5주차 예비보고서

전공 : 국제한국학과 학년 : 4학년 학번 : 20181202 이름 : 김수미

**1. De Morgan의 정리에 대해 조사하시오.**

드모르간(De-morgan)의 법칙이란 식 전체에 부정을 취하는 경우, 이를 풀어서 전개 할 때 AND 연산은 OR 연산으로, OR 연산은 AND 연산으로 바꾼 다음 각 변수에 부정을 취하면 되는 것을 가리키는 법칙이라고 할 수 있다. 즉 아래 식과 같이 나타낼 수 있다.

드모르간의 제 1 법칙 : 혹은

드모르간의 제 2 법칙 : 혹은

드모르간의 제 1 법칙은 식 (A+B)의 전체에 보수를 취한 것은 A의 보수와 B의 보수를 곱한 것(AND)과 같음을 나타내고, 드모르간의 제 2 법칙은 식 (AxB)의 전체에 보수를 취한 것은 A의 보수와 B의 보수를 합한 것(OR)과 같음을 나타낸다.

혹은 역시 드모르간의 법칙을 적용한 것이다.

**2. 논리회로의 간소화에 대해 조사하시오. (예시 포함)**

논리회로는 논리식을 물리적인 회로 위에서 구현하는 것이기 때문에, 다양한 법칙을 이용하여 논리식을 간소화 시킴으로써 논리회로를 간소화 시켜 표현할 수 있게 된다.

아래는 논리식을 간소화 시키는 방법들에 대한 정리이다.

|  |  |  |
| --- | --- | --- |
| 항등원 법칙(Identity Law) |  |  |
| 보수 법칙(Negation Law) |  |  |
| 등멱 법칙(Idempotent Law) |  |  |
| 경계 법칙(Domination Law) |  |  |
| 대합 법칙(Double Negation Law) |  | |
| 교환 법칙(Commutative Law) |  |  |
| 연관 법칙(Associative Law) |  |  |
| 분배 법칙(Distributive Law) |  |  |
| 드모르간 법칙(De-Morgan’s Law) |  |  |
| 흡수 법칙(Absorption Law) |  |  |

예를 들어 과 같은 식은 먼저 위 표의 분배법칙에 나와 있는 식을 이용해 의 형태로 간소화 시킬 수 있으며 이는 다시 보수법칙에 의해 로, 그리고 항등원 법칙에 의해 로 간소화 시킬 수 있다.

**3. 카르노 맵에 대해 조사하시오. (예시 포함)**

카르노맵(Karnaugh Map)이란 Boolean Function을 시각적으로 나타내어 좀 더 쉽게 논리식을 간소화 시킬 수 있는 방법이다. 아래 예시를 통해 카르노맵을 이용해 논리식을 간소화하는 방법을 알아보자.

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

먼저 A, B의 두가지 값에 대해 의 Boolean Function의 진리표는 아래와 같으며, 해당 진리표를 아래의 카르노맵(2변수 카르노맵) 처럼 나타낼 수 있다.

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

이때 카르노맵에서는 1인 값들을 2의 n제곱수 개수 만큼씩 묶을 수 있다. 즉 1을 각각 1개 또는 2개, 4개 등으로 묶을 수 있다. 즉 위 카르노맵에서 1인 값들을 묶는 경우 아래와 같이 나타낼 수 있다.

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

파란색으로 묶은 박스를 살펴보자. 파란색으로 묶인

박스 안의 요소들의 A 값은 1로 일정하고, B 값은

0 또는 1의 값을 가진다. 이는 다시 말해 파란색

박스 안의 요소들은 A의 값은 반드시 1의 값이 되

어야 하지만 B의 값은 0이 되어도, 1이 되어도 상관

이 없다는 의미이기도 하다.

따라서 파란색으로 묶인 박스 안의 요소들은 식 로 나타낼 수 있다(는 이 아닌 반드시 여야 하기 때문에 식에 라고 표기해줘야 하지만, 는 또는 이 모두 올 수 있기 때문에 식에 표기하지 않아도 된다). 파란색 박스 안의 요소들은 가 1일때, 가 0일때 그리고 가 1일때, 가 1일때의 두가지 칸으로 구성되어 있기 때문에 원래 의 식으로 작성되지만 이는 분배법칙 및 보수법칙을 이용하면 로 간소화 시킬 수 있다. 결국 카르노맵에서 살펴본 식과 같이 간소화 되므로, 카르노맵을 이용하면 Boolean Function을 쉽게 간소화할 수 있다. 단 위의 경우는 변수가 2개인 경우이지만, 변수가 4개를 넘어가는 경우 카르노맵을 이용하여 논리식을 간소화 시키는 과정은 매우 복잡해질 수 있다는 단점이 있다.

**4. Quine-McCluskey 최소화 알고리즘에 대해 조사하시오.**

Quine-McCluskey 알고리즘이란 카르노맵처럼 논리식을 간소화하는데 사용할 수 있는 방법이다. 내부적으로는 카르노맵과 거의 동일하지만, 표 위에 그림을 그려서 논리식을 간소화 시키는 카르노맵 방법과 달리 표를 사용하기 때문에 컴퓨터에서 코드식을 사용해 작성할 수 있다. 또한 카르노맵은 논리식의 최소 형태(가장 간소화된 형태)를 보장하지 못하지만, Quine-McCluskey 알고리즘은 논리식의 최소 형태를 보장한다.

알고리즘은 주어진 논리식에서 Prime Implicants(더 큰 Implicant(2의 n제곱 개수 만큼의 묶음)에 포함되지 않는 묶음)를 모두 구한 다음 그 안에서 다시 Essential Prime Implicant(다른 Implicant에 속하지 않고 자신의 Prime Implicant에만 속하는 요소를 적어도 하나 가지는 Prime Implicant)를 구하는 방식으로 논리식을 간소화 시킨다. 카르노 맵과 달리, Quine-McCluskey 알고리즘을 사용하면 변수가 4개를 넘더라도 효율적으로 논리식을 최소화 시킬 수 있다.

**5. 기타이론**

1) Quine-McCluskey 최소화 알고리즘 활용 예시

위에서 Quine-McCluskey 알고리즘의 대략적인 개념을 살펴보았다. 그럼 실제로 Quine-McCluskey 알고리즘을 사용하여 논리식을 어떻게 최소화시킬 수 있는지를 알아보자.

먼저 다음과 같은 Boolean Function을 최소 형태로 나타내어 보자.

먼저 위 minterms를 이용해 해당 함수의 Canonical Sum of Products를 구할 수 있다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Minterm | A | B | C | D | Product |
|  | 0 | 1 | 0 | 0 |  |
|  | 1 | 0 | 0 | 0 |  |
|  | 1 | 0 | 0 | 1 |  |
|  | 1 | 0 | 1 | 0 |  |
|  | 1 | 0 | 1 | 1 |  |
|  | 1 | 1 | 0 | 0 |  |
|  | 1 | 1 | 0 | 1 |  |
|  | 1 | 1 | 1 | 1 |  |

위에서 구한 minterms 들은 아래 표와 같이 정리할 수 있다.

|  |  |  |
| --- | --- | --- |
| 1의 개수 | Minterm | Binary |
| 1개 |  | 0100 |
|  | 1000 |
| 2개 |  | 1001 |
|  | 1010 |
|  | 1100 |
| 3개 |  | 1011 |
|  | 1101 |
| 4개 |  | 1111 |

이제 위 표에서 minterms 들을 서로 결합한다. 만일 두 항이 1bit 만 차이가 난다면 그 자리를 ‘-‘ 로 대체할 수 있으며, 이는 그 자리에는 0값이 오든 1값이 오든 상관이 없다는 뜻이다. 더 이상 결합하지 못하는 항은 Prime Implicant 가 된다.

|  |  |  |  |
| --- | --- | --- | --- |
| 1의 개수 | Implicant | Binary | Prime Implicant |
| 1개 |  | -100 | O |
|  | 100- | X |
|  | 10-0 | X |
|  | 1-00 | X |
| 2개 |  | 10-1 | X |
|  | 101- | X |
|  | 1-10 | X |
|  | 11-0 | X |
| 3개 |  | 1-11 | X |
|  | 111- | X |

|  |  |  |  |
| --- | --- | --- | --- |
| 1의 개수 | Implicant | Binary | Prime Implicant |
| 1개 |  | 10-- | O |
|  | 1--0 | O |
| 2개 |  | 1-1- | O |

이제 위 표에서 구한 Prime Implicants 들을 아래와 같이 하나의 표에 나타내어 Essential Prime Implicants 를 찾을 수 있다.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |
|  | O |  |  |  |  | O |  |  |
|  |  | O | O | O | O |  |  |  |
|  |  | O |  | O |  | O | O |  |
|  |  |  |  | O | O |  | O | O |

노란색으로 표시한 것은 카르노맵에서 살펴보았던, Prime Implicant를 형성하는 1들 중에서 다른 Implicant에 속하지 않고 자신의 Prime Implicant에만 속하는 요소를 표시한 것이다. 하나의 열에서 봤을 때 O 표시가 오직 하나의 Prime Implicant에만 있으면 이는 해당 Prime Implicant에만 속하는 요소임을 알 수 있다.

즉 의 경우 자신의 Prime Implicant에만 속하는 요소를 하나도 가지고 있지 않기 때문에 Essential Prime Implicant가 아니다. 이 외 , , 는 모두 Essential Prime Implicant 가 된다.

따라서 = -100 = , = 10-- = , = 1-1- = 이므로

는 결국 로 간소화 시킬 수 있다.