14주차 예비보고서

전공: 컴퓨터공학과 학년: 2학년 학번: 20191619 이름: 이동석

**1. FSM**

Finite State Machine은 Finite-state Automaton으로도 불린다. 영문을 그대로 번역하면, 유한상태기계 혹 유한 상태 오토마톤이다. 오토마톤은 옥스포드 사전에서 자동적으로 정보 처리를 하는 수학적 모델이며 **유한개의 입력과 유한개의 출력**을 방출하는 기구라고 정의한다. 한번에 단 한 개의 상태만을 가질 수 있다. 우리의 일생생활에서도 손쉽게 그 예로 신호등, 엘리베이터, 알람시계, 마이크로 웨이브 등이있다. 이런 기계들은 일종의 반응시스템으로 어떤 입력이나 시그널에 반응하여 외부로 출력한다. 각 입력에 따른 반응들은 **현재 상태에 따라 다른 결과**를 나타낸다. 이를 Transition(전이)라 부른다.

이런 유한 상태 기계는 다음 기호 Σ,S,s0,δ,F 의해 정의된다.

1. Σ는 입력 기호이다. (유한하고, 비어있지 않은 심볼의 집합이다.)
2. S는 유한하고, 비어있지 않은 상태의 집합이다.
3. s0는 초기상태이며, S의 원소이다.
4. δ는 상태 전이 함수이다. δ : S x Σ -> S
5. F는 final state의 집합이며, S의 부분집합이고 비어있을 수 있다.
6. O는 출력들의 집합이며 마찬가지로 비어있을 수 있다.

티켓을 출력해주는 기계를 예로 들어보자. 우선, 입력에는 돈, 거스름돈, 요구 티켓이 있을 것이다. Σ(m, t, r) 또한 상태에는 지불 / 지불하지 않음 2가지가 있을 것이다. S(1,2) 초기상태를 1로 지정하면 이는 아마도 지불하지 않음이 될 것이다. s0(1) 출력은 티켓이 프린트 되었는가와 거스름돈이 될 것이다. O(p/d) 이제 이를 δ로 표현하면 아래 그림과 같다.

시계이(가) 표시된 사진

자동 생성된 설명

위와 같이 그린 상태도를 가지는 FSM을 변환기(Transducer)라 한다. (무어와 밀리가 여기에 해당한다.) 반대로 Acceptors로 표현하면 각 입력에 대해서 예/아니오로만 표시한다. 때문에, Acceptors는 출력은 없지만, F가 있으며 Transducers는 출력이 있다.

유한 상태 기계에는 DFA(Determinstic Finite Automato)와 NDFA(Nondeterminstic Finite Automata)로 나뉜다.

**2. Mealy machine**

밀리 머신은 FSM 중 하나이며, 그 중 Transducer이다. **현재의 입력과 상태 모두에 의해서 출력과 다음 상태**가 결정된다. 밀리머신은 6개의 튜플로 정의된다. M = (Q, ∑, Γ, δ, θ q0 )

Q는 유한 개의 내부 상태들의 집합이다.

∑는 입력 기호이다.

Γ는 출력 기호이다.

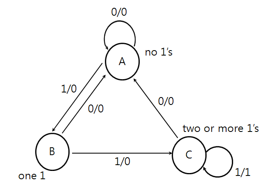
δ는 전이 함수이다. δ : Q X E -> Q

θ는 출력 함수이다. θ : Q X ∑-> Γ

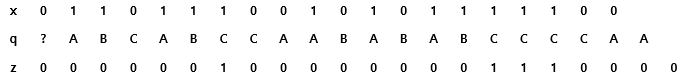
q0는 Q의 원소로 M의 초기상태를 의미한다.

아래의 오른쪽 그림을 Trainsition graph라 부르고, 디지털 회로 개론 시간에서는 상태도라고 불렀다. 밀리 머신의 상태도는 현재 상태에서 입력에 따른 출력을 /를 사용해 같이 표시한다. 예로 A상태에서 x=0을 입력받으면 출력은 0이 되고 다음 상태는 A가 된다. δ(A,0) = A, δ(A, 1) = B, θ(A,0) = 0, θ(C,1) = 1 로 표시할 수 있다.

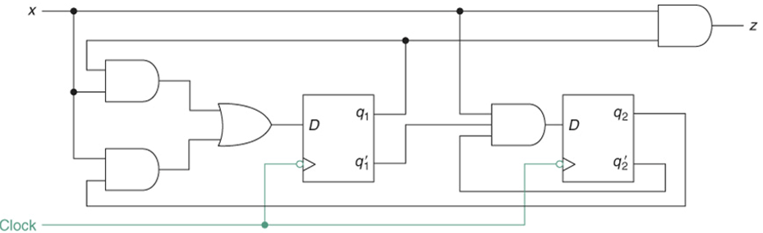
테이블이(가) 표시된 사진

자동 생성된 설명

위 밀리 머신은 1이 3개가 들어오는 시점이 상태 C가 되고 이때 출력 1을 한다. 따라서 아래와 같은 표를 작성할 수 있다.



일반적으로 무어머신의 상태가 밀리 머신보다 많다. 다음은 D플리플롭을 활용해 밀리머신을 나타낸 회로의 예이다. 입력 x가 출력 z와 연결되어 있는 모습을 보고 밀리머신임을 유추할 수 있다.



**3. Moore machine**

무어 머신의 경우 출력은 오직 현재 상태에 의해서 결정된다. 따라서, 입력은 다음 상태를 결정하는데 사용된다. 무어머신은 밀리 머신과 동일하게 6개의 튜플로 정의된다. M = (Q, ∑, Γ, δ, θ q0 ) 다만, 출력 함수 θ의 정의가 다르다.

Q는 유한 개의 내부 상태들의 집합이다.

∑는 입력 기호이다.

Γ는 출력 기호이다.

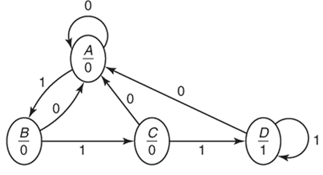
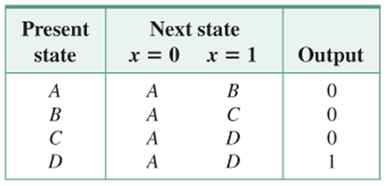
δ는 전이 함수이다. δ : Q X ∑ -> Q

θ는 출력 함수이다. θ : Q -> Γ

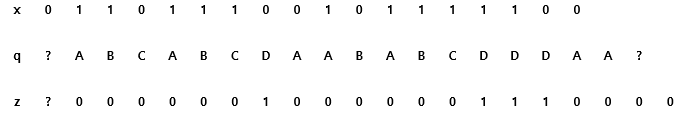
q0는 Q의 원소로 M의 초기상태를 의미한다



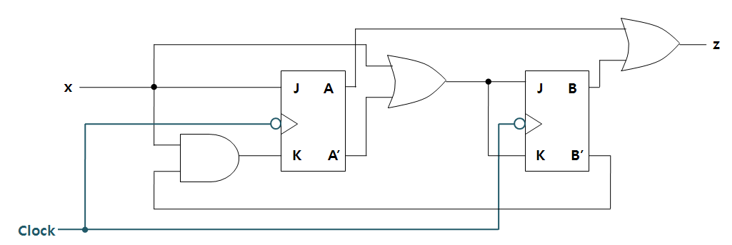
위와 같이 있을 때, 왼쪽과 같이 상태표를 작성하고 오른쪽의 상태 Diagram을 그릴 수 있다. 이때, 상태도에서 vertex는 2가지의 라벨을 가지게 된다. 하나는 상태이며, 하나는 출력 심볼이다.



현재 3개의 연속된 1이 들어왔을 경우 1을 출력하는 머신이다. 따라서, 3개의 1이 들어오면 다음 상태가 D가 되고, 이때 출력이 1이 되는 것이다. 3개째의 1이 들어왔을 때의 상태는 C가 된다. 따라서, 같은 기능을 하더라도 일반적으로 상태가 무어머신이 더 많다.



재밌는 점은 무어머신은 이 때문에 초기출력과 상태모두 ?로 알 수 없다는 점이다. 그에 반해 밀리머신은 출력은 제대로 알 수 있다. 아래 그림은 JK플리플롭을 사용한 무어 머신회로이다. 입력 x가 출력 z에 관여하지 않는 것을 볼 수 있다.

****

NFA와 DFA관계처럼 밀리머신과 무어머신은 서로 변환이 가능하며 이를 equivalent하다고 한다.

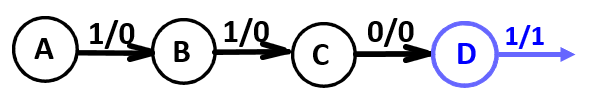
**4. Sequence Detector**

Sequence Detector를 직역하면 순차 검출기이다. 위에서 설명했던 밀리머신이나 무어머신을 활용하면 원하는 특정 시퀀스를 검출할 수 있다. 따라서, 미래 출력값을 예측하는 데 과거 입력 시퀀스는 필수적이다. 이때, 중첩 시퀀스와 비중첩 시퀀스 2가지 경우로 나뉘게 된다. 보통 중첩을 많이 사용하며, 예로 1101을 검출할 때 1101101에는 중첩되는 1이 있게 되며 마지막 1은 다음 시퀀스의 부분 시퀀스가 된다. 반면 비중첩일 경우 끝나게 되면 무조건 처음 상태로 돌아가도록 한다. 다음 순서를 통해 논리회로를 설계한다.

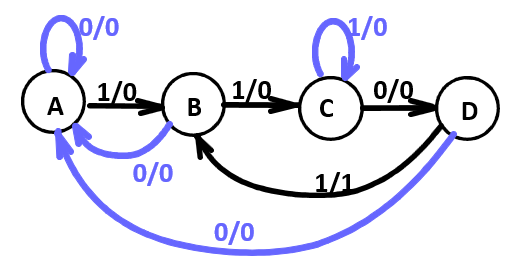
1. 상태도 유도 및 상태표 (Formulation : State Table and State diagram)
2. 상태배정 및 표 (State Assignment)
3. 플리플롭 입력 식 결정 – 테이블의 다음 상태 항목에서 유도
4. 출력 식 결정 – 테이블의 출력항목에서 유도
5. 최적화
6. 기술적 매핑
7. 확인

시퀀스의 첫 부분이 발생하지 않은 상태를 초기상태로 시작한다. 다음으로 시퀀스의 첫번 째를 인식한 상태를 추가하고, 이를 입력 시퀀스의 마지막이 발생했을 때까지 추가한다. 이제, 각 상태에서 적절한 순서가 아닌 기호가 나타났을 경우 상태를 전환하는 심볼을 추가한다.

예를 들어 1101을 검출하는 밀리머신 회로를 만들기 위해선 우선 상태도를 그려야한다. 가장 초기상태를 A라 할 때 1이 들어오면 다음 상태 B로 가며 이때 출력은 0일 것이다. 같은 방식으로 다음 1이 입력으로 들어오면 상태 C가 되며 아직 1101을 만나지 않았으므로 출력역시 0이 된다. 다음으로 0을 만나면 D, 이제 상태 D에서 1이 입력으로 들어오면 1101을 만났기 때문에 1을 출력한다.



이제 D에서 1을 입력받으면 이는 1101을 만난 것을 의미하지만 동시에 상태 B가 되는 것을 의미한다. 따라서 상태 전환 호를 추가해준다. 같은 방식으로 각 상태들에서 입력 0 혹은 1이 들어왔을 때 전이되는 상태를 호로 추가한다.

****

이제 위 상태도를 보고 아래의 상태표를 그린다.

테이블이(가) 표시된 사진

자동 생성된 설명

무어 머신도 동일한 방법을 진행되지만, 무어 머신의 특징에 유의해야한다. 따라서 무어머신의 상태도와 상태표는 다음과 같다.

텍스트, 시계이(가) 표시된 사진

자동 생성된 설명테이블이(가) 표시된 사진

자동 생성된 설명

**5. 기타 이론**

DFA(Determinstic finite accepter로도 불린다)의 구성요소는 FSM과 유사하며 아래와 같다.

Q는 유한개의 내부 상태들이다.

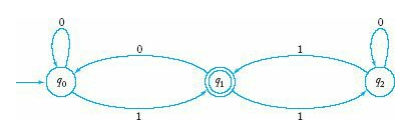
Σ 입력 기호로 심볼들의 유한집합이다.

δ :Q × Σ → Q 전이 함수로, total 함수이다.

q0 ∈ Q 초기 상태이다.

F ⊆Q final states 이다.

예를 들어 다음과 같은 기계가 있다고 하자. M = ({q0, q1, q2}, {0,1}, δ , q0, {q1} ) 이는 순서대로, Q, Σ, δ, q0, F를 의미한다. 따라서, 위 DFA를 아래와 같은 Transition Graph로 표현할 수 있다. 이때 2중으로 된 원은 F를 의미한다.



만약, DFA가 문자열 01을 입력받는 다면 0을 입력받았을 때 q0 -> q0 이 되고 1을 입력받으면 q0 -> q1이 된다. 이때, q1은 F를 나타내므로 이는 accept가 된다. 비슷한 이유로 101이나, 0111 은 모두 accept하며 반대로 100혹은 1100과 같은 문자열은 거절함을 알 수 있다. 더 나아가 Extended Transition Function이 있다.



이는 쉽게 생각해서 δ(q0,a) = q1와 δ(q1,b) = q2 를 만족한다면 δ\*(q0, ab) = q2임을 의미한다.

DFA와 NDFA는 정의에서 차이가 있지만, DFA -> NFA나 NFA -> DFA의 변경이 가능하다.

**6. 참고문헌**

Fundamental of Logic Design, 7th edtion

Formal Languages and Automata, Fifth Edition, Peter Linz

디지털 회로개론 강의자료 Chap 5 Analysis of Sequential Systems. 김주호 교수

디지털 회로개론 강의자료 Chap 6 The Design of Sequential Systems. 김주호 교수

<https://web.stanford.edu/class/cs123/lectures/CS123_lec07_Finite_State_Machine.pdf>