33	25	1469	0.1	980	0.3	972	0.6
C2670	233	140	1995	0.2	1511	0.3	1501	0.6
C5315	178	123	4355	0.2	3274	0.5	3254	1.2
C6288	32	32	4800	0.1	4702	0.5	4704	0.9
C7552	207	108	5968	0.1	4510	0.7	4503	1.1

$Q_1 = fg + a'ef$

$Q_2 = ab + h$

단계 7에서는 더 이상의 논리식들 사이에 공통식이 없기 때문에 결과의 변동이 없다. 따라서 최종적으로 15개의 리터럴을 갖는 논리식을 산출하게 된다. 만일 $Q_1 = fg + a'ef$에 대하여 인수분해를 한 결과 $Q_1 = f(g + a'e)$까지 고려하면 최종 결과는 14개의 리터럴로 논리식을 간략화하게 된다.

4. 실험 결과

제시한 알고리즘은 Pentium IV 1.4GHz CPU PC의 Linux 환경에서 실험하였다. 성능 비교를 위해 이전의 대수 공통식 추출 방법[1-3]과 부울 공통식 추출 방법[10]으로 산출한 결과들과 제안한 방법으로 산출한 결과를 비교하였다. 또 제안한 방법으로 산출한 논리식들이 주어진 논리식들과 동일한지 검사(equivalence checking)를 하기 위해서 제안한 방법으로 산출한 다단 논리식을 SIS를 이용해서 2단 논리식으로 다시 변경하여 주어진 논리식과 동일한지 검사를 하였다. 표 6은 실험 결과를 정리한 것이다. 표 6의 첫 번째 열은 벤치마크 회로[12]의 이름이고, 다음 2개의 열은 벤치마크 회로의 입력 수와 출력 수를 나타낸 것이다. 다음 2개의 열은 SIS의 수행 결과, 다음 2개의 열은 커널 쌍을 이용한 방법으로 산출한 결과들이다. 마지막 2개의 열은 본 논문에서 제안한 방법으로 수행한 결과를 보인 것이다. 실험 결과를 보면 제안한 방법은 cmb 회로를 제외한 모든 경우 대수 공통식 산출 방법보다 우수한 결과를 보였으며, 부울 공통식 산출 방법인 커널 쌍에 의한 방법과 비교해서 대부분의 회로에서 리터럴 개수를 줄이는 효과를 보였다. 특히, rd73과 rd84의 경우 커널-커널 쌍에 의한 것과 비교해서 제안한 방법으로 리터럴 개수를 많이 줄일 수 있었는데, 이 회로들이 대칭회로(symmetric logic circuit)로 2-큐브 쌍으로 구성된 공통식을 많이 포함하기 때문이다. 반면에, cmb 회로에서는 2-큐브 쌍의 산출이 거의 나타나지 않기 때문에 타 방법과 비교해서 보다 좋은 결과를 산출하지 못한 것으로 생각된다. 나머지 벤치마크 회로들에 대해서 제안하는 방법이 커널 쌍에 의한 방법보다 우수한 결과를 산출한 이유는 2-큐브 묶 행렬로부터 산출된 후보식들이 커널 쌍들보다 리터럴 개수를 줄이는 공통식들을 산출하였기 때문이다.

5. 결론

본 논문에서 제시한 방법은 2-큐브 묶 행렬을 산출하는데 부울공리를 적용하기 때문에 공통식을 산출하는데 자연히 부울 나눗셈이 적용되어 간략화된 논리식을 산출한다. 실험 결과에서 보였듯이 수행시간은 다소 늘어나지만 대부분의 회로에서 리터럴 개수를 줄일 수 있었다. 실용적인 측면에서 본다면 다른 최적화 도구들과 함께 제안한 방법을 사용하는 것이 좋을 것으로 생각한다. 즉, 빠른 수행 시간이 필요할 경우라면 대수 공통식 산출 방법을 적용하는 것이 좋을 것이며, 간략화된 논리식을 산출해야 한다면 본 논문에서 제시하는 방법을 사용하는 것이 좋을 것으로 본다. 또, 실험결과에서 보였듯이 일부 회로에서 커널 쌍에 의한 방법이 제안하는 방법보다 우수한 결과를 산출하기도 하기 때문에, 부울 공통식을 산출하는 경우에 제안하는 방법과 커널 쌍에 의한 방법을 함께 사용해서 보다 간략화된 논리식을 선택하는 것도 좋은 방법이라고 생각한다.

References

[1] R. K. Brayton and C. McMullen, “The Decomposition and Factorization of Boolean Epressions.” *Proc. ISCAS*, pp. 49-54, 1982.

[2] R. K. Brayton, R. Rudell, A. Sangiovanni-Vincentelli, and A. R. Wang, “ MIS: A Multiple-Level Logic Optimization System.” *IEEE Trans. CAD*, Vol. 6, No. 6, pp. 1062-1081, 1987.

[3] E. M. Sentovich, K. J. Singh, C. Moon, H. Savoj, R. K. Brayton, R. K., and A. Sangiovanni-Vincentelli, "Sequential Circuit Design Using Synthesis and Optimization." *Proc. ICCD*, pp. 328-333, 1992.

[4] J. Rajski and J. Vasudevamurthy, "The testability-preserving concurrent decomposition and factorization of Boolean expressions," *IEEE Trans. CAD*, Vol. 11, No. 6, pp. 778-79, 1992.

[5] V. K. Singh and A. A. Diwan, "Heuristic for decomposition in multilevel logic optimization," *IEEE Trans. VLSI*, Vol. 1, No. 4, pp. 441-445, 1993.

[6] W.-J. Hsu and W.-Z. Shen, "Coalgebraic division for multilevel logic synthesis," *Proc. of DAC*, pp. 438-442, 1992.

[7] C. Yang and M. Ciesielski, "BDS: A Boolean BDD-Based Logic Optimization System," *IEEE Trans. CAD,* Vol. 21, No. 7, pp. 866-876, 2002

[8] D. Wu and J. Zhu, "FBDD: A Folded Logic Synthesis System," *Proc. of International Conference on ASIC(ASICON),* pp. 746-751, 2005.
[9] D. Wu and J. Zhu, "BDD-based Two Variable Sharing Extraction," *Proc. of Asia and South Pacific Design Automation Conference(ASPDAC),* pp. 1031-1034, 2005.
[10] O.-H. Kwon, "Common Expression Extraction Using Kernel-Kernel pairs" Journal of the Korea Academia-industrial cooperation Society, to be published.
[11] O.-H. Kwon, B. T. Chun, "Boolean Factorization Using Two-cube Non-kernels," Journal of the Korea Academia -industrial cooperation Society, Vol. 11, No. 11, pp. 4597-4603, 2010.
[12] IWLS 2005 Benchmarks, http://iwls.org/iwls2005/benchmarks.html

권 오 형(Oh-Hyeong Kwon) **[정회원]**

- 1999년 2월 : 포항공과대학교 컴퓨터공학과 (컴퓨터공학박사)
- 1989년 7월 ~ 1993년 2월 : 전자통신연구원 연구원
- 1999년 3월 ~ 2003년 2월 : 위덕대학교 컴퓨터공학과 교수
- 2003년 3월 ~ 현재 : 한서대학교 전자컴퓨터통신학부 교수

<관심분야>
회로설계자동화, 논리합성