Chapter 5
Regular Expressions

Plan

1. Definitions
2. Kleene's Theorem
   - Compute the regular expression associated with an automaton
   - From the regular expression to the automaton
3. Star Lemma

Regular languages

A language is said to be regular (rational) if there exists a regular grammar that generates it.

Let the grammar $G = (V_T, V_N, S, R)$,

Definition of a regular grammar:

G is said to be regular if and only if all its production rules have one of the following forms:

$A \rightarrow aB$ or $A \rightarrow a$ with $A, B \in V_N$ and $a \in V_T$.

Regular languages

Definition: A language is regular if and only if there exists a regular grammar that generates it.

Definition: A language is regular if and only if there exists a finite state automaton that recognizes it.

Closure properties of the class of regular languages

- In addition to the regular operations (., *, union, and mirror), the class of regular languages is closed under complement and intersection. .

Rational Languages

Definition:

A language is called a rational language if it can be expressed using a finite number of operations (Regular Expressions).

Definition:

Let X be an alphabet. The regular expressions defined over X and the sets they denote are recursively defined as follows:

1. $\emptyset$ is a regular expression (empty set).
2. a is a regular expression representing the set $\{a\}$.
3. If $w_i \in X$, then w_i is a regular expression representing the set $\{w_i\}$.
4. If E_1 and E_2 are two regular expressions, then $E_1.E_2$, $E_1 \cup E_2$, and E_1^* are also regular expressions.

Example

$L = \{\omega \in \{0,1\}^* \text{ such that } \omega \equiv 0 \mod 2\}$

- $E_1 = (0 \cup 1)^*.0 = X^*.0$ (insignificant zeros).
- $E_2 = 1.(0 \cup 1)^*.0 \cup 0$ (no insignificant zero).

Operator precedence:

Kleene star and plus in the exponent take precedence over concatenation, which in turn takes precedence over plus on the line.

$\uparrow$

* (Itération)

. (Concaténation)

$\cup$ (union)

Example:

$E = 0.1^* \cup 0 = ((0.(1)^*) \cup 0)$

Definition:

Two regular expressions E_1 and E_2 are equivalent if and only if they define the same language, i.e., $L(E_1) = L(E_2)$.

- **Kleene's Theorem:** The class of rational languages is exactly equal to the class of regular languages.

Conversion

From FSA$\leftrightarrow$ RE

From FSA $\leftrightarrow$ RE

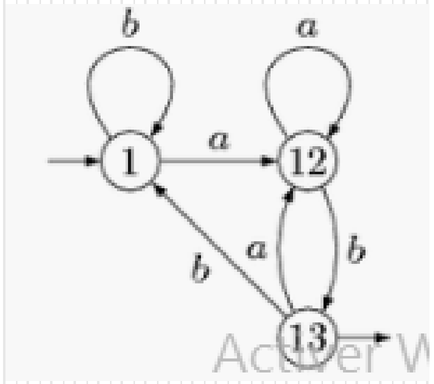
- **Proposition:** For every regular expression E, there exists a finite state automaton (FSA) that recognizes the language denoted by E.

- **Proposition:** For every finite state automaton A, there exists a regular expression E that denotes the language recognized by A.

From the regular expression to the automaton

RE$\longrightarrow$ FSA

$(a+b)^*ab(bb+a)^*$ $\longrightarrow$



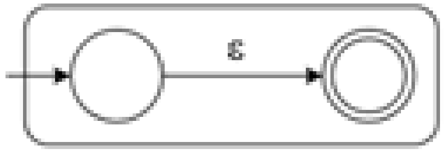
I. Associate an automaton with a regular expression

- It is possible to mechanically (and recursively) associate

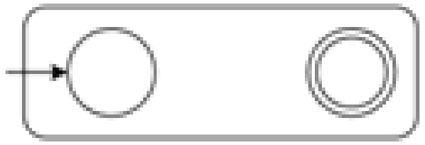
an **ε-transition** with a regular expression. For this, we will use three basic automata and three generic automata

Basic Automata

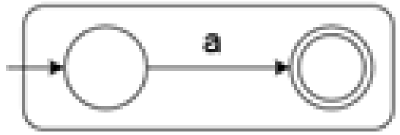
- The first automaton recognizes the language associated with the regular expression **ε**.



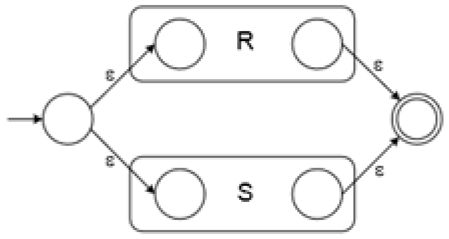
- The second automaton recognizes the language associated with ∅.



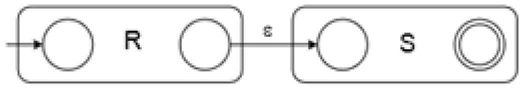
- The third automaton recognizes the language associated with the regular expression **a**.



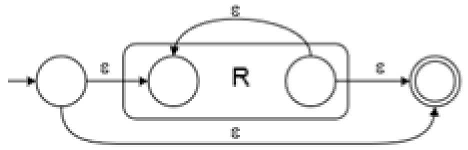
- Regular expressions (RE) are generated through union, concatenation, and closure operations. This results in the following three cases:
- The expression **R + S**:



- The expression **RS**:



- The expression **R***:



- **Note:** For the expression **(R)**, it is sufficient to use the automaton associated with **R**.

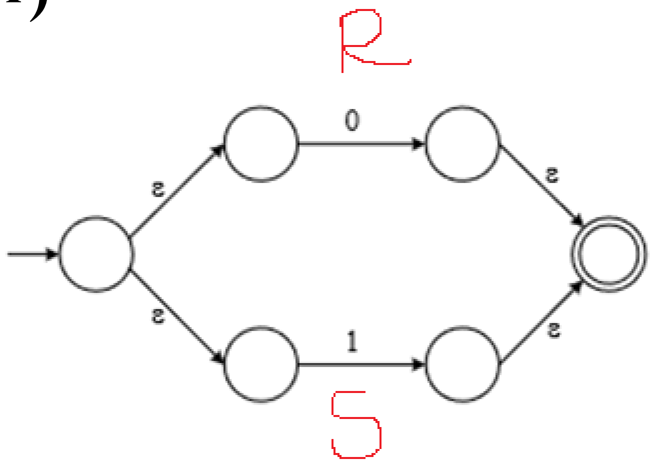
Example

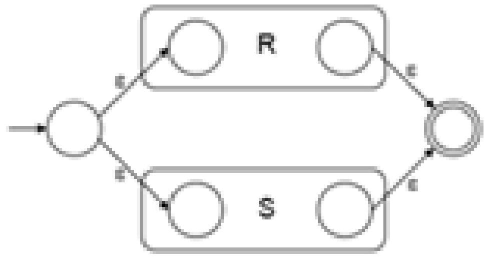
Construct the automaton associated with the regular expression

(0+1)*1(0+1)

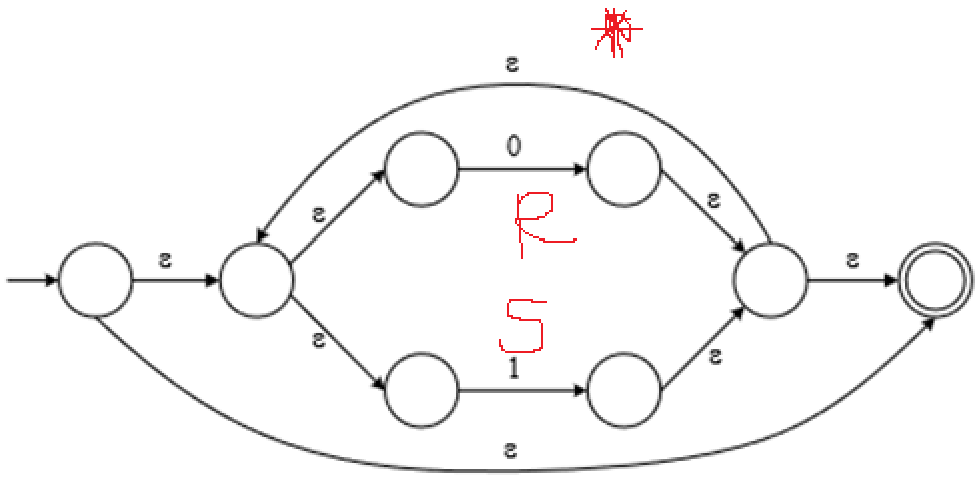
Solution

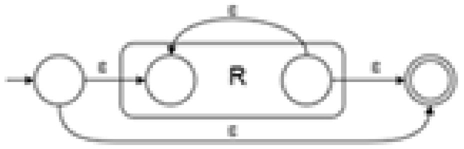
Step 1: (0+1)





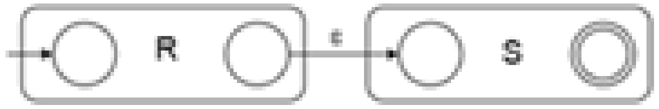
Step 2: (0+1)*

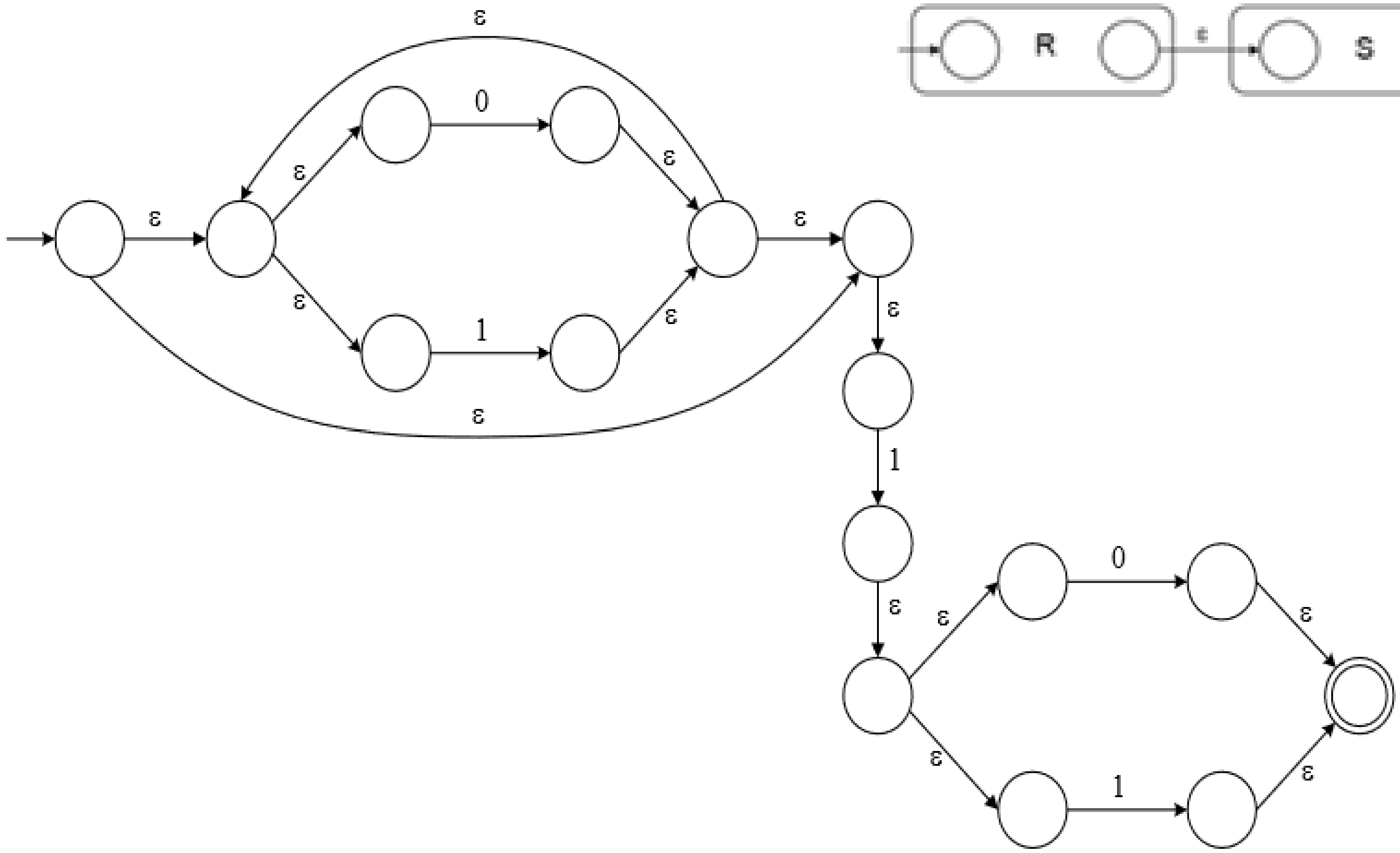




Solution

Step 3: (0+1)*1(0+1)





II. The Nerode derivatives

Derivatives

Definition:

Let L be a language over an alphabet X and $\omega \in X^*$. The derivative of L with respect to ω, denoted $L \parallel \omega$, is defined as:

$$L \parallel \omega = \{z \in X^* \mid \omega.z \in L\}$$

Examples:

Let the languages be:

$L_1 = \{\epsilon, a, ab, aa, ba\}$ and $L_2 = \{a^n \mid n > 0\}$.

- $L_1 \parallel a = \{\epsilon, b, a\}$
- $L_1 \parallel aa = \{\epsilon\}$
- $L_2 \parallel a = L_2$

Derivatives

Nerode's Theorem:

A language L is regular over X^* if and only if the number of derivatives of L is finite

Example:

$L = \{a^i b^j \mid i, j \geq 0\}$

- $L \parallel a = \{a^i b^j \mid i, j \geq 0\} = L$
- $L \parallel b = \{b^j \mid j \geq 0\} = L_1$

Properties of derivatives

- $a_i \parallel a_j = \begin{cases} \varepsilon & \text{If } a_i = a_j \\ \varnothing & \text{Otherwise} \end{cases}$

- $(L_1 \bigcup L_2) \parallel a = L_1 \parallel a \bigcup L_2 \parallel a$

- $(L_1.L_2) \parallel a = \begin{cases} (L_1 \parallel a).L_2 & \text{If } \varepsilon \notin L_1 \\ (L_1 \parallel a).L_2 \cup L_2 \parallel a & \text{Otherwise} \end{cases}$

- $L^* \parallel a = (L \parallel a).L^*;$

Properties of derivatives

- $L \parallel \omega_1.\omega_2 = (L \parallel \omega_1) \parallel \omega_2\,;$
- $L \parallel \omega = \varnothing$ if no word in L starts with ω
- $\varepsilon \parallel a = \varnothing\,;$
- $\omega.L \parallel \omega = L.$

Example1

L = { $a^i b^j$ such that i, j ≥ 0 }

L // ε = L

L // **a** = { $a^i b^j$ such that i, j ≥ 0 } = L

L // **b** = { b^j such that j ≥ 0 } = L_1

L_1 // **a** = b // a L_1 = ∅ L_1 = ∅ = L_2

L_1 // **b** = L_1

L_2 // **a** = L_2 // **b** = L_2

Exemple1

$L = \{ a^i b^j$ such that $i, j \geq 0 \}$

$L \,//\, \varepsilon = L$

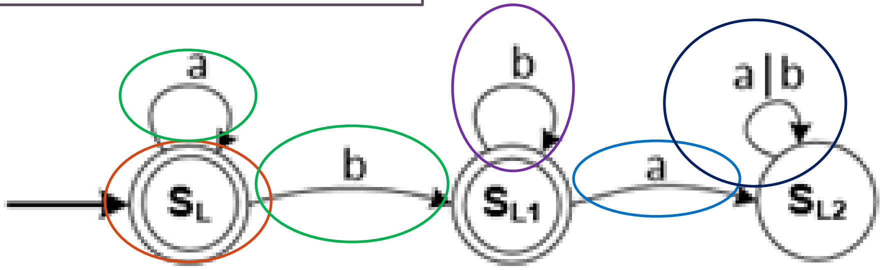
$L \,//\, \mathbf{a} = \{ a^i b^j$ such that $i, j \geq 0 \} = L$

$L \,//\, \mathbf{b} = \{ b^j$ such that $j \geq 0 \} = L_1$

$L_1 \,//\, \mathbf{a} = b \,//\, a\, L_1 = \emptyset\, L_1 = \emptyset = L_2$

$L_1 \,//\, \mathbf{b} = L_1$

$L_2 \,//\, \mathbf{a} = L_2 \,//\, \mathbf{b} = L_2$



Example 2

$L = \{ a^i b^j$ such that $i, j > 0 \}$

L // **a** = $\{ a^i b^j$ such that $i \geq 0, j > 0 \} = L_1$

L // **b** = $\varnothing = L_2$

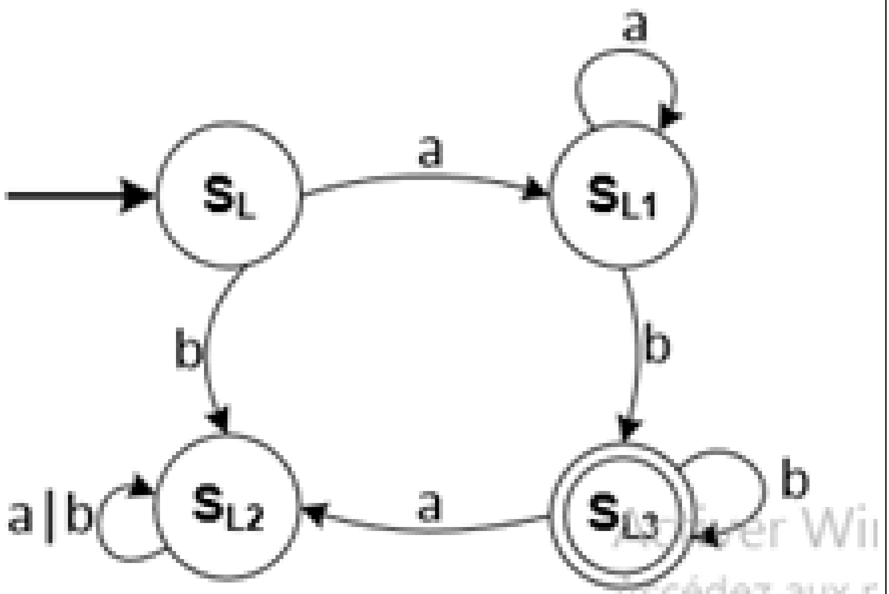
L_1 // **a** = L_1

L_1 // **b** = $\{ b^j$ such that $j \geq 0 \} = L_3$

L_2 // **a** = L_2 // **b** = L_2

L_3 // **a** = $\varnothing = L_2$

L_3 // **b** = L_3



Example 3

Let the language $L = (a+b)^*ab(bb+a)^*$.

Compute $L \parallel a$.

$$L = \underbrace{(a+b)^*}_{L1}\underbrace{ab(bb+a)^*}_{L2}$$

Example 3

$$L \parallel a = (a+b)^* \parallel a.ab(bb+a)^* + ab(bb+a)^* \parallel a$$

$$\boxed{(L_1.L_2) \parallel a = \begin{cases} (L_1 \parallel a).L_2 & \text{If} \quad \varepsilon \notin L_1 \\ (L_1 \parallel a).L_2 \cup L_2 \parallel a & \end{cases}}$$

Example 3

$$L \parallel a = (a+b)^* \parallel \underline{a}.ab(bb+a)^* + ab(bb+a)^* \parallel a$$

$$= \underline{(a+b) \parallel a.(a+b)}^*ab(bb+a)^* + \underline{b(bb+a)^*}$$

$$\boxed{\bullet \quad L^* \parallel a = (L \parallel a).L^* \,;}$$

$$\boxed{\bullet \quad \omega.L \parallel \omega = L.}$$

Example 3

$$L \parallel a = (a+b)^* \parallel a.ab(bb+a)^* + ab(bb+a)^* \parallel a$$

$$=(a+b) \parallel a.(a+b)^*ab(bb+a)^* + b(bb+a)^*$$

$$=\underline{(a \parallel a + b \parallel a)}.a.(a+b)^*ab(bb+a)^* + b(bb+a)^*$$

$$\boxed{\bullet\ (L_1 \bigcup L_2) \parallel a = L_1 \parallel a \bigcup L_2 \parallel a}$$

Example 3

$$L \parallel a = (a+b)^* \parallel \underline{a}.ab(bb+a)^* + ab(bb+a)^* \parallel a$$

$$=(a+b) \parallel a.(a+b)^*ab(bb+a)^* + b(bb+a)^*$$

$$=\underbrace{(\underbrace{a \parallel a}_{\varepsilon} + \underbrace{b \parallel a}_{\varnothing})}_{\varepsilon}.a.(a+b)^*ab(bb+a)^* + b(bb+a)^*$$

- $$\boxed{a_i \parallel a_j = \begin{cases} \varepsilon & \text{If} \quad a_i = a_j \\ \varnothing & \text{Otherwise} \end{cases}}$$

Example 3

$$L \parallel a = (a+b)^* \parallel \underline{a}.ab(bb+a)^* + ab(bb+a)^* \parallel a$$

$$= (a+b) \parallel a.(a+b)^*ab(bb+a)^* + b(bb+a)^*$$

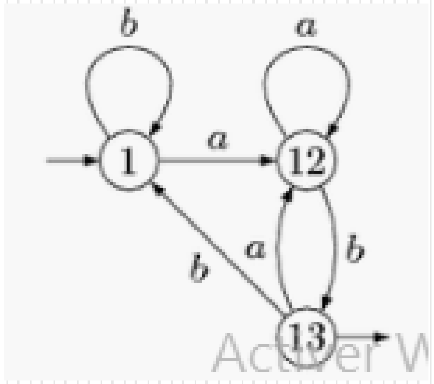
$$= (a \parallel a + b \parallel a).a.(a+b)^*ab(bb+a)^* + b(bb+a)^*$$

$$= (a+b)^*ab(bb+a)^* + b(bb+a)^*. \quad = L_1$$

We stop here because there are no more derivatives to compute, so the result is a new language: L_1 .

Calculate the regular expression associated with an automaton

From FSA⟶ RE



$$\longrightarrow \quad (a+b)^*ab(bb+a)^*$$

Automaton reduction

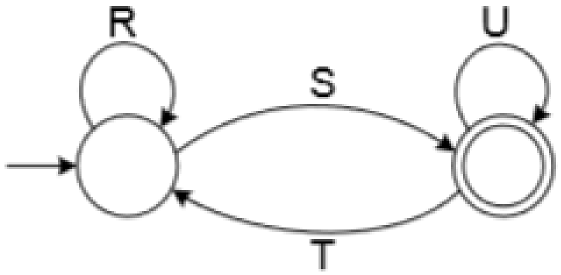
Automaton reduction

The process to construct a regular expression from an automaton is as follows:

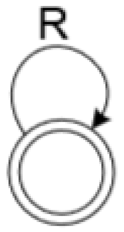
- For each accepting state q, eliminate all intermediate states between e_0(the initial state) and q;

• If q≠ e_0 , we obtain an automaton with two states. The regular expression associated with the language is then:

(R+SU*T)*SU*



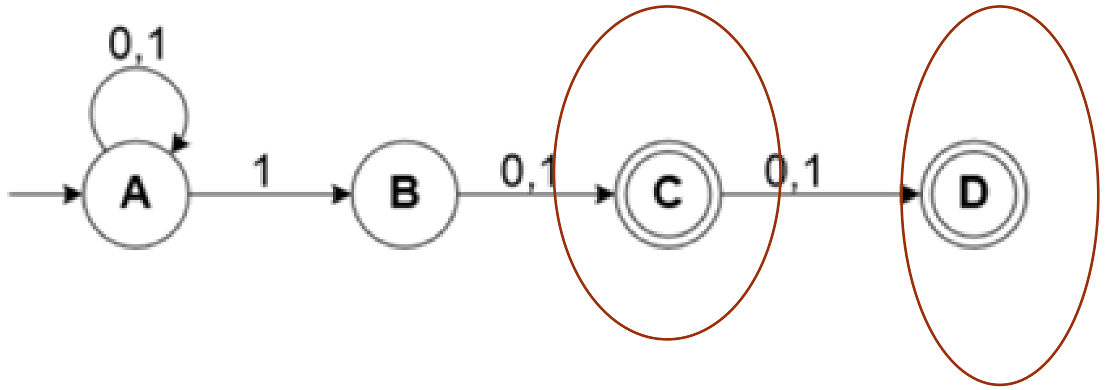
- If e_0 is an accepting state, then we obtain an automaton with a single state. The regular expression associated with the language is then : R*



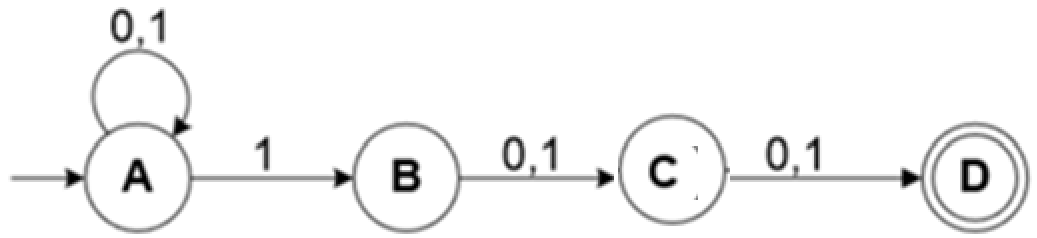
- The regular expression representing the automaton is then the union of all the expressions calculated from the reduced automata by applying rules 2) and 3) for each of the accepting states of the initial automaton.

Example

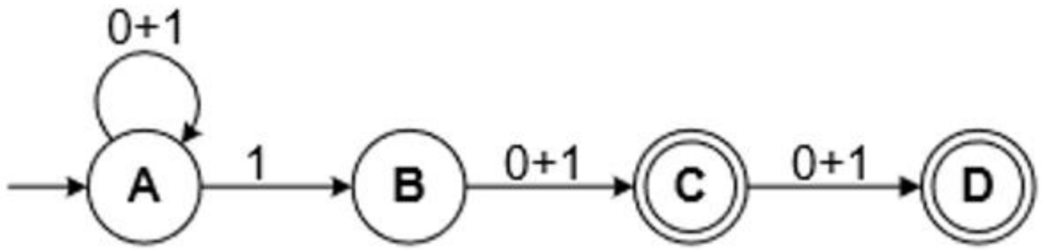
- Find the regular expression of the following non-deterministic finite automaton


$$\begin{tikzcd} \arrow[r] & A \arrow[loop above, "{0,1}"] \arrow[r, "1"] & B \arrow[r, "{0,1}"] & \boxed{C} \arrow[r, "{0,1}"] & \boxed{D} \end{tikzcd}$$

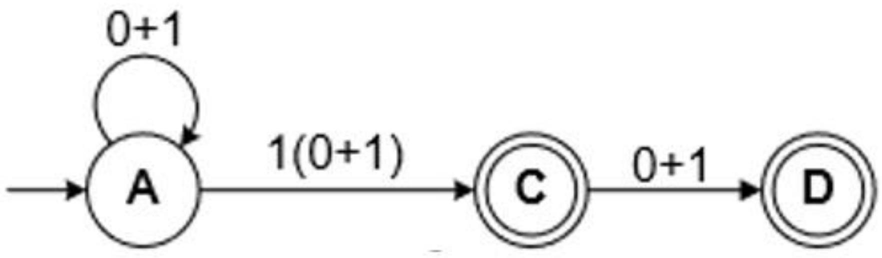
Solution



Step 1

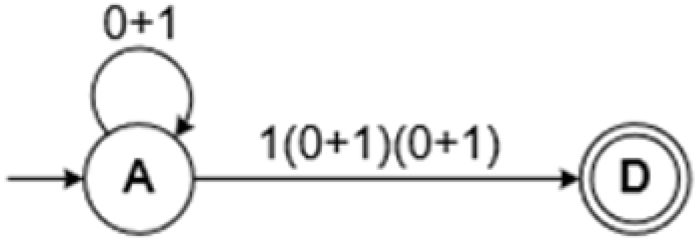


Step 2



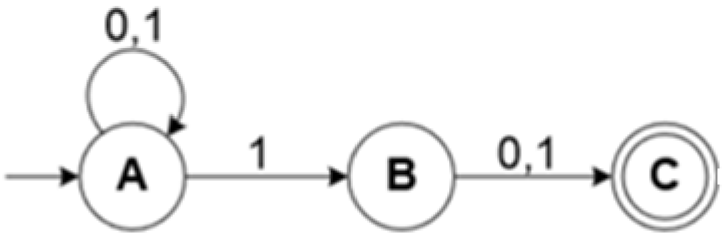
Solution

Step 3

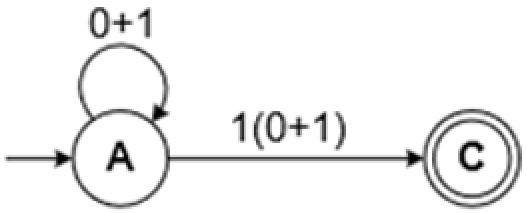

$$\begin{tikzcd} \arrow[r] & A \arrow[loop above, "0+1"] \arrow[r, "1(0+1)(0+1)"] & D \end{tikzcd}$$

The regular expression associated: **(0+1)*1(0+1)(0+1)**

Solution



Step 4: Eleminate the state D



The regular expression associated is: **(0+1)*1(0+1)**

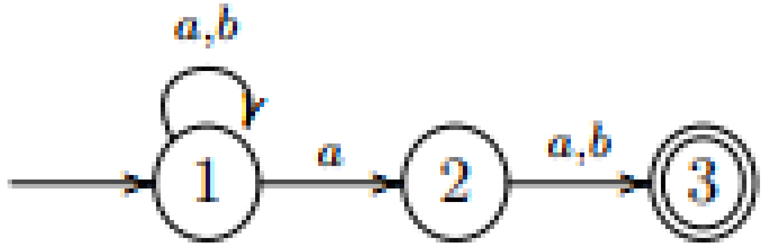
The ER is

(0+1)*1(0+1) + (0+1)*1(0+1)(0+1)

Arden's equation

From finite state automata to Arden's equations

Consider the automaton



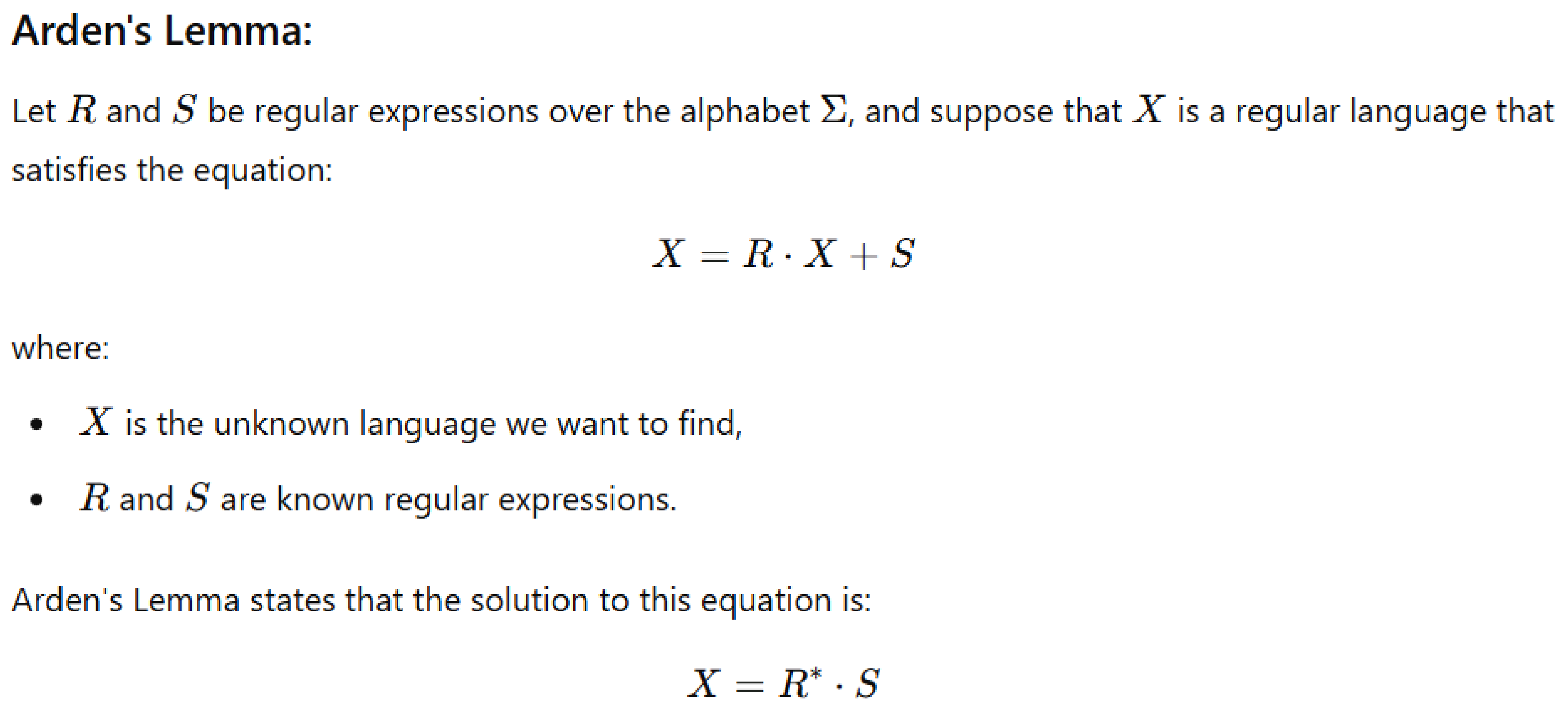
Let L_q be the language recognized by state q of the automaton. The automaton can then be described as a system of equations on languages:

$$\begin{cases} L_1 = (a+b).L_1 + a.L_2 \\ L_2 = (a+b).L_3 \\ L_3 = \epsilon \end{cases}$$

The language recognized by the automaton is the language of its initial state

From Arden's equations to regular expressions

- To obtain the regular expression corresponding to each language L_q, we solve the system of equations using Arden's lemma



Arden's Lemma:

Let R and S be regular expressions over the alphabet Σ, and suppose that X is a regular language that satisfies the equation:

$$X = R \cdot X + S$$

where:

- X is the unknown language we want to find,
- R and S are known regular expressions.

Arden's Lemma states that the solution to this equation is:

$$X = R^* \cdot S$$

Arden’s Lemma

$$X = aX \cup b \;\rightarrow\; X = a^{*}b$$

Application

- By applying Arden's lemma to the previous system of equations, we obtain

$X = aX \cup b \rightarrow X = a^*b$

$$\begin{cases} L_1 = \underbrace{(a+b)}_{A}.L_1 + \underbrace{a.L_2}_{B} = \underbrace{(a+b)^*}_{A}.\underbrace{a.L_2}_{B} = (a+b)^*.a.(a+b) \\ L_2 = (a+b).L_3 = (a+b).\epsilon = (a+b) \\ L_3 = \epsilon \end{cases}$$

Iteration Lemma (Star Lemma)

For any regular Language L, it exist an integer $n \in \mathbb{N}$ such that

$\forall \omega \in L, |\omega| \geq n$ We can decompose it to $uvy, u, y \in X^*$ and $v \in X^+$

such that $uv^*y \in L$ $(uv^iy \in L, i \geq 0)$.

Example Demonstrate th**L={a^{i}b^i , i ≥0}** is not regular Language

Proof by contradiction

- We suppose that L is regular :
- $\forall \omega \in L$ We have

  $\omega = uvy, u, y \in X^*$ et $v \in X^+$ Such That $uv^*y \in L$
- We put : $\omega = a^n b^n$

Exemple

1) $v \in a^+ \Rightarrow |v| = k$, $k > 0$.

$\Rightarrow \omega = a^n b^n = a^{(n-k)} a^k b^n$

$\Rightarrow a^{(n-k)} (a^k)^i b^n \in L$, $i \geq 0$

If $i=0$ Then $a^{(n-k)} b^n \in L$ contradiction for $k>0$.

2) $v \in b^+ \Rightarrow |v| = k$, $k > 0$.

$\Rightarrow \omega = a^n b^n = a^n b^k b^{n-k}$

$\Rightarrow a^n (b^k)^i b^{n-k} \in L$, $i \geq 0$

If $i=0$ Then $a^n b^{n-k} \in L$ contradiction for $k>0$.

Example

3) $v \in a^+b^+ \Rightarrow v = a^{k1}b^{k2}$, k1, k2 > 0.

$\Rightarrow \omega = a^n b^n = a^{n-k1}a^{k1}b^{k2}b^{n-k2}$

$\Rightarrow a^{n-k1}(a^{k1}b^{k2})^i b^{n-k2} \in L$, $i \geq 0$

If i=2 Then $a^{n-k1}a^{k1}b^{k2}a^{k1}b^{k2}b^{n-k2} \in L$

contradiction.

So L is not a regular language ($L \notin Reg(X^*)$).

Methods to show that a language is regular

We can show the regularity of a language L using one of the following methods:

- All finite languages are regular;
- If we find a DFA that recognizes a language L, then L is regular;
- If we find a regular grammar generating L, then the language is regular;
- We can use Nerode's theorem to show that a language is regular;
- We can exploit closure properties to show that a language is regular.

Methods to show that a language is **NOT** regular

To show the irregularity of a language L, it is not enough to be unable to find a DFA recognizing it; we can use the following two methods to do so:

• Proof by contradiction for the star theorem;

• Exploiting the closure properties of non-regular languages: regular languages are closed under certain operations (such as union, intersection, complementation, concatenation, and Kleene star) and non-regular languages may fail to maintain these properties under some operations.