11. 정규 언어

충북대학교
이 재 성

학습내용

- 정규문법 및 정규언어 이론
- 정규 표현
- 정규 표현식

정규 문법과 정규 언어

정규 문법의 사용

- 컴파일러 어휘 분석 과정에서 모형을 만드는데 사용

정규 문법

- Type 3 문법 (*N. Chomsky 계층*)

  RLG : $A \rightarrow tB$, $A \rightarrow t$

  LLG : $A \rightarrow Bt$, $A \rightarrow t$

  - 여기서, $A, B \in V_N$ 이고 $t \in V_T^*$.

- 우선형 형태의 규칙과 좌선형 형태의 규칙이 혼합되어 있으면 정규 문법이 아니다.

  예를 들어,

  $G : S \rightarrow aR \quad S \rightarrow c \quad R \rightarrow Sb$

  $L(G) = \{a^n cb^n \mid n \geq 0\}$ 은 context-free 언어이다.

■ 정의

(1) 각 생성 규칙이 다음과 같을 때 정규문법이라 한다.

i) $A \to aB$, $A \to a$, 여기서 $a \in V_T$, $A, B \in V_N$.

ii) $S \to \varepsilon \in P$ 이면, S는 오른쪽에 나타나지 않아야 한다.

(2) 정규 문법에 의해 생성된 A 언어는 정규 언어(rl)이다.

ex) $L = \{ a^n b^m \mid n, m \geq 1 \}$ 은 정규 언어.

$S \to aS \mid aA$

$A \to bA \mid b$

[정리] 정규문법의 생성 형태는 우선형 문법으로부터 유도할 수 있다.

(증명) $A \rightarrow tB$, 여기서 $t \in V_T$.

$t = a_1 a_2 \ldots a_n$ 이면, $a_i \in V_T$.

$A \rightarrow a_1 A_1$

$A_1 \rightarrow a_2 A_2$

$\ldots$

$A_{n-1} \rightarrow a_n B$.

right-linear grammar :
$A \rightarrow tB$ or $A \rightarrow t$,
where $A, B \in V_N$ and $t \in V_T^*$.

$t = \varepsilon$이면, $A \rightarrow B$ (single production) or $A \rightarrow \varepsilon$ (epsilon production).

$\Rightarrow$ 이 형태의 생성 규칙들은 쉽게 제거할 수 있다.

ex) $S \rightarrow abcA$ $\Rightarrow$ $S \rightarrow aS_1$, $S_1 \rightarrow bS_2$ $S_2 \rightarrow cA$

$A \rightarrow bcA$ $\Rightarrow$ $A \rightarrow bA_1$, $A_1 \rightarrow cA$

$A \rightarrow cd$ $\Rightarrow$ $A \rightarrow cA_1'$, $A_1' \rightarrow d$

동치(Equivalence) 관계

1. 언어 L은 우선형 문법에 의해 생성된다.
2. 언어 L은 좌선형 문법에 의해 생성된다.
3. 언어 L은 정규 문법에 의해 생성된다.

⇨ 1, 2, 3은 모두 같으며, 정규 언어**임**

[예] $L = \{a^n b^m \mid n, m \geq 1\}$: rl

$S \rightarrow aS \mid aA$

$A \rightarrow bA \mid b$

■ 토큰 구조 정의에 정규 언어를 사용

1) 토큰의 구조는 간단하기 때문에 정규 문법으로 표현할 수 있다.
2) context-free 문법보다는 정규 문법으로부터 효율적인 인식기를 쉽게 구현할 수 있다.
3) 컴파일러의 전반부를 모듈러하게 나누어 구성할 수 있다.
   (스캐너+ 파서)

- 문법G가 **정규 문법**이면 언어의 표현을 체계적으로 구하여 **정규 표현**으로 나타낼 수 있다.

G = 정규문법 이면, L: 정규 표현.

- 예 (뒤에 배울 정규표현식 풀이를 이용하여 RG를 RE로 변환):
  - RG:

    LLG: Ident → letter | Ident • letter | Ident • digit

    RLG: Ident → letter • Ident_2

    Ident_2 → letter • Ident_2 | digit • Ident_2
  - RE:

    letter •(letter + digit)*

정규 표현

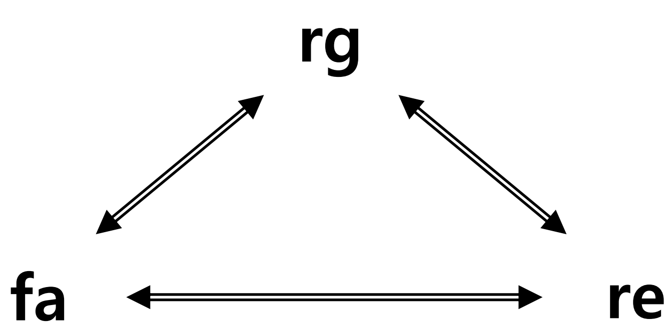
- 정규 언어를 표현하기 위한 하나의 방법

- 정규 언어의 동등한 표현 방법들

  (1) regular grammar(rg)

  (2) regular expression(re)

  (3) finite automata(fa)



정규 표현 정의

- 정의 :
  - I. 기본 소자 : $\emptyset$, ε, $a \in T$
    - (1) $\emptyset$ 는 공집합을 나타내는 정규 표현이다.
    - (2) ε 은 집합$\{\varepsilon\}$를 나타내는 정규 표현이다.
    - (3) $a \in T$ 는 집합$\{a\}$를 나타내는 정규 표현이다.
  - II. 순환식 : $+$, $\bullet$, $*$

    P와 Q가 정규 언어 L_p 와 L_q 를 나타내는 정규 표현이라면,
    - (1) (P + Q)는 $L_p \cup L_q$ 를 나타내는 정규 표현이다. (union)
    - (2) (P • Q)는 $L_p \cdot L_q$ 를 나타내는 정규 표현이다. (concatenation)
    - (3) (P)* 은 다음을 나타내는 정규 표현이다. (closure)

      $\{\varepsilon\} \cup L_p \cup L_p^2 \cup \ldots \cup L_p^n \ldots$

    우선순위 : $+ < \bullet < *$
  - III. 이외에 어떠한 것도 정규 표현이 될 수 없다.

예) (0+1)*는 언어{0,1}*를 의미

(0+1)*011은 0과 1로 이루어진 스트링 뒤에 011이 나오는 형태

■ 정의: α가 정규 표현일 때, L(α)는 α가 나타내는 언어

- α 와 β 가 정규 표현일 때,
  (1) $L(\alpha+ \beta) = L(\alpha) \cup L(\beta)$
  (2) $L(\alpha \beta) = L(\alpha) L(\beta)$
  (3) $L(\alpha^*) = L(\alpha)^*$
- 예 :
  (1) $L(a^*) = \{\varepsilon, a, aa, aaa, \ldots \} = \{a^n \mid n \geq 0\}$
  (2) $L((aa)^*(bb)^*b) = \{a^{2n}b^{2m+1} \mid n,m \geq 0\}$
  (3) $L((a+b)^*b(a+ab)^*) = \{ b, ba, bab, ab, bb, aab, bbb, \ldots \}$

- 정의 : 두 개의 정규 표현이 같은 언어를 표현할 때, 그 정규 표현은 같다고 한다.
  - $L(\alpha) = L(\beta)$ 이면 $\alpha = \beta$.

- 공리 : 정규 표현의 대수학적인 성질
  - α, β, γ 가 정규 표현일 때

A1. $\alpha+\beta = \beta+\alpha$	A2. $(\alpha+\beta) +\gamma = \alpha+ (\beta+\gamma)$
A3. $(\alpha\beta)\gamma = \alpha(\beta\gamma)$	A4. $\alpha(\beta+\gamma) = \alpha\beta +\alpha\gamma$
A5. $(\beta + \gamma)\alpha = \beta\alpha + \gamma\alpha$	A6. $\alpha+\alpha=\alpha$
A7. $\alpha + \phi = \alpha$	A8. $\alpha\phi = \phi = \phi\alpha$
A9. $\varepsilon\alpha = \alpha = \alpha\varepsilon$	A10. $\alpha^* = \varepsilon + \alpha\bullet\alpha^*$
A11. $\alpha^* = (\varepsilon + \alpha)^*$	A12. $(\alpha^*)^* = \alpha^*$
A13. $\alpha^* + \alpha = \alpha^*$	A14. $\alpha^* + \alpha^+ = \alpha^*$
A15. $(\alpha + \beta)^* = (\alpha^* \beta^*)^*$	

정규 표현식

■ 정의 : 정규 표현식

::= 계수가 정규 표현인 식을 정규 표현이라 한다.

ex) α,β가 정규 표현이면, X = αX+ β가 정규 표현식이다. 이때, X는 우측의 식이 그 비단말기호를 생성하는 언어임을 나타낸다.

정규 표현식의 해

$X = \alpha X + \beta.$

- 식의 양변에 $X = \alpha^*\beta$ 를 대입했을 때 각 변은 같은 언어를 나타낸다.

$$
\begin{aligned}
X &= \alpha X + \beta \\
&= \alpha(\alpha^*\beta) + \beta \\
&= \alpha\alpha^*\beta + \beta = (\alpha\alpha^* + \varepsilon)\beta = \alpha^*\beta.
\end{aligned}
$$

- 반복대입

$$
\begin{aligned}
X &= \alpha X + \beta \\
&= \alpha(\alpha X + \beta) + \beta \\
&= \alpha^2 X + \alpha\beta + \beta = \alpha^2 X + (\varepsilon + \alpha)\beta \\
&\quad \ldots \\
&= \alpha^{k+1} X + (\varepsilon + \alpha + \alpha^2 + \ldots \alpha^k)\beta \\
&= (\varepsilon + \alpha + \alpha^2 + \ldots + \alpha^k + \ldots)\beta = \alpha^*\beta.
\end{aligned}
$$

■ 모든 정규 표현식이 유일해를 갖는 것은 아니다.

$$\mathbf{X = \alpha X + \beta}$$

(a) ε 이 α에 속해 있지 않을 때, $\mathbf{X = \alpha^* \beta}$ 는 유일해이다.

(b) ε 이 α에 속해 있을 때, $\mathbf{X = \alpha^*(\beta + L)}$ 는 어떤 언어 L에 대해 유한한 해를 가진다.

⇒ 가장 작은 해 : $X = \alpha^*\beta$.

ex) $X = X + a$: 유일해가 아니다.

⇒ $X = a + b$ or $X = b^*a$ or $X = (a + b)^*$ etc.

$$
\begin{aligned}
X &= X + a \\
&= a + b + a \\
&= a + a + b \\
&= a + b.
\end{aligned}
\qquad
\begin{aligned}
X &= X + a \\
&= b^*a + a \\
&= (b^* + \varepsilon)\, a \\
&= b^*a
\end{aligned}
$$

정규 문법을 정규 표현으로 변환

- 정규 문법 G가 생성하는 언어 L(G)를 정규 표현으로 변환

$$G \xrightarrow{\text{유도}} L$$

G = 정규문법 이면, L: 정규 표현.

- L(A) 여기서 $A \in V_N$ 는 A에 의해 생성된 언어를 나타낸다.
  정의에 따라, S 가 시작 심벌이면, L(G)= L(S).

- Two steps :
  1. G로부터 연립방정식(정규표현식)을 만든다.
     $A \to aB,\ \ A \to a$
     $L(A) = \{a\} \cdot L(B) \cup \{a\} \in A = aB + a$
     보통, $X \to \alpha \,|\beta|\, \gamma \Rightarrow X = \alpha + \beta + \gamma$.
  2. 이 식들을 푼다.

$$X = \alpha X + \beta \Leftrightarrow X = \alpha^* \beta.$$

ex1) $S \rightarrow aS \qquad S \rightarrow bR \qquad S \rightarrow \varepsilon \qquad R \rightarrow aS$

$$\begin{cases} L(S) = \{a\}L(S) \cup \{b\}L(R) \cup \{\varepsilon\} \\ L(R) = \{a\}L(S) \end{cases}$$

ree: $S = aS + bR + \varepsilon$

$R = aS$

$$\begin{aligned} S &= aS + baS + \varepsilon \\ &= (a + ba)S + \varepsilon \\ &= (a + ba)^{*}\varepsilon = (a + ba)^{*} \end{aligned}$$

ex2) $S \rightarrow aA \mid bB \mid b$ $A \rightarrow bA \mid \varepsilon$ $B \rightarrow bS$

ree: $S = aA + bB + b$

$A = bA + \varepsilon \Rightarrow A = b^*\varepsilon = b^*$

$B = bS$

$\therefore\ S = ab^* + bbS + b$

$= bbS + ab^* + b$

$= (bb)^*(ab^*+b)$

참고 문헌

- [1] Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman, “Compilers – Principles, Techniques, and Tools,” Bell Telephone Laboratories, Incorporated, 1986.
- [2] 오세만, “컴파일러 입문”, 정익사, 2004.