CPSC 421/501 Oct 6, 2020

§1.3 - Regular Expression → Regular Languages

(give the algorithm)

- State Theorem: Regular Language → Regular Expr.

(give an example but skip the algorithm)

§1.4 - Instead of "$\overbrace{\text{pumping lemma}}^{\text{avoid}}$"

we will use the "Myhill-Nerode Theorem":

Also → - to prove some languages are <u>not regular</u>

- to find minimum # states in a DFA

for some languages

For Thursday, we'll have student selected

breakout rooms; you'll need <u>Zoom ≥ 5.3.0</u>

§1.3 Upshot $\{\underbrace{\text{Regular Languages}}_{\text{recognized by a DFA}}\} = \{\text{Described by a regular expression}\}$

Regular Expression: If Σ is an alphabet, a regular expression over Σ:

1. If $\sigma \in \Sigma$, "σ" is a regular expression
2. "ε" is a regular expression
3. $\emptyset$ (empty set)

if R_1, R_2 regular expressions, then

4. $(R_1 \cup R_2)$
5. $(R_1 \circ R_2)$
6. (R_1^*)

are also regular expressions

[Sip] gives a bunch of examples

Say $\Sigma = \{a, b\}$, then

$$\Sigma^* aa \Sigma^* = (a \cup b)^* aa (a \cup b)^*$$

$$= (a \cup b)^* \circ a \circ a \circ (a \cup b)^*$$

could be $((a \cup b)^* \circ a) \circ (a \circ (a \cup b)^*)$

describes $\{a,b\}^* \circ \{a\} \circ \{a\} \circ \{a,b\}^*$ notation in §1.1

or $\Sigma^* aa \Sigma^*$ in shorthand

(1) $\Sigma^* aa \Sigma^* = \{ w \in \Sigma^* \mid w$ has aa as a substring $\}$

(2) $(\Sigma^* \cup a) \circ (bb) = \{\Sigma^* bb\} \cup \{abb\} = \Sigma^* bb$

$= \{ w \in \Sigma^* \mid w$ ends in $bb \}$

(3) $(a \cup bb) \circ (aa) = \{ aaa, bbaa \}$

or $(a \cup bb) aa$

Is $ab \in (a \cup b)^*$

Is $ab \in$ language that $(a \cup b)^*$ describes

$(a \cup b)^*$ as set $\{a,b\}^* = \{$all strings over $\{a,b\}\}$

(4) $a^* \cup b^*$ as a set $\{a^*\} \cup \{b^*\}$

$= \{\varepsilon, a, aa, a^3, a^4\} \cup \{\varepsilon, b, b^2, b^3, \ldots\}$

$= \{ w \mid w$ consists only of a's or only of b's $\}$

build NFA

(5) $(a^5)^* \circ (a^7)^* = \{ a^{5m+7n} \mid m,n = 0,1,2,\ldots \}$

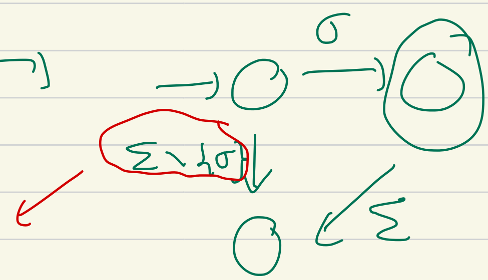
Thm: Any regular expression describes a regular language.

Pf: (1) $\{\sigma\}$ (2) $\{\varepsilon\}$ (3) $\{\ \} = \emptyset$ are regular languages; if R_1, R_2 are regular, then so are

$R_1 \cup R_2$, $R_1 \circ R_2$, and R_1^*

NFA's help us

In more detail $\{\sigma\} \longleftrightarrow$



$\Sigma \setminus \{\sigma\}$

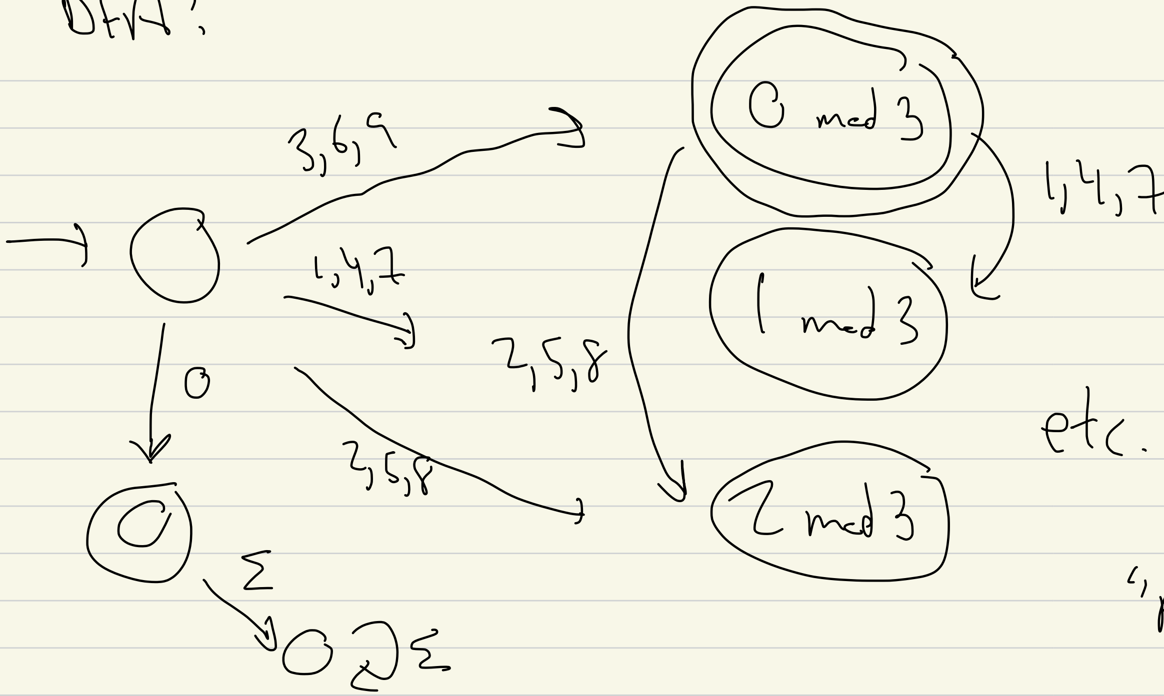
= things in Σ not in $\{\sigma\}$

We'll just assume this w/o proof:

Thm: Any regular language is described by a regular expression.

Example: $L = \{0, 3, 6, 9, 12, \dots\} \subset \underbrace{\{0, 1, \dots, 9\}}_{\Sigma}^*$

DFA:



"Algorithm"

In Section §1.3, there is an algorithm

DFA $\to$ regular expression

The regular expression for $\{0, 3, 6, 9, 12, 15, \ldots\}$

is quite long...

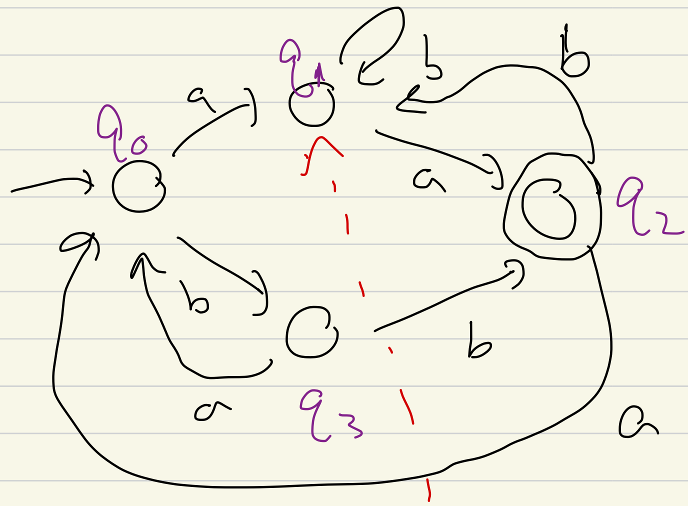
§ 1.4 Non regular languages.

Replace "pumping lemma" with Myhill-Nerode theorem.

Idea:

DFA

e.g.



L

Fundamental Observation:

a	get	to	q_1
ab	"	"	q_1
aab	"	"	q_1

$$\{ w \in \Sigma^* \mid aw \in L \} = \{ w \in \Sigma^* \mid abw \in L \}$$

i.e.

$$aw \in L \iff abw \in L$$

Given Σ alphabet, $L \subset \Sigma^*$ write for $u \in \Sigma^*$

$$\text{AcceptingFutures}_L(u) = \{ w \mid uw \in L \}$$

$=$

e.g., $L_{2\,a\text{'s}} = \{ w \in \{a,b\}^* \mid w \text{ has exactly } 2 \text{ } a\text{'s} \}$

$$\text{AccFut}_L(aaa) = \{ w \mid aaaw \in L \}$$

$$= \emptyset$$

$$\text{AccFut}_L(abab) = \{ w \mid ababw \in L \}$$

$$= \{ w \mid w \text{ has exactly } 0 \text{ } a\text{'s} \} = b^*$$

$$= \{ \varepsilon, b, b^2, b^3, \ldots \}$$

$$\text{AccFut}_L(a) = \{ w \mid w \text{ ~~has~~ has exactly } 1 \text{ } a \}$$

$$\mathrm{AccFut}_L(\varepsilon) = \mathrm{AccFut}_L(b)$$

$$= \mathrm{AccFut}_L(b^2) = \{ w \text{ s.t. } w \text{ has exactly } 2 \text{ a's} \}$$

So: $\mathrm{AccFut}_L(\varepsilon) = L_{2 \text{ a's}}$ 1

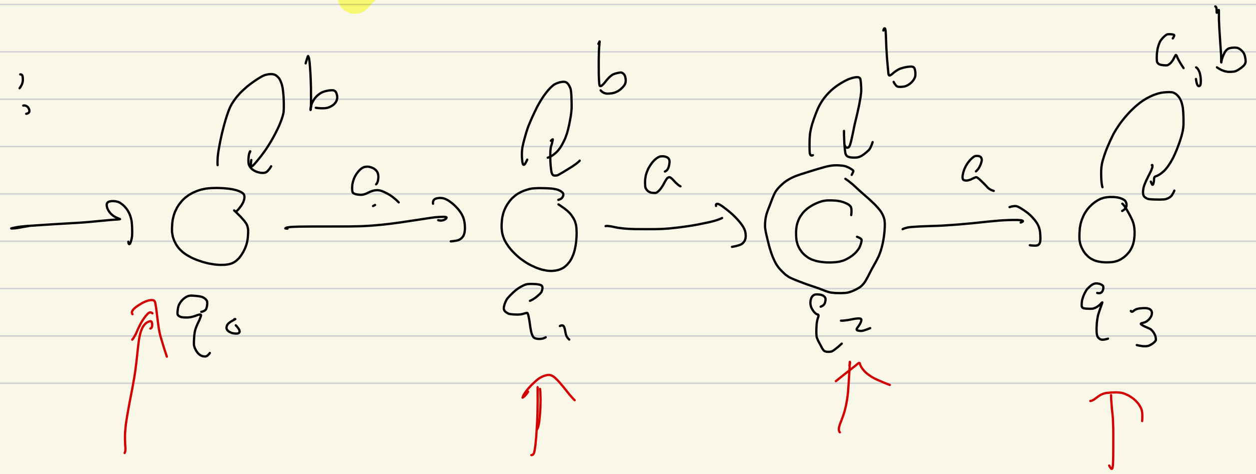
$\mathrm{AccFut}_L(a) = L_{\text{one } a}$ 2

$\mathrm{AccFut}_L(aa) = L_{0 \text{ a's}} = b^*$ 3

$\mathrm{AccFut}_L(aaa) = \emptyset$ 4

$\Rightarrow$ ε, a, aa, aaa have to land in different states of any DFA for $L = L_{2a's}$

DFA:



$\varepsilon, b,$ $b^2, \ldots$ | a, ab $bab, \ldots$ | $aa,$ $aab,$ $baabb, \ldots$ | $aaa,$ $aba^{10},$

size of

$$\{ AccFut_L(u) \mid u \in \Sigma^* \}$$

$=$ minimum number of states in DFA accepting L.

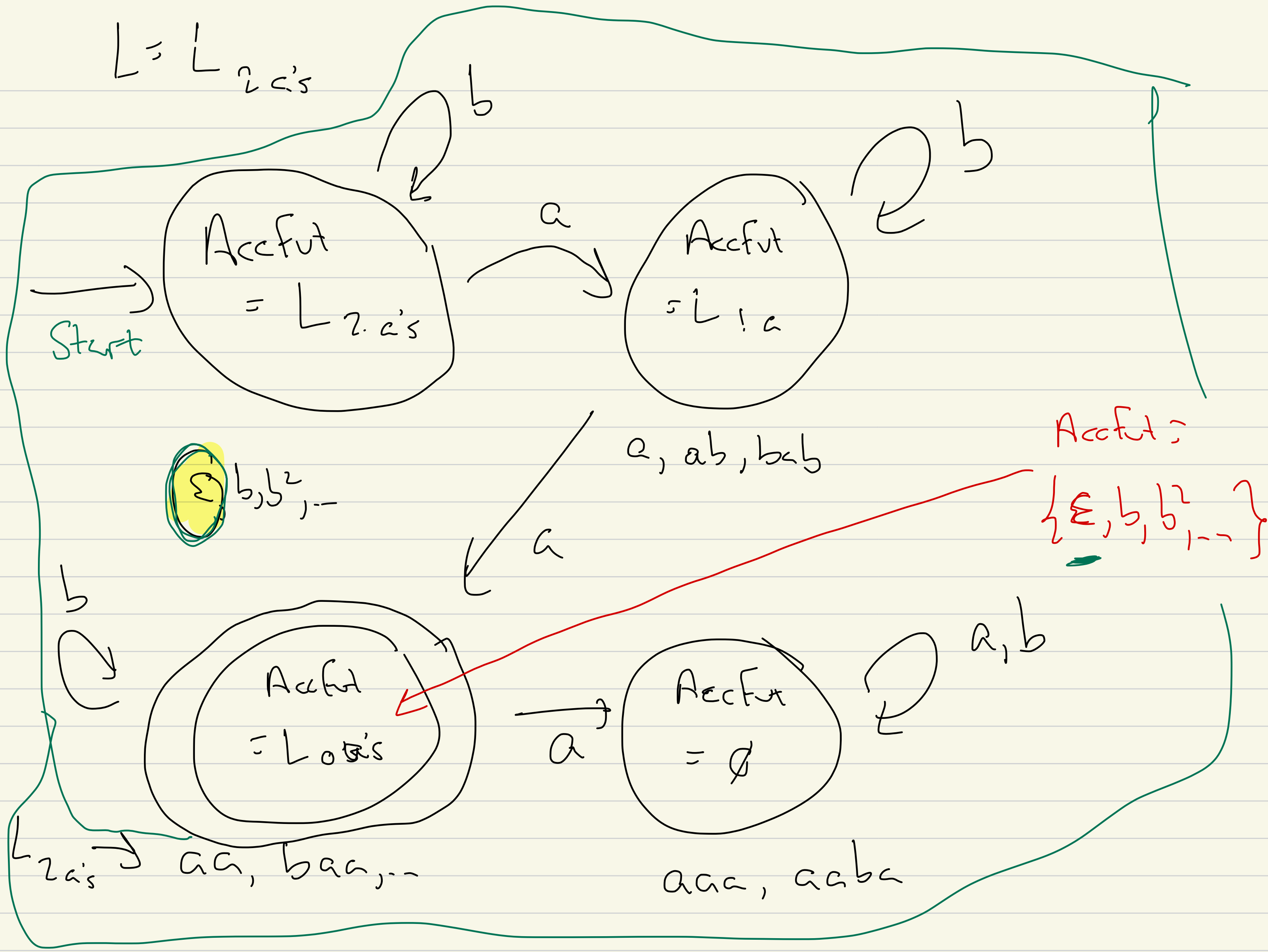
Break! My watch 10:23 ⟶ 10:28

Myhill-Nerode Thm: For an alphabet Σ, $L \subseteq \Sigma^*$

size of $\{ AccFut_L(u) \mid u \in \Sigma^* \}$:

- if finite, $=$ min # states for a DFA for L
- if infinite, L is not regular

$L = L_{2\,a\text{'s}}$



Proof: (1) $\# \{ \mathrm{AccFut}_L(u) \mid u \in \Sigma^* \} \le \#$ states for DFA

(2) $\# \{ \mathrm{AccFut}_L(u) \mid u \in \Sigma^* \} \ge \#$ states for DFA

idea: states $\longleftrightarrow$

Initial state $\Longleftrightarrow \mathrm{AccFut}_L(\varepsilon)$

Final state $\Longleftrightarrow \mathrm{AccFut}_L(w)$ that contain ε

$$L = \left\{ w \;\middle|\; \begin{array}{l} w \text{ contains } \geq 1\ a \\ \text{and } '' \quad \geq 2\ b\text{'s} \end{array} \right\}$$

$$\text{AccFut}_L(\varepsilon) = L = \{ w \mid \varepsilon w \in L \}$$

$$= \{ w \mid w \in L \}$$

start state

$$\text{AccFut}_L(b) = \{ w \mid \text{contains } \geq 1\ a \ \&\ \geq 1\ b \}$$

$$\text{AccFut}_L(a) = \{ \quad '' \quad \ldots \quad (\geq 0\ a\text{'s}) \ \&\ \geq 2\ b\text{'s} \}$$

$$\text{AccFut}_L(ab) = \{ \quad '' \quad \ldots \quad '' \quad \&\ \geq 1\ b \}$$

$$\text{AccFut}_L(bb) = \{ \quad '' \quad \ldots \quad \geq 1\ a\text{'s} \ \&\ (\geq 0\ b\text{'s}) \}$$

$$\text{AccFut}_L(abb) = \{ w \mid \quad \geq 0\ a\text{'s} \ \&\ \geq 0\ b\text{'s} \}$$

$$= \Sigma^*$$

$$\text{AccFut}_L(abbabbbbaa) = \Sigma^* = \text{AccFut}_L(abb)$$

Thm: $L = \{0^n 1^n \mid n \in \mathbb{N}\}$

$= \{01, 0011, 000111,$

$0^4 1^4, \ldots\}$ is nonregular

Rough reason: if you see $0 \ldots 0$, have to know how many

Next time:

$\mathrm{AccFut}_L(\varepsilon) = \{01, 0011, 000111, \ldots\}$

$\mathrm{AccFut}_L(0) = \{1, 011, 00111, \ldots\}$

$\mathrm{AccFut}_L(00) = \{11, 01^3, 0^2 1^4, \ldots\}$

$\vdots$

Update to Zoom 5.3.something

HW 3, Group, Prob 5

Consider

$$\left\{ \text{DFA's } (Q, \Sigma, \delta, q_0, F) \;\middle|\; \begin{array}{l} \text{for some } m \in \mathbb{N}, \\ Q = \{1, 2, \ldots, m\} = [m] \end{array} \right\}$$

(a)

$$Q = \{1\}, \ \{1,2\}, \ \{1,2,3\}, \ldots$$

a notion of "algorithm"