04012022-UNIT1

04 January 2022 13:32

Unit1-Slides

CSO322: Theory of Computation

Unit-1: Formal Languages and Finite Automata

Dr. Lavanya Selvaganesh

Formal Languages

Definition

Let Σ be an alphabet. Let Σ^* be the set of all strings over Σ. A language is a subset of Σ^*.

Σ^*

Formal Languages

Definition

Let Σ be an alphabet. Let Σ^* be the set of all strings over Σ. A language is a subset of Σ^*.

Example

Let $\Sigma = \{a, b\}$. $\Sigma^* = \{\epsilon, a, b, aa, bb, ab, ba, aaa, \ldots\}$.
Examples of languages over Σ are

1. $L_1 = \{\epsilon, a, aa, aab\}$. $\subset \Sigma^*$

Formal Languages

Definition

Let Σ be an alphabet. Let Σ^* be the set of all strings over Σ. A language is a subset of Σ^*.

Example

Let $\Sigma = \{a, b\}$. $\Sigma^* = \{\epsilon, a, b, aa, bb, ab, ba, aaa, \ldots\}$.
Examples of languages over Σ are

1. $L_1 = \{\epsilon, a, aa, aab\}$.
2. $L_2 = \{x \in \Sigma^* : |x| \leq 8\}$. → $|\cdot|$ - length of

Dr. Lavanya Selvaganesh CSO322 2 / 46

Formal Languages

Definition

Let Σ be an alphabet. Let Σ^* be the set of all strings over Σ. A language is a subset of Σ^*.

Example

Let $\Sigma = \{a, b\}$. $\Sigma^* = \{\epsilon, a, b, aa, bb, ab, ba, aaa, \ldots\}$.
Examples of languages over Σ are

1. $L_1 = \{\epsilon, a, aa, aab\}$.
2. $L_2 = \{x \in \Sigma^* : |x| \leq 8\}$.
3. $L_3 = \{x \in \Sigma^* : n_a(x) \geq n_b(x)\}$ → a, aa, ab, ba, aaa, aab, aba, . . .
   where n_a and n_b are the number of a's and b's in x respectively.

Dr. Lavanya Selvaganesh CSO322 2 / 46

Formal Languages

Definition

Let Σ be an alphabet. Let Σ^* be the set of all strings over Σ. A language is a subset of Σ^*.

Example

Let $\Sigma = \{a, b\}$. $\Sigma^* = \{\epsilon, a, b, aa, bb, ab, ba, aaa, \ldots\}$.
Examples of languages over Σ are

1. $L_1 = \{\epsilon, a, aa, aab\}$.
2. $L_2 = \{x \in \Sigma^* : |x| \leq 8\}$.
3. $L_3 = \{x \in \Sigma^* : n_a(x) \geq n_b(x)\}$
   where n_a and n_b are the number of a's and b's in x respectively.
4. $L_4 = \{x \in \Sigma^* : |x| \geq 2$ and x begins and ends with $b\}$

Dr. Lavanya Selvaganesh CSO322 2 / 46

Formal Languages

1. **New languages can be constructed using set operations, since languages are sets of strings.**

since languages are sets of strings.

Formal Languages

1. **New languages can be constructed using set operations, since languages are sets of strings.**
   - For any two languages over an alphabet Σ, their union, intersection and difference are also languages over Σ.

Formal Languages

1. **New languages can be constructed using set operations, since languages are sets of strings.**
   - For any two languages over an alphabet Σ, their union, intersection and difference are also languages over Σ.
   - Complement of a language over Σ is defined by $L' = \Sigma^* - L$.

Formal Languages

1. **New languages can be constructed using set operations, since languages are sets of strings.**
   - For any two languages over an alphabet Σ, their union, intersection and difference are also languages over Σ.
   - Complement of a language over Σ is defined by $L' = \Sigma^* - L$.
   - Suppose $L_1 \subseteq \Sigma_1^*$ and $L_2 \subseteq \Sigma_2^*$, then L_1 and L_2 are languages over $(\Sigma_1 \cup \Sigma_2)^*$ i.e.

$$L_1, L_2 \subseteq (\Sigma_1 \cup \Sigma_2)^*$$

$\Sigma_1^* \subseteq (\Sigma_1 \cup \Sigma_2)^2$

Formal Languages

1. **New languages can be constructed using set operations, since languages are sets of strings.**
   - For any two languages over an alphabet Σ, their union, intersection and difference are also languages over Σ.
   - Complement of a language over Σ is defined by $L' = \Sigma^* - L$.
   - Suppose $L_1 \subseteq \Sigma_1^*$ and $L_2 \subseteq \Sigma_2^*$, then L_1 and L_2 are languages over $(\Sigma_1 \cup \Sigma_2)^*$ i.e.

     $$L_1, L_2 \subseteq (\Sigma_1 \cup \Sigma_2)^*$$

   - However, there is a possibility for complement of L_1 to be either

     $$L_1 = \Sigma_1^* - L_1 \text{ or } L_1' = (\Sigma_1 \cup \Sigma_2)^* - L_1.$$

     Depending on the context, it will be clear which alphabet is referred to.

Formal Languages

2. **Concatenation operation on strings will allow us to construct new languages.**

Formal Languages

2. **Concatenation operation on strings will allow us to construct new languages.**
   - If x and y are elements of Σ^*, the concatenation of x and y is the string xy formed by writing the symbols of x followed by symbols of y.

$\Sigma = \{a, b\}$, $x = aab$, $y = bab$ $\therefore xy = aabbab$, $yx = babaab$

$xy \neq yx$.

Formal Languages

2. **Concatenation operation on strings will allow us to construct new languages.**
   - If x and y are elements of Σ^*, the concatenation of x and y is the string xy formed by writing the symbols of x followed by symbols of y.

Example

If $x = abb$ and $y = ba$, then $xy = abbba$ and $yx = baabb$.

Formal Languages

2. **Concatenation operation on strings will allow us to construct new languages.**
   - If x and y are elements of Σ^*, the concatenation of x and y is the string xy formed by writing the symbols of x followed by symbols of y.

Example

If $x = abb$ and $y = ba$, then $xy = abbba$ and $yx = baabb$.

   - For any string x, $x\epsilon = \epsilon x = x$.

ϵ - empty string.

$\epsilon \in \Sigma^*$

lang. cont.

$L = \{\epsilon\}$ - empty string.

$L = \phi$ - empty language

Formal Languages

2. **Concatenation operation on strings will allow us to construct new languages.**
   - If x and y are elements of Σ^*, the concatenation of x and y is the string xy formed by writing the symbols of x followed by symbols of y.

Example

If $x = abb$ and $y = ba$, then $xy = abbba$ and $yx = baabb$.

   - For any string x, $x\epsilon = \epsilon x = x$.
   - For strings x, y, z; $(xy)z = x(yz)$ i.e. **concatenation is associative**.

$(xy)z$ or $x(yz)$.

$x\epsilon = \epsilon x = x$.

Formal Languages

- - - - .

3. **Substring of a string**
   - x is a substring of another string y, if there are strings w and z, either or both may be null, so that $y = wxz$.

Formal Languages

3. **Substring of a string**
   - x is a substring of another string y, if there are strings w and z, either or both may be null, so that $y = wxz$.

Example

The string "car" is a substring of "descartes", "vicar", "cartridge" and "car", but not a substring of "charity".

Formal Languages

3. **Substring of a string**
   - x is a substring of another string y, if there are strings w and z, either or both may be null, so that $y = wxz$.

Example

The string "car" is a substring of "descartes", "vicar", "cartridge" and "car", but not a substring of "charity".

- A **prefix** of a string is an initial substring.
  Example - Prefixes of $abaa$ – ϵ, a, ab, aba, $abaa$.

Formal Languages

3. **Substring of a string**
   - x is a substring of another string y, if there are strings w and z, either or both may be null, so that $y = wxz$.

Example

The string "car" is a substring of "descartes", "vicar", "cartridge" and "car", but not a substring of "charity".

- A **prefix** of a string is an initial substring.
  Example - Prefixes of $abaa$ – ϵ, a, ab, aba, $abaa$.
- A **suffix** of a string is a final substring.
  Example - Suffixes of $abaa$ – ϵ, a, aa, baa, $abaa$.

Formal Languages

4. **Concatenation of languages is also possible.**
   - If $L_1, L_2 \subseteq \Sigma^*$, $L_1 L_2 = \{xy | x \in L_1, y \in L_2\}$

ε

Formal Languages

4. **Concatenation of languages is also possible.**
   - If $L_1, L_2 \subseteq \Sigma^*$, $L_1 L_2 = \{xy | x \in L_1, y \in L_2\}$

Example $\Sigma = \{0, 1\}$

$L_1 = \{\epsilon, 01, 011\}$,
$L_2 = \{\epsilon, 11, 10, 110\}$ ✓
$L_1 L_2 = \{\epsilon, 01, 10, 11, 110, 011, 0111, 0110, 01110, 01111, 011110\}$
is the concatenation of L_1 and L_2.

- Concatenation of L with $\{\epsilon\}$: $L\{\epsilon\} = \{\epsilon\}L = L$.

$L_2 . L_1 = \{yx : y \in L_2, x \in L_1\}$.