Regular Language

-

Hemant Kumar

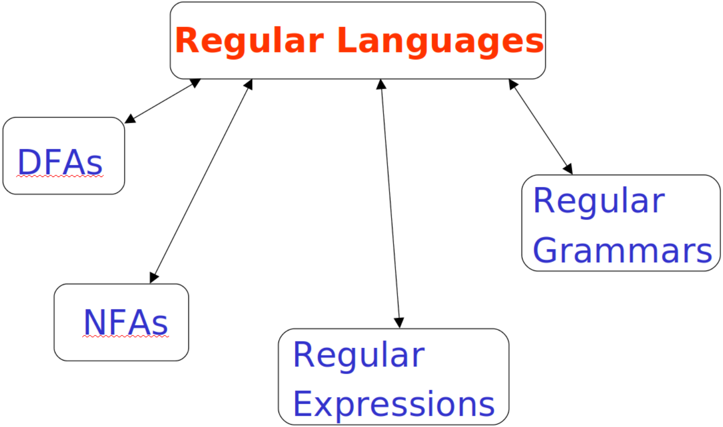
August 2, 2020

Outline

Regular Language

The language accepted by some regular expression or finite automata is known as regular language.

We know that language have two type finite language and infinite language.
Finite language is regular because we can construct it's regular expressions or finite automata.
Infinite language may or may not be regular language.


Regular Languages
DFAs
NFAs
Regular Expressions
Regular Grammars

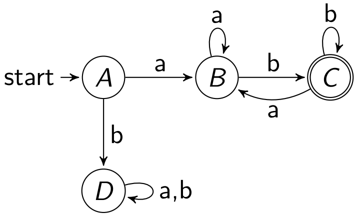
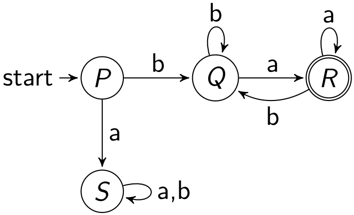
Closure Properties of Regular Languages

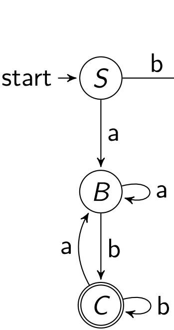
Regular language are closed under

1. Union
2. Intersection
3. Concatenation
4. Star Operation
5. Reverse
6. Complement
7. Difference
8. Homomorphism
9. Inverse Homomorphism
10. Right Quotient

Union

$L1 \cup L_2$ is regular if it's regular expression R_1 and R_2 's union generate a regular expression i.e. $R_3 = R_1 + R_2$ Construct a minimal dfa which start and ends with different symbol. Let input $\Sigma = \{a, b\}$.

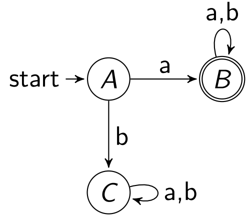
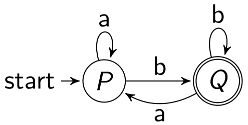
$L_1 = \{ab, aab, abb, abbb, \ldots\}$	$L_2 = \{ba, baa, bba, baaba, \ldots\}$
	

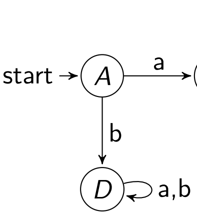


it makes DFA

Concatenation

$L_1 \cdot L_2$ is regular if $R_1 \cdot R_2$ gives a regular expression .

$L_1 = \{a, aa, ab, aaa, aba \ldots\}$	$L_2 = \{b, ab, bb, bab, aab, \ldots\}$
	



Star Operation

L_1^* is regular if R_1^* gives a regular expression .

Complement

$\overline{L} = \Sigma^* - L$

where Σ^* is Universal language.

Theorem : For regular language L , the complement $\overline{L}$ is regular.

Proof: Take a DFA that accepts L and make

- nonfinal states to final
- final states to nonfinal

Resulting DFA accepts $\overline{L}$

Intersection

$L_1 \cap L_2$ is regular if $(\overline{\overline{L_1} \cup \overline{L_1}})$ is regular

Theorem : For regular language L_1 and L_2, the intersection $L_1 \cap L_2$ is regular.

Proof: Apply Demorgan's Law:

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_1}}$$

$$L_1 \;,\; L_2 \qquad \textit{regular}$$

$$\overline{L_1} \;,\; \overline{L_2} \qquad \textit{regular}$$

$$\overline{L_1} \cup \overline{L_2} \qquad \textit{regular}$$

$$\overline{\overline{L_1} \cup \overline{L_1}} \qquad \textit{regular}$$

$$L_1 \cap L_2 \qquad \textit{regular}$$

Difference

Using Demorgon's Law
$L_1 - L_2 = L_1 \cap \overline{L_2}$ is regular language.

Theorem : For regular language L_1 and L_2, the intersection $L_1 - L_2$ is regular.

Proof: Apply Demorgan's Law:

$$L_1 - L_2 = L_1 \cap \overline{L_2}$$

Apply Demorgon's Law

$$L_1 , L_2 \qquad regular$$

$$L_1 , \overline{L_2} \qquad regular$$

$$L_1 \cap \overline{L_2} \qquad regular$$

$$L_1 - L_2 \qquad regular$$

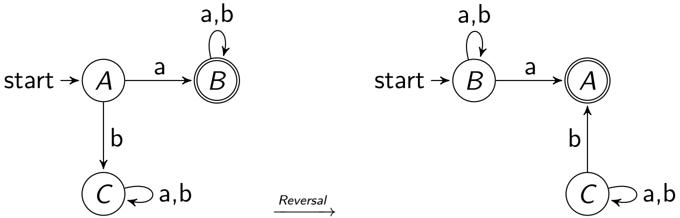
Reverse

$L \xrightarrow{Reveresal} L^R$ Reversal operation on DFA and output can be either DFA or NFA.
For Reversal operation , we make final state to start state and start state to final state and the direction of arrow is revert. Dead state in DFA convert to unreachable or useless state and remove that state.

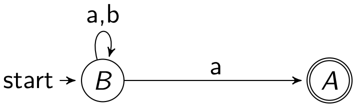
Reversal of a of minimal DFA which start with a.

languages $L_1 = \{a, aa, ab, aaa, aba \ldots\}$

L	$\xrightarrow{Reversal}$	L^R



Here state C is unreachable or useless state so remove it, and final automata is



This is NFA not DFA.

Decision Properties of Regular Language

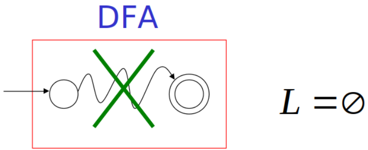
Membership Property

Question: Given regular language L and string ω, how can we check if $\omega \in L$?

Answer: Take the DFA that accepts L and check if ω is accepted

Emptyness Property

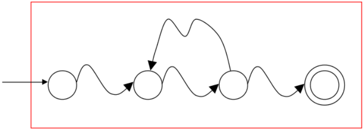
Question: Given regular language L , how can we check if L is empty: $(L = \phi)$?
Answer: Take the DFA that accepts L and check if there is a path from the initial state to a final state



Finiteness Property

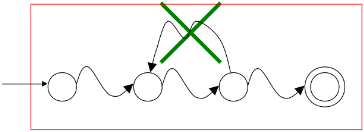
Question: Given regular language L , how can we check if L is finite ?
Answer: Take the DFA that accepts L and check if a walk with a cycle from a initial state to a final state

DFA



L is infinite

DFA



L is finite