5주차 예비보고서

전공: 물리학과 학년: 3학년 학번: 20191286 이름: 김나현

1. De Morgan의 정리에 대해 조사하시오.

드 모르간 정리는 영국의 수학자 Augustus De Morgan, 오거스터스 드 모르간의 이름을 딴 논리곱, 논리합, 부정 연산 간의 관계를 기술하여 정리한 것입니다. 교집합이 공집합이 아닌 두 개의 집합 A와 B가 있다고 가정할 때, A와 B의 논리곱 AB는 두 집합의 교집합 A∩B를 의미하고, A와 B의 논리합 A+B는 두 집합의 합집합 A∪B를 의미할 것입니다. 이때, 두 집합의 교집합의 여집합은 (A∩B)C으로 표현이 가능하고 이는 A의 여집합과 B의 여집합의 합집합, (A C ∪B C)으로 구할 수 있고 두 집합의 합집합의 여집합은 (A∪B)C으로 표현이 가능하고 이는 A의 여집합과 B의 여집합의 교집합, (A C ∩B C)으로 구할 수 있습니다. 이것은 중, 고등학교 수학 시간에 집합에 대해 배울 때 다룬 내용으로, 집합론에서 볼 수 있는 드 모르간의 정리입니다. 논리 회로에서도 이러한 드 모르간의 정리가 적용되는데 두 개의 입력 A와 B가 있다고 가정할 때, OR 연산에 NOT을 붙인 (A+B)’는 A’B’가 되고, 이는 A의 보수와 B의 보수의 AND 연산이라고 생각할 수 있습니다. 마찬가지로, AND 연산에 NOT을 붙인 (AB)’는 A’+B’가 되고, 이는 A의 보수와 B의 보수의 OR 연산으로 생각할 수 있습니다.

드 모르간의 법칙을 증명하기 위해서는 A=X+Y, A’=X’Y’이라는 가정과 a+a’=1이고 aa’=0이라는 사실을 이용해야 합니다. A+A’=1이므로 X+Y+X’Y’가 1이라는 사실과 AA’=0이므로 (X+Y)(X’Y’)=0이라는 사실을 증명하면 X’Y’=(X+Y)’라는 사실을 증명할 수 있습니다. 우선, X+Y+X’Y’는 분배 법칙과 교환 법칙에 의해 (X+Y+X’)(X+Y+Y’)로 변환할 수 있고, X+X’과 Y+Y’는 1이므로 X+Y+X’Y’= (1+X)(1+Y)=1·1=1입니다. 또한, (X+Y)(X’Y’)은 분배 법칙에 의해 (XX’Y’+X’YY’)가 되고 XX’와 YY’는 0이므로 (X+Y)(X’Y’)=0이 됩니다. 따라서, X+Y의 보수는 X’Y’라는 사실이 증명이 되었고, 드 모르간의 정리가 성립한다는 것을 확인할 수 있습니다.

1. 논리 회로의 간소화에 대해 조사하시오(예시 포함).

논리 회로를 간소화하기 전에 Boolean 연산에 사용되는 여러 정리에 대해 알아보도록 하겠습니다. 다음과 같은 정리들의 증명은 여기서 다루지 않을 것이지만 이미 증명이 된 것이기 때문에 사용할 때에는 증명 과정이 필요하지 않고 자유자재로 사용이 가능합니다.

우선, a(b+c)=ab+ac가 되고, a+bc=(a+b)(a+c)가 되는 것을 distributive 법칙이라고 하고, ab+ab’=a가 되고, (a+b)(a+b’)=a가 되는 것은 adjacency 법칙이라고 하며 a+a’b=a+b, a(a’+b)=ab가 되는 것은 simplification 법칙이라고 합니다. a+ab=a가 되는 것과 a(a+b)=a가 되는 것은 absorption 법칙이라고 합니다. 이러한 법칙 외에도 다음 <Figure 1>에 적힌 여러 정리를 사용하여 논리식을 변환하는 것이 가능합니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

<Figure 1> switching algebra

이러한 Boolean algebra를 이용하여 논리 회로를 단순화하는 과정을 알아보겠습니다. 만약 네 개의 입력에 대해 AB+A’CD+A’BD+A’CD’+ABCD와 같은 논리식이 주어지면 모든 게이트가 최대 두 개의 입력까지 가능하다고 가정하면, 14개의 AND, OR, NOT 게이트가 필요할 것임을 알 수 있습니다. 하지만 위의 식을 <Figure 1>의 boolean algebra를 이용하여 단순화하면

AB+ABCD+A’CD+A’CD’+A’BD=AB+AB(CD)+A’C(D+D’)+A’BD=AB(1+CD)+A’C+A’BD=AB+A’C+A’BD=AB+A’BD+A’C=B(A+A’D)+A’C=B(A+D)+A’C

가 되어 4개의 게이트만이 필요하게 됩니다.

1. 카르노 맵에 대해 조사하시오(예시 포함).

카르노 맵을 이해하기 위해서 Canonical form, Minterm과 Maxterm을 이해할 필요가 있습니다. Canonical Minterm이란 세 개의 입력 x, y, z가 있다고 가정할 때 진리표를 보고 x, y, z가 0, 0, 0일 때, 세 입력의 논리곱이 1이 되게 하기 위해 x’y’z’라고 표현하고, x, y, z가 1, 1, 1일 때는 xyz라고 표현하는 것입니다. 이때, 진리표의 순서대로 각 minterm은 m0, m1, m2,..,m7라고 합니다.

<Table 1> 3-input minterms, notation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x | y | z | Minterms | Notation |
| 0 | 0 | 0 | x’y’z’ | m0 |
| 0 | 0 | 1 | x’y’z | m1 |
| 0 | 1 | 0 | x’yz’ | m2 |
| 0 | 1 | 1 | x’yz | m3 |
| 1 | 0 | 0 | xy’z’ | m4 |
| 1 | 0 | 1 | xy’z | m5 |
| 1 | 1 | 0 | xyz’ | m6 |
| 1 | 1 | 1 | xyz | m7 |

이때, F1=m3+m5+m6+m7이라고 하면 F1은 위의 <Table 1>을 보고 F1=x’yz+xy’z+xyz’+xyz라고 표현할 수 있습니다. F1(x,y,z)는 minterm들의 논리합으로 표현할 수 있기 때문에 시그마라는 기호를 이용하여 F1(x,y,z)=Σ(3,5,6,7)이라고 나타낼 수 있습니다. 또한 F1의 보수 F1’은 드 모르간 정리에 의해 F1’=(x’yz+xy’z+xyz’+xyz)’=(x’yz)’(xy’z)’(xyz’)’(xyz)’=(x+y’+z’)(x’+y+z’)(x’+y’+z)(x’+y’+z’)이라고 나타낼 수 있고 x+y+z와 같은 term을 maxterm이라고 하므로 이는 maxterm들의 논리곱이라고 생각할 수 있습니다.

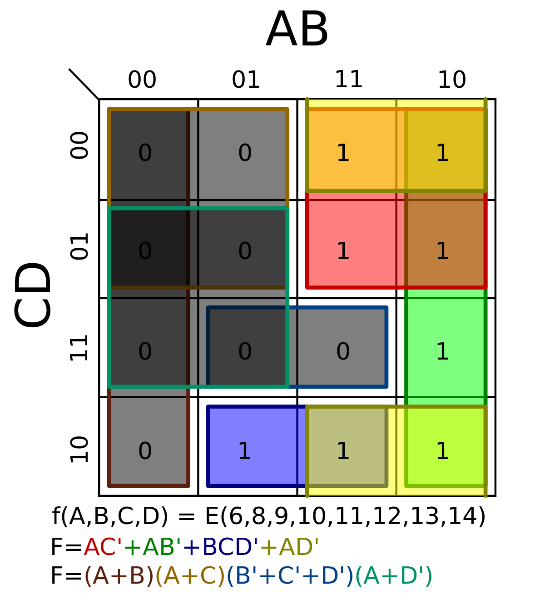
이제 세 개의 입력, x, y, z의 논리합을 0이 되게 하는 표현인 maxterm에 대해 알아보도록 하겠습니다.

<Table 2> 3-input maxterms, notation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x | y | z | Minterms | Notation |
| 0 | 0 | 0 | x+y+z | M0 |
| 0 | 0 | 1 | x+y+z’ | M1 |
| 0 | 1 | 0 | x+y’+z | M2 |
| 0 | 1 | 1 | x+y’+z’ | M3 |
| 1 | 0 | 0 | x’+y+z | M4 |
| 1 | 0 | 1 | x’+y+z’ | M5 |
| 1 | 1 | 0 | x’+y’+z | M6 |
| 1 | 1 | 1 | x’+y’+z’ | M7 |

위의 <Table 2>와 <Table 1>을 비교해보면 세 개의 입력의 논리합을 0으로 만드는 표현과 세 개의 입력의 논리곱을 1로 만드는 표현이 서로 보수 관계임을 생각해볼 수 있습니다. 드 모르간의 법칙에 의해 x, y, z의 입력이 0, 0, 0일 때, 논리곱을 1로 만드는 minterm은 x’y’z’이지만 논리합을 0으로 만드는 maxterm은 (x’y’z’)’=x+y+z이라는 사실을 알 수 있습니다. 따라서, 위에서 예시로 든 F1의 보수 F1’=(x’yz+xy’z+xyz’+xyz)’=(x’yz)’(xy’z)’(xyz’)’(xyz)’=(x+y’+z’)(x’+y+z’)(x’+y’+z)(x’+y’+z’)은 F1’=M3M5M6M7이며 F1’=m0+m1+m2+m4, F1=M0M1M2M4라는 사실을 알 수 있습니다. Maxterm을 이용한 논리 식의 표현은 논리곱을 이용하여 나타내므로 라는 문자를 사용하여 F1’=(3,5,6,7)이라고 나타냅니다.

카르노 맵(Karnaugh map)은 minterm을 이용하여 논리식을 간단히 만들기 위한 수단입니다. 다음 <Figure 2>를 보면 AB가 적힌 가로축은 왼쪽부터 A’B’, A’B, AB, AB’를 나타낸 것이고 CD가 적힌 세로축은 위에서부터 C’D’, C’D, CD, CD’를 의미합니다. 따라서 이 논리식의 결과를 F라고 가정하면 F=ABC’D’+AB’C’D’+ABC’D+AB’C’D+AB’CD+A’BCD’+ABCD’+AB’CD’를 통해 구할 수 있습니다. 하지만 이렇게 일일이 카르노 맵의 값이 1인 minterm을 더하지 않고 아래 <Figure 2>의 빨간 정사각형처럼 네 부분의 minterm을 한 번에 AC’라고 나타낼 수 있습니다. 이는 ABC’D’+AB’C’D’+ABC’D +AB’C’D=ABC’(D+D’)+AB’C’(D+D’)=ABC’+AB’C’=AC’(B+B’)=AC’를 정리한 것으로, minterm을 모두 더하고 식에서 정리하는 것이 아닌 카르노 맵을 보고 값이 1인 개의 묶음을 한 번에 나타내면 더욱 효율적으로 논리 식을 단순화시킬 수 있습니다.



<Figure 2> 4 input 카르노 맵

위의 <Figure 2>를 이용하여 단순화된 논리 식을 구해보자면 앞서 말한 것처럼 빨간 정사각형을 의미하는 AC’, 오른쪽에 길다란 초록 직사각형을 의미하는 AB’, 오른쪽 위, 아래의 노란 정사각형을 의미하는 AD’, 보라색 직사각형을 의미하는 BCD’의 합으로 F를 표현할 수 있습니다.

카르노 맵에서 값이 1인 개의 묶음을 implicant라고 하고, 이 중 다른 implicant에 포함되지 않는 묶음을 Prime implicant라고 합니다. Prime implicant 중 다른 implicant에 속하지 않고 해당 묶음에만 속하는 1이 적어도 하나 이상 있는 경우, 이 prime implicant을 essential prime implicant이라고 합니다.

카로노 맵을 사용할 때는 가능한 한 가장 큰 정사각형, 또는 직사각형으로 묶고 그 사각형에 포함되지 않는 값들을 포함하는 부수적인 정, 직사각형을 만드는 것입니다. 즉, 모든 prime implicant를 찾고 essential prime implicant를 사용하여 식을 적고 최소한의 non-essential prime implicant를 사용하여 cover되지 않은 모든 minterm들을 cover하는 것입니다. 이런 식으로 카르노 맵을 이용하여 논리 식을 적는 것이 카르노 맵을 이용할 수 있는 가장 효과적인, optimal한 방법이기 때문입니다.

카르노 맵을 이용하면 짝수 개의 입력뿐만 아니라, 홀수 개의 입력에 대해서도 논리 식을 효과적으로 단순화할 수 있고 4개 이상, 즉 5개나 6개의 입력이 들어와도 큐브 모양의 입체적인 카르노 맵을 만들어 간단한 논리 식으로 정리할 수 있습니다. 하지만 카르노 맵은 직접 그려 implicant를 찾아야 하므로 사실상 7개 이상의 입력에 대해서는 카르노 맵을 이용하여 단순화된 논리 식을 구할 수 없고, 사람이 일일이 해야 된다는 단점이 있습니다.

1. Quine-McCluskey 최소화 알고리즘에 대해 조사하시오.

최소 SOP(minimum sum of products)를 알기 위해 Prime implicant를 찾는 방법에는 카르노 맵 이외에도 Quine-McCluskey 방법 등 여러 가지 방법이 존재합니다. 현실적으로 입력의 개수가 6개보다 많을 때 사용이 힘들었던 카르노 맵의 방법과 달리, Quine-McCluskey 방법(이하 QM 방법)을 이용하면 입력의 개수가 6개보다 많을 때도 사용이 가능합니다. QM 방법은 minterm의 합을 이용하여 입력이 1인 개수에 따라 인덱스를 매겨 그룹을 만들고, 각 그룹 간에 1비트만 다른 항들을 찾아서 단순화(제거)하여 단순화에 사용된 항들을 표시하는 과정을 통해 이루어집니다. 예시를 통해, 구체적으로 설명해보겠습니다. F(w,x,y,z)=Σm(0, 4, 5, 7, 8, 11, 12, 15)이라고 하면 F를 SOP로 나타내기 위한 minterm들은 w’x’y’z’(0000, m0), w’xy’z’(0100, m4), w’xy’z(0101, m5), w’xyz(0111, m7), wx’y’z’(1000, m8), wx’yz(1011, m11), wxy’z’(1100, m12), wxyz(1111, m15)입니다. 이들을 1인 값의 개수에 따라 그룹을 매겨보면 다음과 같습니다.

A 0000

B 0100, C 1000

D 0101, E 1100

F 0111, G 1011

H 1111

이제 이 그룹을 이용해서 인접한 그룹끼리 단순화해보면

A+B=J=0-00

A+C=K=-000

B+D=L=010-

B+E=M=-100

C+D=NONE

C+E=N=1-00

D+F=O=01-1

D+G=NONE

E+F=NONE

E+G=NONE

F+H=P=-111

G+H=Q=1-11

이므로 J부터 Q까지 1인 비트 수의 개수에 따라 그룹을 다시 나눠보면

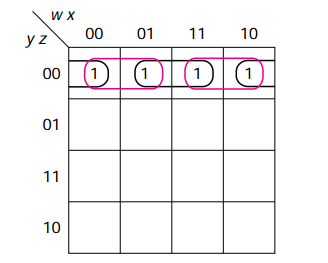
J 0-00, K -000

L 010-, M -100, N 1-00

O 01-1

P -111, Q 1-11

가 되고, -를 무시하고 앞에서 한 과정을 반복하면 J와 N이 만나서 --00이 되고, K와 M이 만나서 똑같이 --00이 되어 다음 <Figure 3>과 같은 prime implicant를 만들 수 있습니다.



<Figure 3> F의 prime implicant

따라서, 여기서 구한 prime implicant과 prime implicant에 사용되지 않은 L, O, P, Q, R을 이용하여 F는 yz+w’xy’+w’xz+wxz+xyz+y’z’으로 단순화하여 표현할 수 있습니다.

Quine-McCluskey 알고리즘을 이용한 논리식 단순화 방법은 컴퓨터에 의해 계산이 가능합니다.

1. 기타 이론

위의 Quine-McCluskey 알고리즘을 설명할 때, 사용한 -라는 기호는 1이든, 0이든 상관이 없다는 의미로, don’t cares이라고 부릅니다. 일반적으로 디지털 논리 회로의 진리표를 작성할 때, don’t cares를 사용하는 경우는 그런 입력이 절대 회로에 들어가지 않다고 가정할 때인데 카르노 맵이나 Quine-McCluskey 방법을 이용하여 논리식을 단순화할 때에는 don’t cares로 처리된 부분을 포함하여 implicant를 만들어도 무관하기 때문에 don’t cares term이 1이라고 생각하여 이를 포함하면 더 큰 prime implicant를 만들 수 있다는 장점이 있습니다.