Lecture 6

Thursday, 11 February, 2021 10:44

Advanced Closure Properties of Reg Langs

- in the past: intersection (DFA), union (regex, DFA, NFA), complement (DFA), Kleene * (regex, NFA)
- can combine: $(L_1 \setminus L_2) \cap (\overline{L_3} \cup L_4)^*$, HW1 P3
- Today: more complicated operations on languages via transforming regexes & automata

- flip : $\{0,1\}^* \to \{0,1\}^*$

  flip(w) inverts every bit of w

  flip(1010100) = 0101011

Given language L $\quad \text{FLIP}(L) = \{ \text{flip}(w) \mid w \in L \}$

$\text{FLIP}(\{01, 001\}) = \{10, 110\}$

Q: if L is regular, is FLIP(L) regular?

1. if L is regular $\exists$ regex R for L.

   try to "flip" the regex.

R	FLIP(R)
$\emptyset$	$\emptyset$
w	$\text{flip}(w)$
$R_1 \cdot R_2$	$\text{FLIP}(R_1) \cdot \text{FLIP}(R_2)$
$R_1 + R_2$	$\text{FLIP}(R_1) + \text{FLIP}(R_2)$
R_1^*	$\text{FLIP}(R_1)^*$

Proof of correctness?

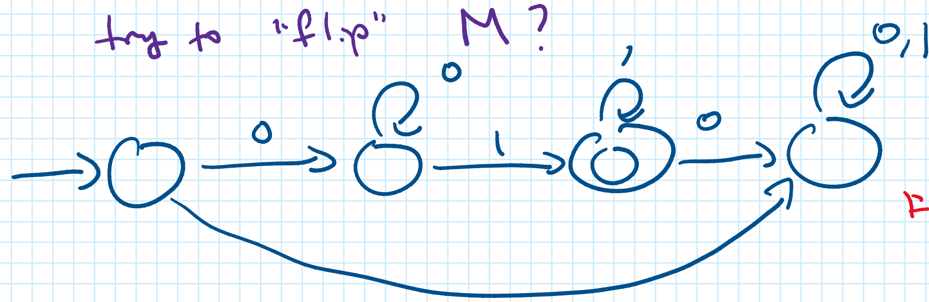
fun w/ (structural) induction

2. if L is regular $\exists$ DFA M for L

   try to "flip" M ? $\quad$ 0,1

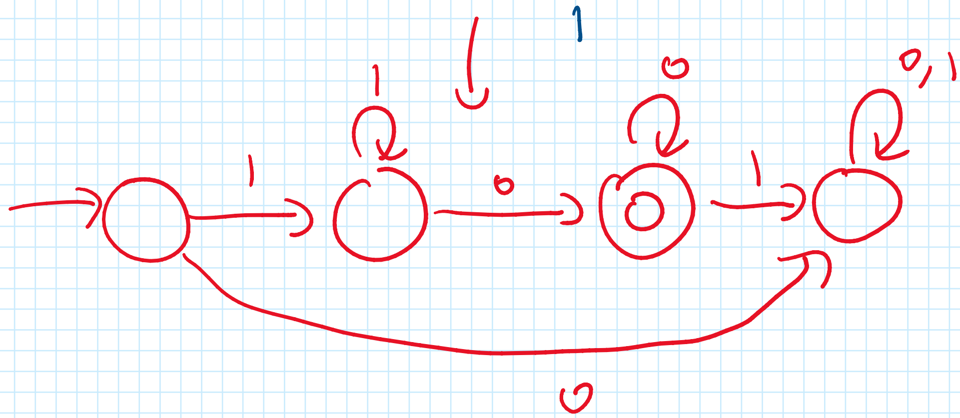
2. If L is regular $\exists$ DFA M for L

try to "flip" M?



$L = L(00^*11^*)$

$\text{FLIP}(L) = L(11^*00^*)$



In general

$M = (Q, \Sigma, \delta, s, A) \longrightarrow M' = (Q, \Sigma, \delta', s, A)$

$\delta'(q, a) = \delta(q, 1-a)$

— w^R is reverse of string w

$\text{REVERSE}(L) = \{ w^R \mid w \in L \}$

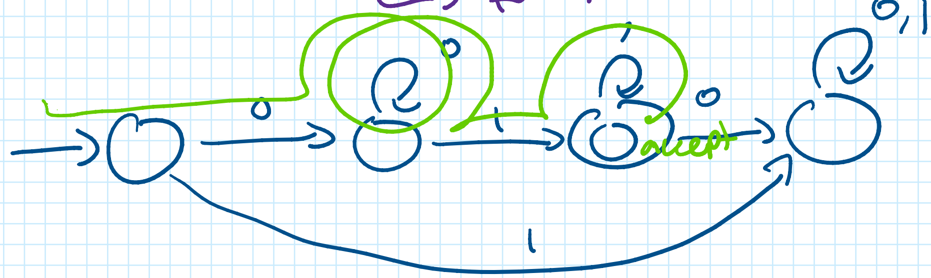
Q: If L is reg, is $\text{REVERSE}(L)$ also reg?

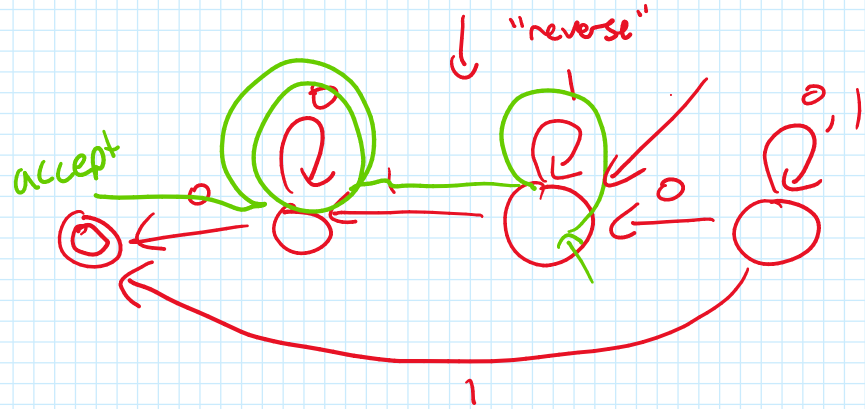
1. converting regex R for L

R	$\text{REVERSE}(R)$
ϕ	ϕ
w	w^R
$R_1 + R_2$	$\text{REVERSE}(R_1) + \text{REVERSE}(R_2)$
$R_1 \bullet R_2$	$\text{REVERSE}(R_2) \bullet \text{REVERSE}(R_1)$

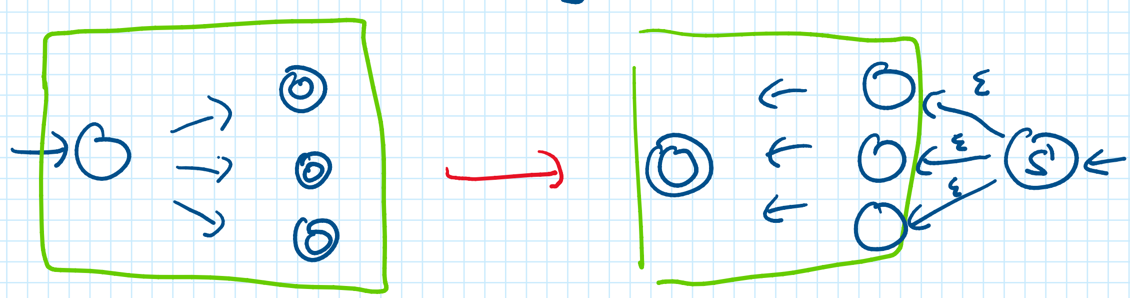
$R_1 + R_2$	$\text{REVERSE}(R_1) + \text{REVERSE}(R_2)$
$R_1 \circ R_2$	$\text{REVERSE}(R_2) \circ \text{REVERSE}(R_1)$
R_1^*	$\text{REVERSE}(R_1)^*$

2. convert a DFA M for L
   ↳ to NFA N





what if multple accepting states?



new starting state w/ ε-transitions to old acc states

Intuitively, given w, simulate M by
"guessing" where M ends on w
following transitions of M "backwards"
accept if were able to get to
where M starts on w

accept if were able to get to
where M starts on w
(starting state)

In general

$$M = (Q, \Sigma, \delta, s, A) \longrightarrow N = (Q', \Sigma, \delta', s', A')$$

$Q' = Q \cup \{s'\}$ new start state

$s' = s'$

$A' = \{s\}$

for $q \in Q$ $\left\{ \begin{array}{l} \delta'(q,a) = \{ r \in Q \mid \delta(r,a) = q \} \\ \delta'(q, \varepsilon) = \emptyset \end{array} \right.$

$\delta'(s', a) = \emptyset$

$\delta'(s', \varepsilon) = A$

$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to P(Q)$

Shortcut: "unspecified transitions are $\emptyset$"

– $\text{prefix}(w) = \{ x \mid \exists y,\ w = x \cdot y \}$

$\text{prefix}(10111)$

$\{\varepsilon, 1, 10, 101, 1011, 10111\}$

$$\text{PREFIX}(L) = \bigcup_{w \in L} \text{prefix}(w)$$

Q: if L is reg. is $\text{PREFIX}(L)$ reg?

Given DFA $M = (Q, \Sigma, \delta, s, A)$ for L

$\hookrightarrow$ NFA N for $\text{PREFIX}(L)$

$x \in \text{PREFIX}(L)$

iff $\exists\, y$ so that

$\delta^*(s, x \cdot y) \in A$

$\Leftrightarrow \delta^*(\delta^*(s,x), y) \in A$

iff there is a path from

$\delta^*(s,x)$ to some state in A.

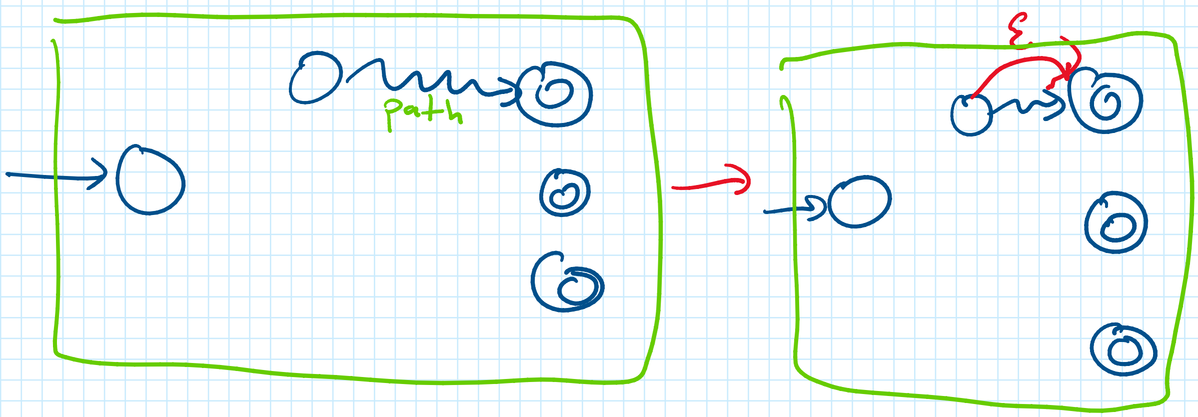
Idea: ε-transition from q to r

if there is a path $q \rightsquigarrow r$

and $r \in A$

effectively "guessing" y then following

$\delta^*(q, y)$ to r.



$M = (Q, \Sigma, \delta, s, A) \longrightarrow N(Q, \Sigma, \delta', s, A)$

$\delta'(q, a) = \{\delta(q, a)\}$

$\delta'(q, \varepsilon) = \{ r \in A \mid$ there is a path from q to r $\Leftrightarrow \exists y \ \delta^*(q,y) = r \}$

— $\text{CYCLE}(L) = \{ y \cdot x \mid x \cdot y \in L \}$

$\text{CYCLE}(\{01, 0011\})$
$= \{01, 10, 0011, 0110, 1100, 1001\}$

$01 = \varepsilon \cdot 01 = 0 \cdot 1 = 01 \cdot \varepsilon$

$0011 = \varepsilon \cdot 0011 = 0 \cdot 011 = 00 \cdot 11 = 001 \cdot 1 = 0011 \cdot \varepsilon$

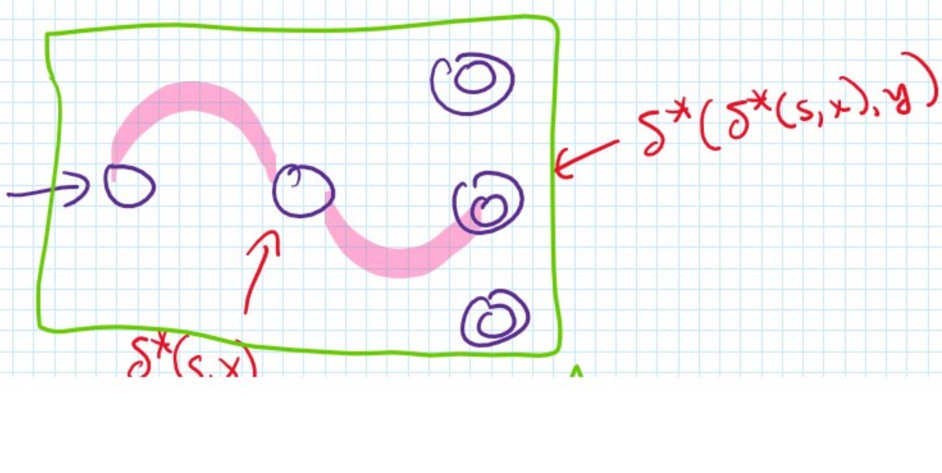
Given DFA $M = (Q, \Sigma, \delta, s, A)$

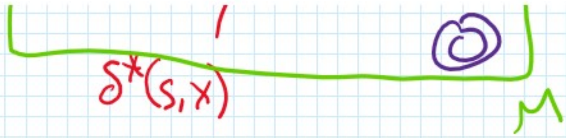
$\hookrightarrow$ NFA $N = (Q', \Sigma, \delta', s', A')$

$w \in \text{CYCLE}(L)$
$\Leftrightarrow \exists$ decomposition of w into $w = y \cdot x$ s.t. $x \cdot y \in L$.

guess the decomposition
aka magic fairy
aka nondeterminism.

how to know if $y \cdot x$ right decomp?





!! we don't read $x \cdot y$. We read $y \cdot x$.

<u>Guess</u> q to be $\delta^*(s,x)$

Read $y \to \delta^*(q,y)$. (if $\delta^*(q,y) \in A$ "guess" that we're done w/ the y part)

<u>Verify</u> $q = \delta^*(s,x)$

by reading $x \to \delta^*(s,x)$

& checking.

Keep track of guess q.

current state in simulation r.

if we're in y part or x part

new start state.

$$Q' = (\underbrace{Q}_{\text{guess}} \times \underbrace{Q}_{\text{current state in simulation}} \times \{\underbrace{\text{before}}_{\text{"y" part}}, \underbrace{\text{after}}_{\text{"x" part}}\}) \cup \{s'\}$$

$$s' = s'$$

$$\delta'(s', \varepsilon) = \{(q, q, \text{before}) \mid q \in Q\}$$

Captures "guess & go to q"

$$\delta'((q, r, \text{before}), a) = \{(q, \delta(r,a), \text{before})\}$$

for $q \in A$ $\delta'((q, r, \text{before}), \varepsilon) = \{(q, s, \text{after})\}$

for $r \in A$ $\delta'((q, r, \text{before}), \varepsilon) = \{(q, s, \text{after})\}$

$$\delta'((q, r, \text{after}), a) = \{(q, \delta(r, a), \text{after})\}$$

everything not specified is φ

$$A' = \{ \underbrace{(q, q, \text{after})} \mid q \in Q \}$$

verifies $\delta^*(s, x) = q$ for guess of x, q.

easier way? maybe? IDK.