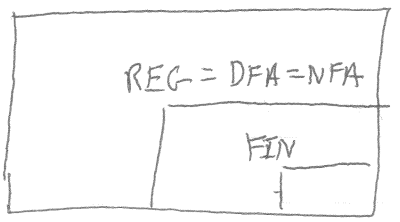
6-1

ALL



Categories of Languages	Computers i.e. $\in$	Specifications or programs i.e. $=$
ALL		
FIN	list of values	
REG	DFA, NFA	???

$\in : \Sigma^* \to Y/N$ — identifies members

$= : \to P(\Sigma^*)$ — produces the entire set

A regular expression over Σ is either:

		where r_1, r_2 are regexp	
ε	$r_1 \cup r_2$	$r_1 \cap r_2$	r_1^R
$\emptyset$	$r_1 \circ r_2$	$\overline{r_1}$ (r_1^c)	Optional
$c \in \Sigma$	r_1^*		

A regexp is a program written using the regular operators =
(operations the regular languages are closed under)

L : regexp $\to P(\Sigma^*)$

$L(\varepsilon) = \{\varepsilon\}$ $\quad L(r_1 \cup r_2) = L(r_1) \cup L(r_2)$

$L(\emptyset) = \emptyset$ $\quad L(r_1 \circ r_2) = L(r_1) \circ L(r_2)$

$L(c) = \{c\}$ $\quad L(r_1^*) = L(r_1)^*$

Third from end is 1:

$(1 \cup 0)^* \circ 1 \circ (1 \cup 0) \circ (1 \cup 0)$

unix:

*1??

perl regexp:

.*1...

$((a \circ b) \cup (c \circ d)) \circ e \circ f \circ$ ␣

$(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)$

(ab|cd)ef\1

\d

6-2/

DFA = REG

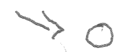
⇒ decompiles a DFA into a REG (hard)

⇐ compiles a REG into a DFA (easy)

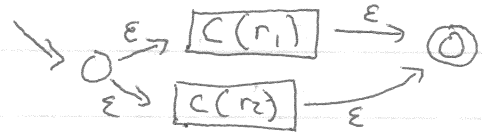
REG → NFA → DFA

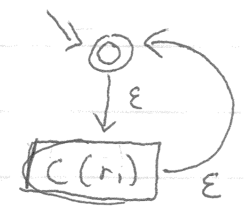
C : REG → NFA

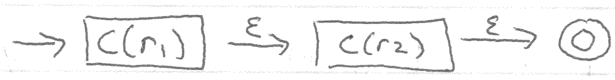
$C(\varepsilon)$ = 

$C(\emptyset)$ = 

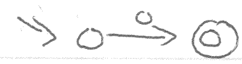
$C(c \in \Sigma)$ = 

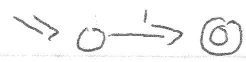
$C(r_1 \cup r_2)$ = 

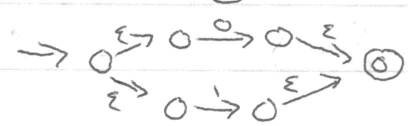
$C(r_1^*)$ = 

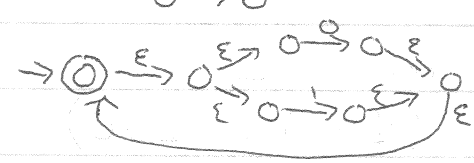
$C(r_1 \circ r_2)$ = 

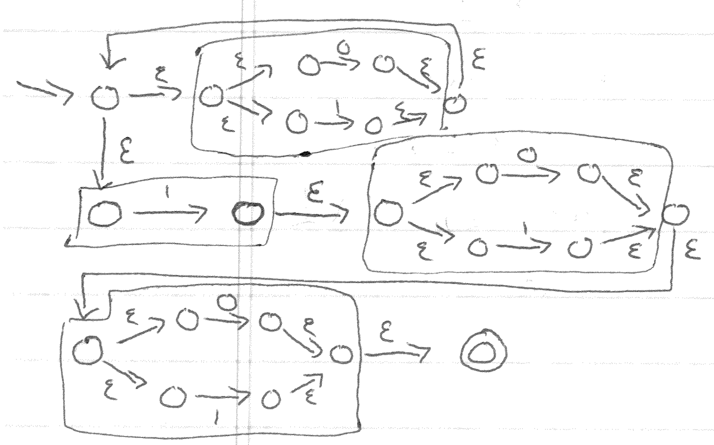
$C((0 \cup 1)^* \circ (1 \circ ((0 \cup 1) \circ (0 \cup 1))))$ =

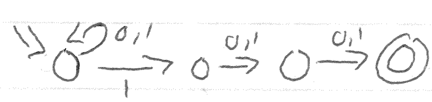
$C(0)$ = 

$C(1)$ = 

$C(0 \cup 1)$ = 

$C((0 \cup 1)^*)$ = 



= 

*1??