Regular Expressions

Table of Contents

[Rules 3](#_Toc132732943)

[Regular Language to Regular Expression 5](#_Toc132732944)

[Finite State Machine to Regular Expression 8](#_Toc132732945)

[Arden’s Method 8](#_Toc132732946)

[Converting Regular Expressions to Finite State Machines 13](#_Toc132732947)

1. is a regular expression when .
2. is a regular expression when is a regular expression and is a regular expression.
3. is a regular expression when is a regular expression and is a regular expression.
4. is a regular expression if is a regular expression.
5. is a regular expression.
6. is a regular expression.
7. is a regular expression.

### Rules

1. Start binds tighter:

, not .

1. Concatenation binds tighter than unions:

, not

, not

Examples:

Let

For , .

For , .

For , .

For , .

For , .

For , .

Rules:

For the last two rules, the proof is as follows:

Let and , where and .

Let and . Again, and .

More rules:

* (think of as )
* (think of as )

Examples:

## Regular Language to Regular Expression

,

## Finite State Machine to Regular Expression

### Arden’s Method

If and are two regular expressions over , and if does not contain , the following equation in , given by , has a unique solution, .

Proof:

or

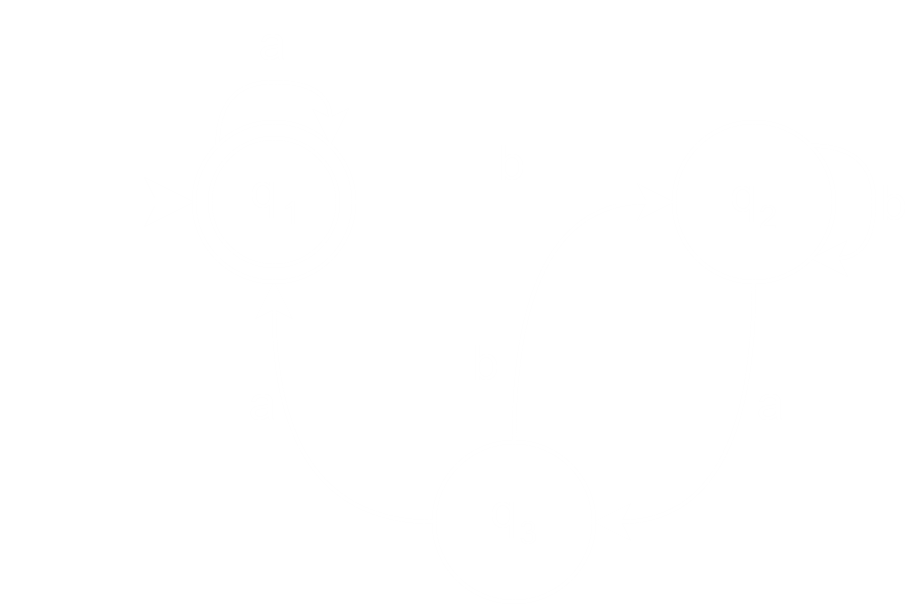
Steps:

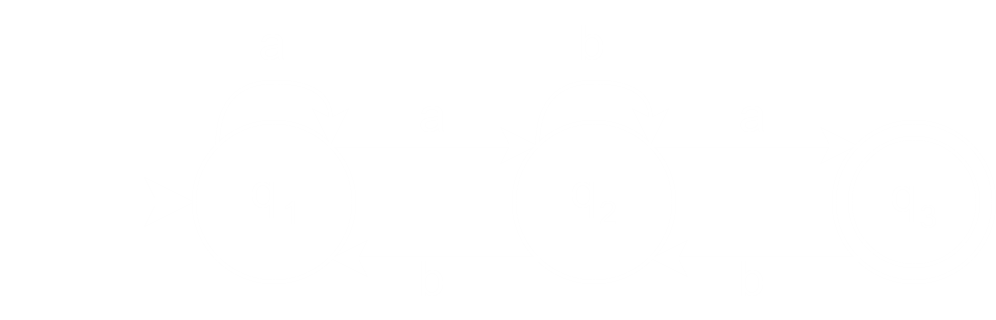
1. Write an equation for each state based on its **incoming edges**.
2. Simplify the equations using Arden’s method and find the regular expression for the final state.

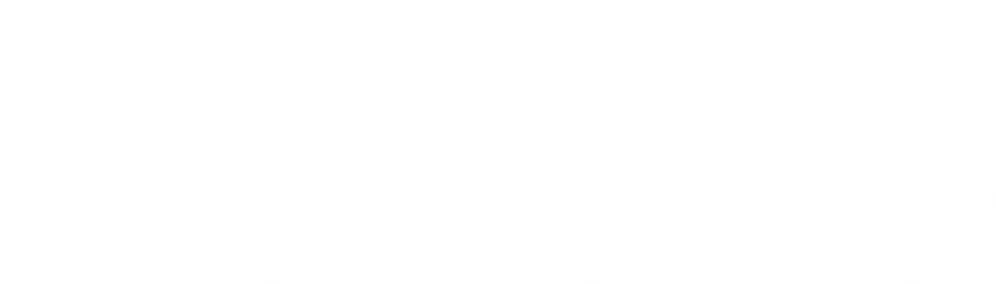
Conditions:

* 1. The finite state machine should not have an edges.
  2. The finite state machine should only have one initial state.









## Converting Regular Expressions to Finite State Machines

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |