Regular Languages

Strings and Languages:
- $\Sigma$ – an “alphabet,” i.e., a finite set of symbols
- $\Sigma^*$ – set of all finite strings over $\Sigma$
- $\epsilon$ – the empty string
- for $x \in \Sigma^*$, $|x|$ denotes the \textit{length} of $x$
- for $x, y \in \Sigma^*$, $xy$ denotes their \textit{concatenation}
- a \textit{language} is a subset of $\Sigma^*$
Finite automata

Finite automaton (general form):

- A finite automaton (FA) is a directed graph $(Q, E)$, where each edge $e \in E$ has an associated set of labels $\ell(e)$
- Each $\ell(e)$ is a finite subset of $\Sigma^*$
- Vertices are called states
- Edges are called transitions
- There is also a special state $q_0 \in Q$ called the start state, and a set of $F \subseteq Q$ called the set of accept sets
- Notation: $M = (\Sigma, Q, E, \ell, q_0, F)$
The language of $M$:

- Let $x \in \Sigma^*$, and $r, s \in Q$
- Write $M : r \xrightarrow{x} s$ if there is a path 
  $$\langle r_0, r_1, \ldots, r_k \rangle$$
  in the graph $(Q, E)$, starting at $r$, ending at $s$, along with labels 
  $$x_1 \in l(r_0, r_1), \ldots, x_k \in l(r_{k-1}, r_k)$$
  such that $x = x_1 \cdots x_k$

- We say $M$ accepts $x$ if $M : q_0 \xrightarrow{x} s$ for some $s \in F$

- The language of $M$ is 
  $$L(M) := \{ x \in \Sigma^* : M \text{ accepts } x \}$$
Example:

\[ \Sigma = \{0, 1\} \]

\[ L(M) = \{ x \in \{0, 1\}^* : x \text{ contains either 0001 or 11 as a substring} \} \]
Regular languages:

• A language is called *regular* if it is accepted by some FA

Equivalent FA’s:

• FA’s $M_1$ and $M_2$ are called equivalent if $L(M_1) = L(M_2)$

NFA’s:

• A *nondeterministic finite automata (NFA)* is a FA where all labels are either $\varepsilon$ or $\alpha \in \Sigma$

• Any FA can be easily converted to an NFA
Converting a general FA to an NFA:

- Transition from 0, 1 to 0001, 11
- Transition from 0, 1 to 0001
- Transition from 0, 1 to 0
- Transition from 0, 1 to 1
- Transition from 0, 1 to 1
Deterministic FA’s:

- A FA $M$ is called *deterministic* if
  - every label is of the form $a \in \Sigma$,
  - for every $q \in Q$, and $a \in \Sigma$, there exists a unique $r \in Q$ such that $(q, r) \in E$ and $a \in l(q, r)$
  - This defines a function $\delta : Q \times \Sigma \to Q$
    $$\delta(q, a) \mapsto r$$
    called the *transition function*
  - In this case, $M$ is called a *deterministic finite automata (DFA)*, and is traditionally denoted $M = (\Sigma, Q, \delta, q_0, F)$
Theorem:

• Every FA has an equivalent DFA

Proof:

• Let $M = (\Sigma, Q, E, \ell, q_0, F)$
• We may assume labels are $\varepsilon$ or $a \in \Sigma$
• Define the DFA $D = (\Sigma, Q', \delta, q'_0, F')$, where
  
  ○ $Q' := \text{power set of } Q$
  ○ $q'_0 := \{ q \in Q : \quad M : q_0 \xrightarrow{\varepsilon} q \}$
  ○ for each $R \subseteq Q$ and $a \in \Sigma$:
    
    $$\delta(R, a) := \left\{ q \in Q : \quad M : r \xrightarrow{a} q \text{ for some } r \in R \right\}$$
  ○ $F' := \{ R \subseteq Q : R \cap F \neq \emptyset \}$
Theorem:

- regular languages are closed under union, intersection, and complement

Proof:

- Let $M = (\Sigma, Q, \delta, q_0, F)$ and $M' = (\Sigma, Q', \delta' q'_0, F')$ be two DFA’s
- DFA for $\Sigma^* \setminus L(M)$: 
  $$\overline{M} := (\Sigma, Q, \delta, q_0, Q \setminus F)$$
Proof (cont’d):
• Define a DFA with states $Q \times Q'$ and transition function:
  $$((q, q'), a) \mapsto (\delta(q, a), \delta'(q', a))$$
• To get $L(M) \cup L(M')$, choose final states
  $$\{(q, q') : q \in F \text{ or } q' \in F'\}$$
• To get $L(M) \cap L(M')$, choose final states
  $$\{(q, q') : q \in F \text{ and } q' \in F'\}$$
Regular expressions

Recursive definition of regular expressions (RE’s):

- Atoms: $\varepsilon$, $\emptyset$, $a \in \Sigma$
- Recursive rule: if $E_1$ and $E_2$ are regular expressions, then so are $(E_1E_2)$, $(E_1 | E_2)$, and $(E_1)^*$

The language of a RE:

- $L(\varepsilon) = \{ \varepsilon \}$, $L(\emptyset) = \emptyset$, and $L(a) = \{ a \}$
- $L(E_1E_2) = L(E_1)L(E_2)$
  
  $= \{ x_1x_2 : x_1 \in L(E_1), x_2 \in L(E_2) \}$
- $L(E_1 | E_2) = L(E_1) \cup L(E_2)$
- $L(E_1)^* = \bigcup_{k=0}^{\infty} L(E_1)^k$
Example:

\[(0 \mid 1)^*(0001 \mid 11)(0 \mid 1)^*\]

Short-hand notation:

- \(\Sigma\) denotes \((a_1 \mid \cdots \mid a_m)\), where \(\Sigma = \{a_1, \ldots, a_m\}\)
- \(E^+\) denotes \(EE^*\)
- \(E^n\) denotes \(E \cdots E\) (\(n\) times)
Theorem:

- A language is regular if and only if it is the language of some regular expression

Proof:

- $\iff$: given a regular expression, recursively construct an equivalent FA
\[\varepsilon:\] 
\[\emptyset:\] 
\[\alpha \in \Sigma:\] 
\[E_1E_2:\] 
\[E_1^*:\] 
\[E_1 | E_2:\]
Proof (cont’d):

- \(\implies\): given a FA \(M\), build an equivalent RE
- Idea: Floyd-Warshall
- Number states \(1, \ldots, n\)
- Assume initial state is 1 and that \(n\) is the unique final state
- For \(k = 0, \ldots, n\), recursively define RE’s \(E_{ij}^{(k)}\) such that \(L(E_{ij}^{(k)})\) consists of all strings which drive \(M\) from state \(i\) to state \(j\) via intermediate states in \(\{1, \ldots, k\}\)
- \(L(M) = L(E_{1n}^{(n)})\)
Proof (cont’d):

• More simplifying assumptions:
  ◦ for each state \( i \), there is an edge \((i, i)\) and \( \varepsilon \in \ell(i, i) \)
  ◦ if there is no edge \((i, j)\), define \( \ell(i, j) := \emptyset \)

• Recursive construction of \( E_{ij}^{(k)} \):
  ◦ \( E_{ij}^{(0)} = \) regular expression for \( \ell(i, j) \)
  ◦ for \( k > 0 \):
    \[ E_{ij}^{(k)} = E_{ij}^{(k-1)} \mid E_{ik}^{(k-1)} \left( E_{kk}^{(k-1)} \right)^* E_{kj}^{(k-1)} \]
The pumping lemma: a tool to prove that a language is not regular

Theorem (Pumping Lemma):

• Let $A$ be a regular language

• $\exists p \in \mathbb{Z}_{>0} \ \forall s \in A \ with \ |s| \geq p \ \exists x, y, z \in \Sigma^*$:

0. $s = xyz$

1. $|xy| \leq p$

2. $|y| > 0$

3. $xy^kz \in A$ for all $k \in \mathbb{Z}_{k \geq 0}$
Proof:

- Let $M$ be a DFA recognizing $A$
- Let $p = \# \text{ of states in } M$
- Consider any $s \in A$, where $s = s_1 \cdots s_n$ (each $s_i \in \Sigma$) and $n \geq p$
- Since $s \in A$, $s$ drives $M$ through a path $\langle q_0, \ldots, q_n \rangle$, starting at the start state $q_0$ and ending at a final state $q_n$
- Since $|s| \geq p$, this path must contain a cycle
- In particular, there are indices $0 \leq i < j \leq p$ such that $q_i = q_j$
- Set $x := s_1 \cdots s_i$, $y := s_{i+1} \cdots s_j$, $z := s_{j+1} \cdots s_n$
Pumping Lemma Proof:

$q_0 \xrightarrow{x} q_i = q_j \xrightarrow{y} q_n$

Diagram:

- $q_0$ to $x$ to $q_i = q_j$ to $y$ to $q_n$
Example:

- \( A = \{0^n \# 0^n : n \geq 0\} \)
- Use pumping lemma to show \( A \) is not regular
- Proof by contradiction: suppose \( A \) were regular
- Let \( p = \text{pumping length} \)
- Choose \( s = 0^p \# 0^p \)

\[
\begin{array}{cccccc}
0 & 0 & \ldots & 0 & \# & 0 & 0 & \ldots & 0 \\
\hline
x & y & & & & & & & z
\end{array}
\]

Neither \( xz \) nor \( xy^2z \) are in \( A \)
- \( \Rightarrow \Leftarrow \)
Example:

- $B = \{x#y : x, y \in \{0, 1\}^*, x \neq y\}$
- Use closure properties and previous example to show that $B$ is not regular
- Proof by contradiction: assume $B$ is regular
- $\overline{B} \cap (0^* #0^*)$ is also regular
- But $\overline{B} \cap (0^* #0^*) = A$ (in previous example)
- $A$ is not regular
- $\Rightarrow \Leftarrow$