20/09/2023

# MATH 2301

* Admin : Final exam on 8 Nov → More on Wattle soon

* Recap : Alphabets $\Sigma$ (Typically $\{0,1\}$)
  Languages $L \subseteq \Sigma^*$
  Operations : concatenation, union, intersection, star

* Star of a language (Kleene star)

  $L$ a language on some $\Sigma$

  $$L^* = \{\varepsilon\} \cup \{w_1 w_2 \cdots w_n \mid w_i \in L\}$$

  E.g. $L = \{00, 110\}$

  $$L^* = \{\varepsilon, 00, 110, 0000, 00110, 11000, 110110, \cdots\}$$

  Note : $\Sigma \subseteq \Sigma^*$, so $\Sigma$ itself is a language

  $\Sigma^*$ in the above sense is the same as the $\Sigma^*$ we discussed earlier.

* Dictionary order (a total order on $\Sigma^*$)

  Given an ordering on $\Sigma$, for example, $(0,1)$ we can produce a total order on $\Sigma^*$, called the dictionary order.

  Let $v = a_1 \cdots a_m$ ← length $m$

  $w = b_1 \cdots b_n$ ← length $n$

If $m < n$ then $v < w$

If $n < m$ then $w < v$

If $m = n$ : start by comparing $a_1$ & $b_1$

If $a_1 < b_1$ $v < w$

If $b_1 < a_1$ $w < v$

If $a_1 = b_1$, compare $a_2$ & $b_2$ and so on.

[dictionary ordering]

---

Back to $*$ (Star)

$L = \emptyset$, then $L^* = \{\varepsilon\}$

$L = \{\varepsilon\}$, then $L^* = \{\varepsilon\}$

$L^*$ is infinite except the two cases above [check]

---

$$L^* = \{\varepsilon\} \cup L \cup (L \circ L) \cup (L \circ L \circ L) \cup (L \circ L \circ L \circ L) \cup \ldots$$

---

$*$ Use this stuff for pattern-matching. (regular expr)

Fix alphabet $\Sigma$.

A regular expression is a word made up of symbols from $\Sigma$ as well as certain additional symbols, which follow some recursive rules.

Three basic constructors (base cases) (for regexes $r$)

① $r = a$, for some $a \in \Sigma$

② $r = \varepsilon$ (empty string)

3. $r = \phi$ (empty set)

E.g. $\Sigma = \{0, 1\}$, the following are valid regular expressions:

- 0
- 1
- $\varepsilon$
- $\phi$

$\boxed{\text{REGEX = regular expression}}$

---

Three inductive/recursive constructors for regexes $r$:

4. $r = r_1 r_2$, e.g. $r = 00$, $01$, $1110$

where $r_1$ and $r_2$ are valid regexes.

5. $r = r_1 \mid r_2$, e.g. $r = 0 \mid 110$, $r = 101 \mid 11 \mid 1100$

(OR)

where $r_1$, $r_2$ are valid regular expressions.

6. $r = (r_1)^*$, where $r_1$ is a valid regular expression

E.g. $(010)^*$

Note: Parentheses are used for grouping.

## * Order of operations

When parsing regular expressions, order is:

1. brackets / parentheses first
2. star next
3. concatenation next
4. OR last

## * Examples : $\Sigma = \{0,1\}$

An exercise in parsing a complicated regex.

$$r = (01 \mid \emptyset^* \mid 111)^* \, 010 \, (0 \mid 1 \mid \varepsilon \mid 00)$$

(Analogy: $(\underbrace{x + 5y + 25z^2}) ^3 (\underbrace{56x^2 + 12y^2})$)

$$r = \underbrace{(01 \mid \emptyset^* \mid 111)^*}_{r_1} \, \underbrace{010}_{r_2} \, \underbrace{(0 \mid 1 \mid \varepsilon \mid 00)}_{r_3}$$

[concatenation constructor]

$r_1 = (01 \mid \emptyset^* \mid 111)^* = s^*$, where $s = 01 \mid \emptyset^* \mid 111$

[star constructor]

$r_2 = 010$

$r_3 = (0 \mid 1 \mid \varepsilon \mid 00)$

$$s = \underbrace{01}_{s_1} \mid \underbrace{\emptyset^*}_{s_2} \mid \underbrace{111}_{s_3}$$

[OR constructor]

$s_1 = 01$ → concatenation of 0 & 1

$s_2 = \emptyset^*$

$s_3 = 111$

## More examples

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

$r = 0$, $r = 1$, $r = \varepsilon$, $r = \phi$.

$r = 00$, $101$, $\varepsilon | 01 | \phi$, $(\varepsilon | (01)^* | \phi)^* 11$

etc.