Lecture 6 - May 22

Lexical Analysis

Meanings, Precedence of RE Operators
RE Constructions and Specifications
DFA: Introduction

$L = \{ab, bc\}$

Given some language L

$\boxed{L^*}$ ↳ as many concat/repetition as needed, including no concat/repet.

concat. L to itself zero time

$L^0 = \{\boxed{\varepsilon}\}$ → a string resulted from zero concat.

$L^1 = \{x \mid x \in L\}$

$L^2 = LL = \{x_1 x_2 \mid x_1 \in L \wedge x_2 \in L\}$

$\vdots$

$L^i = \underbrace{LL \cdots L}_{i \text{ times}} = \{x_1 x_2 \cdots x_i \mid (\forall j \cdot 1 \leq j \leq i \Rightarrow x_j \in L)\}$

Q1. $|L^i| = \underbrace{|L|}_{x_1} \cdot \underbrace{|L|}_{x_2} \cdot \ldots \cdot \underbrace{|L|}_{x_i} = |L|^i$

Q2. Given $L = \underbrace{\{0\}}_{\text{alphabet}}{}^* = \{\varepsilon, 0, 00, \ldots\}$ → strings of arb. length

language

What is L^*? → strings from arb. # of concat

$L^* = L$ → not necessarily true for any L

Constructions of REs

Meaning of each reg. exp. operation is defined formally by its corresponding language.

Recursive Case: Given that E and F are regular expressions:

- The union $E + F$ is a regular expression.

$$L(E + F) = L(E) \cup L(F)$$

an RE operator

- The concatenation EF is a regular expression.

$$L(\boxed{EF}) = \boxed{L(E)L(F)}$$

concat. of reg. exp.

concat. of reg. languages

- Kleene closure of E is a regular expression.

reg. languages

$$L(E^*) = (L(E))^*$$

reg. exp.

- A parenthesized E is a regular expression.

$$L((E)) = L(E)$$

Base Case:

- Constants ϵ and $\varnothing$ are regular expressions.

shortest reg. exp.

a singleton language with just ϵ

$$L(\epsilon) = \boxed{\{\epsilon\}}$$
$$L(\varnothing) = \varnothing$$

empty language

- An input symbol $a \in \boxed{\Sigma}$ is a regular expression.

$\{a, \ldots\}$

match verbatim

$$L(a) = \{a\}$$

singleton language

$L(a) = \{a\}$

only input string "a" can be matched/recognized

$\epsilon \notin L(a)$ $\quad b \notin L(a)$ $\quad aa \notin L(a)$

RE Construction: Exercise

Given a **language** L,
derive the following **languages** constructed from **REs**:

1. $\varnothing + L$ 2. $\varnothing L$ 3. $\varnothing^*$ 4. $\varnothing^* L$

r.e. r.e.

empty

reg. exp. op.

$^* \quad \phi^* L = \{x_1 x_2 \mid x_1 \in \phi^* \wedge x_2 \in L\}$

$\epsilon x_2 = x_2 \quad = \{x_2 \mid x_2 \in L\} \quad \{\epsilon\}$

$= L$

$\phi + L = L = L + \phi$

union

$** \quad \phi L = \{x_1 x_2 \mid \boxed{x_1 \in \phi} \wedge x_2 \in L\}$

F

F

$= \phi$

ϕ^*

$= \boxed{\phi^0} \cup \phi^1 \cup \phi^2 \cup \ldots$

no concat

false

$\{x \mid \boxed{x \in \phi}\}$

ϕ ∵ no x can satisfy false

false

$\{x_1 x_2 \mid \boxed{x_1 \in \phi \wedge x_2 \in \phi}\}$

$= \{\epsilon\} \cup \phi \cup \phi \cup \ldots$

$= \{\epsilon\}$

Implication

P	Q	P ⇒ Q
F	F	T
F	T	T
T	F	F
T	T	T

RE Specification: Exercise

Write a regular expression for the following language

set of strings = { w | w has alternating $0'$s and $1'$s } = $\mathcal{L}$

language

$01 \in \mathcal{L}$ $010 \in \mathcal{L}$ $\varepsilon \in \mathcal{L}$ $0 \in \mathcal{L}$

$00 \notin \mathcal{L}$

∵ no violation of alternating 0s and 1s

V1

$1(01)^* + (01)^* + (10)^* + 0(10)^*$

$\{1, 101, 10101, \dots\}$ $\{\varepsilon, 01, 0101, \dots\}$ $\{\varepsilon, 10, 1010, \dots\}$ $\{0, 010, 01010, \dots\}$

V2

$(1 + \varepsilon)(01)^* + (0 + \varepsilon)(10)^*$

RE: Operator Precedence

$*$ (highest)
concat
$+$ (lowest)

- Are RE_1 and RE_2 equivalent?
- A string in $L(RE_1)$ but <u>not</u> in $L(RE_2)$?
- A string in $L(RE_2)$ but <u>not</u> in $L(RE_1)$?

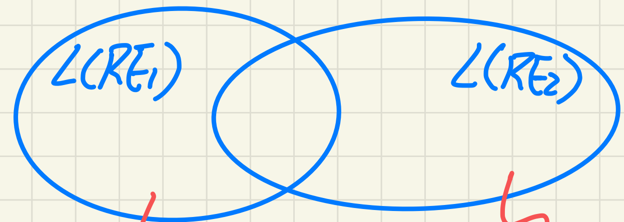
$\boxed{10^*}$ vs. $(10)^*$

$10^* \equiv 1(0^*)$

Witness of inequality:

$100 \in 10^*$

$100 \notin (10)^*$



$\exists w \cdot w \in L(RE_1) \land w \notin L(RE_2)$

$\exists w \cdot w \in L(RE_2) \land w \notin L(RE_1)$

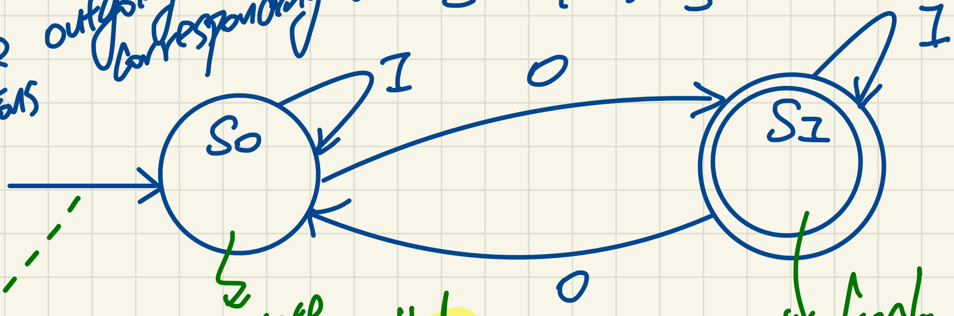
Exercises.

$01^* + 1$ vs. $0(1^* + 1)$

$0 + 1^*$ vs. $(0 + 1)^*$

$\{ w \mid w \text{ has an odd \# of } 0\text{s} \}$

Each state must have outgoing transitions corresponding to $\Sigma = \{0, 1\}$



start state (with a unlabelled transition)

final, accept state (at least one)

S_0 : read even # of 0s

S_1 : read odd # of 0s

DFA: Exercise

Draw the transition diagram of a DFA which accepts/recognizes the following language:

$\{ w \mid w \neq \varepsilon \wedge w \text{ has equal \# of alternating 0's and 1's} \}$