

Check Compass
grades

Check addition etc

Problems \implies us

Regular Expressions (Regex)

Base $\left\{ \begin{array}{l} a \\ \epsilon \\ \emptyset \end{array} \right.$ $a \in \Sigma$

Recursive $\left\{ \begin{array}{l} R_1 \cup R_2 \\ R_1 R_2 \\ R_1^* \end{array} \right.$ $R_1 + R_2$ if R_1, R_2 are regex
 $R_1 \circ R_2$
 $()$

$$ab + ba$$

$$((ab) + (ba))$$

- 1) lexer strings → tokens
- 2) parser token → structure

floating number constant
variable name

$$[0-9]^* \cdot [0-9]^*$$

$$\begin{aligned} & \overline{[0-9]^* \cdot [0-9]^*} \\ (\epsilon V + V-) & \left[\left([0-9][0-9]^* \cdot [0-9]^* \right) \right. \\ & \left. \cup \left(\underbrace{[0-9]^* \cdot [0-9][0-9]^*}_{\text{redundant}} \right) \right] \end{aligned}$$

Variable names

$[a-z] ([a-z][0-9])^*$

Regex \equiv NFAs / DFAs

1) Regex \Rightarrow NFA

2) DFA \Rightarrow Regex

Regex \rightarrow NFA

"Structural induction"

<u>Base cases</u>	<u>L_g</u>	NFA
$a \in \Sigma$	$\{a\}$	$\rightarrow \bigcirc \xrightarrow{a} \odot$
ϵ	$\{\epsilon\}$	$\rightarrow \odot$
\emptyset	\emptyset	$\rightarrow \bigcirc$

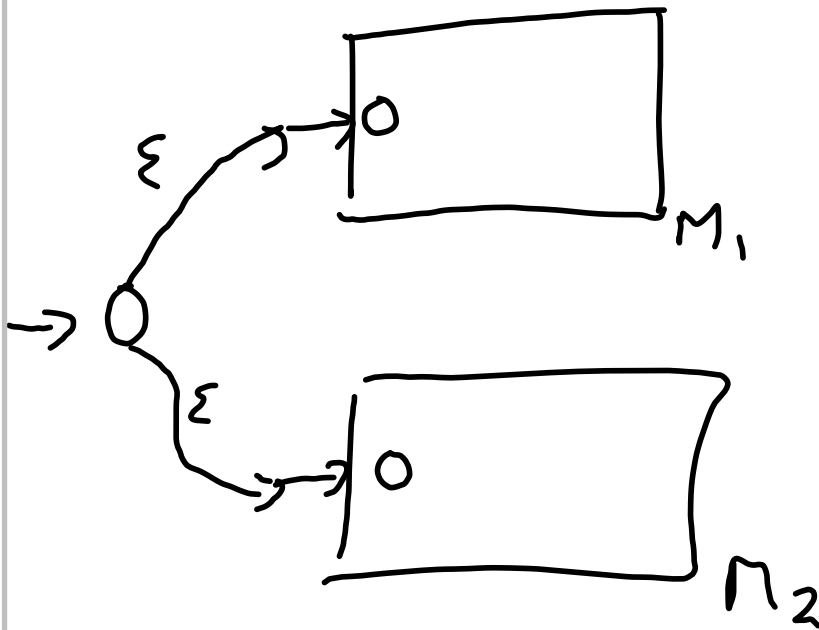
Let R_1 & R_2 be regex

Suppose we have NFAs

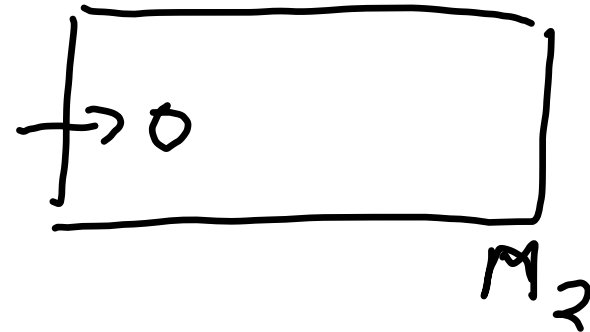
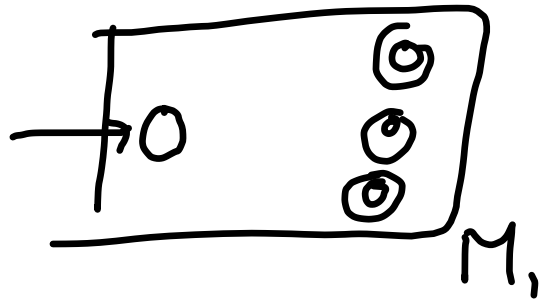
M_1 & M_2 accepting

Their languages

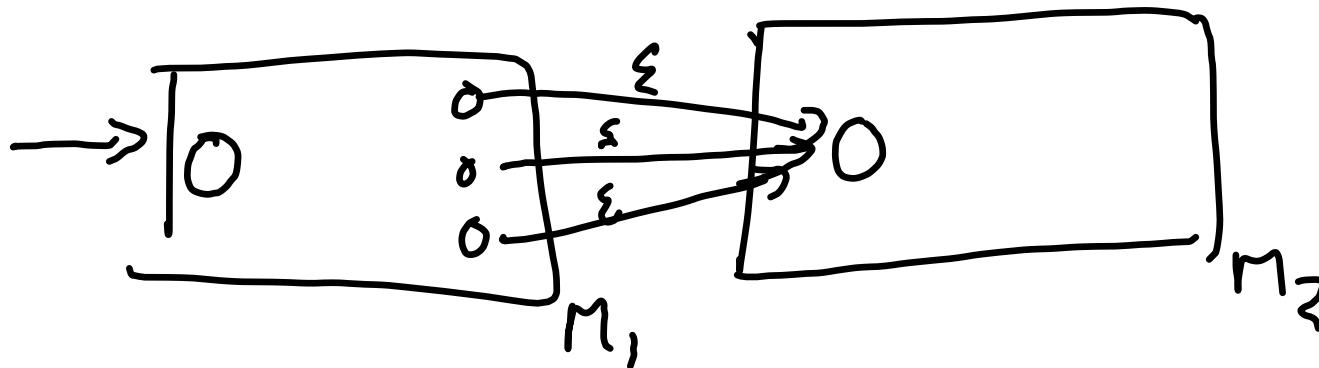
$R_1 \cup R_2$



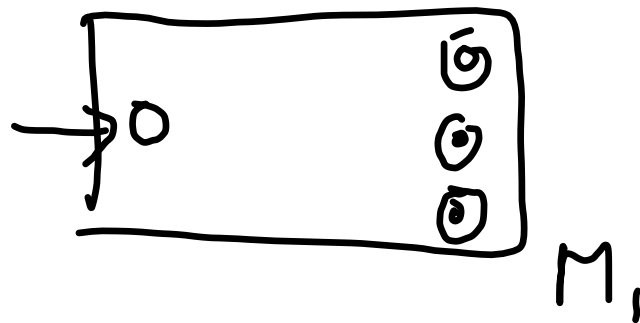
$R_1 \circ R_2$



Joined up

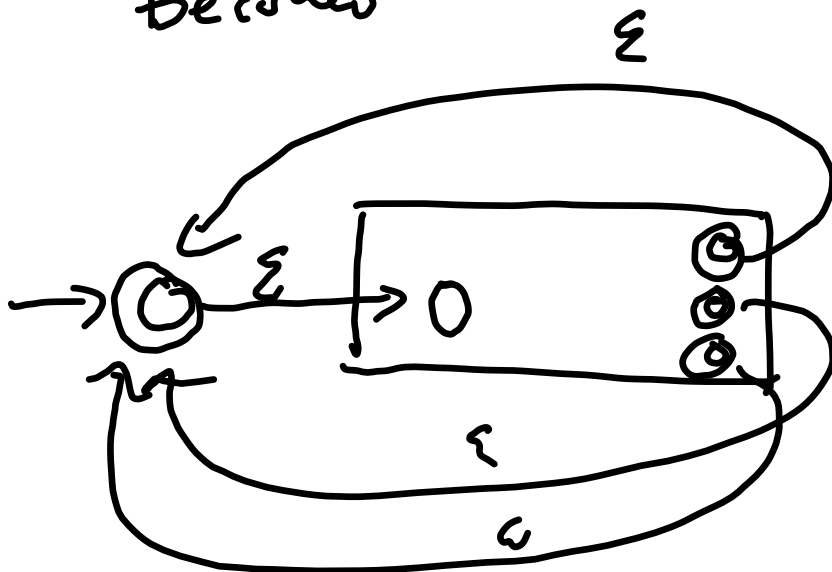


$R, \#$



recognize
 $L(R)$

becomes



recognize
 $L(R, \#)$

DFA \rightarrow Regex

- Normalize



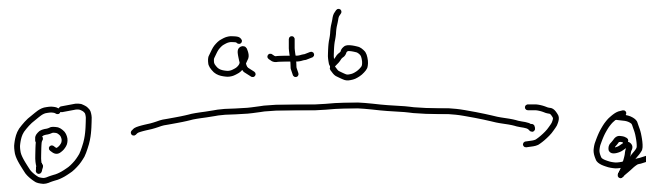
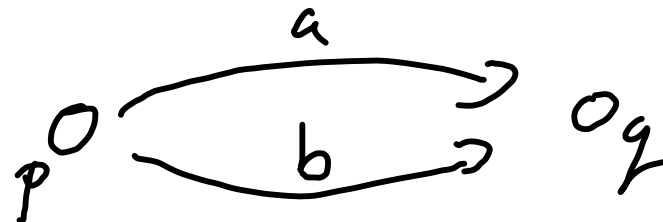
- Remove states one by one
making longer & longer
regexes as transitions

Generalized Non-deterministic Finite Automata

GNFA

Like NFA but each transition
has regex on it

One transition from each
state p to each state q



Special form for this proof

- GNFA's

- each pair of states (p, q)
There is exactly one transition
from p to q

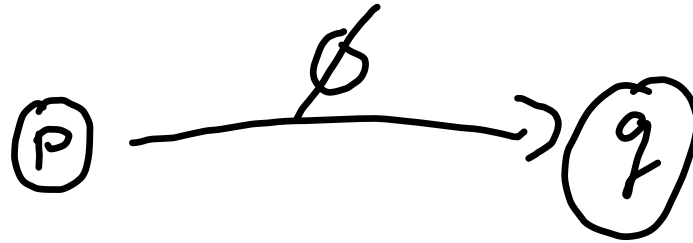
- except; start state has only
outgoing transitions

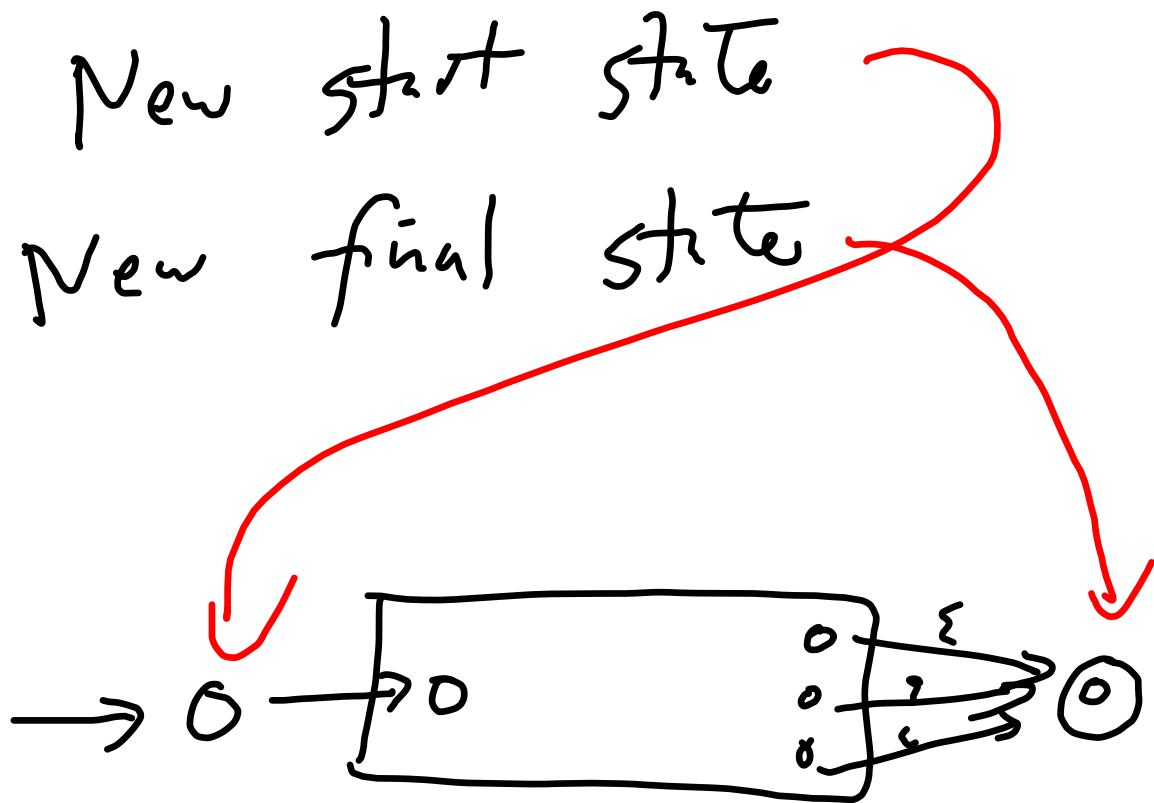
\exists one final state with
only incoming
transitions

\mathbb{P}

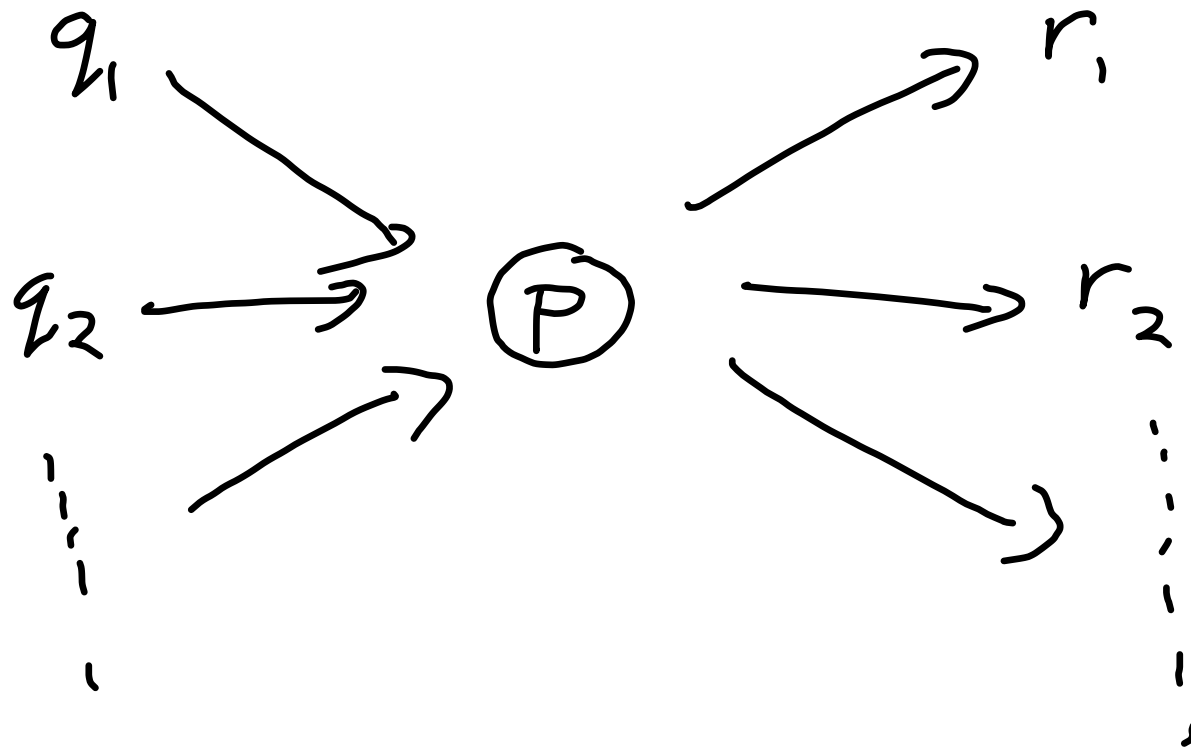
\mathbb{Q}

beomes

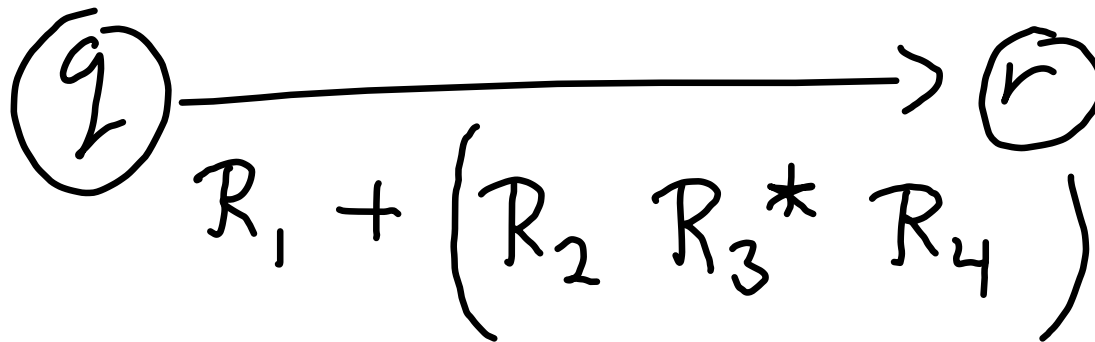
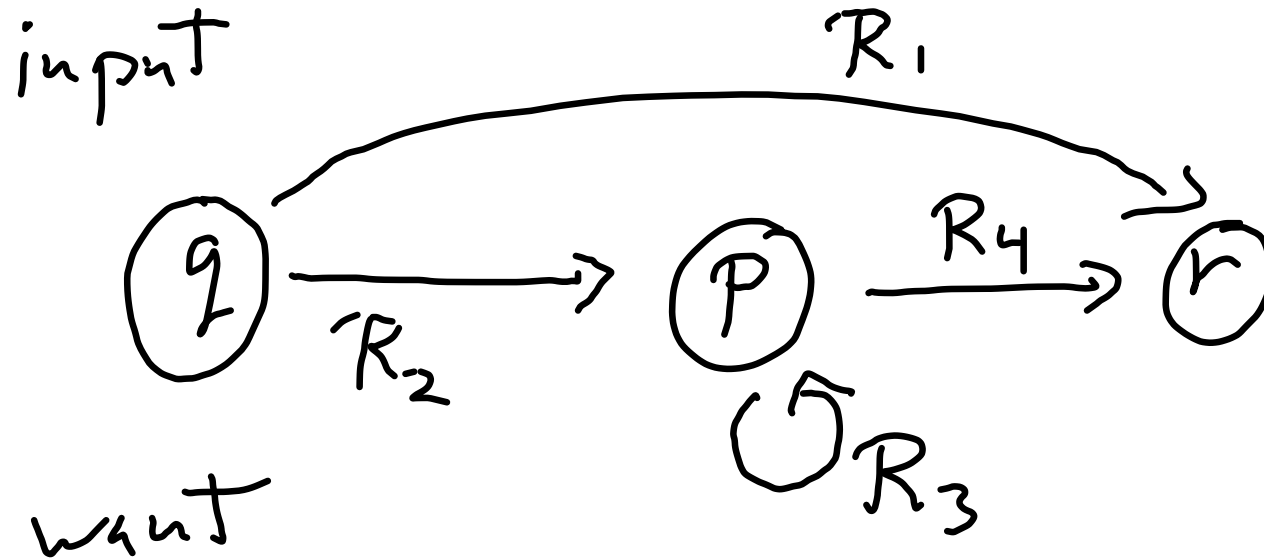




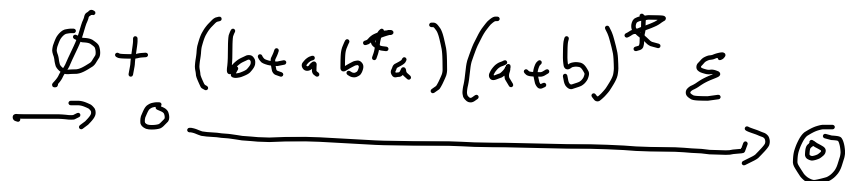
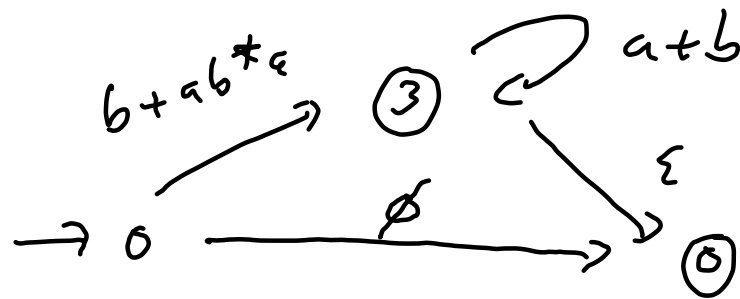
To Eliminate state P
(p not start, not final)



For every other states q & r
do \mathcal{R} follow



eliminate state 3



$$w \epsilon = w$$

$$\phi + w = w$$

$$\begin{aligned} \phi + (b + a b^* a) (a + b)^* \epsilon \\ = (b + a b^* a) (a + b)^* \end{aligned}$$

$$w \phi = \phi$$

$\epsilon + w$ leave as is

Regular Expression
represent exactly
the regular languages
i.e. languages accepted
by NFAs & DFAs

Closure under homomorphism

Simple example $a \rightarrow A$
 $b \rightarrow B$

$ab^*a \rightarrow A \star^* A$

homomorphism maps a character
to a string

$a \rightarrow A$
 $b \rightarrow B$

$a \rightarrow \epsilon$

$a \rightarrow 001$

Not ASCII

$a \rightarrow 001$

$b \rightarrow 010$

$c \rightarrow 011$

⋮

$ab^*a \rightarrow 001(010)^*001$

Defn: A homomorphism is
a function from Σ^* to
 Γ^* s.t.

\forall strings w, x, y
if $w = xy$

then $h(w) = h(x)h(y)$

$$\text{Then } h(w) = h(x)h(y)$$

So if $w = c_1 c_2 \dots c_n$

$$\text{Then } h(w) = h(c_1)h(c_2)\dots h(c_n)$$

Non-regular lgs

$$L = \{ a^n b^n : n \geq 0 \}$$

is not regular

Therefore

$$\{ 0^n 1^n : n \geq 0 \} \text{ is not regular}$$

$$h: \begin{array}{l} a \rightarrow 0 \\ b \rightarrow 1 \end{array} \text{ homomorphism}$$

~~Also~~

~~$$\{ (ac)^n x^n : n \geq 0 \}$$~~

~~is not regular~~

inverse
homomorphism

~~$$h: \begin{array}{l} a \rightarrow ac \\ b \rightarrow x \end{array}$$~~

To show that

$$L' = \{(ac)^n x^n : n \geq 0\}$$

not regular

Suppose L' is regular

Apply homomorphism h to L'

$$h: a \rightarrow a$$

$$c \rightarrow \varepsilon$$

$$x \rightarrow b$$

$$\text{Then } h(L') = \{a^n b^n : n \geq 0\}$$

but L' regular + closure under
homomorphism

$$\Rightarrow h(L') \text{ regular}$$

but $\{a^n b^n : n \geq 0\}$ is not
regular

by forward reference to Tuesday

So we have a contradiction.

So L' can't have been regular