

15-453

FORMAL LANGUAGES, AUTOMATA AND COMPUTABILITY

THE PUMPING LEMMA

Let L be a regular language with $|L| = \infty$

Then there exists a positive integer P such that

if $w \in L$ and $|w| \geq P$

then $w = xyz$, where:

1. $|y| > 0$
2. $|xy| \leq P$
3. $xy^iz \in L$ for any $i \geq 0$

USING THE PUMPING LEMMA

Use the pumping lemma to prove that
 $B = \{0^n 1^n \mid n \geq 0\}$ is not regular

Hint: Assume B is regular, and try pumping $s = 0^P 1^P$

Use the pumping lemma to prove that
 $C = \{w \mid w \text{ has an equal number of 0s and 1s}\}$
is not regular

Hint: Try pumping $s = 0^P 1^P$

REGULAR EXPRESSIONS

TUESDAY SEP 18

WHAT DOES **D** LOOK LIKE?

$D = \{ w \mid w \text{ has equal number of occurrences of } 01 \text{ and } 10 \}$

REGULAR EXPRESSIONS

σ is a regular expression representing $\{\sigma\}$

ϵ is a regular expression representing $\{\epsilon\}$

\emptyset is a regular expression representing \emptyset

If R_1 and R_2 are regular expressions representing L_1 and L_2 then:

(R_1R_2) represents $L_1 \cdot L_2$

$(R_1 \cup R_2)$ represents $L_1 \cup L_2$

$(R_1)^*$ represents L_1^*

PRECEDENCE

***** **.** **∪**

EXAMPLE

$$R_1^*R_2 \cup R_3 =$$

{ w | w has exactly a single 1 }

What language does
 \emptyset^* represent?

{ w | w has length ≥ 3 and its 3rd symbol is 0 }

$\{ w \mid \text{every odd position of } w \text{ is a } 1 \}$

EQUIVALENCE

L can be represented by a regexp

\Leftrightarrow

L is a regular language

L can be represented by a regexp

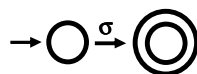
\Rightarrow

L is a regular language

Given regular expression R, we show there exists NFA N such that R represents L(N)

Induction on the length of R:

Base Cases (R has length 1):

$R = \sigma \rightarrow$ 

$R = \epsilon \rightarrow$ 

$R = \emptyset \rightarrow$ 

Inductive Step:

Assume R has length $k > 1$ and that any regular expression of length $< k$ represents a language that can be recognized by an NFA

Three possibilities for R:

$$R = R_1 \cup R_2$$

$$R = R_1 R_2$$

$$R = (R_1)^*$$

L can be represented by a regexp

\Rightarrow

L is a regular language

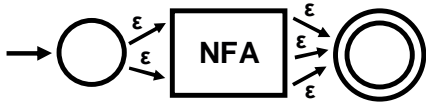
Transform $(1(0 \cup 1))^*$ to an NFA

L can be represented by a regexp

\Leftarrow

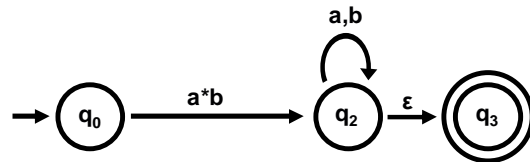
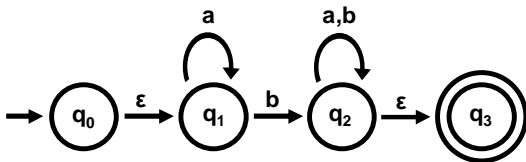
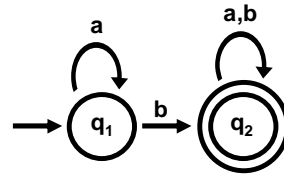
L is a regular language

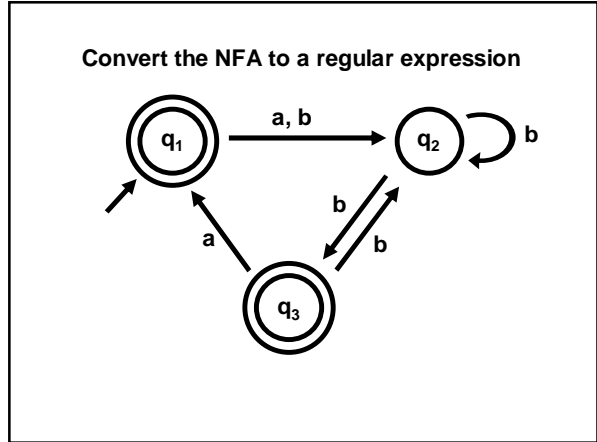
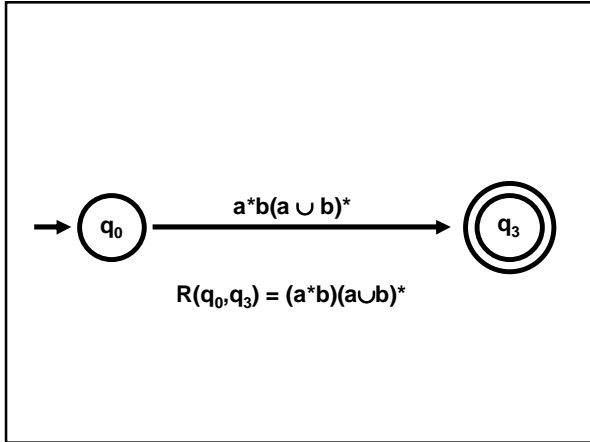
Proof idea: Transform an NFA for L into a regular expression by removing states and re-labeling the arrows with regular expressions



Add unique and distinct start and accept states

While machine has more than 2 states:
Pick an internal state, rip it out and re-label the arrows with regular expressions to account for the missing state





Formally: Add q_{start} and q_{accept} to create G
 Run CONVERT(G):
 If #states = 2
 return the expression on the arrow
 going from q_{start} to q_{accept}

Formally: Add q_{start} and q_{accept} to create G
 Run CONVERT(G):
 If #states > 2
 select $q_{rip} \in Q$ different from q_{start} and q_{accept}
 define $Q' = Q - \{q_{rip}\}$
 define R' as:
 $R'(q_i, q_j) = R(q_i, q_{rip})R(q_{rip}, q_{rip})^*R(q_{rip}, q_j) \cup R(q_i, q_j)$
 return CONVERT(G')

CONVERT(G) is equivalent to **G**

Proof by induction on **k** (number of states in **G**)

Base Case:

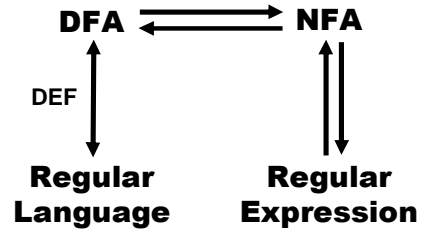
✓ $k = 2$

Inductive Step:

Assume claim is true for $k-1$ states

We first note that **G** and **G'** are equivalent

But then, by the induction hypothesis, **G'** is equivalent to **CONVERT(G')**



WWW.FLAC.WS

Read Chapter 1.3 of the book for next time