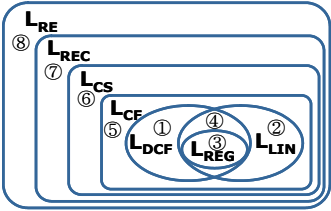


1. 다음 npda $M = (\{q_0, q_1, q_2\}, \{a, b\}, \{a, b, z\}, \delta, q_0, z, \{q_1, q_2\})$ 이 주어졌다. (10점)
 $\delta(q_0, a, z) = \{(q_1, a), (q_2, \lambda)\}$, $\delta(q_1, b, a) = \{(q_1, b)\}$, $\delta(q_1, b, b) = \{(q_1, b)\}$, $\delta(q_1, a, b) = \{(q_2, a)\}$
(1) 이 npda에 의해 accept되는 길이 4이하인 string을 모두 찾으시오.
(2) 이 npda에 의해 accept되는 language $L(M)$ 을 찾으시오.
(1) a ab aba abb abba abbb
(2) $L(ab^*) + L(abb^*a)$
2. Turing Machine $M = (\{q_0, q_1, q_f\}, \{a, b\}, \{a, b, \square\}, \delta, q_0, \square, \{q_f\})$ 이 주어졌다. (15점)
 $\delta(q_0, a) = (q_1, a, R)$, $\delta(q_1, b) = (q_0, b, R)$, $\delta(q_0, \square) = (q_f, \square, L)$
(1) M에 의해 accept되는 길이 10 이하인 string을 모두 찾으시오.
(2) M에 의해 accept되는 language $L(M)$ 를 집합으로 표현하시오.
(3) 위에서 정의된 δ 를 적절히 변경하여 L^R 을 accept하는 TM을 찾으시오.
(1) ab abab ababab abababab ababababab
(2) $L(M) = \{(ab)^n \mid n \geq 0\}$
(3) $\delta(q_0, b) = (q_1, b, R)$, $\delta(q_1, a) = (q_0, a, R)$, $\delta(q_0, \square) = (q_f, \square, L)$
3. 다음 각 Language가 오른쪽 그림의 영역 ①부터 ⑧ 중 어디에 속하는지 말하고, 간단히 증명하시오. (각 10점, 총 20점)
(1) $L = \{a^n b^n c^j \mid n \leq j\}$
답: ⑥
LBA가 존재하여 Context Sensitive임 (LBA구성 방법 간단히 설명)
Context Free가 아님은 Pumping Lemma를 이용하여 증명.
(2) $L = \{w \mid n_a(w) \geq n_b(w)\}$
답: ①
DPDA가 존재하므로 Deterministic Context Free임. (DPDA작성 방법 간단히 설명)
Linear가 아님은 Pumping Lemma를 이용하여 증명.
- 
4. 다음 명제의 참, 거짓을 판단하고 간단히 증명하시오. (각 5점, 총 30점)
(1) L_1 과 L_2 가 context sensitive 이면 $L_1 \cup L_2$ 를 accept하는 Linear Bounded Automata가 존재한다.
참. 각 Language에 대해 Noncontracting grammar가 존재. 각 Grammar의 start 변수가 S_1, S_2 일 때, $S \rightarrow S_1 \mid S_2$ 를 추가하면 $L_1 \cup L_2$ 를 위한 Noncontracting grammar가 되어 context sensitive가 되고, 따라서 LBA가 존재.
(2) L_1 이 deterministic context free이고 L_2 가 regular이면 $L_1 \cap L_2$ 는 deterministic context free 이다.
참. L_1 을 위한 DPDA와 L_2 를 위한 DFA를 결합하여 $L_1 \cap L_2$ 를 위한 DPDA를 만들 수 있음.
(3) $L = \{w \in \{a, b\}^* \mid n_a(w) = n_b(w), w \text{ does not contain a substring } aab\}$ 는 context free 이다.
참. $L_1 = \{w \in \{a, b\}^* \mid n_a(w) = n_b(w)\}$, $L_2 = L((a+b)^*aab(a+b)^*)$ 일 때. $L = L_1 \cap L_2$ 이고, L_1 은 CFL, L_2 는 regular. CFL은 regular intersection에 닫혀 있으므로 L 은 context free.

- (4) Recursive languages 는 concatenation에 닫혀있다.
참. $L = L_1 L_2$ 일 때, L_1, L_2 는 각각 membership algorithm이 존재. 임의의 string w 에 대해, $w = v_1 v_2$ 로 나눌 수 있는 가능한 모든 경우에 대해 각각 v_1, v_2 를 L_1, L_2 의 membership algorithm에 적용하여 L 을 위한 membership algorithm을 만들 수 있음.
(5) Language L_1 에 대한 enumeration procedure가 존재하면 L_1 은 recursive이다.
거짓. L_1 은 enumeratation procedure가 존재하므로 Recursively Enumerable language임. Recurively Enumerable하면서 Recursive하지 않은 Language는 존재.
- (6) $L = \{a^n b^n \mid n \geq 1\}$ 를 생성하는 $LL(k)$ grammar는 존재하지 않는다.
거짓. $S \rightarrow aS1b, S1 \rightarrow aS1b \mid \lambda$ 는 L 을 생성하는 $LL(2)$ grammar.
5. Turing Machine의 변형으로, 한 번에 1개 이상의 cell을 이동할 수 있는 Head를 가진 TM을 생각한다.
즉, 각 transition에서 head를 움직일 방향(L/R) 뿐 아니라 움직일 칸(cell)의 수도 정할 수 있다. (10점)
(1) 이 Turing machine의 formal definition을 쓰시오.
(2) 이 Turing machine은 standard turing machine과 equivalent한 power를 가지는가? 이에 대해 판단하고 그 이유를 간단히 쓰시오.
(1) $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$, $\delta: Q \times \Sigma \rightarrow Q \times \Gamma \times \{L, R\} \times N$ (N 은 자연수의 집합)
(2) equivalent한 power를 가짐. 움직이는 칸 수가 2개 이상인 경우, 중간 state를 따로 두어 움직인 칸 수 만큼 cell을 옮기는 transition을 추가함으로써 standard TM으로 새로운 TM을 simulate할 수 있음.
6. 함수 $f(n) = n \bmod 4$ (n 은 음이 아닌 정수)를 계산하는 Turing Machine을 만들고자 한다. (15점)
(1) TM에서 사용할 입출력값을 표현하는 코딩 방식을 정하고, 그에 맞추어 입력 symbol 집합 Σ 와 tape symbol 집합 Γ 를 정의하시오.
(2) (1)에서 정한 코딩방식을 이용하여 함수 $f(n)$ 에 대한 TM의 입출력을 정의하시오.
(3) $f(n)$ 을 계산하는 TM의 처리 과정을 단계별로 기술하시오.
(1) unary notation이나 binary notation등 자유롭게 사용 가능. (unary notation의 경우 $w(n) = 1^n$)
단 (2), (3)의 답과 일치해야 하며, 사용한 symbol들을 정확히 Σ 와 Γ 로 정의해야 함
(2) unary notaion을 사용한 경우의 예답
 $n \bmod 4$ 가 0보다 큰 경우: $q_0 w(n) \vdash q_f 0 w((n \bmod 4))$
 $n \bmod 4$ 가 0인 경우: $q_0 w(n) \vdash q_f 0$
(3) q_0 에서 시작하여 1을 하나씩 볼 때마다 오른쪽으로 움직이면서 state를 $q_1 \rightarrow q_2 \rightarrow q_3 \rightarrow q_0$ 로 바꿈.
 $q_0 \sim q_3$ state에서 blank를 만나면 $qr_1 \sim qr_3$ 으로 각각 두고 왼쪽으로 움직이면서 state를 $qr_3 \rightarrow qr_2 \rightarrow qr_1 \rightarrow qr_0$ 로 바꿈
 qr_0 에서 1을 만나면 0으로 바꾸로 state를 qd 로 바꿈
 qd 에서 1을 만나면 왼쪽으로 움직이면서 cell의 값을 모두 blank로 수정.
 qd 에서 blank를 만나면 qr 로 바꾸고 0을 찾아 오른쪽으로 이동.
 qr 에서 0을 만나면 state를 qf 로.
(처리과정은 반드시 위와 같을 필요 없으며, (2)에서 정의한 내용과 일관된 결과를 내면 됨)