5주차 예비보고서

전공: 컴퓨터공학과/생명과학과 학년: 9학기 학번: 20181435 이름: 박다희

**1.**

드모르간(De-morgan) 정리는 NOT 연산과 AND연산 또는 OR연산과의 관계를 나타내는 정리로, 집합과 논리학 이론에서 중요하게 다루고 있는 개념 중 하나이다. 이 정리는 주어진 식 전체에 부정을 취하면 합집합은 교집합으로, 교집합은 합집합으로, OR연산은 AND연산으로, AND연산은 OR연산으로 바뀌고 각각의 변수에도 부정을 취하게 된다는 정리이다. 드모르간의 정리는 크게 두 가지 형태로 나뉘는데, 그 중 하나인 드모르간 제 1법칙은 NOT(A AND B) = (NOT A) OR (NOT B) 또는 이고, 또 다른 하나인 드모르간 제 2법칙은 NOT(A OR B) = (NOT A) AND (NOT B) 또는 이다. 수학 논리학이나 집합론에선 드모르간 제 1법칙은 이고, 드모르간 제 2법칙은 이다. 드모르간 정리를 사용하면 논리적으로 복잡한 식도 간단하게 바꿀 수 있고 더 잘 이해할 수 있다. 활용되는 분야는 불 대수, 논리 회로 설계 그리고 집합 연산 등으로 다양하게 있다.

**2.**

다양한 법칙을 이용하면 논리회로를 나타내는 논리식을 간소화 시킬 수 있다. 논리식을 간소화 시키면 동작 속도를 더 빠르게 만들 수 있고, 소비하는 전력을 감소시킬 수 있다는 장점이 있다. 다음은 논리식을 간소화하는 법칙들에 대한 설명이다.

1. 교환 법칙 : X +Y = Y+ X , X \*Y = Y\* X
2. 결합 법칙 : (X +Y)+Z = X +(Y+Z) , (X \*Y)\*Z = X \*(Y\*Z)
3. 분배 법칙 : X\*(Y+Z) = X\*Y+X\*Z , X+(Y\*Z) = (X+Y)(X+Z)
4. 동치 법칙 : X+X = X , X\*X = X
5. 흡수 법칙 : X+X\*Y = X , X\*(X+Y) = X
6. 부정 법칙 : (X’)’ = X
7. 보수 법칙 : X+X’ = 1 , X\*X’ = 0
8. 등멱 법칙 : X + X = X , X\*X = X
9. 경계 법칙 : X + 1 = 1 , X \* 0 = 0

(10s드모르간 법칙 : (X+Y)’ = X’\*Y’ , (XY)’ = Z’+Y’

(11) 항등원 법칙 : X + 0 = X , X\*1 = X

위와 같은 법칙들을 사용하여 간소화 시키는 예를 하나 들어보자면, 다음과 같다. 예를 들어 xyz + x’y’z + xy’z라는 식이 있을 때 분배 법칙을 사용하여 xz(y + y’) + x’y’z으로, 보수 법칙을 사용하여 xz + x’y’z으로, 분배 법칙을 2번 이어서 이용하여 z(x + x’y’)였다가 z(x + x’)(x + y’)로, 그리고 보수법칙과 분배 법칙 순으로 간소화를 하면 z(x + y’)에서 최종적으로 xz + y’z로 간추릴 수 있다. 이 논리식을 간소화하기 전의 논리 회로와 간소화하고 나서의 논리 회로를 비교해보면 다음과 같다.

텍스트, 스크린샷, 소프트웨어, 도표이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명

간소화하기 전의 논리회로가 왼쪽의 그림이고, 간소화 후의 논리회로는 오른쪽의 그림이다. 이렇게 논리식을 간소화하면 더 간단한 논리 회로를 얻을 수 있다.

**3.**

카르노맵은 복잡한 논리식을 시각적으로 보다 더 간단하고 직관적이게 나타내어 간소화 시키는 방법으로 작거나 중간 크기의 논리 회로를 설계할 때 주로 사용한다. 카르노맵은 진리표를 바탕으로 입력 값들과 출력 값들 사이의 관계를 나타낸다. 카르노맵을 만들려면 먼저 변수의 개수만큼 2를 거듭제곱해주고 그 수만큼 표를 만든 뒤에 진리표의 결과를 각각의 표에 대응시킨다. 그 다음 인접한 칸에 같은 출력 값이 있는 패턴을 찾고, 그 논리식을 간소화 시킨다. 이에 대한 간단한 예시를 들어보면, 먼저 A, B에 대해 F = A + B 라는 논리 함수의 진리표를 구하고, 이에 대한 카르노 맵을 차례로 그리면 다음과 같다.

|  |  |  |
| --- | --- | --- |
| A | B | F |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | A | 0 | 1 | |
| B |  |
| 0 | | 0 | | 1 |
| 1 | | 1 | | 1 |

이제 인접한 칸에서 같은 값을 가진 것들을 찾고 패턴을 파악해야 하는데, 이때 카르노맵에서는 값이 1인 칸을 2의 거듭제곱 수 만큼 씩 묶을 수 있기 때문에 이번 예시에서는 2개나 4개로만 묶을 수 있다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | A | 0 | 1 | |
| B |  |
| 0 | | 0 | | 1 |
| 1 | | 1 | | 1 |

1이 4개인 경우는 없어서 2개로 묶을 수 있는 경우를 빨간 박스와 파란 박스로 묶어보았다. 먼저 파란색 박스의 경우, 논리식은 AB’ + AB로 나타내지만 패턴을 찾아 간소화 해보면 다음과 같다. 파란색 박스의 패턴은 A의 값이 1이라는 것이고, 이 말은 출력 값이 1이 되려면 B의 값에는 영향을 받지 않고 A값이 1이어야 한다는 말이다. 따라서 파란색 박스에서의 출력 값은 B 또는 B’에 영향을 받지 않으므로 논리식으로 나타낼 때에 A로 나타낼 수 있고, 이는 원래의 논리식인 AB’ + AB을 분비법칙과 보수법칙을 사용하여 간소화한 논리식과 같은 결과가 나타남을 확인할 수 있다. 이처럼 카르노맵을 이용하면 보다 더 쉽게 논리식을 간소화할 수 있지만, 변수의 개수가 많아지면 카르노맵도 복잡해져 간소화 시키는 과정도 복잡해진다는 단점이 있다.

**4. Quine-McCluskey 최소화 알고리즘**

Quine-McCluskey 최소화 알고리즘은 카르노맵과 같이 논리식을 더 간단하게 나타내는 방법들 중 하나이다. 카르노맵은 변수의 개수가 많아지면 그 효율성이 떨어진다는 단점이 있어서 4개의 변수 이하일때만 사용하는 것이 효율이 좋은 반면, Quine-McCluskey 최소화 알고리즘은 변수의 개수가 4개가 넘어가더라도 논리식을 간소화하는데 효율성이 좋다. Quine-McCluskey 최소화 알고리즘은 먼저 값들을 인덱스 별로 구분한 뒤에 AB + AB’ = A를 이용하여 제거할 수 있는 변수들을 없앤 후에 남은 결과 값들을 주항으로 만드는 것이다. 그 다음에는 주항 표를 이용하여 주항 집합을 구하면 된다. 이처럼 Quine-McCluskey 최소화 알고리즘은 카르노맵과 다르게 표만 사용하기 때문에 컴퓨터로 코드식을 이용할 수 있으며 논리식의 최소 형태를 보장해준다. Quine-McCluskey 최소화 알고리즘의 예시는 다음과 같다. 만약 F = a’b’c’d’ + a’b’c’d + a’b’cd’ + a’bc’d + a’bcd’ + a’bcd + ab’c’d’ + ab’c’d + ab’cd’ + abcd’라는 식이 있을 때, a,b,c,d를 순서대로 묶어서 하나의 이진수라고 생각하면 F(a,b,c,d) = (0000, 0001, 0010, 0101, 0110, 0111, 1000, 1001, 1010, 1110), 곧 F(a,b,c,d) = (0,1,2,5,6,7,8,9,10,14)로 나타낼 수 있다. 이 이진수들을 1의 개수에 따라 그룹을 나눠서 표를 작성하면 다음과 같다.

|  |  |  |
| --- | --- | --- |
| 1의 개수 | Minterm | Binary |
| 0개 |  | 0000 |
| 1개 |  | 0001 |
|  | 0010 |
|  | 1000 |
| 2개 |  | 0101 |
|  | 0110 |
|  | 1001 |
|  | 1010 |
| 3개 |  | 0111 |
|  | 1110 |

이렇게 완성된 표에서 minterms를 서로 합칠 건데, 만약 두개의 항이 한 개의 비트만 차이가 나면 -로 나타낼 수 있고, 이 기호의 의미는 그 자리의 값이 무슨 값이든 상관 없다는 뜻이다. 더 결합하지 못하는 항이 생기면 그 항은 prime implicant이고 이를 바탕으로 표를 작성하면 다음과 같다.

|  |  |  |  |
| --- | --- | --- | --- |
| 1의 개수 | Implicant | Binary | Prime implicant |
| 0개 |  | 000- | X |
|  | 00-0 | X |
|  | -000 | X |
| 1개 |  | 0-01 | O |
|  | -001 | X |
|  | 0-10 | X |
|  | -010 | X |
|  | 100- | X |
|  | 10-0 | X |
| 2개 |  | 01-1 | O |
|  | 011- | O |
|  | -110 | X |
|  | 1-10 | X |

한 번 더 앞의 과정을 반복하면 다음과 같은 표가 나온다.

|  |  |  |
| --- | --- | --- |
| Implicant | Binary | Prime implicant |
|  | -00- | O |
|  | -0-0 | O |
|  | -00- | O |
|  | -0-0 | O |
|  | --10 | O |
|  | --10 | O |

여기서 중복되는 것을 제외하면 prime implicant는 -00-, -0-0, --10 이렇게 3종류이고, 앞선 표에서의 prime implicant로 01-1, 011-, 0-01 이렇게 3개를 더해서 총 6개의 prime implicant가 나왔다. 이를 표로 나타내면 다음과 같다.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |
| -00- | O | O |  |  |  |  | O | O |  |  |
| -0-0 | O |  | O |  |  |  | O |  | O |  |
| --10 |  |  | O |  | O |  |  |  | O | O |
| 0-01 |  | O |  | O |  |  |  |  |  |  |
| 01-1 |  |  |  | O |  | O |  |  |  |  |
| 011- |  |  |  |  | O | O |  |  |  |  |

이때 prime implicant중에서 다른 implicant에 속하지 않으면서 자신의 prime implicant에만 해당하는 요소를 노란 색으로 표시하였다. 이를 essential prime implicant라 하고, 또한 이 essential prime implicant가 포함하는 행과 열은 모두 삭제 가능하므로 남는 항은 01-1만 남게 된다. -00-은 b’c’이고, --10은 cd’, 01-1은 a’bd이므로 결과적으로 F = a’b’c’d’ + a’b’c’d + a’b’cd’ + a’bc’d + a’bcd’ + a’bcd + ab’c’d’ + ab’c’d + ab’cd’ + abcd’은 b’c’+cd’+a’bd로 간소화 시킬 수 있다.

**5. 기타 이론**

Boolean식은 각각의 변수들의 합 또는 곱으로 표현할 수 있는데 이를 Canonical Form이라고 한다. 그 종류에는 minterm과 maxterm이 있는데 이는 논리 함수를 간소화 시키고 표현할 때 사용된다. 먼저 minterm은 각 변수들의 곱으로 표현되는데 만약 n개의 변수가 있다면 2^n개의 서로 다른 minterm이 존재하게 된다. 가능한 모든 조합으로 결과를 도출해내는데 예를 들어 3개의 변수 a,b,c가 있다면 이들의 minterm은 xyz, x’yz, xy’z, xyz’, xy’z’, x’yz’, x’y’z, x’y’z’가 된다. Maxterm은 각 변수들의 합으로 표현되는데 n개의 변수가 있다면 2^n개의 maxterm이 존재하게 된다. 예를 들어 a, b, c 이렇게 3개의 변수가 있다면 이들의 maxterm은 x+y+z, x’+y+z, x+y’+z, x+y+z’, x’+y’+z, x’+y+z’, x+y’+z’, x’+y’+z’이다. Minterm과 maxterm은 서로 보수 관계이고, 이는 드모르간의 법칙을 사용하여 나타낼 수 있다.