cse303
ELEMENTS OF THE THEORY OF COMPUTATION

Professor Anita Wasilewska
LECTURE 4
CHAPTER 1

SETS, RELATIONS, and LANGUAGES

7. Alphabets and Languages
8. Finite Representation of Languages
CHAPTER 1

PART 7: Alphabets and Languages
Introduction

Data are encoded in the computers’ memory as strings of bits or other symbols appropriate for manipulation.

The mathematical study of the Theory of Computation begins with understanding of mathematics of manipulation of strings of symbols.

We first introduce two basic notions: Alphabet and Language.
Alphabet

Definition
Any finite set is called an alphabet

Elements of the alphabet are called symbols of the alphabet

This is why we also say:
Alphabet is any finite set of symbols
Alphabet

Alphabet Notation
We use a symbol $\Sigma$ to denote the alphabet.

Remember
$\Sigma$ can be $\emptyset$ as empty set is a finite set.

When we want to study non-empty alphabets we have to say so, i.e. to write:

$$\Sigma \neq \emptyset$$
Alphabet Examples

E1  $\Sigma = \{\dagger, \emptyset, \partial, \oint, \otimes, \vec{a}, \nabla\}$

E2  $\Sigma = \{a, b, c\}$

E3  $\Sigma = \{n \in \mathbb{N} : n \leq 10^5\}$

E4  $\Sigma = \{0, 1\}$ is called a binary alphabet
Alphabet Examples

For simplicity and consistence we will use only as symbols of the alphabet letters (with indices if necessary) or other common characters when needed and specified.

We also write $\sigma \in \Sigma$ for a general form of an element in $\Sigma$.

$\Sigma$ is a finite set and we will write

$$\Sigma = \{a_1, a_2, \ldots, a_n\} \text{ for } n \geq 0$$
Finite Sequences Revisited

Definition
A finite sequence of elements of a set $A$ is any function $f : \{1, 2, \ldots, n\} \rightarrow A$ for $n \in \mathbb{N}$

We call $f(n) = a_n$ the n-th element of the sequence $f$
We call $n$ the length of the sequence $a_1, a_2, \ldots, a_n$

Case $n=0$
In this case the function $f$ is empty and we call it an empty sequence and denote by $e$
Words over $\Sigma$

Let $\Sigma$ be an alphabet

We call finite sequences of the alphabet $\Sigma$ words or strings over $\Sigma$

We denote by $e$ the empty word over $\Sigma$

Some books use symbol $\lambda$ for the empty word
Words over $\Sigma$

**E5**  Let $\Sigma = \{a, b\}$
We will write some words (strings) over $\Sigma$ in a **shorthand** notation as for example

$$aaa, \ ab, \ bbb$$

instead using the formal definition:

$$f : \{1, 2, 3\} \rightarrow \Sigma$$

such that $f(1) = a, f(2) = a, f(3) = a$ for the word $aaa$
or

$$g : \{1, 2\} \rightarrow \Sigma$$

such that $g(1) = b, g(2) = b$ for the word $bb$ .. etc..
Words in \( \Sigma^* \)

Let \( \Sigma \) be an **alphabet**. We denote by \( \Sigma^* \) the set of **all finite** sequences over \( \Sigma \).

Elements of \( \Sigma^* \) are called **words** over \( \Sigma \).

We write \( w \in \Sigma^* \) to express that \( w \) is a **word** over \( \Sigma \).

**Symbols** for words are

\[
\begin{align*}
w, z, v, x, y, z, \alpha, \beta, \gamma & \in \Sigma^* \\
x_1, x_2, \ldots & \in \Sigma^* \\
y_1, y_2, \ldots & \in \Sigma^*
\end{align*}
\]
Words in $\Sigma^*$

**Observe** that the set of all finite sequences include the empty sequence i.e. $e \in \Sigma^*$ and we hence have the following

**Fact**
For any **alphabet** $\Sigma$, $\Sigma^* \neq \emptyset$
Chapter 1

Some Short Questions and Answers
Q1 Let $\Sigma = \{a, b\}$
How many are there all possible \textbf{words} of length 5 over $\Sigma$?

A1 By definition, words over $\Sigma$ are \textbf{finite sequences};
Hence words of a \textbf{length 5} are functions
\[ f : \{1, 2, \ldots, 5\} \rightarrow \{a, b\} \]
So we have by the \textbf{Counting Functions Theorem} that
there are $2^5$ words of a length 5 over $\Sigma = \{a, b\}$
Counting Functions Theorem

Counting Functions Theorem
For any finite, non empty sets $A$, $B$, there are $|B|^{|A|}$ functions that map $A$ into $B$

The proof is in Lecture 2, Part 5
Short Questions

Q2
Let \( \Sigma = \{a_1, \ldots, a_k\} \) where \( k \geq 1 \)
How many are there possible words of length \( \leq n \) for \( n \geq 0 \) in \( \Sigma^* \)?

A2
By the **Counting Functions Theorem** there are

\[
k^0 + k^1 + \cdots + k^n
\]

words of length \( \leq n \) over \( \Sigma \) because for each \( m \)
there are \( k^m \) words of length \( m \) over \( \Sigma = \{a_1, \ldots, a_k\} \)
and \( m = 0, 1 \ldots n \)
Q3  Given an alphabet $\Sigma \neq \emptyset$

How many are there words in the set $\Sigma^*$?

A3

There are infinitely countably many words in $\Sigma^*$ by the Theorem 5 (Lecture 2) that says: "for any non empty, finite set $A$, $|A^*| = \aleph_0"$

We hence proved the following

Theorem

For any alphabet $\Sigma \neq \emptyset$, the set $\Sigma^*$ of all words over $\Sigma$ is countably infinite
Languages over $\Sigma$

Language Definition
Given an alphabet $\Sigma$, any set $L$ such that

$$L \subseteq \Sigma^*$$

is called a language over $\Sigma$

Fact 1
For any alphabet $\Sigma$, any language over $\Sigma$ is countable
Languages over $\Sigma$

**Fact 2**
For any alphabet $\Sigma \neq \emptyset$, there are uncountably many languages over $\Sigma$

More precisely, there are exactly $C = |R|$ of languages over any non-empty alphabet $\Sigma$
Languages over $\Sigma$

**Fact 1**
For any alphabet $\Sigma$, any language over $\Sigma$ is **countable**

**Proof**
By definition, a set is **countable** if and only if is finite or countably infinite

1. Let $\Sigma = \emptyset$, hence $\Sigma^* = \{e\}$ and we have two languages $\emptyset$, $\{e\}$ over $\Sigma$, both finite, so **countable**

2. Let $\Sigma \neq \emptyset$, then $\Sigma^*$ is **countably infinite**, so obviously any $L \subseteq \Sigma^*$ is finite or countably infinite, hence **countable**
Languages over $\Sigma$

Fact 2
For any alphabet $\Sigma \neq \emptyset$, there are exactly $C = |R|$ of languages over any non-empty alphabet $\Sigma$

Proof
We proved that $|\Sigma^*| = \aleph_0$
By definition $L \subseteq \Sigma^*$, so there is as many languages over $\Sigma$ as all subsets of a set of cardinality $\aleph_0$—that is as many as $2^{\aleph_0} = C$
Languages over $\Sigma$

Q4 Let $\Sigma = \{a\}$

There is $\aleph_0$ languages over $\Sigma$

NO

We just proved that that there is uncountably many, more precisely, exactly $C$ languages over $\Sigma \neq \emptyset$ and we know that

$$\aleph_0 < C$$
Some Basic Definitions
Some Basic Definitions

Definition
Given an alphabet $\Sigma$ and a word $w \in \Sigma^*$
We say that $w$ has a length $n = l(w) = |w|$ when

$$w : \{1, 2, \ldots, n\} \longrightarrow \Sigma$$

We re-write $w$ as

$$w : \{1, 2, \ldots, |w|\} \longrightarrow \Sigma$$
Some Basic Definitions

We define now a position of $\sigma \in \Sigma$ in a word $w \in \Sigma^*$ as follows

**Definition**

Given $\sigma \in \Sigma$ and a word $w \in \Sigma^*$

$\sigma \in \Sigma$ occurs in the j-th position in $w \in \Sigma^*$ if and only if $w(j) = \sigma$ for $1 \leq j \leq |w|$
Some Examples

E6 Consider a word $w$ written in a shorthand as

$$w = anita$$

By formal definition we have

- $w(1) = a$,
- $w(2) = n$,
- $w(3) = i$,
- $w(4) = t$,
- $w(5) = a$

and $a$ occurs in the 1st and 5th position.

E7 Let $\Sigma = \{0, 1\}$ and $w = 01101101$ (shorthand).

Formally $w : \{1, 2, \ldots, 8\} \rightarrow \{0, 1\}$ is such that

- $w(1) = 0$,
- $w(2) = 1$,
- $w(3) = 1$,
- $w(4) = 0$,
- $w(5) = 1$,
- $w(6) = 1$,
- $w(7) = 0$,
- $w(8) = 1$

1 occurs in the positions 2, 3, 5, 6 and 8.

0 occurs in the positions 1, 4, 7.
Informal Concatenation

Informal Definition

Given an alphabet $\Sigma$ and any words $x, y \in \Sigma^*$

We define informally a **concatenation** $\circ$ of words $x, y$ as a word $w$ obtained from $x, y$ by writing the word $x$ followed by the word $y$

We write the **concatenation** of words $x, y$ as

$$w = x \circ y$$

We use the symbol $\circ$ of **concatenation** when it is needed formally, otherwise we will write simply

$$w = xy$$
Formal Concatenation

Definition
Given an alphabet $\Sigma$ and any words $x, y \in \Sigma^*$
We define:

$$w = x \circ y$$

if and only if

1. $|w| = |x| + |y|
2. $w(j) = x(j)$ for $j = 1, 2, \ldots, |x|
2. $w(|x| + j) = y(j)$ for $j = 1, 2, \ldots, |y|$
Formal Concatenation

Properties
Directly from definition we have that

\[ w \circ e = e \circ w = w \]

\[ (x \circ y) \circ z = x \circ (y \circ z) = x \circ y \circ z \]

Concatenation of words is associative

Remark: we need to define a concatenation of two words only and then we define

\[ x_1 \circ x_2 \circ \cdots \circ x_n = (x_1 \circ x_2 \circ \cdots \circ x_{n-1}) \circ x_n \]

and prove by Mathematical Induction that

\[ w = x_1 \circ x_2 \circ \cdots \circ x_n \text{ is well defined for all } n \geq 2 \]
Substring

Definition
A word \( v \in \Sigma^* \) is a **substring** (sub-word) of \( w \) iff there are \( x, y \in \Sigma^* \) such that

\[
w = xvy
\]

**Remark:** the words \( x, y \in \Sigma^* \), i.e. they can also be empty

**P1** \( w \) is a substring of \( w \)

**P2** \( e \) is a substring of any string (any word \( w \))
as we have that \( ew = we = w \)

**Definition** Let \( w = xy \)
x is called a **prefix** and y is called a **suffix** of \( w \)
Power $w^i$

**Definition**
We define a **power** $w^i$ of $w$ recursively as follows

\[
    w^0 = e
\]

\[
    w^{i+1} = w^i \circ w
\]

This type of definition is called **definition by induction**

**E8**

\[
    w^0 = e, \quad w^1 = w^0 \circ w = e \circ w = w, \quad w^2 = w^1 \circ w = w \circ w
\]

**E9**

\[
    anita^2 = anita^1 \circ anita = e \circ anita \circ anita = anita \circ anita
\]
Reversal $w^R$

Definition

Reversal $w^R$ of $w$ is defined by induction over length $|w|$ of $w$ as follows

1. If $|w| = 0$, then $w^R = w = e$
2. If $|w| = n + 1 > 0$, then $w = ua$ for some $a \in \Sigma$, and $u \in \Sigma^*$ and we define

$$w^R = au^R \text{ for } |u| < n + 1$$

Short Definition of $w^R$

1. $e^R = e$
2. $(ua)^R = au^R$
Reversal Proof

We prove now as an example of Inductive proof the following simple fact

Fact
For any $w, x \in \Sigma^*$

$$(wx)^R = x^Rw^R$$

Proof by Mathematical Induction over the length $|x|$ of $x$ with $|w| = \text{constant}$

Base case $n=0$

$|x| = 0$, i.e. $x=\epsilon$ and by definition

$$(we)^R = (w)^R = ew^R = e^Rw^R$$
Reversal Proof

Inductive Assumption

\[(wx)^R = x^R w^R\] for all \(|x| \leq n\)

Let now \(|x| = n + 1\), so \(x = ua\) for certain \(a \in \Sigma, \ u \in \Sigma^*\) and \(|u| = n\)

We evaluate

\[ (wx)^R = (w(ua))^R = ((wu)a)^R \]

\[= \text{def} \ a(wu)^R = \text{ind} \ au^R w^R = \text{def} \ (ua)^R = x^R w^R \]
Languages over $\Sigma$

Definition
Given an alphabet $\Sigma$, any set $L$ such that $L \subseteq \Sigma^*$ is called a language over $\Sigma$

Observe that $\emptyset$, $\Sigma$, $\Sigma^*$ are all languages over $\Sigma$

We have proved

Theorem For any language $L$ over $\Sigma$, $L$ is finite or infinitely countable

By definition, languages are sets so we can define them in ways we did for sets, by listing elements (for small finite sets) or by giving a property $P(w)$ defining $L$, i.e. by setting

$$L = \{w \in \Sigma^* : P(w)\}$$
Languages Examples

E1  \( L_1 = \{ w \in \{0, 1\}^* : \text{w has an even number of 0's} \} \)
E2  \( L_2 = \{ w \in \{a, b\}^* : \text{w has ab as a sub-string} \} \)
E3  \( L_3 = \{ w \in \{0, 1\}^* : |w| \leq 2 \} \)
E4  \( L_4 = \{ e, 0, 1, 00, 01, 11, 10 \} \)

Observe that \( L_3 = L_4 \)

Languages are sets so we can define all set operations union, intersection, complement, generalized union, Cartesian product etc and them as we did for sets

Given \( L_1, L_2 \subseteq \Sigma^* \), we consider

\[
L_1 \cup L_2, L_1 \cap L_2, L_1 - L_2, -L = \Sigma^* - L, L_1 \times L_2, \ldots \text{etc.}
\]
Special Operations on Languages

We define now a special operation on languages, different from any of the set operation

**Definition**

Given $L_1, L_2 \subseteq \Sigma^*$

A language

$$L_1 \circ L_2 = \{w \in \Sigma^* : w = xy \text{ for some } x \in L_1, y \in L_2\}$$

is called a **concatenation** of $L_1$ and $L_2$
The **concatenation domain** issue

We can have that the languages $L_1$, $L_2$ are defined over **different domains**, i.e. we have two alphabets $\Sigma_1 \neq \Sigma_2$ and

$$L_1 \subseteq \Sigma_1^* \quad \text{and} \quad L_2 \subseteq \Sigma_2^*$$

In this case we always take

$$\Sigma = \Sigma_1 \cup \Sigma_2 \quad \text{and get} \quad L_1, L_2 \subseteq \Sigma^*$$
Concatenation Examples

E5 Let $L_1$, $L_2$ be languages defined below. Describe $L_1 \circ L_2$

$L_1 = \{ w \in \{ a, b \}^* : |w| \leq 1 \}$

$L_2 = \{ w \in \{ 0, 1 \}^* : |w| \leq 2 \}$

Observe that we have here $\Sigma_1 = \{ a, b \}$ and $\Sigma_2 = \{ 0, 1 \}$, so we take $\Sigma = \Sigma_1 \cup \Sigma_2 = \{ a, b, 0, 1 \}$ and $L_1, L_2 \subseteq \Sigma^*$

We write now a general formula for $L_1 \circ L_2$ as follows

$L_1 \circ L_2 = \{ w \in \Sigma^* : w = xy \text{ where } x \in \{ a, b \}^*, y \in \{ 0, 1 \}^*, |x| \leq 1, |y| \leq 2 \}$
E5 revisited  Let $L_1$, $L_2$ be languages defined below. Describe $L_1 \circ L_2$

$L_1 = \{ w \in \{a, b\}^* : |w| \leq 1 \}$
$L_2 = \{ w \in \{0, 1\}^* : |w| \leq 2 \}$

As $L_1$, $L_2$ are finite we can just list their elements and then evaluate (or list) the $L_1 \circ L_2$

$L_1 = \{e, a, b\}$
$L_2 = \{e, 0, 1, 01, 00, 11, 10\}$

$L_1 \circ L_2 = \{ ey : y \in L_2 \} \cup \{ ay : y \in L_2 \} \cup \{ by : y \in L_2 \}$

Observe that we can write yet another general formula for $L_1 \circ L_2$

$L_1 \circ L_2 = L_2 \cup (\{a\} \circ L_2) \cup (\{b\} \circ L_2)$
Concatenation Examples

E6  Let $L_1$, $L_2$ be languages defined below for $\Sigma = \{0, 1\}$.
Describe $L_1 \circ L_2$

$L_1 = \{w \in \Sigma^*: w$ has an even number of 0’s$\}$

$L_2 = \{w \in \Sigma^*: w = 0xx, x \in \Sigma^*\}$

$L_1 \circ L_2 = \{w \in \Sigma^*: w$ has an odd number of 0’s$\}$

$L_2 \circ L_1 = \{w \in \Sigma^*: w$ starts with 0$\}$

Observe that $1000 \in L_1 \circ L_2$ and $1000 \notin L_2 \circ L_1$ and so we have that $L_1 \circ L_2 \neq L_2 \circ L_1$

We hence proved the following

Fact

Concatenation of languages is not commutative
E8 Let $L_1$, $L_2$ be languages defined below for $\Sigma = \{0, 1\}$

Describe the language $L_2 \circ L_1$

$L_1 = \{w \in \Sigma^* : w = x1, \ x \in \Sigma^*\}$

$L_2 = \{w \in \Sigma^* : w = 0x, \ x \in \Sigma^*\}$

$L_2 \circ L_1 = \{w \in \Sigma^* : w = 0xy1, \ x, y \in \Sigma^*\}$

Observe that $L_2 \circ L_1$ can be also defined by a property as follows

$L_2 \circ L_1 = \{w \in \Sigma^* : w \text{ starts with } 0 \text{ and ends with } 1\}$
Distributivity of Concatenation

Theorem
Concatenation is distributive over union of languages

More precisely, given languages \( L, L_1, L_2, \ldots, L_n \) over \( \Sigma \), then for any for \( n \geq 2 \),

\[
(L_1 \cup L_2 \cup \cdots \cup L_n) \circ L = (L_1 \circ L) \cup \cdots \cup (L_n \circ L)
\]

\[
L \circ (L_1 \cup L_2 \cup \cdots \cup L_n) = (L \circ L_1) \cup \cdots \cup (L \circ L_n)
\]

Proof by Mathematical Induction over \( n \in \mathbb{N}, \ n \geq 2 \)
We prove the base case for the first equation and leave the
Inductive argument and a similar proof of the second equation
as an exercise.
Distributivity of Concatenation Proof

**Base Case** $n = 2$
We have to prove that

$$(L_1 \cup L_2) \circ L = (L_1 \circ L) \cup (L_2 \circ L)$$

$w \in (L_1 \cup L_2) \circ L$ (by definition of $\circ$) iff 
$(w \in L_1 \text{ or } w \in L_2)$ and $w \in L$ (by distributivity of and over or) iff 
$(w \in L_1 \text{ and } w \in L) \text{ or } (w \in L_2 \text{ and } w \in L)$ (by definition of $\circ$) iff 
$(w \in L_1 \circ L) \text{ or } (w \in L_2 \circ L)$ (by definition of $\cup$) iff 
$w \in (L_1 \circ L) \cup (L_2 \circ L)$
Kleene Star - $L^*$

Kleene Star $L^*$ of a language $L$ is yet another operation specific to languages.

It is named after Stephen Cole Kleene (1909 - 1994), an American mathematician and world famous logician who also helped lay the foundations for theoretical computer science.

We define $L^*$ as the set of all strings obtained by concatenating zero or more strings from $L$.

Remember that concatenation of zero strings is $e$, and concatenation of one string is the string itself.
We define $L^*$ formally as

$$L^* = \{ w_1 w_2 \ldots w_k : \text{for some } k \geq 0 \text{ and } w_1, \ldots, w_k \in L \}$$

We write it usually without a concatenation sign $\circ$ as follows

$$L^* = \{ w_1 w_2 \ldots w_k : k \geq 0, w_i \in L, i = 1, 2, \ldots, k \}$$

or in a form of Generalized Union

$$L^* = \bigcup_{k \geq 0} \{ w_1 w_2 \ldots w_k : w_1, \ldots, w_k \in L \}$$

**Remark** we write $xyz$ for $x \circ y \circ z$. We use the concatenation symbol $\circ$ when we want to stress that we talk about some properties of the concatenation
Kleene Star Properties

P1 \( e \in L^* \), for all \( L \)
Follows directly from the definition as we have case \( k = 0 \)

P2 \( L^* \neq \emptyset \), for all \( L \)
Follows directly from P1, as \( e \in L^* \)

P3 \( \emptyset^* \neq \emptyset \)
Because \( L^* = \emptyset^* = \{e\} \neq \emptyset \)

P4 \( L^* = \Sigma^* \) for some \( L \)
Take \( L = \Sigma \)

P6 \( L^* \neq \Sigma^* \) for some \( L \)
Take \( L = \{00, 11\} \) over \( \Sigma = \{0, 1\} \)
\( 01 \not\in L^* \) and \( 01 \in \Sigma^* \)
Example

The property $P4$ provided us with a quite trivial example of $L$, such that $L^* = \Sigma^*$, namely $L = \Sigma$

Here is a non-trivial example of $L$, such that $L \neq \Sigma$ and nevertheless $L^* = \Sigma^*$

Example

Take $\Sigma = \{0, 1\}$ and $L = \{w \in \Sigma^* : w$ has an unequal number of 0 and 1\}$

Prove that $L^* \neq \Sigma^*$

Some words in and out of $L$

\[100 \in L, \quad 00111 \in L \quad 100011 \notin L\]
Example Proof

Given
$L = \{w \in 0, 1^* : w \text{ has an unequal number of 0 and 1}\}$
We now prove that

$L^* = \{0, 1\}^* = \Sigma^*$

Proof
By definition we have that $L \subseteq \{0, 1\}^*$ and $\{0, 1\}^{**} = \{0, 1\}^*$
By the the following property of languages:

$P$: If $L_1 \subseteq L_2$, then $L_1^* \subseteq L_2^*$

and get that

$L^* \subseteq \{0, 1\}^{**} = \{0, 1\}^*$ i.e. $L^* \subseteq \Sigma^*$
Now we have to show that $\Sigma^* \subseteq L^*$, i.e.

$$\{0, 1\}^* \subseteq \{w \in 0, 1^* : w \text{ has an unequal number of 0 and 1}\}$$

Observe that

$0 \in L$ because 0 regarded as a string over $\Sigma$ has an unequal number appearances of 0 and 1

The number of appearances of 1 is zero and the number of appearances of 0 is one

$1 \in L$ for the same reason a $0 \in L$

So we proved that $\{0, 1\} \subseteq L$

We now use the property $P$ and get

$$\{0, 1\}^* \subseteq L^* \text{ i.e } \Sigma^* \subseteq L^*$$

What ends the proof that $\Sigma^* = L^*$
We define

\[ L^+ = \{ w_1 w_2 \ldots w_k : \text{for some } k \geq 1 \text{ and some } w_1, \ldots, w_k \in L \} \]

We write it also as follows

\[ L^+ = \{ w_1 w_2 \ldots w_k : k \geq 1 \ w_i \in L, \ i = 1, 2, \ldots, k \} \]

Properties

\[ P1 : \quad L^+ = L \circ L^* \quad P2 : \quad e \in L^+ \text{ iff } e \in L \]
We know that 
\[ e \in L^* \quad \text{for all } L \]

Show that

For some language \( L \) we have that \( e \in L^+ \) and

for some language \( L \) we can have that \( e \not\in L^+ \)

E1

Obviously, for any \( L \) such that \( e \in L \) we have that \( e \in L^+ \)

E2

If \( L \) is such that \( e \not\in L \) we have that \( e \not\in L^+ \) as \( L^+ \) does not have a case \( k=0 \)
CHAPTER 1
PART 8: Finite Representation of Languages
Introduction

We can represent a finite language by finite means for example listing all its elements.

Languages are often infinite and so a natural question arises if a finite representation is possible and when it is possible when a language is infinite.

The representation of languages by finite specifications is a central issue of the theory of computation.

Of course we have to define first formally what do we mean by representation by finite specifications, or more precisely by a finite representation.
Idea of Finite Representation

We start with an example: let

\[ L = \{a\}^* \cup (\{b\} \circ \{a\}^*) = \{a\}^* \cup (\{b\}\{a\}^*) \]

Observe that by definition of Kleene’s star

\[ \{a\}^* = \{e, a, aa, aaa \ldots\} \]

and \(L\) is an infinite set

\[ L = \{e, a, aa, aaa \ldots\} \cup \{b\}\{e, a, aa, aaa \ldots\} \]

\[ L = \{e, a, aa, aaa \ldots\} \cup \{b, ba, baa, baaa \ldots\} \]

\[ L = \{e, a, b, aa, ba, aaa baa, \ldots\} \]
Idea of Finite Representation

The expression \( \{a\}^* \cup (\{b\}\{a\}^*) \) is built out of a finite number of symbols:

\[ \{, \}, (, ), *, \cup \]

and describe an infinite set

\[ L = \{e, a, b, aa, ba, aaa baa, \ldots\} \]

We write it in a simplified form - we skip the set symbols \{'\,\,\}\ as we know that languages are sets and we have

\[ a^* \cup (ba^*) \]

We will call such expressions a finite representation

Idea (formal definition to come) of a finite representation of a language \( L \) is the following

We use symbols

\[ (, ), *, \cup, \emptyset, \quad \text{and symbols } \sigma \in \Sigma \]

to write expressions that describe the language \( L \)
Example of a Finite Representation

Let $L$ over $\Sigma = \{0, 1\}$ be a language defined as follows (by a property)

$$L = \{ w \in \{0, 1\}^* : \ w \text{ has two or three occurrences of 1 the first and the second of which are not consecutive } \}$$

The language $L$ can be expressed as

$$L = \{0\}^*\{1\}\{0\}^*\{0\} \circ \{1\}\{0\}^*\{\{1\}\{0\}^* \cup \emptyset^*\}$$

We will define and write the finite representation of $L$ in as

$$L = 0^*10^*010^*(10^* \cup \emptyset^*)$$

We will call the expression above (and others alike) a regular expression
Problem with Finite Representation

Question
Can we finitely represent all languages over an alphabet \( \Sigma \neq \emptyset \)?

Observe the following
1. Different languages must have different representations
2. Finite representations are finite strings over a finite set, so we have that there are \( \aleph_0 \) possible finite representations
3. There are uncountably many (exactly \( C = |R| \)) possible languages over \( \Sigma \neq \emptyset \)

Proof As \( \Sigma \neq \emptyset \) and we have proved that \( |\Sigma^*| = \aleph_0 \)
By definition \( L \subseteq \Sigma^* \), so there are as many languages as subsets of \( \Sigma^* \) that is as many as \( |2^{\Sigma^*}| = 2^{\aleph_0} = C \)
Problem with Finite Representation

Question
Can we finitely represent all languages over an alphabet \( \Sigma \neq \emptyset \)?

Answer
No, we can’t
We have proved that there are countably many (exactly \( \aleph_0 \) ) possible finite representations and there are uncountably many (exactly \( C = |R| \) ) possible languages over \( \Sigma \neq \emptyset \)

NOT ALL LANGUAGES CAN BE FINITELY REPRESENTED
There are \( C \) languages that can not be finitely represented
We can represent only a small countable portion of languages
We will define and STUDY here only two classes of languages: regular and context free languages
Regular Expressions Definition

Definition
We define a $\mathcal{R}$ of regular expressions over an alphabet $\Sigma$ as follows

$\mathcal{R} \subseteq (\Sigma \cup \{(, ), \emptyset, \cup, \ast\})^*$ and $\mathcal{R}$ is the smallest set such that

1. $\emptyset \in \mathcal{R}$ and $\Sigma \subseteq \mathcal{R}$, i.e. we have that

   $\emptyset \in \mathcal{R}$ and $\forall \sigma \in \Sigma$ ($\sigma \in \mathcal{R}$)

2. If $\alpha, \beta \in \mathcal{R}$, then

   $(\alpha \beta) \in \mathcal{R}$ \hspace{1cm} \text{concatenation}

   $(\alpha \cup \beta) \in \mathcal{R}$ \hspace{1cm} \text{union}

   $\alpha^* \in \mathcal{R}$ \hspace{1cm} \text{Kleene’s Star}
Regular Expressions Theorem

Theorem
The set \( \mathcal{R} \) of regular expressions over an alphabet \( \Sigma \) is countably infinite

Proof
Observe that the set \( \Sigma \cup \{(, ), \emptyset, \cup, *\} \) is non-empty and finite, so the set \( (\Sigma \cup \{(, ), \emptyset, \cup, *\})^* \) is countably infinite, and by definition

\[
\mathcal{R} \subseteq (\Sigma \cup \{(, ), \emptyset, \cup, *\})^*
\]

hence we \( |\mathcal{R}| \leq \aleph_0 \)

The set \( \mathcal{R} \) obviously includes an infinitely countable set

\( \emptyset, \emptyset \emptyset, \emptyset \emptyset \emptyset, \ldots, \ldots, \)

what proves that \( |\mathcal{R}| = \aleph_0 \)
Regular Expressions

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

1. $\emptyset \in \mathcal{R}, \ 1 \in \mathcal{R}, \ 0 \in \mathcal{R}$
2. $(01) \in \mathcal{R}, \ 1^* \in \mathcal{R}, \ 0^* \in \mathcal{R}, \ \emptyset^* \in \mathcal{R}, \ (\emptyset \cup 1) \in \mathcal{R}, \ldots,$
   $\ldots, \ (((0^* \cup 1^*) \cup \emptyset)1)^* \in \mathcal{R}$

Shorthand Notation   when writing regular expressions we will keep only essential parenthesis

For example, we will write

$$((0^* \cup 1^* \cup \emptyset)1)^*$$ instead of $(((0^* \cup 1^*) \cup \emptyset)1)^*$

$$1^*01^* \cup (01)^*$$ instead of $(((1*0)1^*) \cup (01)^*)$
Regular Expressions and Regular Languages

We use the regular expressions as a representation languages.
Languages represented by regular expressions are called regular languages.
The idea of representation is explained in the following.

Example the regular expression (written in a shorthand notion)

\[ 1^*01^* \cup (01)^* \]

represents a language

\[ L = \{1\}^*\{0\}\{1\}^* \cup \{01\}^* \]
Definition of Representation

Definition  The representation relation between regular expressions and languages they represent is established by a function \( L \) such that if \( \alpha \) is any regular expression, then \( L(\alpha) \) is the language represented by \( \alpha \).

Formal Definition
The function \( L : R \rightarrow 2^{\Sigma^*} \) is defined recursively as follows:

1. \( L(\emptyset) = \emptyset \), \( L(\sigma) = \{\sigma\} \) for all \( \sigma \in \Sigma \)
2. If \( \alpha, \beta \in R \), then

\[
L(\alpha \beta) = L(\alpha) \circ L(\beta) \quad \text{concatenation}
\]
\[
L(\alpha \cup \beta) = L(\alpha) \cup L(\beta) \quad \text{union}
\]
\[
L(\alpha^*) = L(\alpha)^* \quad \text{Kleene's Star}
\]
Definition
A language \( L \subseteq \Sigma^* \) is regular if and only if

\[ L = \mathcal{L}(\alpha) \]

where \( \mathcal{L} : \mathcal{R} \rightarrow 2^{\Sigma^*} \) is the representation function.

We use a shorthand notation

\[ L = \alpha \quad \text{for} \quad L = \mathcal{L}(\alpha) \]
Examples

E1  Given  \( \alpha \in \mathcal{R} \), for  \( \alpha = ((a \cup b)^* a) \)
Evaluate  \( L \) over an alphabet  \( \Sigma = \{a, b\} \), such that  \( L = \mathcal{L}(\alpha) \)
We write  \( \alpha \) in the **shorthand** notation as  \( \alpha = (a \cup b)^* a \)
and evaluate  \( L = (a \cup b)^* a \) as follows

\[
\mathcal{L}((a \cup b)^* a) = \mathcal{L}((a \cup b)^*) \circ \mathcal{L}(a) = \mathcal{L}((a \cup b)^*) \circ \{a\} =
\]

\[
(a \cup b)^* \{a\} = (\mathcal{L}(a) \cup \mathcal{L}(b))^* \{a\} = (\{a\} \cup \{b\})^* \{a\}
\]

Observe that  \( (\{a\} \cup \{b\})^* \{a\} = \{a, b\}^* \{a\} = \Sigma^* \{a\} \), so

\[
L = \mathcal{L}((a \cup b)^* a) = \Sigma^* \{a\}
\]

\[
L = \{w \in \{a, b\}^* : w \text{ ends with } a\}
\]
Examples

E2  Given $\alpha \in \mathcal{R}$, for $\alpha = ((c^*a) \cup (bc^*))^*$

Evaluate $L = \mathcal{L}(\alpha)$, i.e describe $L = \alpha$

We write $\alpha$ in the shorthand notation as $\alpha = c^*a \cup (bc^*)^*$
and evaluate $L = c^*a \cup (bc^*)^*$ as follows

$$
\mathcal{L}((c^*a \cup (bc^*))^*) = \mathcal{L}(c^*a) \cup (\mathcal{L}(bc^*))^* = \{c\}^*\{a\} \cup (\{b\}\{c\}^*)^*
$$

and we get that

$$
L = \mathcal{L}(c^*a \cup (bc^*)^*) = \{c\}^*\{a\} \cup (\{b\}\{c\}^*)^*
$$
Examples

E3  Given $\alpha \in \mathcal{R}$, for

$$\alpha = (0^* \cup (((0^*(1 \cup (1)))((00^*)(1 \cup (1)))^*)0^*))$$

Evaluate $L = \mathcal{L}(\alpha)$, i.e describe the language $L = \alpha$

We write $\alpha$ in the **shorthand** notation as

$$\alpha = 0^* \cup 0^*(1 \cup 11)((00^*(1 \cup 11))^*)0^*$$

and evaluate

$$L = \mathcal{L}(\alpha) = 0^* \cup 0^*(\{1, 11\})(00^*(\{1, 11\})^*)0^*$$

Observe that $00^*$ contains at least one 0 that separates $0^*(\{1, 11\})$ on the left with $(00^*(\{1, 11\})^*)$ that follows it, so we get that

$$L = \{w \in \{0, 1\}^* : w \text{ does not contain a substring } 111\}$$
Class **RL** of Regular Languages

**Definition**

Class **RL** of regular languages over an alphabet $\Sigma$ contains all $L$ such that $L = L(\alpha)$ for certain $\alpha \in \mathbb{R}$, i.e.

$$RL = \{ L \subseteq \Sigma^* : L = L(\alpha) \text{ for certain } \alpha \in \mathbb{R} \}$$

**Theorem**

There $\aleph_0$ regular languages over $\Sigma \neq \emptyset$ i.e.

$$|RL| = \aleph_0$$

**Proof**

By definition that each regular language is $L = L(\alpha)$ for certain $\alpha \in \mathbb{R}$ and the interpretation function $L : \mathbb{R} \rightarrow 2^{\Sigma^*}$ has an infinitely countable domain, hence $|RL| = \aleph_0$
Class **RL** of Regular Languages

We can also think about languages in terms of closure and get immediately from definitions the following

**Theorem**

Class **RL** of regular languages is the closure of the set of languages

$$\left\{ \{\sigma\} : \sigma \in \Sigma \right\} \cup \{\emptyset\}$$

with respect to union, concatenation and Kleene Star
Languages that are NOT Regular

Given an alphabet \( \Sigma \neq \emptyset \)

We have just proved that there are \( \aleph_0 \) regular languages, and we have also there are \( C \) of all languages over \( \Sigma \neq \emptyset \), so we have the following

**Fact**
There is \( C \) languages that are not regular

**Natural Questions**

Q1 How to prove that a given language is regular?
A1 Find a regular expression \( \alpha \), such that \( L = \alpha \), i.e. \( L = \mathcal{L}(\alpha) \)
Languages that are NOT Regular

Q2 How to prove that a given language is NOT regular?
A2 Not easy!

We will have a Theorem, called Pumping Lemma which provides a criterium for proving that a given language is not regular.

E1 A language

\[ L = 0^*1^* \]

is is regular as it is given by a regular expression \( \alpha = 0^*1^* \).

E2 We will prove, using the Pumping Lemma that the language

\[ L = \{0^n1^n : n \geq 1, n \in N\} \]

is not regular
General Problem

Given a language $L$ over $\Sigma$ and a word $w \in \Sigma^*$, HOW TO RECOGNIZE whether

$$w \in L \quad \text{or} \quad w \notin L$$

Next SUBJECT:
Automata - LANGUAGE RECOGNITION devices