∫(͡° ͜ʖ ͡°)dx

CS 719

Topics in Mathematical Foundations of Formal Verifications

Notes By: Aryaman Maithani

Spring 2020-21

Lecture 1 (11-01-2021)

11 January 2021 09:34

Recap of Regular Languages

Different formalisms surprisingly describe the same class of lang- regular languages. Regular expressions, DFA, NFA, MSO logic.

Notation and Setup (for the rest of course)

Fix a finite alphabet Σ.
A (finite) word over Σ is a finite sequence $a_0 a_1 \cdots a_n$ of elements of Σ. $u, v, w \ldots$ are used for words.

$$w = a_0 a_1 \ldots a_n \quad \text{where each } a_i \in \Sigma.$$

The empty sequence corresponds to the unique word of length 0 and is denoted by ϵ, the empty word.

Σ^* = the set of all finite words over Σ. $(\epsilon \in \Sigma^*)$
$\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$ = the set of all non-empty words over Σ.

Concatenation of words:

$$\cdot : \Sigma^* \times \Sigma^* \longrightarrow \Sigma^*$$
$$(u, v) \longmapsto u \cdot v$$

defined in the usual manner.

→ The $\cdot$ operation on Σ^* is associative.
That is, $\forall u, v, w \in \Sigma^* : (u \cdot v) \cdot w = u \cdot (v \cdot w)$.

→ ϵ acts as an (the) identity for $\cdot$

This is an example of a monoid $(X, \star)$

$$\forall w \in \Sigma^* : \quad \epsilon \cdot w = w \cdot \epsilon = w.$$

(Another example of monoid: $(\mathbb{N}, +)$
$\mathbb{N} = \{0, 1, \ldots\}$ in this course)

(Later we'll look a finite monoids.)

$$\ell : \Sigma^* \longrightarrow \mathbb{N}$$
$$u \longmapsto \text{length of } u = \ell(u)$$

Note $\ell(u \cdot w) = \ell(u) + \ell(w)$
$\ell(\epsilon) = \ell(0)$

Thus, ℓ is a monoid morphism.

Def^n. A language L is simply a subset of Σ^*.

Given languages $L_1, L_2 \subset \Sigma^*$, we define

$$L_1 \cdot L_2 = \{ w_1 \cdot w_2 \mid w_1 \in L_1, \ w_2 \in L_2 \}.$$

Regular Expressions

$$r \equiv \phi \mid \epsilon \mid \underset{\Sigma}{\underset{\cap}{a}} \mid r_1 + r_2 \mid r_1 \cdot r_2 \mid r^*$$

$r \rightsquigarrow L(r)$ language associated to r

$L(r)$ is defined by structural induction on r.

- $L(\phi) = \phi$
- $L(\epsilon) = \{\epsilon\}$
- $L(a) = \{a\}$ $\quad (a \in \Sigma)$
- $L(r_1 + r_2) = L(r_1) \cup L(r_2)$
- $L(r_1 \cdot r_2) = L(r_1) \cdot L(r_2)$ (RHS defined earlier)

• $L(r^*) = \{\varepsilon\} \cup L(r) \cup L(r)\cdot L(r) \cup L(r)\cdot L(r)\cdot L(r) \cup \cdots$

$$= \bigcup_{i=0}^{\infty} L^i \qquad \left(L^0 = \{\varepsilon\},\ L^1 = L(r),\ \underset{i \geq 1}{L^{i+1} = L^i \cdot L} \right)$$

[Example. $(ab)^* = \{ \varepsilon, ab, abab, \ldots \}$.]

Def^n: A language $L \subseteq \Sigma^*$ is said to be regular if there exists a regular expression r such that
$$L(r) = L.$$

Thm. Regular languages are closed under union, intersection, complementation, concatenation.

(As per our def^n using regular expressions, union & concatenation are obvious.)

Some of the above is easier to prove under diff. formalisms. One first shows that two diff. formalisms are actually same.

Def^n. (Extended reg. expressions)

$$r \equiv \phi \mid \epsilon \mid \underset{\Sigma}{a} \mid r_1 + r_2 \mid r_1 \cap r_2 \mid \neg r \mid r_1 \cdot r_2 \mid r^*$$

these we can add, in view of th^m, w/o changing the class of languages

Q: What subclass of language will we get if we restrict ourselves to a subset of the operators?

Defⁿ: (Star-free reg. expressions) Exclude the $*$ operator.

Q. Which regular languages admit a star free representation?

(Non?) Example : $r = (ab)^*$

Can we rewrite this without $*$?