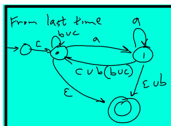
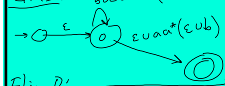
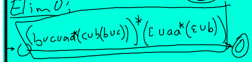
From last time



<u>Elim 1</u>: $buc \cup aa^*(c \cup b(buc))$

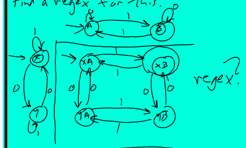


<u>Elim 0</u>:



<u>Ex</u>: $\{w \in \{0,1\}^* : w$ has even # 0s & odd # of 1's$\}$

Find a regex for this.



<u>Closure properties of the regular languages.</u>

"If A, B, ... are regular, then so is (some op applied to A, B, ...)

<u>Ex</u>: string reversal.

Let $w \in \Sigma^*$ be a string.

Define $w^R = w_n w_{n-1} \cdots w_1$ where $w_1, \ldots, w_n \in \Sigma$ and $w = w_1 \cdots w_n$

For $L \subseteq \Sigma^*$, define L^R in the usual way;

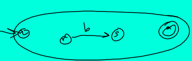
$$L^R := \{w^R : w \in L\}$$

<u>Prop</u>: If $L \subseteq \Sigma^*$ is regular, then L^R is regular.

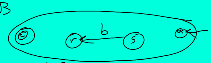
<u>Proof</u>: (2 ways)

<u>Method 1</u>: Convert an automaton A (recognizing some L) into an automaton B recognizing L^R.

$A \longmapsto$ clean NFA A'



B



Note: $(L^R)^R = L$ //

<u>Method 2</u>: Convert a regex r for L into a regex r^R for L^R

How does reversal interact with union, concat, *?

L_1, L_2 langs (arbitrary)

$\cancel{(L_1 \cup L_2)^R = L_1^R \cup L_2^R}$

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

$(L_1^*)^R = (L_1^R)^*$

Def of r^R by recursion on the syntax of r:

	r	$r' = r^R$
	$\emptyset$	$\emptyset$
$(a \in \Sigma)$	a	a
s, t regexs	$s \cup t$	$s' \cup t'$
	st	$t's' \leftarrow$ ☆
	s^*	$(s')^*$

<u>Ex</u>:

$r = (ab \cup cab)^*(cca \cup a^*)$

$r' = (acc \cup a^*)(ba \cup bac)^*$

Ex: For language $L \subseteq \Sigma^*$, define $pp(L)$

(pp = "principal prefix")

as

$pp(L) = \{w \in \Sigma^* : (\exists a \in \Sigma)\, wa \in L\}$

$pp(L)$ = the result of taking each nonempty string in L & stripping off the last char.

Prop: If L is regular, then $pp(L)$ is regular.

Proof: Method 1: Let $A = \langle Q, \Sigma, \delta, q_0, F \rangle$ be a DFA s.t. $L(A) = L$.

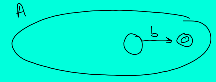
Define

$B := \langle Q, \Sigma, \delta, q_0, F' \rangle$

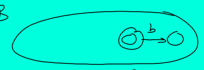
where

$F' := \{q \in Q : (\exists a \in \Sigma)\, \delta(q,a) \in F\}$

A



B



$L(B) = pp(L)$ [must argue both directions] //

Method 2: regex conversion:

r	$r' = pp(r)$
$\emptyset$	$\emptyset$
$a \in \Sigma$	ε
$s \cup t$	$s' \cup t'$
st	$\begin{cases} st' \cup s' & \text{if } \varepsilon \in t \\ st' & \text{otherwise} \end{cases}$
s^*	$s^* s'$

Extreme case: $s = \emptyset$ ($s^* = \varepsilon$)
get $\emptyset^* \emptyset = \emptyset$ ok. //

Ex: $(ab \cup bc)^* = r$

$pp(r) = r' = (ab \cup bc)^* (ab \cup bc)'$
$= (ab \cup bc)^* ((ab)' \cup (bc)')$
$= (\;\;"\;\;)^* (ab' \cup bc')$
$= (\;\;"\;\;)^* (a \cup b)$ //

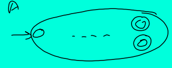
Ex:

DROP-ONE(L)
$= \{wx : w, x \in \Sigma^* \text{ and } (\exists a \in \Sigma)[wax \in L]\}$

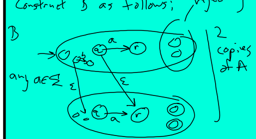
Prop: L regular $\Longrightarrow$ DROP-ONE(L) regular.

Proof: Let A be an NFA recognizing L.

A



Construct B as follows;



Do this for every non-ε-transition in A. //

Method 2: convert regex $r = L$ to $r' =$ DROP-ONE(L):

r	r'
$\emptyset$	$\emptyset$
$a \in \Sigma$	ε
$s \cup t$	$s' \cup t'$
st	$s't \cup st'$
s^*	$s^* s' s^*$