KAIST 전산학과 최광무

2.1.3 $\epsilon-move$ 를 허용한다

(해결책) $\epsilon-closure(\epsilon^*)$를 구한다.

오토마타를 더 확장해보자. $\epsilon-move$ 를 허용한다. 즉 기존에 NFA에서 입력문자열을 보지 않고 상태를 바꿀 수 있게 한다.

이 상황을 상태변화함수 δ로 표시하면 아래와 같다.

$\delta:\ Q\times(\Sigma\cup\{\epsilon\})\ \rightarrow\ 2^Q$.

즉 상태변화함수의 두 번째 도메인에 $\{\epsilon\}$을 추가하여, $\delta(q,\epsilon)\ =\ \{p_1,p_2,\cdots,p_n\}$을 허용한다.

(**정의** 10) $\epsilon-move$ 를 허용하는 오토마타 $M_{\epsilon-NFA}\ =\ (Q,\Sigma,\delta_{\epsilon-NFA},q_0,F)$ 는 아래와 같이 정의한다.

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

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

(**정의** 11) $\mathbb{M}_{\varepsilon\text{-NFA}}$를 $M_{\epsilon-NFA}$전체의 오토마타 클래스라 부른다.

(**쉬운정리** 3) $\mathbb{M}_{\text{DFA}}\subsetneq\mathbb{M}_{\text{부분}}\subsetneq\mathbb{M}_{\text{NFA}}\subsetneq\mathbb{M}_{\varepsilon\text{-NFA}}$.

$\mathbb{M}_{\text{NFA}}\subsetneq\mathbb{M}_{\varepsilon\text{-NFA}}$이므로 쉽다.

(**정의** 10)와 (**쉬운정리** 3)으로 확장의 첫 번째와 두 번째 작업인 확장된 오토마타 클래스의 (A) **정의**와 (B) **확장**을 마치었다. 확장에 마지막 작업으로 **같은 일을 하면서** 상태변화함수 δ가 전체함수인 오토마타 클래스로 바꾸어보자.

(**정의** 12) 상태변화함수 δ의 **첫 번째 정의역**을 상태(Q)에서 상태집합(2^Q)으로 확장하고 **두 번째 정의역**을 $\{\epsilon\}$ 분리하여 따로 생각하자.

$\delta':\ 2^Q\times\Sigma\ \rightarrow\ 2^Q$[1)]

$$\delta'(P,a)\ \stackrel{\text{def}}{=}\ \{q\in Q|\ p\in P,\delta(p,a)=q,a\in\Sigma\}$$

$$=\ \bigcup_{p\in P}\delta_{\epsilon-NFA}(p,a)\ =\ \delta_{\epsilon-NFA}(P,a)$$

(**정의** 13) 정의 12에서 분리된 $\epsilon-move$를 ϵ으로 간단하게 써보자.

$\epsilon:\ 2^Q\ \rightarrow\ 2^Q$

$$\epsilon(P)\ \stackrel{\text{def}}{=}\ \{q\in Q|\ p\in P,\delta(p,\epsilon)=q\}$$

$$=\ \bigcup_{p\in P}\delta_{\epsilon-NFA}(p,\epsilon)\ =\ \delta_{\epsilon-NFA}(P,\epsilon)$$

(**정의** 14) 반복 $\epsilon-move$ $\epsilon^i(i\geqq 0)$을 정의하자.

$\epsilon^i:\ 2^Q\ \rightarrow\ 2^Q$

$\epsilon^0(P)\ \stackrel{\text{def}}{=}_{\text{B}}\ P$ 단 $P\subseteq Q(P\in 2^Q)$.

$\epsilon^{i+1}(P)\ \stackrel{\text{def}}{=}_{\text{R}}\ \epsilon(\epsilon^i(P))$ 단 $i\geqq 1$, $P\subseteq Q(P\in 2^Q)$.

(**정의** 15) $\epsilon-move$의 반복합 ϵ^*를 정의하자.

$\epsilon^*:\ 2^Q\ \rightarrow\ 2^Q$

1) 앞 장의 NFA (**정의** 8)과 같다.

$$\epsilon^*(P) = \bigcup_{i \in N_0} \epsilon^i(P) \qquad \text{단 } P \subseteq Q(P \in 2^Q).$$

즉 ϵ^*는 ε-NFA에서 추가로 정의된 $\delta(q,\epsilon)$을 여러 번 하는 것이다.

임의의 $M_{\epsilon-NFA}$를 **같은 일을 하는** M_{DFA}로 바꾸어주는 알고리즘을 보자.

(알고리즘) **function** ε-NFA_to_DFA($Q_{\epsilon-NFA}$(ε-NFA상태집합), Σ, $\delta_{\epsilon-NFA}$(ε-NFA상태변화함수), $q_0 \in Q_{\epsilon-NFA}$, $F_{\epsilon-NFA} \subseteq Q_{\epsilon-NFA}$) **returns** ($Q_{DFA}$(DFA상태집합), Σ, δ_{DFA}(DFA상태변화함수), $q_0^{DFA} \in Q_{DFA}$, $F_{DFA} \subseteq Q_{DFA}$);

```
variable P ⊆ Q_{ε-NFA}; a ∈ 입력문자집합(Σ);
  q_0^{DFA} = ε*({q_0}); q_0^{DFA} ⊆ Q_{DFA} 2);
  repeat
     for P ∈ Q_{DFA} do
        for a ∈ Σ do
           ε*(δ_{ε-NFA}(P, a)) ⊆ Q_{DFA}; (δ_{DFA}(P, a)=ε*(δ_{ε-NFA}(P, a))) ⊆ δ_{DFA}; 3)
        od
        if (P ∩ F_{ε-NFA}) ≠ ∅ → P ⊆ F_{DFA} else skip fi
        od
     until no more new states are added to Q_{DFA}
  return M_{DFA} = (Q_{DFA}, Σ, δ_{DFA}, q_0^{DFA}, F_{DFA});
 end function NFA_to_DFA;
```

(부분정리) $Q_{DFA} = 2^{Q_{\epsilon-NFA}} = \{P \mid P \subseteq Q_{\epsilon-NFA}\} = \{P \mid P \in Q_{DFA}\}$

$P \subseteq Q_{\epsilon-NFA} (\equiv P \in 2^{Q_{\epsilon-NFA}} \equiv P \in Q_{DFA})$, $a \in \Sigma$.

$\delta_{DFA}(P,a) = \epsilon^*(\delta_{\epsilon-NFA}(P,a))$.

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

(증명1) 위의 (알고리즘)은 임의의 ε-NFA에서 위의 (부분정리)을 만족하는 DFA를 만든다.

(증명2) $L(M_{\epsilon-NFA}) = L(M_{DFA})$

(생략)

(**중요정리** 3) $\mathbb{M}_{\mathrm{DFA}}$ 와 $\mathbb{M}_{부분}$, $\mathbb{M}_{\mathrm{NFA}}$, $\mathbb{M}_{\varepsilon\text{-NFA}}$는 모두 **같은 일을 하는**(**같은**) 오토마타 클래스이다.

2.1.4 입력문자열(Σ^*)에 대한 상태변환을 허용한다

(해결책) 중간에 입력문자 하나씩 보고 상태를 바꾸는 중간상태들을 추가한다.

---

2) 이러한 표현은 declarative 언어에서 쓰는 표현으로 앞 장의 procedural 언어의 $Q_{DFA} := Q_{DFA} \cup \{q_0^{DFA}\}$와 같다. 또한 엄밀히 쓰면 $\{q_0^{DFA}\} \subseteq Q_{DFA}$로 써야하나, 집합을 위한 { }기호를 빼고 $q_0^{DFA} \subseteq Q_{DFA}$로 쓰기도 한다.

3) $Q = \epsilon^*(\delta_{\epsilon-NFA}(P,a))$;라 하고, $Q \subseteq Q_{DFA}$; $(\delta_{DFA}(P,a) = Q) \subseteq \delta_{DFA}$와 같다.

오토마타를 더 확장해보자. $\epsilon-move$뿐만이 아니라 길이 2 이상의 입력문자열에 의한 상태변화도 허용한다. 즉 기존에 ε-NFA에서 입력문자 여러 개를 보고 한꺼번에 상태를 바꿀 수 있게 한다.

이 상황을 상태변화함수 δ로 표시하면 아래와 같다.

$$\delta:\ Q \times \Sigma^* \rightarrow 2^Q.$$

즉 상태변화함수의 두 번째 도메인을 Σ^*로 확장하여, $\delta(q, a_1 a_2 \cdots a_k) = \{p_1, p_2, \cdots, p_n\}$을 허용한다.

(정의 12) 확장된 오토마타 $M_{XFA} = (Q, \Sigma, \delta_{XFA}, q_0, F)$는 아래와 같이 정의한다.

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

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

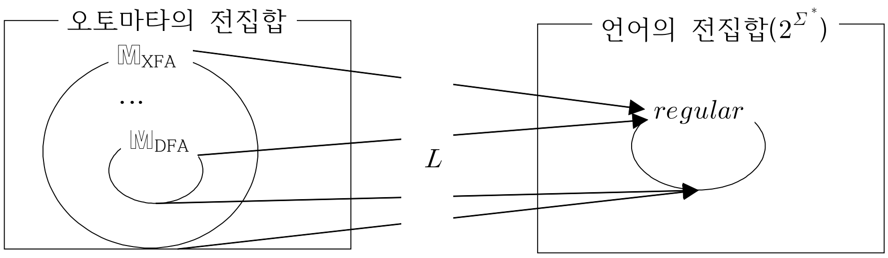
(정의 13) $\mathbb{M}_{\text{XFA}}$를 M_{XFA}전체의 오토마타 클래스라 부른다.

(쉬운정리 4) $\mathbb{M}_{\text{DFA}} \subsetneq \mathbb{M}_{\text{부분}} \subsetneq \mathbb{M}_{\text{NFA}} \subsetneq \mathbb{M}_{\varepsilon\text{-NFA}} \subsetneq \mathbb{M}_{\text{XFA}}$.

$\mathbb{M}_{\varepsilon\text{-NFA}} \subsetneq \mathbb{M}_{\text{XFA}}$이므로 쉽다.

(정의 12)와 **(쉬운정리 4)**로 확장의 첫 번째와 두 번째 작업인 확장된 오토마타 클래스의 (A) **정의**와 (B) **확장**을 마치었다. 확장에 세 번째 작업으로 **같은 일을 하면서** 상태변화함수 δ가 입력문자 하나에 대하여 전체함수인 오토마타로(DFA) 바꾸는 작업은, (1) 길이가 0인 ε-move는 2.1.3의 알고리즘을 이용하고, (2) 길이 2 이상인 문자열(길이 n)에 관한 상태변환은 중간에 문자 하나만 보고 상태를 바꾸는 중간 상태 $(n-1)$개를 넣어줌으로 쉽게 해결되므로 생략한다.

(중요정리 4) $\mathbb{M}_{\text{DFA}}$ 와 $\mathbb{M}_{\text{부분}}$, $\mathbb{M}_{\text{NFA}}$, $\mathbb{M}_{\varepsilon\text{-NFA}}$, $\mathbb{M}_{\text{XFA}}$는 모두 **같은 일을 하는**(**같은**) 오토마타 클래스이다.



확장된 오토마타 $\mathbb{M}_{\text{XFA}}$를 줄여서 오토마타 $\mathbb{M}_{\text{FA}}$라 부르기도 한다.

(마지막 중요정리) $\mathbb{M}_{\text{DFA}}$, $\mathbb{M}_{\text{부분}}$, $\mathbb{M}_{\text{NFA}}$, $\mathbb{M}_{\varepsilon\text{-NFA}}$, $\mathbb{M}_{\text{FA}}$는 모두 **같은 언어 클래스**(**정규**언어)를 정의하는 오토마타 클래스이고 서로 바꿀 수 있으므로, **구분 없이** 사용한다.

$\mathbb{M}_{\text{DFA}}$	$\delta_{DFA}:\ Q \times \Sigma \rightarrow Q$	$Q \times \Sigma^* \rightarrow Q$
$\mathbb{M}_{\text{부분}}$	$\delta_{부분}:\ Q \times \Sigma \rightarrow Q \cup \{\varnothing\}$	$Q \times \Sigma^* \rightarrow Q$

KAIST 전산학과 최광무

$\mathbb{M}_{\mathrm{NFA}}$	$\delta_{NFA}: Q \times \Sigma \to 2^Q$	$2^Q \times \Sigma^* \to 2^Q$
$\mathbb{M}_{\varepsilon\text{-NFA}}$	$\delta_{\epsilon-NFA}: Q \times \{\Sigma \cup \{\epsilon\}\} \to 2^Q$	$2^Q \times \Sigma^* \to 2^Q$
$\mathbb{M}_{\mathrm{FA}}$	$\delta_{FA}: Q \times \Sigma^* \to 2^Q$	$2^Q \times \Sigma^* \to 2^Q$