

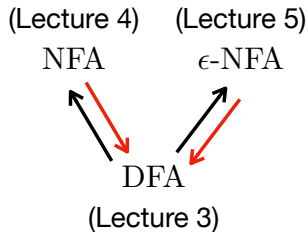
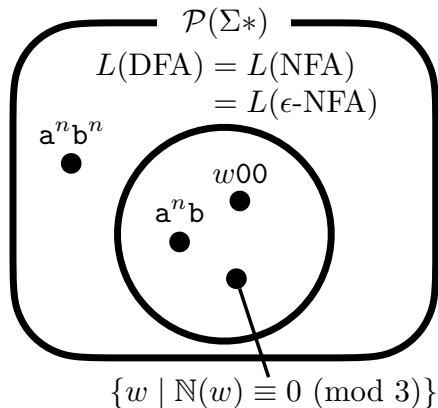
Lecture 6 – Regular Expressions and Languages

COSE215: Theory of Computation

Jihyeok Park



2023 Spring



→ : Subset Construction

1. Operations in Languages

- Union

- Concatenation

- Kleene Star

2. Regular Expressions

- Definition

- Language of Regular Expressions

- Extended Regular Expressions

- Examples

- The **union** of languages:

$$L_1 \cup L_2$$

- The **concatenation** of languages:

$$L_1 L_2 = \{w_1 w_2 \mid w_1 \in L_1 \wedge w_2 \in L_2\}$$

- The **Kleene star** of a language:

$$L^* = L^0 \cup L^1 \cup L^2 \cup \dots = \bigcup_{n \geq 0} L^n$$

$$L_1 = \{a^n \mid n \geq 0\} \quad L_2 = \{b^n \mid n \geq 0\}$$

$$L_1 \cup L_2 = \{a^n \text{ or } b^n \mid n \geq 0\}$$

$$L_1 L_2 = \{a^n b^m \mid n, m \geq 0\}$$

$$L_1^* = L_1 = \{a^n \mid n \geq 0\}$$

Definition (Regular Expressions)

A **regular expression** over a set of symbols Σ is inductively defined as follows:

- **(Basis Case)** \emptyset , ϵ , and $a \in \Sigma$ are regular expressions.
- **(Induction Case)** If R_1 and R_2 are regular expressions, then so are $R_1 \mid R_2$, $R_1 \cdot R_2$, R^* , and (R) .

$R ::=$	\emptyset	(Empty)	
	ϵ	(Epsilon)	
	a	(Symbol)	$(a \mid \epsilon) b^*$
	$R \mid R$	(Union)	
	$R \cdot R$	(Concatenation)	
	R^*	(Kleene Star)	
	(R)	(Parentheses)	

Precedence of operators:

$* > \cdot > \mid$

```
// The type definitions of symbols
type Symbol = Char
// The definition of regular expressions
trait RE
case class REEmpty() extends RE
case class REEpsilon() extends RE
case class RESymbol(symbol: Symbol) extends RE
case class REUnion(left: RE, right: RE) extends RE
case class REConcat(left: RE, right: RE) extends RE
case class REStar(re: RE) extends RE
case class REParen(re: RE) extends RE
// An example of regular expression
val re: RE = REConcat(
  REParen(
    REUnion(
      RESymbol('a'),
      REEpsilon(),
    )
  ),
  REStar(
    RESymbol('b')
  )
)
```

Definition (Language of Regular Expressions)

For a given regular expression R on a set of symbols Σ , the **language** $L(R)$ of R is inductively defined as follows:

$$\begin{array}{ll}
 L(\emptyset) &= \emptyset & L(R_1 \mid R_2) &= L(R_1) \cup L(R_2) \\
 L(\epsilon) &= \{\epsilon\} & L(R_1 R_2) &= L(R_1) L(R_2) \\
 L(a) &= \{a\} & L(R^*) &= L(R)^* \\
 & & L((R)) &= L(R)
 \end{array}$$

$$\begin{aligned}
 L((\epsilon \mid a) b^*) &= L((\epsilon \mid a)) \cdot L(b^*) \\
 &= L(\epsilon \mid a) \cdot L(b)^* \\
 &= (L(\epsilon) \cup L(a)) \cdot L(b)^* \\
 &= (\{\epsilon\} \cup \{a\}) \cdot \{b^n \mid n \geq 0\} \\
 &= \{b^n \text{ or } ab^n \mid n \geq 0\}
 \end{aligned}$$

More operators:

$$\begin{array}{lcl} R & ::= & \dots \\ & | & R^+ \quad (\text{Kleene plus}) \\ & | & R^? \quad (\text{Optional}) \end{array}$$

Actually, they are just **syntactic sugar** for the existing operators:

$$L(R^+) = L(RR^*) = L(R) \cdot L(R^*)$$

$$L(R^?) = L(\epsilon | R) = L(\epsilon) \cup L(R)$$

For examples,

$$\begin{aligned} L(a^+) &= L(aa^*) \\ &= L(a) \cdot L(a^*) \\ &= \{a^n \mid n \geq 1\} \end{aligned}$$

$$\begin{aligned} L(b^?) &= L(\epsilon) \cup L(b) \\ &= \{\epsilon, b\} \end{aligned}$$

- $L = \{\epsilon, a\}$
- $L = \{w \in \{0, 1\}^* \mid w \text{ contains at least two } 0's\}$
- $L = \{w \in \{0, 1\}^* \mid w \text{ contains exactly two } 0's\}$
- $L = \{w \in \{0, 1\}^* \mid w \text{ has three consecutive } 0's\}$
- $L = \{w \in \{a, b\}^* \mid a \text{ and } b \text{ alternate in } w\}$

- $L = \{a^n b^m \mid n \geq 3 \wedge m \equiv 0 \pmod{2}\}$
- $L = \{a^n b^m \mid n + m \equiv 0 \pmod{2}\}$
- $L = \{w \in \{0, 1\}^* \mid \text{the number of 0's is divisible by 3}\}$
- $L = \{w \in \{0, 1\}^* \mid \mathbb{N}(w) \equiv 0 \pmod{3}\}$
where $\mathbb{N}(w)$ is a natural number represented by w .

$$(0 \mid 1(01^*0)^*1)^*$$

- $L = \{a^n b^n \mid n \geq 0\}$ – IMPOSSIBLE (\nexists RE R . $L(R) = L$)

- Please see <https://github.com/ku-plrg-classroom/docs/tree/main/fa-examples>.
- You don't have to submit it. This is just exercise for your practice.
- The goal is to implement the finite automata (FA) objects in the `Implementation.scala` file.

1. Operations in Languages

- Union

- Concatenation

- Kleene Star

2. Regular Expressions

- Definition

- Language of Regular Expressions

- Extended Regular Expressions

- Examples

- Equivalence of Regular Expressions and Finite Automata

Jihyeok Park

jihyeok_park@korea.ac.kr

<https://plrg.korea.ac.kr>