Type 3 Grammer

정규문법의 정의

RLG (우선형 문법) 우측 맨 끝에 V_n이 있거나 없거나
right linear grammer

LLG (좌선형 문법) 좌측 맨 끝에 V_n이 있거나 없거나

*정규문법은 반드시 RLG와 LLG 중 하나의 형태여야 한다
V_n이 위치가 오른쪽 끝과 왼쪽 끝에 섞여 있으면 정규문법이 아니다

좀더 단순하게 재정의
정규문법은 V_T 하나 V_n 하나, 또는 V_T 하나만 있는 생성규칙이다.

근데 제약조건 하나가 있다.
$S \to \varepsilon \in P$ 이면 다른 P의 RHS로 S가 등장하면 안된다

ex) $S \to abcA \Rightarrow S \to aS_1 \quad S_1 \to bS_2 \quad S_2 \to cA$

$A \to bcA \Rightarrow A \to bA_1 \quad A_1 \to cA$

$A \to cd \Rightarrow A \to cA_2 \quad A_2 \to d$

terminal이 ε이면, $A \to B$ (single production) 이거나
$A \to \varepsilon$ (epsilon production) 이다

이것들은 제거가 가능하다

정규문법에 의해 정의되면 정규언어 이다

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

$S \rightarrow Sb \mid Ab$ (좌선형 문법)
$A \rightarrow Aa \mid a$

* 좌상향 문법과 우상향 문법중 우상향 문법을 이용하는게 훨씬 편하다

정규언어의 기술방법
regular grammer
regular expression
finite automata
셋다 상호변환가능

〈regular expression (정규표현)〉 String들의 집합

기본요소: $\emptyset$, ε, $a \in T$
$\{\}$ 원소 0개 / $\{\varepsilon\}$ 원소1개 / $\{a\}$

연산자: $+, *, ^*$

$e_1 + e_2$ 는 $L_1 \cup L_2$ 이다 (union)
$e_1 * e_2$ 는 $L_1 * L_2$ 이다 (concatenation)
$(e_1)^*$ 는 $L_1^* = \{\varepsilon\} \cup L_1 \cup L_1^2 \cup \dots L_1^n \dots$ (closuer)

우선순위 $+ < * < ^*$

위 6가지로만 정규표현을 나타낸다

정규표현은 수학규칙도 성립한다

1 $\alpha+\beta=\beta+\alpha$ 2 $(\alpha+\beta)+\gamma=\alpha+(\beta+\gamma)$

3 $(\alpha\beta)\gamma=\alpha(\beta\gamma)$ 4 $\alpha(\beta+\gamma)=\alpha\beta+\alpha\gamma$

5 $(\beta+\gamma)\alpha=\beta\alpha+\gamma\alpha$ 6 $\alpha+\alpha=\alpha$

7 $\alpha+\emptyset=\alpha$ 8 $\alpha\emptyset=\emptyset=\emptyset\alpha$

9 $\varepsilon\alpha=\alpha=\varepsilon\alpha$ 10 $\alpha^*=\varepsilon+\alpha*\alpha^*$

11 $\alpha^*=(\varepsilon+\alpha)^*$ 12 $(\alpha^*)^*=\alpha^*$

13 $\alpha^*+\alpha=\alpha^*$ 14 $\alpha^*+\alpha^+=\alpha^*$

15 $(\alpha+\beta)^*=(\alpha^*\beta^*)^*$

정규표현식 (regular expression equation)

α β가 정규표현일때

정규표현식 $X=\alpha X+\beta$ 의 해는 $\alpha^*\beta$ 이다.

(해는 = 결과)

$$X=\alpha X+\beta$$

$$=\alpha(\alpha^*\beta)+\beta$$

$$=\alpha\alpha^*\beta+\beta=(\alpha\alpha^*+\varepsilon)\beta=\alpha^*\beta$$

정규표현식을 어디다 쓰나?

정규문법이 있으면 정규문법이 기술하는 정규언어가 있고
그 정규언어를 정규표현으로 기술할수 있다

정규문법을 통해 정규표현을 계산할수있다
→ 각 생성규칙을 정규표현식 써준다

정규문법을 식 형태로 쓰는게 정규표현식이다.
이때 식은 V_n의 개수만큼 나온다

이 정규표현식을 통해 시작기호에 대한 정규표현을 구해준다

[예제] $S \to aS$ $S \to bR$ $S \to \varepsilon$ $R \to aS$

$S = aS + bR + \varepsilon$
$R = aS$
] 연립방정식 하듯이 생각하면 된다

$S = aS + baS + \varepsilon$
$= (a+ba)S + \varepsilon$
$= (a+ba)^* \varepsilon = (a+ba)^*$

[예제]

L_{mn} 은 정규언어, L_{nn}은 정규언어X context free임

$L_{mn} = \{a^m b^n \mid m, n \geq 1\}$

$L_{nn} \{a^n b^n \mid n \geq 1\}$ — $S \to aSb$

V_n이 중간에 있음

V_n을 꼭 대문자 하나로 표현해야 하느냐?

No! V_n이 많아지면 알파벳 만으로 표현하기는 부족하다.

식별자(identifier)에 대한 정규표현

↳ 변수명, 함수명, 상수명

<letter> (<letter> + <digit>)*

정규표현의 등가성

정규표현 형태가 달라도 이것으로 표현되는 스트링 집합이 동일하면, 동일한 언어에 대한 정규표현이다.

ex) $aa^* \equiv a^*a$

$(ab)^*a \equiv a(ba)^*$