Sept 29 (i)

§1.2 NFA - non-determinism

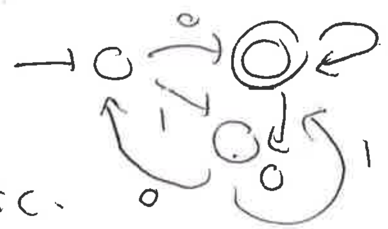
§1.3 Regular languages

[§1.4 Which languages aren't regular]

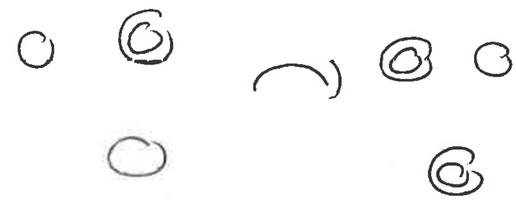
Examples: Given L, L' regular, over Σ

[1.1: $L^{comp} = \Sigma^* \setminus L = \{ s \in \Sigma^* \mid s \notin L \}$ is regular

Machine



etc.



swap accepting with non-accepting

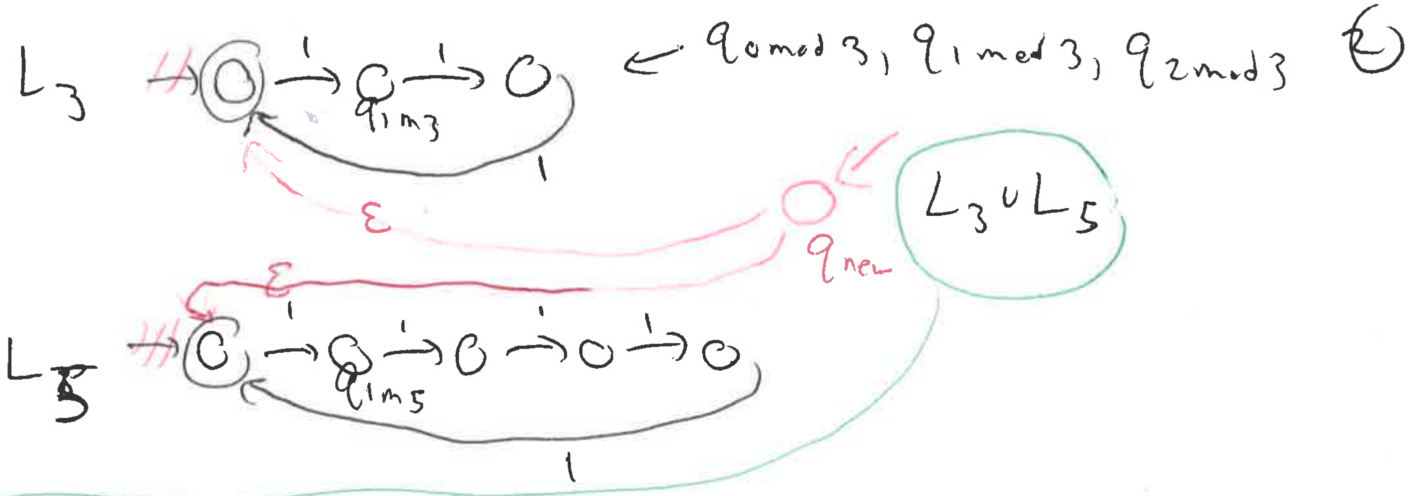
- $L \circ L'$ is regular
- $L \cup L'$ regular

To prove this (statements about <u>DFAs</u>...) we introduce <u>NFA's</u>

$L_3 = \{ 1^m \mid m \text{ divis by } 3 \}$

$L_5 = \quad \cdot \quad - \quad - \quad - \quad 5 ..$

$L_3 \circ L_5$ or $L_3 \cup L_5$ is trick without non-determinism



We say s is accepted by an NFA if there is some path through the NFA reads s and winds up accepting.

An NFA recognizes the language of string that it accepts.

So what?

So... want to take an NFA and construct an equiv DFA.

NFA = $q_{new}, q_{0m3}, q_{1m3}, q_{2m3}$
$q_{0m5}, q_{1m5}, q_{2m5}, q_{3m5}, q_{4m5}$

DFA: $\rightarrow \bigcirc$

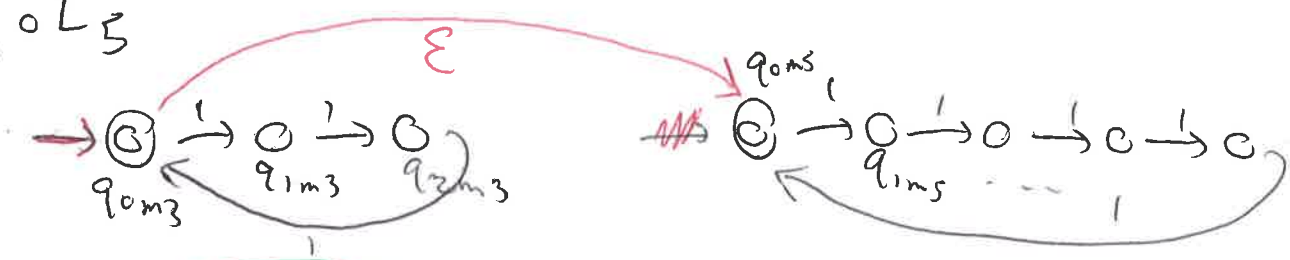
where could be when read first 1! ?

either is $\{q_{1m3}, q_{1m5}\}$

read next 1! $\{q_{2m3}, q_{2m5}\}$

So DFA: Set of States = Power Set (States of the NFA)
Set of all Subsets

$L_3 \circ L_5$ ③



start $\{q_{0m3}, q_{0m5}\}$ before you read your first 1, symbol

after — — — but before 2nd symbol ...

1

$\{q_{1m3}, q_{1m5}\}$ $\xrightarrow{1}$ $\{q_{2m3}, q_{2m5}\}$

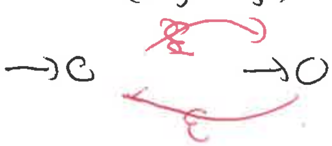
1

$\{q_{0m3}, q_{3m5}, q_{0m5}\}$

$\{q_{1m3}, q_{4m5}, q_{1m5}\}$

$\{q_{2m3}, q_{0m5}, q_{2m5}\}$ → ...

But $(L_3 \cup L_5)^*$



$NFA = (S, \Sigma, \delta, q_0, F)$ $\delta : Q \times \Sigma_\varepsilon \to Power(Q)$

set

$DFA = (Power(S), \Sigma, \tilde{\delta}, \hat{q_0}, \tilde{F})$: $\tilde{F} = \{T \subset S \mid s.t.\ T \cap F \neq \emptyset\}$

8 states NFA ⇝ DFA could have 2^8 states ...

Implement: just write down subset

q_{0m3}	q_{1m3}	q_{2m3}	q_{0m5}	— —
True	False	True	True	

← Could I be here?

So implementing non-determinism (finite automata) not so bad...

DFAs (§1.1) $\leftrightarrow$ NFAs (§1.2) $\longleftrightarrow$ Regular Expressions (1.3)

Regular Expression:

[Sip] : - Built on $\emptyset$, ε, $\sigma \in \Sigma$ regular expressions

- If R_1, R_2 are regular, so is $(R_1 \cup R_2)$

$(R_1 \circ R_2)$

(R_1^*)

(usually there are other ways to make regular expressions.

$\Sigma = \{0,1\}$: $((\underbrace{1 \circ 1 \circ 1}) \cup (1 \circ 1 \circ 1 \circ 1 \circ 1))^*$

$(1^3 \cup 1^5)^*$

$R_1 \cup R_2$ refers to any string in union of strings referred to by R_1 or R_2

$R_1 \circ R_2$ - - - - - - - - concatenation - - -

R_1^* refers to any string s that can be written as $x_1 \ldots x_k$ for some k, some $x_1, \ldots, x_k$ all in R_1

Outside definition is [Sip], convenient $\neg$ negation

shortcuts of symbols

$* \leftrightarrow \Sigma^*$

=

$\{\{011, 11\}^* \circ \{000\}^* \circ 111\}^*$

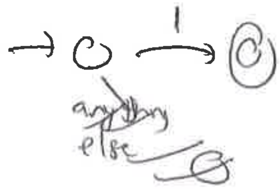
=

1.3: each regular ~~expression~~ expression describes regular language, and conversely

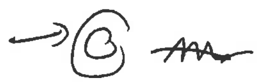
$\longrightarrow$ easy

If we regular expression:

$\emptyset \longleftrightarrow$ 

$1 \longleftrightarrow$ 

NFA

$\varepsilon \longrightarrow$ 
NFA