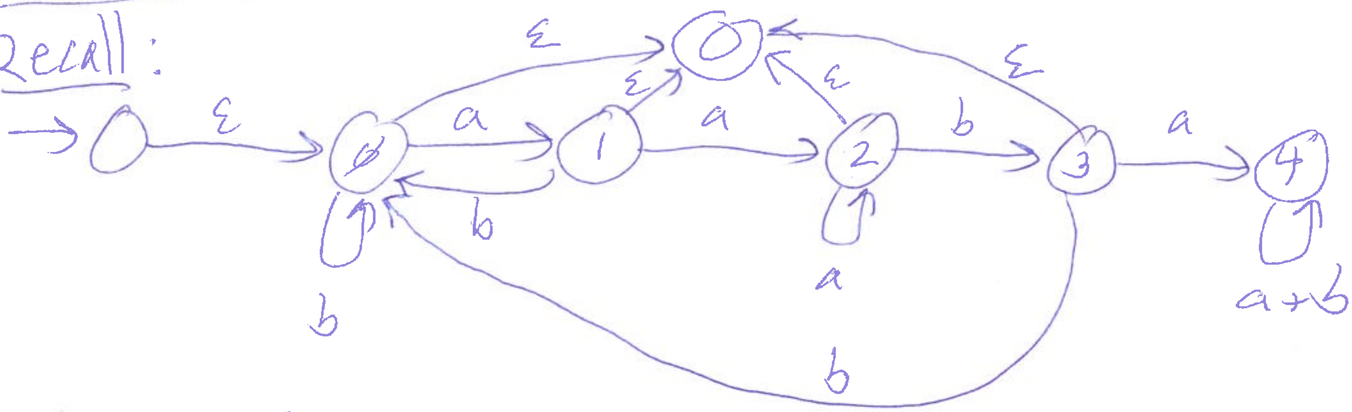
CSCE 355
2/12/2024

ε-NFA $\to$ regex example (cont.)

Closure properties of REG_Σ

1

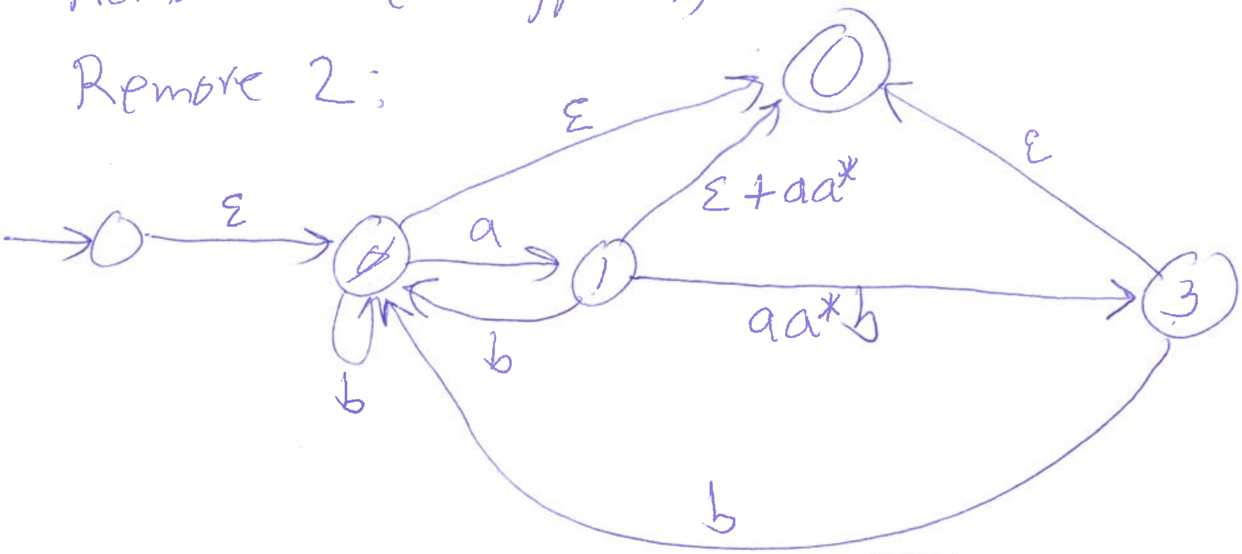
Recall:



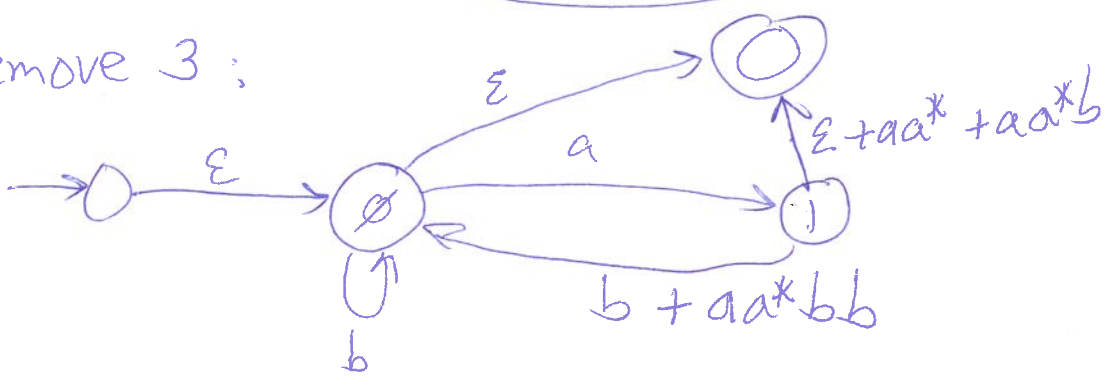
$L = \{ w \mid w \text{ does not have } aaba \text{ as a substring} \}$

Remove 4 (no bypasses)

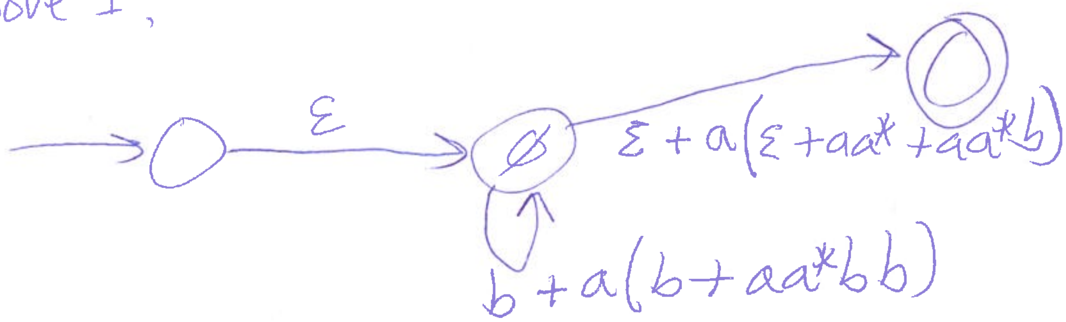
Remove 2:



Remove 3:



Remove 1:



$$\longrightarrow \bigcirc \xrightarrow{(b+a(b+aa^*bb))^*(\varepsilon+a(\varepsilon+aa^*+aa^*b))} ⊚$$

Final

Closure properties

Reg lang's are closed under complement & intersection (hence union, relative complement, symmetric difference).

Def: Let w be any string, say,

$w = w_1 w_2 \cdots w_n$ $(w_i \in \Sigma)$ and $n \geq 0$.

The reversal w^R of w is the string

$$w_n w_{n-1} \cdots w_1.$$

w is a palindrome if $w = w^R$ (~~is~~ eg. aba, ..)

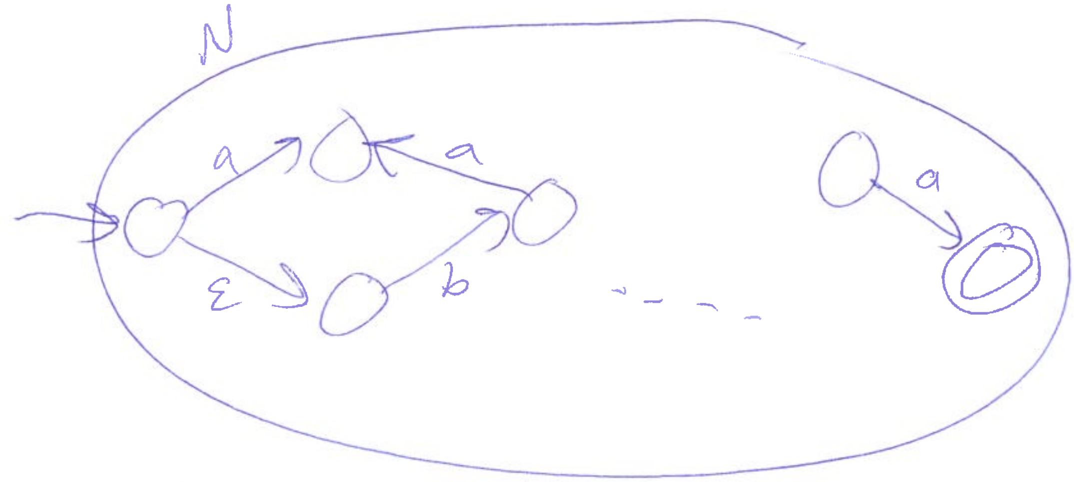
If $L \subseteq \Sigma^*$ is a language, define

$$L^R := \{w^R : w \in L\}$$

Prop: If L is regular then L^R is regular.

Proof #1: Let L be regular and let

N be an ε-NFA that recognizes L. WLOG
N is clean, so N has a unique accepting state.



Construct N' ε-NFA recognizing L^R, by from N

1) swapping the start state with the accepting state
2) Reversing the directions of all the arrows

N accepts w $\iff$ $\exists$ path $s_0, \ldots, s_n$ from start to accepting state in N reading w

$\iff$ $\exists$ path $s_n, \ldots, s_0$ from start to accepting state in N' reading w^R

$\iff$ N' accepts w^R

$\therefore$ Since w was arbitrary,

$$L(N') = \cancel{L(N)} \; L(N)^R = L^R$$

$\therefore$ L^R is regular. $\blacksquare$ Proof #1

Proof #2: Let r be a regex such that $L(r) = L$.
We give rules to convert any r into a regex r^R such that $L(r^R) = L(r)^R = L^R$

Idea: How does the reversal operator interact with the regex-building operators? union, concat, $*$-operator

$$\rightarrow (L_1 \cup L_2)^R = L_1^R \cup L_2^R$$

$$\rightarrow (L_1 L_2)^R = L_2^R L_1^R$$

$$\rightarrow (L^*)^R = (L^R)^*$$

Rules for building r^R:

	r	r^R
Atomic	$\emptyset$	$\emptyset$
$(a \in \Sigma) \rightarrow$	a	a
s, t regexes over Σ	$s + t$	$s^R + t^R$
	st	$t^R s^R \leftarrow$
	s^*	$(s^R)^*$

Proof of correctness is by induction on the syntax (or length) of r (omitted). //

Ex: $((ab+bc^*)^*a)^R = \underline{a^R}((ab+bc^*)^*)^R$

$= a((ab+bc^*)^R)^* = a((ab)^R + (bc^*)^R)^*$

$= a(b^Ra^R + (c^*)^Rb^R)^* = a(ba + (c^*)^Rb)^*$

$= \underline{a(ba + (c^R)^*b)^*} = \underline{a(ba + c^*b)^*}$

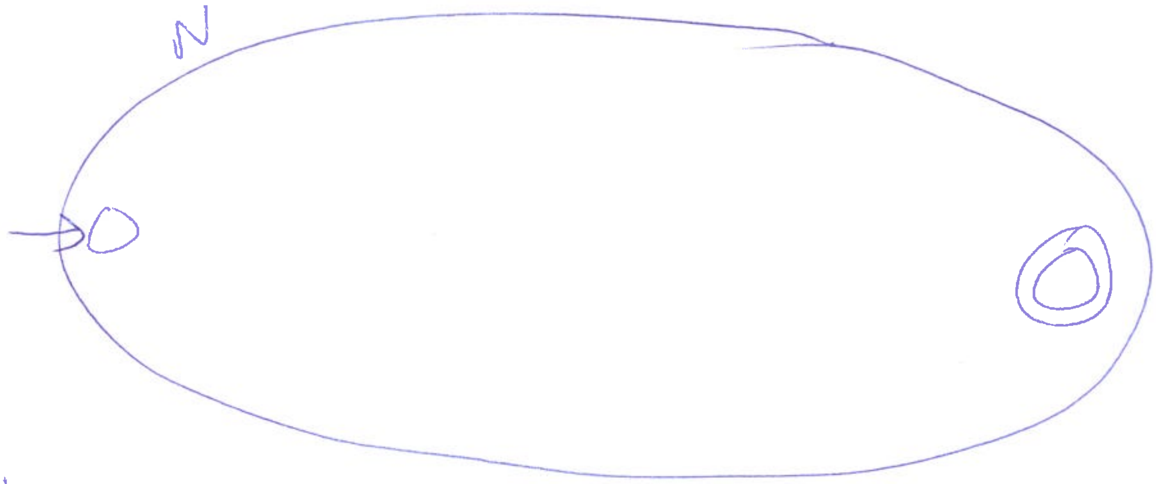
Ex: Let L be a language. Let

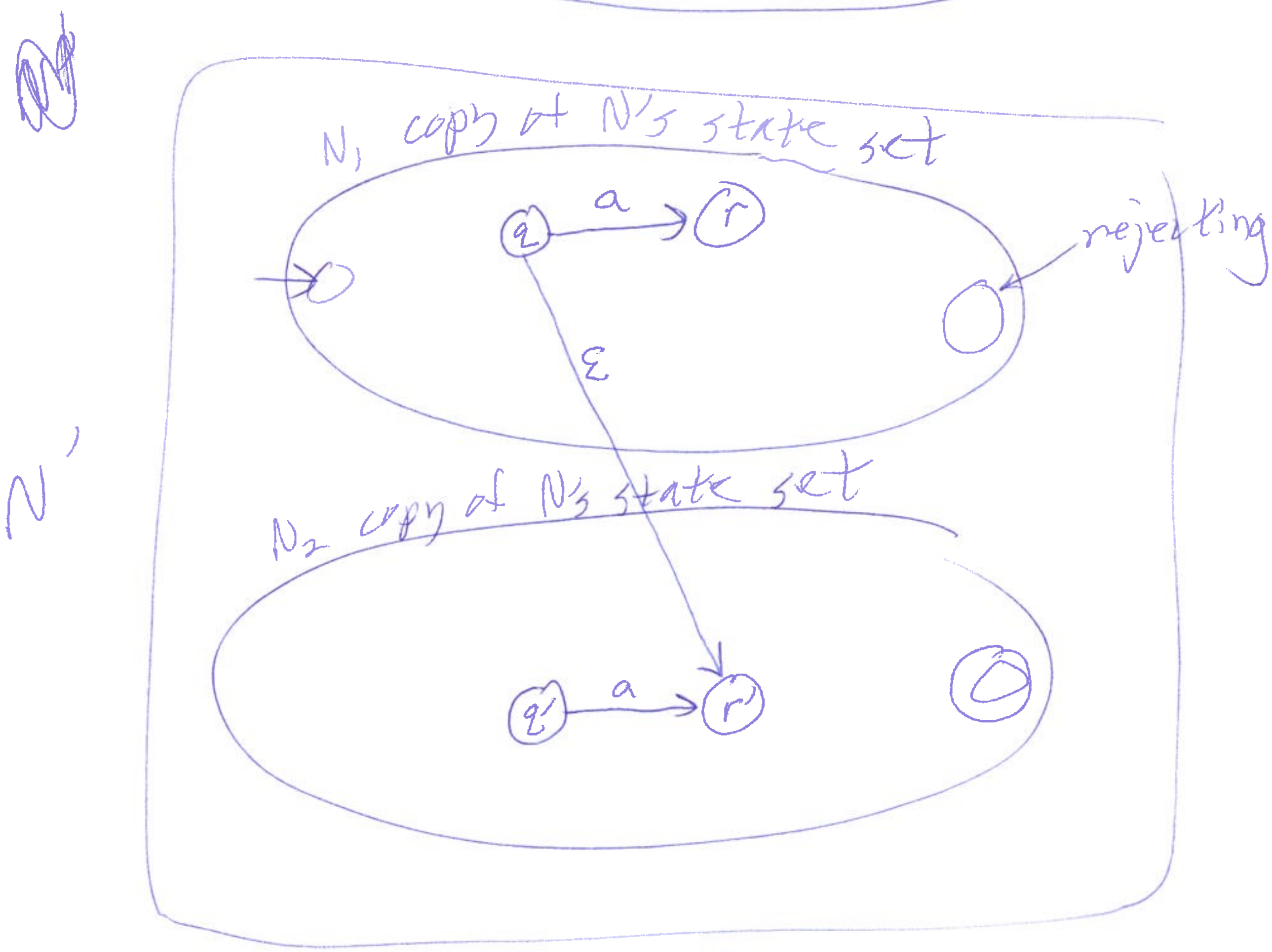
$\text{DROP-ONE}(L) := \{w : w$ results from a string in L by removing a single symbol from anywhere in the string$\}$

$= \{xy : xay \in L$ for some $a \in \Sigma$ (& $x, y \in \Sigma^*$)$\}$

Prop: If L is regular, then DROP-ONE(L) is regular.

Proof #1: Given an ε-NFA ~~recogni~~ N recognizing L, construct an ε-NFA N' recognizing DROP-ONE(L) as follows:





For every non-ε transition $q \xrightarrow{a} r$

add an ε-move from $q \xrightarrow{\varepsilon} r'$

q ↑ top copy

r' ↑ bottom copy

(Proof of correctness omitted) //

Proof #2: Rules to convert a regex r for L into a regex r' for DROP-ONE(L) by induction on the syntax of r:

	r	r'	
	$\emptyset$	$\emptyset$	
($a \in \Sigma$)	a	ε	($\varepsilon := \emptyset^*$)
s, t regexes over Σ	$s+t$	$s' + t'$	
	st	$s't + st'$	
	s^*	$s^* s' s^*$	

(Proof of correctness by induction on syntax (omitted))

//