

오토마타를 차츰 확장해보자. 기존에 2. Finite State Automata에서 정의한 오토마타를 결정적(Deterministic) 오토마타(짧게는 DFA)라고 부르고 이를 다시 정의하여 보자.

(정의 1) 결정적 오토마타(Deterministic Finite State Automata: DFA) $M_{DFA} =$

$(Q, \Sigma, \delta_{DFA}, q_0, F)$ 는 아래와 같이 정의한다.

- | | | |
|-----|---|-----------|
| (1) | Q | 상태 집합 |
| (2) | Σ | 입력문자 집합 |
| (3) | $\delta_{DFA}: Q \times \Sigma \rightarrow Q$ | 상태변화함수 집합 |
| (4) | $q_0 \in Q$ | 처음 상태 |
| (5) | $F \subseteq Q$ | 끝나는 상태 집합 |

오토마타를 확장하는 작업은 다음 세 가지로 이루어진다.

(A) 상태변화함수 δ 을 확장한다.

$$\delta: Q \times \Sigma \rightarrow Q \quad \Rightarrow^* \quad \delta^*: Q \times \Sigma^* \rightarrow 2^Q$$

(B) 확장한 오토마타 클래스¹⁾가 확장 전 오토마타 클래스를 포함하는 확장클래스(super class)임을 밝힌다.

(C) 확장한 오토마타와 같은 일을 하는 확장하기 전 형태를 가진 오토마타가 존재함을 증명하여 확장한 오토마타 클래스와 확장하기 전 오토마타 클래스가 같은 일을 하는 클래스임을 밝힌다.

2.1.1. 부분함수를 허용하여 보자.

(해결책) 죽은 상태(dead state)를 더한다.

상태변화함수 δ 가 전체함수(total function) 이어야하나, 이를 부분함수(partial function)도 허용하도록 확장하여 보자. 즉 어떤 상태 $q \in Q$ 와 입력문자 $a \in \Sigma$ 에서 상태변화함수 $\delta(q, a)$ 값이 없는 경우이다. 이 경우 현 상태 $q \in Q$ 에 올 때까지 오토마타가 읽은 문자열이 $x \in \Sigma^*$ 라면 현재문자($a \in \Sigma$)을 보면, 다음에 어떤 문자열($\forall y \in \Sigma^*$)이 오른쪽에 와도 전체문자열($xay \in \Sigma^+$)은 오토마타가 받아들이는 언어가 아니라고($xay \notin L(M)$) 판단하여 오토마타 동작을 멈추고 **false**를 보고한다.

이 확장을 상태변화함수 δ 로 표시하면 아래와 같다.

$$\delta: Q \times \Sigma \rightarrow Q \cup \{\emptyset\}.$$

즉 상태변화함수 $\delta(q, a)$ 값이 없는 경우를 $\delta(q, a) = \emptyset$ 로 표현한다.

(정의 2) 부분함수를 허용하는 오토마타 $M_{\text{부분}} = (Q, \Sigma, \delta_{\text{부분}}, q_0, F)$ 는 아래와 같이 정의한다.

- | | |
|---------------------|--|
| (1), (2), (4), (5)의 | Q, Σ, q_0, F 는 기본정의 M_{DFA} 와 같다. 단 |
| (3) | $\delta_{\text{부분}}: Q \times \Sigma \rightarrow Q \cup \{\emptyset\}$ 상태변화함수만이 다르다. |

1) 오토마타는 집합이 아니므로, 오토마타 집합이나 불려도 좋으나, 앞으로 나올 언어 클래스와 형평을 고려하여 오토마타 클래스라고 부른다.

(정의 3) \mathbb{M}_{DFA} 는 M_{DFA} 전체의 집합이고, $\mathbb{M}_{부분}$ 은 $M_{부분}$ 전체의 집합이라 하자. \mathbb{M}_{DFA} 를 Deterministic 오토마타 클래스, $\mathbb{M}_{부분}$ 을 부분함수를 허용하는 오토마타 클래스라 부른다.

(쉬운정리(Fact)²) 1) $\mathbb{M}_{DFA} \subsetneq \mathbb{M}_{부분}$.
 모든 $M_{DFA} \in \mathbb{M}_{DFA}$ 는 $\mathbb{M}_{부분}$ 의 일종($\delta_{DFA}(q, a) = \emptyset$ 인 경우가 없는 경우)이고, \mathbb{M}_{DFA} 가 아닌($\delta_{부분}(q, a) = \emptyset$ 인 경우가 있는 경우) $\mathbb{M}_{부분}$ 이 존재한다.

(정의 4) $\mathbb{M}_{DFA} \subsetneq \mathbb{M}_{부분}$ 이면, \mathbb{M}_{DFA} 는 $\mathbb{M}_{부분}$ 의 **적당한(proper) 부분클래스(subclass)**라고 부르고, $\mathbb{M}_{부분}$ 은 \mathbb{M}_{DFA} 의 **적당한 확장클래스(superclass)**라고 부른다.

(정의 2)와 (쉬운정리 1)로 확장의 첫 번째와 두 번째 작업인 확장된 오토마타 클래스의 (A) 정의와 (B) 확장을 마치었다. 확장에 마지막 작업으로 상태변화함수 δ 가 전체함수인 오토마타 클래스로 바꾸어보자.

상태변화함수 값이 존재하지 않는 경우($\delta(q, a) = \emptyset$)에 새로운 상태($d \notin Q$) d 를 하나 더한 후, 새로운 상태 d 로 상태 변화하는 함수($\delta'(q, a) = d$)로 확장하여 **전체함수**를 만든다. 새로운 상태 d 에서도 전체함수로 만들기 위하여 모든 입력문자 $a \in \Sigma$ 에 대하여 $\delta'(d, a) = d$ 를 추가한다. 새로 더한 상태 d 는 한번 들어가면 끝나는 상태(F)로 빠져나올 수 없는 블랙홀과 같은 **죽은 상태(dead state)**이다.

(입력) 부분함수를 허용하는 오토마타 $M_{부분} = (Q, \Sigma, \delta, q_0, F)$
 (출력) 전체함수만을 허용하는 오토마타 $M_{DFA} = (Q \cup \{d\}, \Sigma, \delta', q_0, F)$ ³ 단 $d \notin Q$.

(알고리즘) $\delta' = \delta \cup \{\delta'(q, a) = d \mid \delta(q, a) = \emptyset\} \cup \{\delta'(d, a) = d \mid a \in \Sigma\}$

(증명) $L(\mathbb{M}_{DFA}) = L(\mathbb{M}_{부분})$
 (1) $L(\mathbb{M}_{DFA}) \subseteq L(\mathbb{M}_{부분})$
 $\mathbb{M}_{DFA} \subsetneq \mathbb{M}_{부분}$ 이므로 당연하다(trivial).
 (2) $L(\mathbb{M}_{부분}) \subseteq L(\mathbb{M}_{DFA})$
 $\forall M_{부분} \in \mathbb{M}_{부분}: \exists M_{DFA} \in \mathbb{M}_{DFA}, L(M_{부분}) = L(M_{DFA})$
 (2.1) $L(M_{부분}) \subseteq L(M_{DFA})$
 (2.2) $L(M_{DFA}) \subseteq L(M_{부분})$

(정의 5) 오토마타 클래스 \mathbb{M} 이 정의하는 언어 클래스
 $L(\mathbb{M}) = \{L \subseteq \Sigma^* \mid M \in \mathbb{M}, L = L(M)\}$

(정의 6) 두 개의 오토마타 클래스 \mathbb{M}_1 과 \mathbb{M}_2 가 정의하는 언어 클래스가 같을 때, $L(\mathbb{M}_1) = L(\mathbb{M}_2)$, 오토마타 클래스 \mathbb{M}_1 과 \mathbb{M}_2 는 **같은 일을 하는(동등한; isomorphic)** 오토마타 클래스라 정의한다.

(중요정리(Theorem) 1) \mathbb{M}_{DFA} 와 $\mathbb{M}_{부분}$ 는 **같은 일을 하는(같은)** 오토마타 클래스이다.

2) 쉬운정리, 부분정리, (중요)정리는 Fact, Lemma, Theorem의 번역이다.
 3) 새로 정의 되는 M_{DFA} 는 (1) 상태집합은 Q 에서 $Q \cup \{d\}$ 로 늘어나고, (2) 입력문자집합 Σ 과 (4) 초기상태 q_0 , (5) 끝나는 상태 F 는 원래의 $M_{부분}$ 과 같고, (3) 상태변화함수만이 δ 에서 δ' 으로 바뀐다.

2.1.2. Nondeterministic Finite State Automata(NFA; 비결정적, 여러 상태로 상태 변화를 허용하는 오토마타)

(해결책) NFA에서 **상태집합**을 변환(같은 일을 하도록)할 DFA의 **상태 하나**로 한다.

특정 상태 $q \in Q$ 와 입력문자 $a \in \Sigma$ 에서 상태변화함수 $\delta(q, a)$ 가 없는 경우를 생각하였는데 이번에는 **여러 개의 상태로** 상태변화를 허용하는 경우로 확장하여 보자. 즉

$$\delta(q, a) = \{p_1, p_2, \dots, p_n\} \quad \text{단 } n \geq 0^4, \quad q, p_1, p_2, \dots, p_n \in Q, \quad a \in \Sigma$$

로 정의하여 오토마타의 정의의 유연성을 높일 수 있다. p_1, p_2, \dots, p_n 까지 n 개의 상태가 정의되어 있으면 p_1, \dots, p_n 중 어떤 상태로 상태를 바꾸어도 되고, n 가지 상태변화 중 적어도 **한 번만** 끝나는 상태에 도착하여도 그 문자열은 오토마타가 받아들이는 문장(sentence)으로 본다.

(정의 7) 비결정적(NFA) 오토마타 $M_{NFA} = (Q, \Sigma, \delta_{NFA}, q_0, F)$ 는 아래와 같이 정의한다.

(1), (2), (4), (5)의 Q, Σ, q_0, F 는 기본정의 M_{DFA} 와 같다. 단

(3) $\delta_{NFA}: Q \times \Sigma \rightarrow 2^Q$ 상태변화함수만이 다르다.

NFA는 상태변화함수의 치역으로 상태의 부분집합의 집합(power set)을 허용하는 \mathbb{M}_{NFA} 는 상태 하나만 허용하는 \mathbb{M}_{DFA} 와 공집합을 허용하는 $\mathbb{M}_{\text{부분}}$ 의 슈퍼클래스다.

(쉬운정리 2) $\mathbb{M}_{DFA} \subseteq \mathbb{M}_{\text{부분}} \subseteq \mathbb{M}_{NFA}$.

마지막으로 임의의 NFA를 **같은 일을 하는** DFA로 바꾸어 보자.

(정의 8) 상태변화함수 δ 의 **첫 번째 정의역**을 상태(Q)에서 상태집합(2^Q)으로 확장하자.

$$\delta': 2^Q \times \Sigma \rightarrow 2^Q$$

$$\delta'(P, a) \triangleq \{q \in Q \mid p \in P, \delta(p, a) = q\}$$

$$= \bigcup_{p \in P} \delta(p, a)$$

(입력) 비결정적 오토마타 $M_{NFA} = (Q_{NFA}, \Sigma, \delta_{NFA}, q_0, F_{NFA})$

(출력) 전체함수만을 허용하는 DFA $M_{DFA} = (Q_{DFA}, \Sigma, \delta_{DFA}, \{q_0\}, F_{DFA})$

(알고리즘) **function** NFA_to_DFA(Q_{NFA} (NFA상태집합), Σ, δ_{NFA} (NFA상태변화함수), $q_0 \in$

$Q_{NFA}, F_{NFA} \subseteq Q_{NFA}$) **returns** ($Q_{DFA} \subseteq 2^{Q_{NFA}}$ (DFA상태집합: NFA상태의 power set)⁵⁾, Σ, δ_{DFA} (DFA상태변화함수), $q_0^{DFA} \in Q_{DFA}, F_{DFA} \subseteq Q_{DFA}$);

variable P, Q $\subseteq Q_{NFA}$ (= P, Q $\in Q_{DFA}$)⁶⁾; a \in 입력문자집합(Σ);

$$q_0^{DFA} := \{q_0\}; \quad Q_{DFA} := \{\{q_0\}\}; \quad \delta_{DFA} := \emptyset; \quad F_{DFA} := \emptyset;$$

4) $n = 0$ 이면 $\{p_1, p_2, \dots, p_n\} = \emptyset$ 으로 본다. 즉 NFA 확장은 첫 번째 확장인 부분함수도 포함한다.

5) 변환 후 DFA 상태는 변환 전 NFA 상태의 power set의 부분집합이다.

6) P, Q는 NFA 상태의 부분집합이지만,

```

repeat
  for P ∈ QDFA7) do
    for a ∈ Σ do
      Q := ∅; for p ∈ P do Q := Q ∪ δ(p, a) od
      QDFA := QDFA ∪ {Q}8);    δDFA := δDFA ∪ {δ(P, a) = Q}9);
    od
    if (P ∩ FNFA) ≠ ∅ → FDFA := FDFA ∪ {P} else skip fi
  od
until no more new Q is added to QDFA
return MDFA = (QDFA, Σ, δDFA, q0DFA, FDFA);
end function NFA_to_DFA;
    
```

(정의 9) 확장된 상태변화함수 δ'의 두 번째 정의역을 입력문자(Σ)에서 입력문자열(Σ*)로 확장하자. δ'* : 2^Q × Σ* → 2^Q

$$\delta'^*(P, \epsilon) \stackrel{\text{def}}{=} P \quad \text{단 } P \subseteq Q (P \in 2^Q).$$

$$\delta'^*(P, xa) \stackrel{\text{def}}{=} \delta'(\delta'^*(P, x), a) \quad \text{단 } P \subseteq Q (P \in 2^Q), x \in \Sigma^*, a \in \Sigma (xa \in \Sigma^+).$$

(부분정리) Q_{DFA} ⊆ 2^{Q_{NFA}}.

$$P \subseteq Q_{NFA} (\equiv P \in 2^{Q_{NFA}} \equiv P \in Q_{DFA}), a \in \Sigma \text{ 일 때}$$

$$\delta_{DFA}(P, a) = \{q \in Q_{NFA} \mid p \in P, q \in \delta_{NFA}(p, a)\} = \delta_{NFA}'(P, a)$$

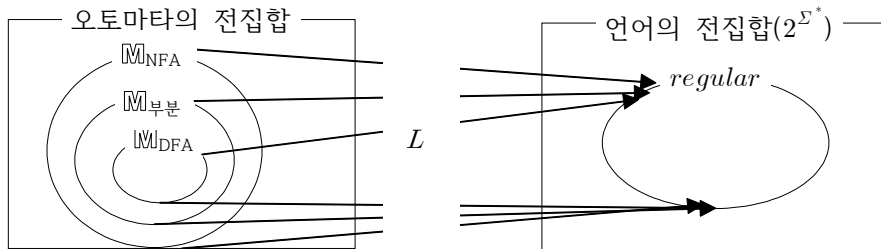
$$F_{DFA} = \{P \in 2^{Q_{DFA}} \mid P \cap F \neq \emptyset\}$$

(증명1) 위의 (알고리즘)은 임의의 NFA를 위의 (부분정리)를 만족하는 DFA로 만든다.

(증명2) L(M_{NFA}) = L(M_{DFA})

(생략)

(중요정리 2) M_{DFA} 와 M_{부분}, M_{NFA}는 모두 같은 일을 하는(같은) 오토마타 클래스이다.



7) 동시에, P, Q는 (DFA의 상태집합이 NFA 상태부분집합의 집합(Power set)이므로(Q_{DFA} = 2^{Q_{NFA}})) DFA 상태 하나이기도하다. 이것을 이해하는 것이 이 알고리즘을 이해하는 핵심이다.

8) 각주 6), 7)과 같다.

9) 각주 6), 7)과 같다.