오토마타



|  |  |  |  |
| --- | --- | --- | --- |
| 제출일 | 2019.04.18 | 전공 | 컴퓨터 소프트웨어공학과 |
| 과목 | 오토마타 | 학번 | 20174016 |
| 담당교수 | 박두순 교수님 | 이름 | 김혜린 |

**1.2절 연습문제**

**5.** = { ab, aa, baa }라 하자. 다음 문자열들 중 어떤 것이 에 속하는가: abaabaaabaa, aaaabaaaa, baaaaabaaaab, baaaaabaa? 어떤 문자열이 에 속하는가?

풀이)

- abaabaaabaa : (ab)(aa)(baa)(ab)(aa) 로 구성된 문자열로 에는 속하지만 에는 속하지 않는다. 에 해당하는 문자열이다.

- aaaabaaaa : (aa)(aa)(baa)(aa) 로 구성된 문자열로 에 모두 속한다.

- baaaaabaaaab : (baa)(aa)(ab)(aa)(ab) 로 구성된 문자열로 에는 속하지만 에는 속하지 않는다. 에 속하는 문자열이다.

- baaaaabaa : (baa)(aa)(ab)(aa) 로 구성된 문자열로 에 모두 속한다.

**12.** 언어 = { , n은 짝수 }에 대한 문법을 찾아라.

풀이)

문법을 생성할 때는 주어진 언어의 가장 짧은 언어부터 시작하여 찾는다. 따라서 주어진 언어의 가장 짧은 언어인 aa부터 시작한다. 그래서 처음에는 S -> aA , A -> a 를 생각했었다. 그런데 이렇게 하면 n의 값이 2일 경우에만 해당하고 4, 6, 8, .. 다음 경우는 해당되지 않는다.

반복되게 하려면 시작 기호 S에 a가 연속적으로 나와야 할 것 같았고 생성 규칙에 자기 자신을 포함하도록 해야 겠다는 생각이 들었다. 생성 규칙에 자기 자신을 넣으면 계속 같은 언어가 반복될 테니 a가 연속적으로 나올 수 있다. 근데 계속 자기 자신이 속해 있으면 안되므로 **ε**을 포함시키므로써 자신이 나오는 경우를 멈추고 원하는 언어가 나올 수 있게 하였다. 결론은 아래와 같다.

P : S -> aSa | **ε**

따라서 문법 G = ( {S}, {a}, P, S ) 이다.

**15.** 다음 생성규칙들을 갖는 문법에 의하여 생성되는 언어의 간단한 묘사를 제시하여라.

S -> aaA,

A -> bS,

S -> **λ.**

풀이)

생성 규칙을 축약하여 표현하면

S -> aaA | **λ**

A -> bS.

이다. 시작 기호 S부터 시작하면

(1) S -> aabS ( A -> bS )

S -> aab**λ ( S -> λ )**

(2) S -> aabS ( A -> bs)

S -> aabaabS ( S -> aabS)

S -> aabaab**λ ( S -> λ )**

**(3)** S -> aabS ( A -> bs)

S -> aabaabS ( S -> aabS)

S -> aabaabaabS ( S -> aabS)

S -> aabaab**aabλ ( S -> λ )**

반복적인 형태의 언어가 만들어진다. L(G) = { | n 1 }

**16.** 다음 생성규칙들을 갖는 문법은 어떤 언어를 생성하는가?

S -> Aa,

A -> B,

B -> Aa.

풀이)

(1) S -> Aa

S -> Ba ( A -> B)

S -> Aaa ( B -> Aa)

(2) S -> Aa

S -> Ba ( A -> B)

S -> Aaa ( B -> Aa)

S -> Baa ( A -> B )

S -> Aaaa ( B -> Aa)

(3) S -> Aa

S -> Ba ( A -> B)

S -> Aaa ( B -> Aa)

S -> Baa ( A -> B )

S -> Aaaa ( B -> Aa )

S -> Baaa ( A -> B )

S -> Aaaaa ( B -> Aa )

반복적인 형태의 언어가 나타난다. L(G) = { A | n 1 }

**20.** 아래에 주어진 생성규칙에 의해 생성되는 언어를 기술하여라.

S -> aSb | bSa | a.

풀이)

(1) S -> aSb

S -> aab ( S->a )

(2) S -> aSb

S -> aaSbb ( S -> aSb )

S -> aaabb ( S -> a )

(3) S -> aSb

S -> aaSbb ( S -> aSb )

S -> aaaSbbb ( S -> aSb )

S -> aaaabbb ( S -> a )

S => aab => aaabb => aaaabbb => .. 형태의 문자열 생성

따라서 L() = { | n 1 } 언어 하나와

(1) S -> bSa

S -> baa ( S->a )

(2) S -> bSa

S -> bbSaa ( S -> bSa )

S -> bbaaa ( S -> a )

(3) S -> bSa

S -> bbSaa ( S -> bSa )

S -> bbbSaaa ( S -> bSa )

S -> bbbaaaa ( S -> a )

S => baa => bbaaa => bbbaaaa => .. 형태의 문자열 생성

따라서 L() = { | n 1 } 언어로 총 두개의 언어가 생성된다.