Regular Expressions

Regular expressions are yet another way to describe some languages using regular operations. We already saw operations like concatenation and star.

→ We can write expressions like $01^* = 0 \cdot 1^*$.
→ Similar to mathematical operations: $(3+5)*4$
→ We can combine regular operations and rules of precedence will apply.

$3+5*4$

Precedence: Star, Concatenation, Union
unless parenthesis is there.

$0 \cup (01)^*$

Def 1.52: R is a regular expression over Σ if

R is (1) a for some $a \in \Sigma$.

(2) ε empty string

(3) ϕ empty set

(4) $R_1 \cup R_2$ where R_1 and R_2 are regular expressions

(5) $R_1 \circ R_2$ " "

(6) R_1^* where R_1 is a reg. exp.

$\hookrightarrow \{x_1 x_2 \ldots x_k \mid x_i \in R_1, \ k \geq 0\}$

Regular expression denotes a set (or a language) of strings and not a single string. For instance,

Reg exp a denotes $\{a\}$

Reg exp ε denotes $\{\varepsilon\}$

Reg exp 01^* denotes $\{0, 01, 011, 0111, \ldots\}$

Reg exp ϕ denotes the empty language.

Notation : Σ set of all strings of length 1.

R^+ is repetition where R appears $(k \neq 0)$ at least once.

$\{x_1 x_2 \ldots x_k \mid x_i \in R, \ k \geq 1\}$

$R^+ = \underline{R R^*}$

$R^+ \cup \{\varepsilon\} = R^*$

R^k = k repeats of R.

Like in DFA/NFA, we also use $L(R)$ to denote the language represented by R.

Examples: 0^*10^* : Binary strings containing exactly one 1

$\Sigma = \{0,1\}$, $\Sigma^* 1 \Sigma^*$: Binary strings containing atleast one 1.

$\Sigma^* 00 \Sigma^*$: Binary strings containing 00 as a substring

$(\Sigma\Sigma)^*$: Binary strings of even length.

Read examples 1.51 and 1.53 from Sipser.

1.51 : $(0 \cup 1)^*$ — All binary strings.

1.53 : Select examples below.

→ $0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1$ — All binary strings that start and end with the same symbol.

→ $(0 \cup \varepsilon)(1 \cup \varepsilon)$

$= 01 \cup 0\varepsilon \cup \varepsilon 1 \cup \varepsilon\varepsilon = \{01, 0, 1, \varepsilon\}$

Some more rules / conventions

$1^* \phi = \phi$ $\quad\quad$ $1^* \varepsilon = 1^*$

$\phi^* = \{\varepsilon\}$

$(0 \cup \varepsilon)1^* = 01^* \cup \varepsilon 1^* = 01^* \cup 1^* = \begin{Bmatrix} 0, 01, 011, 0111, \ldots \\ \varepsilon, 1, 11, 111, \ldots \end{Bmatrix}$

$R \cup \phi = R$ $\qquad R \cup \varepsilon =$ need not be R

$R \circ \phi = \phi$ $\qquad R \circ \varepsilon = R$

If $R = 0$, $L(R) = \{0\}$

$L(R \cup \varepsilon) = \{0, \varepsilon\}$

$L(R \circ \phi) = \phi$.

A parser/compiler can analyse a reg. exp if it is in the correct form.

Q: What are the class of languages that can be described using regular exp.?

Theorem 1.54: A language is regular iff some regular expression describes it. $\qquad$ if and only if

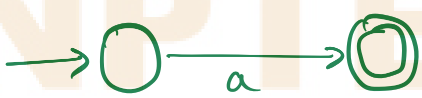
This is yet another characterization for regular languages. We prove it in two parts:

(1) IF ($\Leftarrow$) and (2) ONLY IF ($\Rightarrow$)

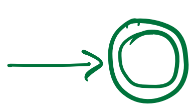
Lemma 1.55: If a regular expression describes a language, then it is regular.

Proof: Given a regular expression R, we will construct an NFA that recognizes $L(R)$.

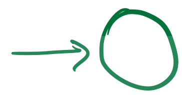
(1) $R = a$ for some $a \in \Sigma$. Then $L(R) = \{a\}$



(2) $R = \varepsilon$. Then $L(R) = \{\varepsilon\}$.



(3) $R = \phi$. Then $L(R) = \phi$.



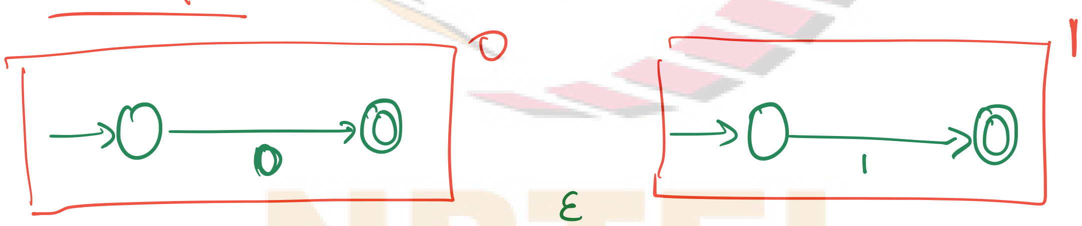
(4) $R = R_1 \cup R_2$

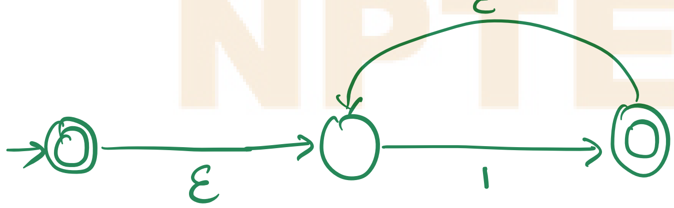
(5) $R = R_1 \circ R_2$

(6) $R = R_1^*$

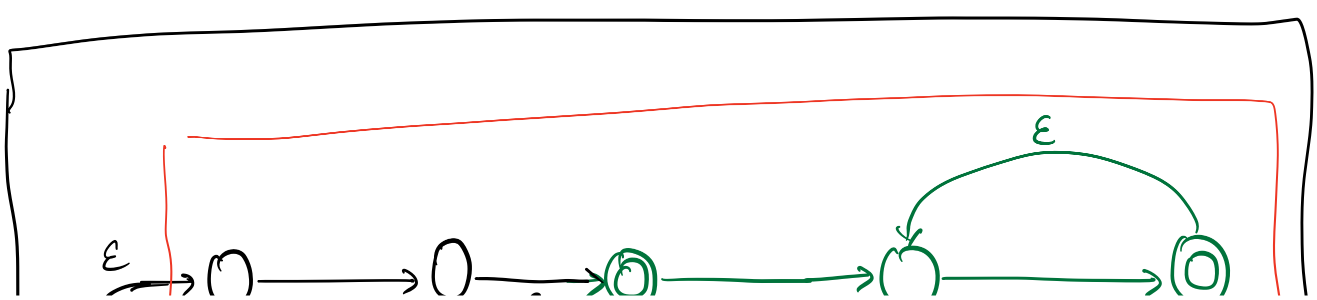
Follows using closure properties of regular languages.

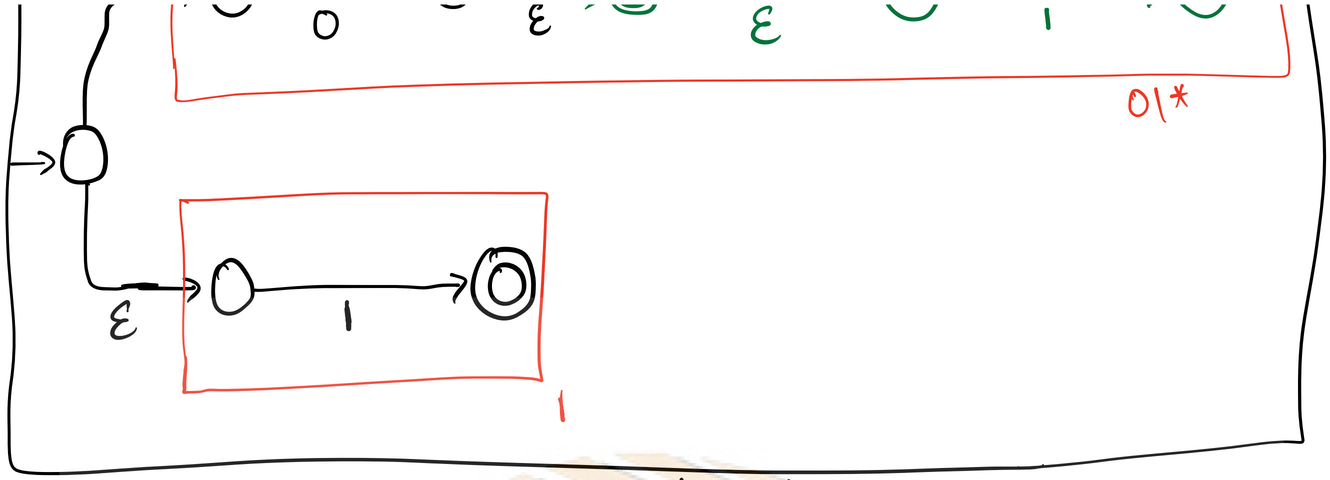
Example: $01^* \cup 1$





→ Accepts 1^*





Accepts $01^* \cup 1$

Read examples 1.56 and 1.58. (Sipser)

1.56: $(ab \cup a)^*$

1.58: $(a \cup b)^* aba$.