1 Operations on Languages

Operations on Languages

• Recall: A language is a set of strings
• We can consider new languages derived from operations on given languages
  – e.g., \( L_1 \cup L_2, L_1 \cap L_2, \ldots \)
• A simple but powerful collection of operations:
  – Union, Concatenation and Kleene Closure

Union is a familiar operation on sets. We define and explain the other two operations below.

**Concatenation of Languages**

**Definition 1.** Given languages \( L_1 \) and \( L_2 \), we define their concatenation to be the language \( L_1 \circ L_2 = \{ xy \mid x \in L_1, \ y \in L_2 \} \)

**Example 2.**
- \( L_1 = \{ \text{hello} \} \) and \( L_2 = \{ \text{world} \} \) then \( L_1 \circ L_2 = \{ \text{helloworld} \} \)
- \( L_1 = \{ 00, 10 \}; \ L_2 = \{ 0, 1 \}. \ L_1 \circ L_2 = \{ 000, 001, 100, 101 \} \)
- \( L_1 = \) set of strings ending in 0; \( L_2 = \) set of strings beginning with 01. \( L_1 \circ L_2 = \) set of strings containing 001 as a substring
- \( L \circ \{ \epsilon \} = L. \ L \circ \emptyset = \emptyset. \)

**Kleene Closure**

**Definition 3.**

\[
L^n = \begin{cases} 
\{ \epsilon \} & \text{if } n = 0 \\
L^{n-1} \circ L & \text{otherwise}
\end{cases}
\]

\[L^* = \bigcup_{i \geq 0} L^i\]

i.e., \( L^i \) is \( L \circ L \circ \cdots \circ L \) (concatenation of \( i \) copies of \( L \)), for \( i > 0 \).

**\( L^* \), the Kleene Closure of \( L \):** set of strings formed by taking any number of strings (possibly none) from \( L \), possibly with repetitions and concatenating all of them.

- If \( L = \{ 0, 1 \} \), then \( L^0 = \{ \epsilon \}, L^2 = \{ 00, 01, 10, 11 \}. \ L^* = \) set of all binary strings (including \( \epsilon \)).
- \( \emptyset^0 = \{ \epsilon \}. \) For \( i > 0, \emptyset^i = \emptyset. \ \emptyset^* = \{ \epsilon \} \)
- \( \emptyset \) is one of only two languages whose Kleene closure is finite. Which is the other? \( \{ \epsilon \}^* = \{ \epsilon \}. \)
2 Regular Expressions

2.1 Definition and Identities

Regular Expressions
A Simple Programming Language

A regular expression is a formula for representing a (complex) language in terms of “elementary” languages combined using the three operations union, concatenation and Kleene closure.

Regular Expressions
Formal Inductive Definition

Syntax and Semantics
A regular expression over an alphabet $\Sigma$ is of one of the following forms:

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\emptyset$</td>
<td>$L(\emptyset) = {}$</td>
</tr>
<tr>
<td>$\epsilon$</td>
<td>$L(\epsilon) = {\epsilon}$</td>
</tr>
<tr>
<td>$a$</td>
<td>$L(a) = {a}$</td>
</tr>
</tbody>
</table>

Induction

| $(R_1 \cup R_2)$ | $L((R_1 \cup R_2)) = L(R_1) \cup L(R_2)$ |
| $(R_1 \circ R_2)$ | $L((R_1 \circ R_2)) = L(R_1) \circ L(R_2)$ |
| $(R_1^*)$        | $L((R_1^*)) = L(R_1)^*$ |

Notational Conventions

Removing the brackets To avoid cluttering of parenthesis, we adopt the following conventions.

- Precedence: $\ast, \circ, \cup$. For example, $R \cup S^* \circ T$ means $\text{\textit{(R \cup ((S^*) \circ T) )}}$
- Associativity: $(R \cup (S \cup T)) = ((R \cup S) \cup T)$ and $(R \circ (S \circ T)) = ((R \circ S) \circ T) = R \circ S \circ T$.

Also will sometimes omit $\circ$: e.g. will write $RS$ instead of $R \circ S$.

Regular Expression Examples
Regular Languages

Definition 4. A language \( L \subseteq \Sigma^* \) is a regular language iff there is a regular expression \( R \) such that \( L(R) = L \).

Some Regular Expression Identities
We say \( R_1 = R_2 \) if \( L(R_1) = L(R_2) \).

- Commutativity: \( R_1 \cup R_2 = R_2 \cup R_1 \) (but \( R_1 \circ R_2 \neq R_2 \circ R_1 \) typically)
- Associativity: \( (R_1 \cup R_2) \cup R_3 = R_1 \cup (R_2 \cup R_3) \) and \( (R_1 \circ R_2) \circ R_3 = R_1 \circ (R_2 \circ R_3) \)
- Distributivity: \( R \circ (R_1 \cup R_2) = R \circ R_1 \cup R \circ R_2 \) and \( (R_1 \cup R_2) \circ R = R \circ R_1 \cup R \circ R_2 \circ R \)
- Concatenating with \( \epsilon \): \( R \circ \epsilon = \epsilon \circ R = R \)
- Concatenating with \( \emptyset \): \( R \circ \emptyset = \emptyset \circ R = \emptyset \)
- \( R \cup \emptyset = R \). \( R \cup \epsilon = R \) iff \( \epsilon \in L(R) \)
- \( (R\epsilon)^* = R\epsilon^* \)
- \( \emptyset^* = \epsilon \)

Useful Notation

Definition 5. Define \( R^+ = RR^* \). Thus, \( R^* = R^+ \cup \epsilon \). In addition, \( R^+ = R^* \) iff \( \epsilon \in L(R) \).