Finite Representation of Languages. Regular Expressions
(Chapter 1, Section 1.8)

Language: any set of strings over an alphabet \( \Sigma \), i.e. any subset of \( \Sigma^* \).

Grammars - finite sets of rules used to describe languages.

Grammars are finite representations. Each grammar is in fact a finite sequence of symbols, i.e. a string over some alphabet. We know that given an alphabet \( \Sigma \), the set of all possible strings \( \Sigma^* \) is countably infinite. Hence the set of all possible grammars is countably infinite.

Since each language is a subset of \( \Sigma^* \), the set of all possible languages is \( 2^{\Sigma^*} \), the power set of \( \Sigma^* \). The power set of \( \Sigma^* \) is not countable (the power set of any countably infinite set is not countable). This means that the set of all possible languages is not countable, though each language is countable.

Thus on one hand we have the set of all possible grammars to be countably infinite and on the other hand we have the set of all possible languages to be uncountable. Hence we cannot have a representation for every language.

Within the set of all possible languages we will consider the subset of those languages that can be represented by a grammar description. We will see that these languages and their grammars can be grouped into four types. We start with the simplest representation called regular expressions/regular grammars and the corresponding languages - regular languages.

1. Regular Expressions

Expressions consisting of string concatenations combined with the symbols \( \cup \) and \( * \), possibly using \( '(' \) and \( ')' \) are called regular expressions.

Example:
If \( \Sigma = \{0,1\} \), regular expressions are:

- \( 0, 1, 010101, \) any combination of 0s and 1s
- \( 0 \cup 1, (0 \cup 1)1^* \)
- \( (0 \cup 1)*01 \)
- \( (0^* \cup 1^*) \)

Before discussing the meaning of a regular expression, we'll give the formal definition:
Formal definition:

**Alphabet** $\sum \cup \{ (,), \emptyset, U, * \}$

1. $\emptyset$ and each member of $\sum$ is a regular expression.
2. If $\alpha$ and $\beta$ are regular expressions, then $(\alpha \beta)$ is a regular expression.
3. If $\alpha$ and $\beta$ are regular expressions, then $(\alpha \ U \beta)$ is a regular expression.
4. If $\alpha$ is a regular expression, then $\alpha^*$ is a regular expression.
5. Nothing else is a regular expression.

2. Regular Languages

Each regular expression can be mapped to a language, using a function $L$

1. $L(\emptyset) = \emptyset$ (the empty set) and $L(\text{a}) = \{\text{a}\}$, for each $\text{a}$ in the alphabet
2. If $\alpha$ and $\beta$ are regular expressions, then $L((\alpha \beta)) = L(\alpha) \ L(\beta)$
3. If $\alpha$ and $\beta$ are regular expressions, then $L((\alpha \ U \beta)) = L(\alpha) \cup L(\beta)$
4. If $\alpha$ is a regular expression, then $L(\alpha^*) = (L(\alpha))^*$

**Note:** $U$ is used in regular expressions, $\cup$ is used to denote set union

Applying this function to a regular expression, we obtain strings in a language.
A language, whose strings can be obtained from a regular expression is called a regular language.

More about the correspondence between a regular expression and the language that it describes (details about the function $L$)

1. Rule 1 above says that $L(\text{a}) = \{\text{a}\}$. This means that the regular expression $\text{a}$ corresponds to a language that consists only of one letter - the letter $\text{a}$

2. Rule 2 says that $L(\alpha \beta) = L(\alpha) \ L(\beta)$, where $\alpha$ and $\beta$ are regular expressions. Let $\alpha = \text{a}, \beta = \text{b}$. We have already seen that the regular expression $\text{a}$ corresponds to $L1 = \{\text{a}\}$, similarly the regular expression $\text{b}$ will correspond to the set of one letter $L2 = \{\text{b}\}$. The regular expression $\text{ab}$ will correspond to the concatenation of $L1$ and $L2$, i.e. to $L3 = \{\text{ab}\}$

Another example: Let $\alpha = \text{ab}, \beta = \text{bc}$. Then the composition $\{\text{ab}\} \ {\text{bc}}$ will result in a language consisting of one string only - the string $\text{abbc}$

3. Rule 3 says that $L((\alpha \ U \beta)) = L(\alpha) \cup L(\beta)$. Let $\alpha = \text{a}, \beta = \text{b}$. Then the regular expression $\alpha \ U \beta$ will describe the set $\{\text{a}\} \cup \{\text{b}\} = \{\text{a,b}\}$. 


More examples:

<table>
<thead>
<tr>
<th>Expression</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>ab U b</td>
<td>L = {ab,b}</td>
</tr>
<tr>
<td>ab U ac</td>
<td>L = {ab,ac}</td>
</tr>
<tr>
<td>a U ab U b</td>
<td>L = {a,b,ab}</td>
</tr>
</tbody>
</table>

4. Rule 4 says that then \( \mathcal{L}(a^*) = (\mathcal{L}(a))^* \). Let \( a = a \). The expression \( a^* \) corresponds to \( \{a\}^* \), i.e. the empty strings and all possible strings that consist only of \( a \).

More examples that use all rules

<table>
<thead>
<tr>
<th>Expression</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a U b) *</td>
<td>L = {a,b}*</td>
</tr>
<tr>
<td>(ab U bc) *</td>
<td>L = {ab,bc}*</td>
</tr>
<tr>
<td>(a^* U b)</td>
<td>L = {a}^* \cup {b} , i.e. the language consists of the empty string, all possible strings of ( a ), and the letter ( b )</td>
</tr>
<tr>
<td>(a^* U b^*)</td>
<td>L = {a}^* \cup {b}^* , i.e. the language consists of the empty string, all possible strings of ( a ) and all possible strings of ( b )</td>
</tr>
<tr>
<td>a(a^* U b)</td>
<td>L = { aw : w \in {a}^* \cup {b} } i.e. L consist of strings that start with an ( a ), followed by zero or more as, or by one ( b )</td>
</tr>
<tr>
<td>a(a U b) *a</td>
<td>L = (awa: w \in {a,b}*) i.e. L consists of all string that start with an ( a ), followed by any possible string consisting of ( a )'s and ( b )'s, including the empty string, and end with an ( a ).</td>
</tr>
<tr>
<td>b^<em>ab</em></td>
<td>L = {xay: x,y \in {b}^* } Note that while the regular expression has ( b^* ) both to the left and to the right of ( a ), when representing ( L ) as a set of strings, we use ( x ) and ( y ) - different letters for the left and the right parts. This is so because ( b^* ) corresponds to any string of ( bs ), and the left part of a string may be different from the right part.</td>
</tr>
<tr>
<td>(a^<em>ba</em>) U bc*</td>
<td>L = { xby : x,y \in {a}^* \cup {bz, z \in {c}^* }</td>
</tr>
<tr>
<td>ab^* (b U bc*)</td>
<td>L = {axy: x \in {b}^<em>, y \in {b } \cup {b}{c}^</em> }</td>
</tr>
</tbody>
</table>
Examples:

1. Consider the regular expression \((a \cup b)^*\)

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

Thus \((a \cup b)^*\) represents the set \{a,b\}^*

2. Consider the regular expression: \((a^* \cup b^*)\)

\[
L((a^* \cup b^*)) = L(a^*) \cup L(b^*) = L(a^*) \cup L(b^*) = \{a\}^* \cup \{b\}^* = \{w : w \in \{a\}^* \cup \{b\}^*\}
\]

This is the language consisting of \(e\), sequences of \(a\)'s and sequences of \(b\)'s.

3. Consider the regular expression \((a^* \cup b^*)^*\)

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

\(a^*\) gives all strings containing zero, one, or more \(a\)'s, and \(b^*\) gives all strings containing zero, one, or more \(b\)'s.

The iteration \{a^*,b^*\}^* would give all possible strings over \{a,b\}, hence \{a^*,b^*\}^* = \{a,b\}^*

Since \((a^* \cup b^*)^*\) represents the set \{a,b\}^*, and \((a \cup b)^*\) represents the same set, we conclude that \((a^* \cup b^*)^* = (a \cup b)^*\)
4. Consider the regular expression \((a \cup b^*)\)

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

The set \{a,b^*\} contains the empty string, the string \(a\), and all strings containing one or more \(b\)'s.

5. Consider the regular expression \((a \cup b^*)^*\)

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

\(b^*\) gives all strings containing zero or more \(b\)'s.

The iteration \{a, b^\}^* would give all possible strings over \{a,b\}, hence
\{a, b^\}^* = \{a,b\}^*

Since \((a \cup b^*)^*\) represents the set \{a,b\}^*, and \((a \cup b)^*\) represents the same set, we conclude that
\[(a \cup b^*)^* = (a \cup b)^*\]

Similarly, we can conclude that
\[(a^* \cup b)^* = (a \cup b)^*\]

6. Consider the regular expression \((a^*ba)\)

\[
\mathcal{L}\left((a^*ba)\right) = \mathcal{L}(a^*) \mathcal{L}(b) \mathcal{L}(a) = \mathcal{L}(a)^* \mathcal{L}(b) \mathcal{L}(a) = \{a^*\}\{b\}\{a\} = \{a^*ba\}
\]

The set \{a^*ba\} contains strings that end with \(ba\), preceded by zero or more \(a\)'s.
7. Consider the regular expression \((a^*ba)^*\)

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

The set \{a^*ba\} contains strings that end with \(ba\), preceded by zero or more \(a\)'s. Its Kleene star \{a^*ba\}^* will consists of strings where there are no two or more consecutive \(b\)'s, and all strings end with \(a\). Each \(b\) is either first in the string, or surrounded by one or more \(a\)'s. E.G.: ba, aaaba, aaababa, aaabaaaabaababaaaa, baaaaaba,

**Important:** In the above examples we have shown that the expressions:

\((a U b)^*\), \((a^* U b^*)^*\), \((a U b^*)^*\), and \((a^* U b)^*\) represent the same set \{a,b\}^*. 

Learning goals

After studying regular expressions and regular languages you should be able to:

1. **Recognize a regular expression when you see it, and write syntactically correct regular expressions.**
2. **Apply the function** $L$ **to a regular expression to find the language represented by that regular expression.**
3. **Describe a language by a regular expression.**
   - E.g. The language over \{a,b\} whose strings end in \(b\), is generated by the regular expression $(a U b)*b$
   - The language over \{a,b\} whose strings contain at least three consecutive \(a\)'s, is generated by the expression $(a U b)*aaa(a U b)*$
4. **Remember that** $(a U b)*$, $(a* U b*)*$, $(a U b*)*$, and $(a* U b)*$ represent the same set \{a,b\}*. 

This is true also if instead of a single symbol \(a\) or \(b\), we have a string, e.g. $(ab U bc)*$ and $((ab)* U (bc)*)*$ represent the set \{ab, bc\}*

These are all strings in \{a,b,c\}* such that:
- start either with \(a\) or \(b\)
- end either with \(b\) or \(c\)
- there are no repeated \(a\)'s or \(c\)'s
- \(a\) can be preceded by \(b\) or \(c\), and followed only by \(b\)
- \(c\) can be preceded only by \(b\) and followed by \(a\) or \(b\)
- there are no three or more consecutive \(b\)'s

**Useful rules** when manipulating regular expressions:
If \(\alpha\) and \(\beta\) are regular expressions, then
1. $\alpha * \alpha * = \alpha *$
2. $(\alpha *)* = \alpha *$
3. $\alpha * U \alpha * = \alpha *$
4. $\alpha * U \alpha * \beta * = \alpha * \beta *$