Finite Automata

Model where the size of the memory is fixed / limited compared to the size of the input.

Command	Output
`\reals`	$\mathbb{R}$
`\integers`	$\mathbb{Z}$
`\naturals`	$\mathbb{N}$
`\rationals`	$\mathbb{Q}$
`\complex`	$\mathbb{C}$

- States: q_1, q_2, q_3
- Transitions: arrows with labels on them
- Start state is the state pointed at with an arrow (q_1)
- Accept states are the double circled state (q_3)

Finite automata accept strings as input and the output is either **accept** or **reject**

- Begin at start state and then follow arrows with the corresponding input symbols
- **Accept** if end with accept state and **reject** if otherwise
- Example: 01011 is accept and 0101 is reject

We want to analyze which strings end in accept and which end in reject

- For this finite automaton, all strings that have 11 in them as a substring are accepted
- Language of M_1 = $L(M_1) = \{w \mid w \text{ has substring } 11\} = A$
- We say that M_1 recognizes A

Formal Definitions

Finite Automata

A finite automaton M is a 5-tuple $M = (Q, \Sigma, \delta, q_0, F)$

- Q - set of states (must be finite)
- Σ - input alphabet (must be finite)
- $\delta : Q \times \Sigma \to Q$ - transition function
- $q_0 \in Q$ - start state
- F - set of accept states

For string $w = w_1 w_2 \ldots w_k$ with $w_i \in \Sigma$, we say that M accepts w if there exists a sequence of states $r_0, r_1, r_2, \ldots, r_n \in Q$ such that:

- $r_0 = q_0$
- $r_i = \delta(r_{i-1}, w_i)$ for $1 \leq i \leq n$
- $r_n \in F$

Language

A set of strings (finite or infinite)

- $L(M)$ is the language of M
- $L(M) = \{w | M \text{ accepts } w\}$
- M recognizes $L(M)$

The empty string ε is the string with length 0 and the empty language $\emptyset$ is the set with no strings.

Regular Language

A language is **regular** if some finite automaton recognizes it

- $A = \{w | w \text{ contains substring } 11\}$ is regular (see above automaton)
- $B = \{w | w \text{ has an even number of } 1\text{s}\}$ is regular (can make automaton for practice)
- $C = \{w | w \text{ has equal number of } 0\text{s and } 1\text{s}\}$ is **not** regular (we will prove)

Goal

Understand the regular languages

Regular Operations

Let A and B be languages:

- Union: $A \cup B = \{w | w \in A \text{ or } w \in B\}$
- Concatenation: $A \circ B = AB = \{xy | x \in A \text{ and } y \in B\}$
- Star: $A^* = \{x_1 \dots x_k | \text{ each } x_i \in A \text{ for } k \geq 0\}$

Note: $\varepsilon \in A^*$

Example: $A = \{\text{good}, \text{bad}\}$ and $B = \{\text{boy}, \text{girl}\}$

- $A \cup B = \{\text{good}, \text{bad}, \text{boy}, \text{girl}\}$
- $A \circ B = \{\text{goodboy}, \text{goodboy}, \text{badboy}, \text{badgirl}\}$
- $A^* = \{\varepsilon, \text{good}, \text{bad}, \text{goodgood}, \text{goodbad}, \dots\}$

Regular Expressions

Built from a set of atomic symbols (akin to numbers in arithmetic expressions) that are combined using symbols (arithmetic operations)

Symbols are from Σ (any letter in the alphabet) or are either $\emptyset$ or ε
These symbols are combined using $\cup, \circ, *$

Examples with $\Sigma = \{0, 1\}$:

- $(0 \cup 1)^* = \Sigma^*$ gives all strings over Σ
- $\Sigma^* 1$ gives all strings that end with 1
- $\Sigma^* 11 \Sigma^*$ gives all strings that contain 11 = $L(M_1)$

Goal

Show finite automata are equivalent to regular expressions

Proposition: Closure under Union

If A_1, A_2 are regular languages, so is $A_1 \cup A_2$

Proof:
Let $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ recognize A_1 and $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ recognize A_2

We wish to construct $M = (Q, \Sigma, \delta, q_0, F)$ recognizing $A_1 \cup A_2$

- $Q = Q_1 \times Q_2 = \{(q_1, q_2) | q_1 \in Q_1, q_2 \in Q_2\}$
- $q_0 = (q_1, q_2)$
- $\delta((q, r), a) = (\delta_1(q, a), \delta_2(r, a))$
  - $F = (F_1 \times Q_2) \cup (Q_1 \times F_2)$

Proposition: Closure under Concatenation

If A_1, A_2 are regular languages, so is $A_1 \circ A_2$

Proof:
Let $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ recognize A_1 and $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ recognize A_2

We wish to construct $M = (Q, \Sigma, \delta, q_0, F)$ recognizing $A_1 \circ A_2$. This M would need to figure out if there is some way to cut an input word W such that $W = xy, x \in A_1, y \in A_2$

Might think that we could construct such an M such that we first try to go until M_1 accepts and then we switch to M_2

- This could fail because there might be multiple possible places to cut W such that the first half is accepted by M_1
- We **don't** want to just cut W at the first possible point

To solve this, we introduce a new topic: Nondeterministic Finite Automata