

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

(정의 5.1) 문맥자유(context-free) 문법(grammar)<sup>1)</sup>  $G = (N, \Sigma, P, S)$ 는

(1)  $N^2$ 은 nonterminal 혹은 variable<sup>3)</sup>이라 불리는 문자(symbol)에 집합이다.

(2)  $\Sigma$ 는 terminal 혹은 입력문자라 불리는 문자에 집합이다.

단  $N \cap \Sigma = \emptyset$ 이고  $V = N \cup \Sigma$ 로 쓰고  $V$ 를 문법의 기본문자라 부르자.

(3)  $P$ 은 (문법) 규칙(rule, production)이라고 부르는 순서쌍  $(A, \alpha)$ 의 집합이다.

규칙 순서쌍  $(A, \alpha) \in P$ 는  $A \rightarrow \alpha \in P$ 로 쓰이기도 하고

규칙의 좌변은  $A \in N^4$ 이고 우변은  $\alpha \in (N \cup \Sigma)^* = V^*$ 이다.

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

(정의 5.2) 넌터미널  $A \in N$ 를 규칙에 좌변으로 가지는 규칙이,  $A \rightarrow \alpha_1, A \rightarrow \alpha_2, \dots, A \rightarrow \alpha_k$

일 때  $A \rightarrow \alpha_1, A \rightarrow \alpha_2, \dots, A \rightarrow \alpha_k$ 를  $A$ 규칙( $A$ -productions)라고 부르고, 짧게

$A \rightarrow \alpha_1 \mid \alpha_2 \mid \dots \mid \alpha_k$ 로 쓰기도 한다.

문법규칙의 좌변은 **넌터미널 문자**이고 우변은 넌터미널과 입력문자를 모두 포함하는 **기본문자열**이다. 처음 넌터미널  $S$ 에서 시작하여 문법규칙  $P$ 중에 처음 넌터미널  $S$ 가 좌변인  $S$ 규칙  $S \rightarrow \sigma^5$ 를 찾아  $S$ 를 그 규칙의 우변  $\sigma \in (N \cup \Sigma)^* = V^*$ 로 바꾸고, 바꾼 **기본문자열**  $\sigma$ 에 **넌터미널 문자**  $A \in N$ 이 있으면 다시  $A$ 규칙에서 찾아 그  $A$ 규칙의 우변으로 바꾸는 과정의 연속이 문맥자유문법의 유도이다. 이 유도는 기본문자열이 넌터미널 문자를 포함하지 않으면 끝난다.

(예 5.1) 영어 문법 중 3형식 문장만 문법으로 표현해보자.

$\langle \text{문장} \rangle \rightarrow SVO$

$S \rightarrow \langle \text{관사} \rangle \langle \text{명사} \rangle \mid \langle \text{대명사} \rangle$

$V \rightarrow \langle \text{타동사} \rangle$

$O \rightarrow \langle \text{관사} \rangle \langle \text{명사} \rangle \mid \langle \text{대명사} \rangle$

$\langle \text{관사} \rangle \rightarrow \text{the} \mid \text{a} \mid \epsilon$

$\langle \text{명사} \rangle \rightarrow \text{boy} \mid \text{girl} \mid \text{예쁜이}$

$\langle \text{대명사} \rangle \rightarrow \text{I} \mid \text{you} \mid \text{he} \mid \text{she}$

$\langle \text{타동사} \rangle \rightarrow \text{love} \mid \text{loves}$

$\langle \text{문장} \rangle \Rightarrow SVO \Rightarrow \langle \text{대명사} \rangle VO \Rightarrow \langle \text{대명사} \rangle V \langle \text{관사} \rangle \langle \text{명사} \rangle$

$\Rightarrow \text{I} V \langle \text{관사} \rangle \langle \text{명사} \rangle \Rightarrow \text{I love} \langle \text{관사} \rangle \langle \text{명사} \rangle \Rightarrow \text{I love} \langle \text{명사} \rangle$

$\Rightarrow \text{I love 예쁜이}$

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

2) 교과서에서는  $N$ 대신  $V$ 를 쓰고 있지만, 우리는  $V$ 를 다른 용도로 쓰기 위하여  $N$ 을 쓴다.

3) Synatactic category라고 부르기도 한다.

4) 이것이 문맥자유(context-free)라고 부르는 이유이다.

5)  $S$ 규칙이 하나 이상일 수 있으므로 이 유도과정은 nondeterministic하다.  $\sigma$ 는 그리스 문자로  $\Sigma$ 의 소문자로 영어의 s에 해당한다.

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

문법  $G = (N, \Sigma, P, S)$ 에서  $\alpha, \gamma \in V^*, B \in N, B \rightarrow \beta \in P$ 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  $G$ 가 잘 알려져 있으면  $G$ 를 생략하고  $\Rightarrow$ 로만 쓰기도 한다<sup>6)</sup>.

(정의 5.4) 유도  $\Rightarrow$ 의 반복  $\Rightarrow^n (n \geq 0)$ 를 아래와 같이 recursive하게 정의한다.

$$\begin{aligned} \Rightarrow^0 &\stackrel{\text{B}}{=} \{id_{V^*}\}, \\ \Rightarrow^n &\stackrel{\text{R}}{=} \Rightarrow^{n-1} \circ \Rightarrow \quad n \geq 1. \end{aligned}$$

(정의 5.5) 유도  $\Rightarrow$ 의 반복합  $\Rightarrow^*$ 를 아래와 같이 정의한다.

$$\Rightarrow^* \stackrel{\text{B}}{=} \bigcup_{i \in N_0} \Sigma^i = \Rightarrow^0 \cup \Rightarrow^1 \cup \Rightarrow^2 \cup \dots \quad \text{단 } N_0 \stackrel{\text{B}}{=} \{0, 1, 2, \dots\}.$$

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

(정의 5.7) 문법  $G = (N, \Sigma, P, S)$ 에 **문장**의 집합 언어  $L(G)$ 를 아래와 같이 정의한다.

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

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

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

(사실 5.1) 정규문법은 문맥자유문법이나, 문맥자유문법 중에는 정규문법이 아닌 문법이 있다.

(증명) 문법  $G_{pal} = (\{P\}, \{0, 1\}, P_{pal}, P)$ 은 정규문법이 아니고 문맥자유문법이다.

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

(정리 5.1) 정규문법은 문법자유문법의 적절한 하위계급이다.

6) 앞 페이지의 세 번째 문단에 있는 유도에 관한 문장을 수학 기호를 이용하여 엄밀하게(formal)하게 정의한 것이 (정의 5.3)이다. 이 정의가 잘 되어있는지 자세히 살펴보아라.

5.2 문맥자유문법과 문법나무 그리고 유도순서

(정의 5.9) 쓴 문법열(rule string)을 표기하도록 유도를 확장, 즉  $\Rightarrow^\pi$  단  $\pi \in P^*$ .

(basis)  $\alpha \Rightarrow^\epsilon \alpha$ , For  $\alpha \in V^*$ .

(recursion) If  $\alpha \Rightarrow^\pi \beta \wedge \beta \Rightarrow^r \gamma$ , then  $\alpha \Rightarrow^{\pi \cdot r} \gamma$ . 단  $\pi \in P^*$ ,  $r \in P$ .

(정의 5.9) 문법나무(parse tree) 문법  $G = (N, \Sigma, P, S)$ 에서 문장  $S \Rightarrow^* x$ 에 대한 문법나무를 뿌리 깊은 나무(rooted tree)  $T_G(x) = (N \cup \Sigma, E, S)$ 로 정의한다.

$$E = \{1 \leq \forall i \leq n : (A, X_i) \mid A \rightarrow X_1 X_2 \cdots X_n \in \pi\}.$$

(관찰 5.1) 문법  $G = (N, \Sigma, P, S)$ 에서 문장  $S \Rightarrow^* x$ 에 대한 문법나무를 뿌리 깊은 나무 (rooted tree)  $T_G(x) = (N \cup \Sigma, E, S)$ 로

$S \Rightarrow^\pi x \in T^*$ 이고  $\pi \in P^*$ 일 때, 사용된 문법규칙으로 subtree를 만들면 전체 문장  $x \in T^*$ 에 대한 트리를 만들 수 있고, 이를 파스트리라 부른다.

문맥자유문법의 파싱(parsing)은 임의의 터미널 문자열  $x \in T^*$ 가 주어진 문법  $G$ 에 문장이면 해당하는 파스트리를 만들어주고, 문장이 아니면 이를 알려주는 과정이다.

문맥자유문법의 문장형태는 일반적으로 여러 개에 년 터미널을 가지는데, 어떤 년 터미널이 유도에 먼저 참여할까 하는 것은 파스트리에 최종모양과는 관계가 **없다**. 따라서 문장형태에서 가장 왼쪽에 있는 년 터미널부터 다시 쓰기를 하는 왼쪽부터 고쳐쓰기(leftmost derivation)과 오른쪽부터 고쳐쓰기(rightmost derivation)에 두 가지 극단적인 경우를 생각할 수 있다. 이를 각각  $\Rightarrow_{lm}$ 와  $\Rightarrow_{rm}$ 로 표시한다.

$$S \Rightarrow_{lm}^{\pi_L^1} xB\gamma \Rightarrow_{lm}^{B \rightarrow \beta} x\beta\gamma \Rightarrow_{lm}^{\pi_L^2} xy\gamma \Rightarrow_{lm}^{\pi_L^3} xyz, \quad \pi_L^1, \pi_L^2, \pi_L^3 \in P^*,$$

단  $x \in T^*$ ,  $\gamma \in (N \cup T)^*$ ,  $A \rightarrow \beta \in P$ ,  $y, z \in T^*$ ,  $\beta \Rightarrow^* y$ ,  $\gamma \Rightarrow^* z$ .

$$S \Rightarrow_{rm}^{\pi_R^1} \alpha Bz \Rightarrow_{rm}^{\alpha \rightarrow \beta} \alpha\beta z \Rightarrow_{rm}^{\pi_R^2} \alpha yz \Rightarrow_{rm}^{\pi_R^3} xyz, \quad \pi_R^1, \pi_R^2, \pi_R^3 \in P^*,$$

단  $z \in T^*$ ,  $\alpha \in (N \cup T)^*$ ,  $A \rightarrow \beta \in P$ ,  $y, x \in T^*$ ,  $\beta \Rightarrow^* y$ ,  $\alpha \Rightarrow^* x$ .

이 때  $\pi_L = \pi_L^1(A \rightarrow \beta)\pi_L^2\pi_L^3 \in P^*$ 라 하고  $\pi_R = \pi_R^1(A \rightarrow \beta)\pi_R^2\pi_R^3 \in P^*$ 라 하면  $\pi_L$ 과  $\pi_R$ 를 각각 문장  $x$ 에 대한 left parse(좌분석)와 right parse(우분석)라고 부른다. 문장  $x$ 에 대한 좌분석이나 우분석이 정해지면 해당하는 문법트리도 정해진다.

(정의 5.1)  $A$ 가 기본문자집합(vocabulary)이고, 관계  $P \subseteq A^* \times A^*$ 가 집합  $A^*$ 에서 정의된 이진 관계(binary relation)일 때 순서쌍  $R = (A, P)$ 를 기본문자  $A$ 에 관한 **고쳐 쓰기 틀**(rewriting system)이라고 부른다. 이 때 관계  $P$ 를 고쳐 쓰기의 production rule(규칙)이라고 부르고,  $\alpha, \beta \in A^*$ 에 대해  $(\alpha, \beta) \in P$ ,  $\alpha P \beta$  또는  $\alpha \rightarrow \beta \in P$ 로 쓰고  $\alpha$ 를 규칙의 좌변(Lefthand side; LHS)  $\beta$ 를 규칙의 우변(Righthand side; RHS)이라 각각 부른다.

(정의 5.2) **고쳐 쓰기**(유도; rewriting, derivation)

**고쳐 쓰기 틀**  $R = (A, P)$ 에서 규칙  $\alpha \rightarrow \beta \in P$ 에 대하여,  $\gamma, \delta \in A^*$ 가 임의의 문자열일 때,  $\gamma\alpha\delta \Rightarrow \gamma\beta\delta$ 로 쓰고 문자열  $\gamma\alpha\delta$ 를 규칙  $\alpha \rightarrow \beta \in P$ 를 써서  $\gamma\beta\delta$ 로 **다시 썼다**(rewrite; derive)고 하고, 특별히 **다시 쓴** 규칙  $r = \alpha \rightarrow \beta \in P$ 를 표현하고 싶을 때는  $\gamma\alpha\delta \Rightarrow^r \gamma\beta\delta$ 로 표현한다.

임의의 문자열의 부분문자열(substring)이 규칙  $\alpha \rightarrow \beta \in P$ 의 좌변  $\alpha$ 과 같으면 나머지 부분(prefix  $\gamma$ 과 suffix  $\delta$ )을 제외한  $\alpha$ 만 규칙의 우변  $\beta$ 로 바꾸어서 **다시 쓸**( $\gamma\alpha\delta \Rightarrow^{\alpha \rightarrow \beta} \gamma\beta\delta$ ) 수 있다.

다시 쓰는 관계가 가진 기본문자집합을 일부 분류하여 다음과 같이 **문법**을 정의한다.

(정의 5.3) 다시 쓰기 틀  $R = (N \cup T, P)$ 에서 **문법**  $G = (N, T, P, S)$ 을 아래로 정의한다.

- i)  $N$ 은 **년** 터미널(nonterminal; variable) 문자에 집합,
- ii)  $T$ 는 **터미널**(terminal) 문자에 집합, 단  $N \cap T = \emptyset$ ,
- iii) 규칙  $P$ 는  $(N \cup T)^*$ 에서 정의된 관계이고, 특히 **문법규칙**이라고도 부른다.
- iv)  $S \in N$ 은 **시작문자**(start symbol).

문법은 다시 쓰기 틀에서 문장(sentence)에 구분을 위하여 기본문자를 **터미널**과 **년 터미널**로 나누고, 문장 다시 쓰기의 **시작**을 위하여 시작문자  $S$ 를 별도로 정의한 것이다.