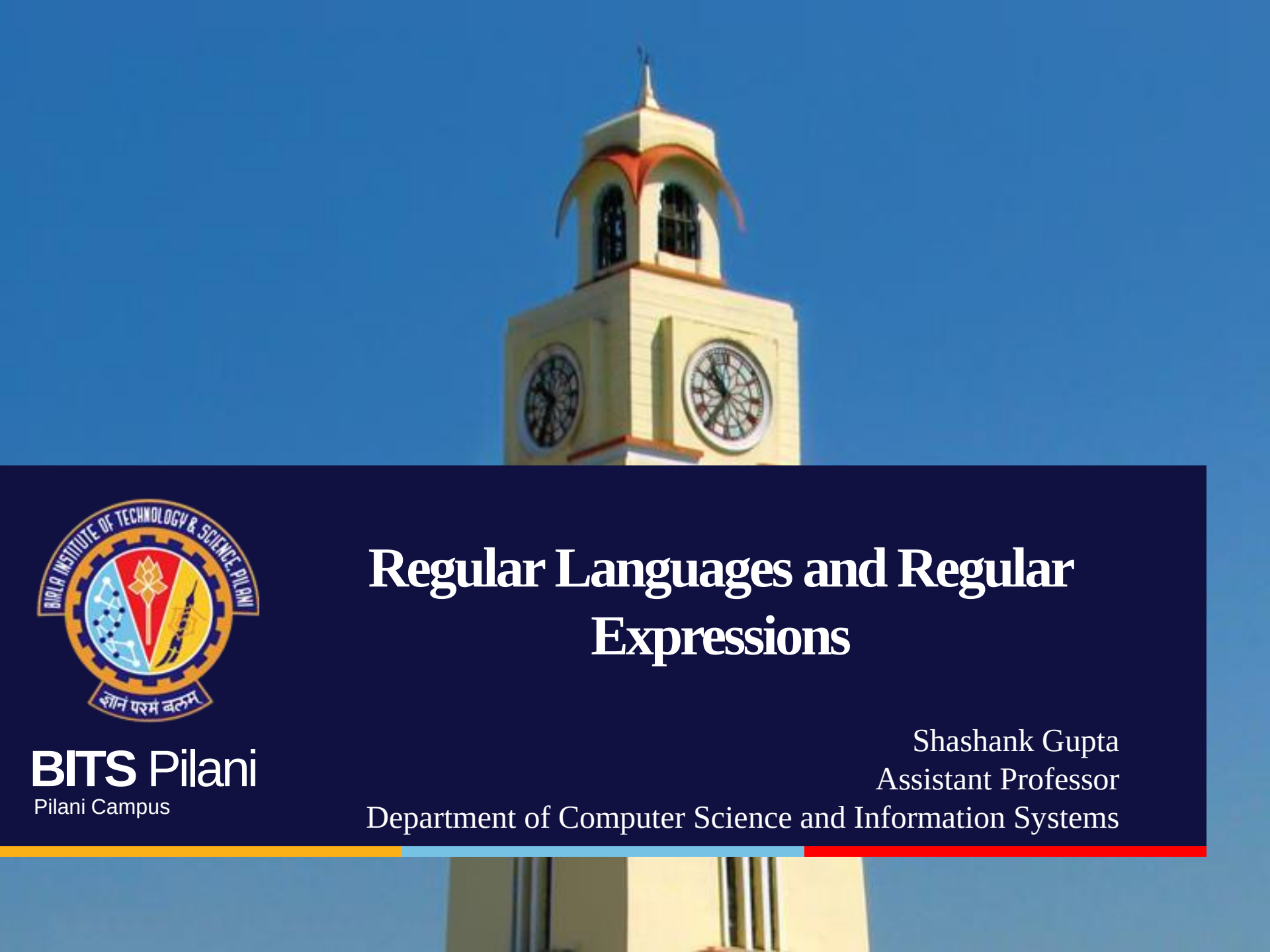

Regular Languages and Regular Expressions

BITS Pilani
Pilani Campus

Shashank Gupta
Assistant Professor
Department of Computer Science and Information Systems

Regular Expressions

It is a declarative way of representing a regular language.

- Algebraic representation of a regular language.
- In order to represent regular language using regular expression, some operators will be utilized.

It has applications in designing and development of compilers.

Regular Language

- Regular Language (also known as Type 3 language) is a restricted formal language that can be denoted by the regular expression.

$$L1 = \{01, 001, 011, 0011, - - - -\}$$

$$L2 = \{0^n 1^m \mid n, m \geq 1\}$$

Conditions for Formal Language to be Regular

- Conditions
  - Finite in Length
  - OR
  - Non-existence of Memory Element

Language	Identification (Regular or Non-Regular)
$L = \{a^n b^n \mid n \leq 1\}$	Regular
$L = \{a^n b^m c^k \mid n, m, k \geq 1\}$	Regular
$L = \{a^n b^m \mid n, m \geq 1, m > n\}$	Non-Regular
$L = \{a^n b^m c^{n+m} \mid n, m \geq 1\}$	Non-Regular

Rules for Generating the Regular Expression

Regular Language	Pattern of the Language	Regular Expression
L = {∈} **(Epsilon)**	Null String	∈
L = {a}	Single Character	a
L = {bb}	Single String	b.b **(Concatenation)**
L = {a, b}	Two Characters	a + b **(Union)**
L = {a, aa, aaa, aaaa,---}	Repetition of a's	a^+ **(Positive Closure)**
L = {∈, a, aa, aaa, aaaa,----}	Repetition of a's (including the null string i.e. ∈)	a* **(Kleene Closure)** OR ∈ + a^+
L = {a, b, ab, ba, aa, aaa, aba, baab, aba----}	Any combination of a's and b's	$(a + b)^+$
L = {∈, a, b, ab, ba, aa, aaa, aba, baab, aba,--}	Any combination of a's and b's (including the null string i.e. ∈)	(a + b)* OR ∈ + $(a + b)^+$

Regular Expressions

For every regular expression R, there is a unique language L(R) corresponding to it.

- It's converse is not true.
- We often drop this . (concatenation) operator symbol from the regular expression. (ab is also a regular expression).

Precedence of the operators of regular expression is from high to low. (), Kleene Closure/Positive Closure, Concatenation, Union.

- The language corresponding to the RE $\phi^* = \{\in\}$ Note that ϕ is the regular expression for Null set {} (the empty set) and this set does not contain the epsilon.

Designing of Regular Expression

- Design a regular expression that accepts all strings of 0's and 1's where each string starts with '0'.

Step 1
- L = { 0, 01, 00, 010,----}

Step 2
- Minimal length String is '0'

Step 3
- 0. (0+1)*

Designing of Regular Expression (Cont.…..)

- Construct a regular expression that accepts all strings of 0's and 1's where each string starts and ends with different symbol.

Step 1

- L = { 01, 10, 011, 110,---}

Step 2

- Minimal length String is '01' and '10'

Step 3

- 0. (0+1)*. 1 + 1. (0+1)*. 0

Designing of Regular Expression (Cont……)

Construct a regular expression that accepts all strings of 0's and 1's where the length of the string is exactly three.

Step 1

- L = { 000, 111, 010, 110,--}

Step 2

- Minimal Length String is Three

Step 3

- ((0 + 1) . (0 + 1) . (0 + 1))

Designing of Regular Expression (Cont……)

Construct a regular expression that accepts all strings of 0's and 1's where the length of the string is exactly divisible by 3.

Step 1

- L= { ∈, 010, 010101, ----}

Step 2

- Minimal length String is Epsilon.

Step 3

- ((0 + 1) . (0 + 1) . (0 + 1))*
- **OR**
- ∈ + ((0 + 1) . (0 + 1) . (0 + 1))$^{+}$