5.1 문맥자유문법(cfg)과 유도(derivation), 언어(language)

(**정의** 5.1) **문맥자유(Context-free) 문법(grammar)**[1] $G = (N, T, P, S)$ 는

(1) N[2]은 nonterminal 혹은 variable[3]이라 불리는 문자(symbol)에 집합이다.

(2) T는 terminal 혹은 입력문자라 불리는 문자에 집합이다.

단 $N \cap T = \varnothing$ 이고 $V = N \cup T$ 로 쓰고 V를 문법의 기본문자라 부르자.

(3) P은 (문법) 규칙(rule, production)이라고 부르는 순서쌍 (A, α)의 집합이다.

규칙 순서쌍 $(A, \alpha) \in P$ 는 $A \to \alpha \in P$[4] 로 쓰이기도 하고

규칙 좌변은 $A \in N$[5]이고 우변은 $\alpha \in (N \cup T)^* = V^*$이다.

(4) $S \in N$은 처음(start, axiom)문자라 부르는 특별한 넌 터미널이다.

(**정의** 5.2) 넌 터미널 $A \in N$를 규칙의 좌변으로 가지는 규칙이, $A \to \alpha_1$, $A \to \alpha_2$, $\cdots$, $A \to \alpha_k$일 때 $A \to \alpha_1$, $A \to \alpha_2$, $\cdots$, $A \to \alpha_k$를 A규칙(A-productions)라고 부르고, 짧게

$A \to \alpha_1 \mid \alpha_2 \mid \cdots \mid \alpha_k$로 쓰기도 한다.

문맥자유 문법규칙의 좌변은 넌 터미널 문자(N) 하나이고 우변은 넌 터미널(N)이나 터미널(T) 문자 여럿(기본문자열(V^*))이다. 따라서 문법규칙을 사용하면 터미널과 넌 터미널 문자들이 여러 개 나타난다. 그 중에 터미널은 문법규칙에 좌변에는 오지 않음에 유의하라.

(**정의** 5.3) **유도(derivation)** $\Rightarrow$는 기본문자열 V^*에서 정의된 관계($\Rightarrow \subseteq V^* \times V^*$)이다.

문법 $G = (N, T, P, S)$ 에서 $\alpha, \gamma \in V^*$, $B \in N$, $B \to \beta \in P$라 하자. 이 때 기본문자열 $\alpha B \gamma$가 기본문자열 $\alpha \beta \gamma$를 유도한다(derive)하고 $\alpha B \gamma \Rightarrow_G \alpha \beta \gamma$ 로 쓰고 문법 G가 잘 알려져 있으면 $\Rightarrow_G$에서 G를 생략하고 $\Rightarrow$로만 쓰기도 한다. $\Rightarrow_G = \{(\alpha B \gamma, \alpha \beta \gamma) \mid B \to \beta \in P\}$

처음에 넌 터미널 S에서 시작하여 문법규칙 P중에 처음 문자 S가 좌변인 S규칙 $S \to \sigma$[6]를 찾아 S를 그 규칙의 우변 $\sigma \in (N \cup T)^* = V^*$ 로 바꾸고, 바꾼 기본문자열 σ에 넌 터미널 문자 $A \in N$이 있으면 다시 A규칙에서 찾아 그 A규칙의 우변으로 바꾸는 과정의 연속이 문맥자유문법의 **유도**(derivation)이다. 이 유도는 넌 터미널 문자를 포함하지 않고 터미널 문자열(T^*) 뿐 이면, 끝이 난다.

(**정의** 5.4) 유도 $\Rightarrow_G$의 $n (n \geq 0)$번 **반복** $\Rightarrow_G^n$을 아래와 같이 recursive하게 정의한다.

$\Rightarrow_G^0 \stackrel{\text{def}}{=}_B id_{V^*}$[7] $\qquad n = 0,$

---

1) 언어학자이고 철학자이며 전산학에도 큰 영향을 준 N. Chomsky가 1950년대 말 처음으로 시작하였다. 생성문법(generative grammar)이라고 부르기도 한다.

2) 교과서에서는 N대신 V를 쓰고 있지만, 우리는 V를 다른 용도($N \cup T$)로 쓰기 위하여 N을 쓴다.

3) Syntactic category라고 부르기도 한다.

4) 문법규칙 P를 $P \subseteq N \times (N \cup T)^*$로 정의할 수도 있다.

5) 이것이 문맥자유(context-free)라고 부르는 이유이다.

6) S규칙이 하나 이상일 수 있으므로 이 유도과정은 nondeterministic하다. σ는 그리스 문자로 영어 소문자 s에 해당한다.

$\Rightarrow_G^n \overset{\text{def}}{=}_R \Rightarrow_G^{n-1} \circ \Rightarrow_G \qquad n \geqq 1.$

(**정의** 5.5) 유도 $\Rightarrow_G$의 **반복 합** $\Rightarrow_G^*$를 아래와 같이 정의한다.

$\Rightarrow_G^* \overset{\text{def}}{=} \bigcup_{i \in N_0} \Sigma^i \overset{\text{def}}{=} \Rightarrow_G^0 \cup \Rightarrow_G^1 \cup \Rightarrow_G^2 \cup \cdots \qquad$ 단 $N_0 \overset{\text{def}}{=} \{0, 1, 2, \cdots\}.$

(**정의** 5.6) 문법 $G = (N, T, P, S)$에서 $S \Rightarrow_G^* \alpha (\alpha \in V^*)$이면 기본문자열 α를 **문장형태**(sentential form)이라하고, 특히 문장형태 $S \Rightarrow_G^* x$가 입력문자열($x \in T^*$)만으로 이루어져 있을 때 **문장**(sentence)이라 한다.

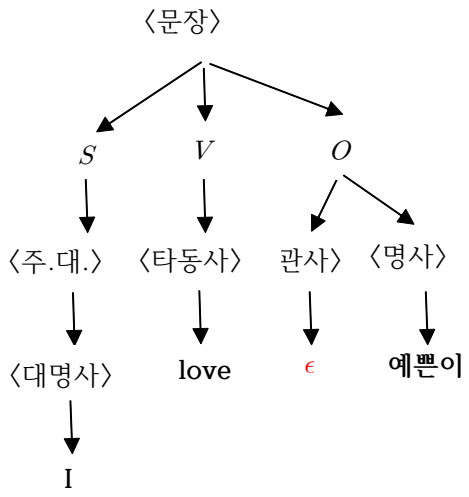
(예 5.1) 영어 문법 중 3형식 문장 중 일부를 문법 $G_3 = (N, T, P, \langle$문장$\rangle)$로 쓰자.

$N = \{$ 〈문장〉, S, V, O, 〈관사〉, 〈명사〉, 〈주격대명사〉, 〈타동사〉, 〈목적격대명사〉 $\}$

$T = \{$ **the**, **a**, **boy**, **girl**, **예쁜이**, **I**, **you**, **he**, **she**, **love**, **loves**, **me**, **him**, **her** $\}$

$P = \{$ 〈문장〉 $\rightarrow SVO$,
$S \rightarrow$ 〈관사〉〈명사〉 | 〈주격대명사〉,
$V \rightarrow$ 〈타동사〉,
$O \rightarrow$ 〈관사〉〈명사〉 | 〈목적격대명사〉,
〈관사〉 $\rightarrow$ **the** | **a** | ϵ,
〈명사〉 $\rightarrow$ **boy** | **girl** | **예쁜이**,
〈주격대명사〉 $\rightarrow$ **I** | **you** | **he** | **she**,
〈타동사〉 $\rightarrow$ **love** | **loves**,
〈목적격대명사〉 $\rightarrow$ **me** | **you** | **him** | **her** $\}$

(예 5.2) 〈문장〉 $\Rightarrow SVO$
$\Rightarrow$ 〈주격대명사〉VO
$\Rightarrow$ 〈대명사〉VO
$\Rightarrow$ I VO
$\Rightarrow$ I 〈타동사〉O
$\Rightarrow$ I love〈관사〉〈명사〉
$\Rightarrow$ I love ϵ 〈명사〉
$=$ I love 〈명사〉
$\Rightarrow$ I love **예쁜이**



문장 "I love 예쁜이"의 파스(parse) 나무(tree)

7) 임의의 집합 A에 관하여 $id_A \overset{\text{def}}{=} \{(a, a) \mid a \in A\}$이다. id_{V^*}는 무엇일까?

(**정의** 5.7) 문법 $G = (N, T, P, S)$의 문장들의 집합을 문법의 **언어** $L(G)$라 하고, 아래와 같이 정의한다.

$$L(G) = \{x \in T^* \mid S \Rightarrow_G^* x\}.$$

(**정의 5.8**) 문맥자유(context-free) 언어(language)

임의의 언어 L을 만들어내는 **문맥자유문법** G가 있을 때, $L = L(G)$, 언어 L을 **문맥자유언어**라 부른다.

(**정의 5.9**) Regular(정규) Grammar $G = (N, T, P, S)$는

(1) N nonterminal 어휘, (2) T terminal 어휘, (4) $S \in N$ 처음(start, axiom)문자는 context-free grammar와 같다.

(3) P는 (문법) 규칙(rule, production) 순서쌍 $A \to \alpha$의 집합인데, 문법규칙 $A \to \alpha \in P$의 좌변은 N으로 같은데, 우변 α는 $(T^* \cdot N)$ 또는 T^*로 제한된다. 즉 $A, B \in N$이고 $x, y \in T^*$일 때 $A \to xB$ 또는 $A \to y \in P$로 제한된다[8].

문맥자유 문법의 문법규칙은 $P_{cfg} \subseteq N \times (N \cup T)^*$이고 정규문법의 문법규칙은 $P_{rg} \subseteq N \times (T^* \cdot N \cup T^*)$인데, 문법규칙의 좌변은 넌 터미널 N으로 같고, 우변은 정규문법 $(T^* \cdot N) \cup T^*$이 문맥자유문법$(N \cup T)^*$의 **부분집합**이므로 정규문법은 문맥자유문법의 한 **종류**라고 볼 수 있다.

(예 5.3) 문법 $G_{ee} = (\{짝짝, 짝홀, 홀짝, 홀홀\}, \{0, 1\}, P_{ee}, 짝짝)$는 문맥자유문법이고 정규문법이다.

$$\begin{aligned} P_{ee}: \quad & 짝짝 \to 0홀짝 \mid 1짝홀 \mid \epsilon, \\ & 홀짝 \to 0짝짝 \mid 1홀홀, \\ & 짝홀 \to 0홀홀 \mid 1짝짝, \\ & 홀홀 \to 0짝홀 \mid 1홀짝. \end{aligned}$$

$$짝짝 \Rightarrow^{짝짝 \to 0홀짝} 0홀짝 \Rightarrow^{홀짝 \to 0짝짝} 00짝짝 \Rightarrow^{짝짝 \to 1짝홀} 001짝홀 \Rightarrow^{짝홀 \to 0홀홀} 0010홀홀 \Rightarrow^{홀홀 \to 1홀짝} 00101홀짝 \Rightarrow^{홀짝 \to 1짝짝} 001010짝짝 \Rightarrow^{짝짝 \to \epsilon} 001010.$$

(예 5,4) Palindrome 문법 $G_{pal} = (\{P\}, \{0, 1\}, P_{pal}, P)$은 문맥자유문법이나 정규문법은 아니다.

$$P_{pal}: \quad P \to \epsilon \mid 0 \mid 1 \mid 0P0 \mid 1P1.$$

(정리 5.1) 임의의 finite automaton $A_{fa} = (Q, \Sigma, \delta, q_0, F)$와 equivalent한 정규문법 $G_{rg} = (N, T, P, S)$을 다음과 같이 만들 수 있다(역도 가능; and vice versa).

(1) $Q \leftrightarrow N$: $q \in Q \quad \leftrightarrow A_q \in N$

(2) $\Sigma \quad = T.$

---

8) 문맥자유문법(CFG)의 문법규칙 $P_{cfg} \subseteq N \times (N \cup T)^*$이고 정규문법(RG) $P_{rg} \subseteq N \times (T^* \cdot N) \cup T^*$이다.

(3) $\delta,\ F \leftrightarrow\ P$: $p \in \delta(q,x)$ $\leftrightarrow\ A_q \to xA_p \in P$

$f \in F$ $\to\ A_f \to \epsilon \in P$

$f \in \delta(q,x),\ f \in F$ $\leftarrow\ A_q \to x \in P$

(4) $q_0 \in Q$ $\leftrightarrow\ A_{q_0} = S \in N$

(최종증명) $L(A_{fa}) = L(G_{rg})$.

(증명) $\forall x \in \Sigma^*$: $p \in \delta^*(q,x)$, iff $A_q \Rightarrow^* xA_p$.

(증명1) $\forall x \in \Sigma^*$: $p \in \delta^*(q,x) \Rightarrow A_q \Rightarrow^* xA_p$.

(증명2) $\forall x \in \Sigma^*$: $A_q \Rightarrow^* xA_p \Rightarrow p \in \delta^*(q,x)$.

(최종증명) $\forall x \in \Sigma^*$: $\delta^*(q_0,x) \in F \Leftrightarrow S \Rightarrow^* x$

(중요사실) 오토마타의 글래스 $\mathbb{M}_{\text{FA}}$(Finite State automata; 유한상태기계)와 문법의 클래스 $\mathbb{G}_{\text{RG}}$(정규문법; Regular Grammars)는 모두 같은 정규언어(Regular Languages) 클래스를 만들어내므로 같다.

(**사실 5.1**) 정규문법은 문맥자유문법의 **적절한 하위계급**(properly contained class)이다.
(**정리 5.2**) **문맥자유언어는 정규언어**의 적절한 상위계급이다.