Finite Automata

- We will now introduce the concept of regular langauges.
- Accepted by a computation device known as finite automata
- Other methods to describe regular languages include, regular expressions and a special type of grammars called regular grammars.

Some Examples

1. ON-OFF switch (push button).
Two states, on and off. Each input toggles the state.
2. Recognition of words (lexical analyzers).
3. Many home devices such as VCRs, . . .

Deterministic Finite Automata (DFA)

A deterministic Finite Automaton consists of:

- A finite set of states, often denoted by Q.
- A finite set of input symbols (letters), often denoted by Σ.
- A transition function, that takes two arguments: a state from Q and a letter from Σ, and returns a state.

Transition function is often denoted by δ.

- A starting state, q_0, which is one of the states of Q.
- A set $F \subseteq Q$ of final/accepting states.

$A = (Q, \Sigma, \delta, q_0, F)$.

Deterministic Finite Automata: Example

DFA accepting strings which contain odd number of 1s (here we are taking the alphabet as $\{0, 1\}$).

$A = (\{q_0, q_1\}, \{0, 1\}, \delta, q_0, \{q_1\})$.
$\delta(q_0, 0) = q_0$.
$\delta(q_0, 1) = q_1$.
$\delta(q_1, 0) = q_1$.
$\delta(q_1, 1) = q_0$.

Deterministic Finite Automata: Another Example

DFA accepting strings which contain 00 as a substring.
This is same as accepting the set: $\{w \mid w$ is of form $x00y$ for some strings x and $y\}$ (here we are taking the alphabet as $\{0, 1\}$).

$A = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_2\})$.
$\delta(q_0, 0) = q_1$.
$\delta(q_0, 1) = q_0$.
$\delta(q_1, 0) = q_2$.
$\delta(q_1, 1) = q_0$.
$\delta(q_2, 0) = q_2$.
$\delta(q_2, 1) = q_2$.

Transition Diagrams

- Using circles to denote states, arrows to denote transition.
- Starting state is denoted by using an arrow labeled start.
- Final/accepting states are denoted by using double circles.

Transition Tables

	0	1
q_0	q_1	q_0
q_1	q_2	q_0
q_2	q_2	q_2

Extending transition function to strings

Basis:

$$\hat{\delta}(q, \epsilon) = q.$$

Induction:

$$\hat{\delta}(q, xa) = \delta(\hat{\delta}(q, x), a).$$

Language Accepted by a DFA

$$L(A) = \{w \mid \hat{\delta}(q_0, w) \in F\}$$

Nondeterministic Finite State Automata (NFA)

- $A = (Q, \Sigma, \delta, q_0, F)$.
- The transition function maps the input (state, symbol) to a set of states (a subset of Q)

Example: $L = \{w \mid n\text{-th symbol in } w \text{ from the end is } 1\}$.

Extending transition function to strings for NFA

Basis:

$$\hat{\delta}(q, \epsilon) = \{q\}.$$

Induction:

$$\hat{\delta}(q, xa) = \cup_{p \in \hat{\delta}(q,x)} \delta(p, a).$$

Language Accepted by an NFA

$$L(A) = \{w \mid \hat{\delta}(q_0, w) \cap F \neq \emptyset\}$$

Equivalence of DFA and NFA

Clearly, a DFA is also an NFA.
So, we only need to show that a language accepted by NFA is also accepted by some DFA.
Suppose NFA $A = (Q, \Sigma, \delta, q_0, F)$ is given.
Define DFA $A_D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$ as follows.
$Q_D = \{S \mid S \subseteq Q\}$.
$F_D = \{S \mid S \subseteq Q \text{ and } S \cap F \neq \emptyset\}$.
$\delta_D(S, a) = \cup_{q \in S}\, \delta(q, a)$.

Equivalence of DFA and NFA

Claim: For any string w, $\hat{\delta_D}(\{q_0\}, w) = \hat{\delta}(q_0, w)$.
Proof: By induction on length of w.
Base Case: $w = \epsilon$.
In this case clearly, $\hat{\delta_D}(\{q_0\}, \epsilon) = \{q_0\} = \hat{\delta}(q_0, \epsilon)$.
Induction Step:

$$
\begin{aligned}
\hat{\delta_D}(\{q_0\}, wa) &= \delta_D(\hat{\delta_D}(\{q_0\}, w), a) \\
&= \bigcup_{q \in \hat{\delta_D}(\{q_0\}, w)} \delta(q, a) \\
&= \bigcup_{q \in \hat{\delta}(q_0, w)} \delta(q, a) \\
&= \hat{\delta}(q_0, wa)
\end{aligned}
$$

QED Claim.

Thus, $\hat{\delta_D}(\{q_0\}, w) \in F_D$ iff $\hat{\delta}(q_0, w) \cap F \neq \emptyset$.
It follows that DFA A_D accepts w iff NFA A accepts w.

Dead and Unreachable States in a DFA

Dead State is a state from which one cannot reach a final state, whatever the sequence of inputs.
Formally, q is a dead state if, for all $w \in \Sigma^*$, $\hat{\delta}(q, w) \notin F$.
Unreachable states are states which cannot be reached from starting state, whatever the sequence of inputs.
Formally, q is an unreachable state if, for all $w \in \Sigma^*$, $\hat{\delta}(q_0, w) \neq q$.

NFA with ϵ transitions

$A = (Q, \Sigma, \delta, q_0, F)$.
δ maps $Q \times (\Sigma \cup \{\epsilon\})$ to subsets of Q.

Example: An integer is $(+, -, \epsilon)$ followed by digits.

ϵ Closures

Definition of $Eclose(q)$

1. $q \in Eclose(q)$.
2. If state $p \in Eclose(q)$, then each state in $\delta(p, \epsilon)$, is in $Eclose(q)$.
3. We iterate step 2, until no more changes are done to $Eclose(q)$.

Definition of extended transition function $\hat{\delta}$

$\hat{\delta}(q, \epsilon) = Eclose(q)$.
$\hat{\delta}(q, wa) = S$, where S is defined as follows:
Let $R = \cup_{p \in \hat{\delta}(q,w)} \delta(p, a)$
Then, $S = \cup_{p \in R} Eclose(p)$.
$L(A) = \{w \mid \hat{\delta}(q_0, w) \cap F \neq \emptyset\}$.

Equivalence of ϵ-NFA and DFA.

Tutorial problem.

Idea:

Suppose $A = (Q, \Sigma, \delta, q_0, F)$.

Form $A_D = (Q_D, \Sigma, \delta_D, q_D, F_D)$ as follows:

Essentially let $Q_D = \{S \mid S \subseteq Q\}$.

$q_D = Eclose(q_0)$.

Transition: $\delta_D(S, a) = \cup_{p \in S} \hat{\delta}(p, a)$

$F_D = \{S \mid S \cap F \neq \emptyset\}$.

Regular Expressions

Basis: The constant ϵ and $\emptyset$ are regular expressions, and $L(\epsilon) = \{\epsilon\}$ and $L(\emptyset) = \emptyset$.

If a is any symbol in Σ, then a is a regular expression, and $L(a) = \{a\}$.

Induction: If r_1 and r_2 are regular expressions, then so are:

(a) $r_1 + r_2$

$L(r_1 + r_2) = L(r_1) \cup L(r_2)$.

(b) $r_1 \cdot r_2$

$L(r_1 \cdot r_2) = \{xy \mid x \in L(r_1) \text{ and } y \in L(r_2)\}$

(c) r_1^*

$L(r_1^*) = \{x_1 x_2 \ldots x_k \mid \text{for } 1 \leq i \leq k,\ x_i \in L(r_1)\}$.

Note: in above k can be 0, and thus, $\epsilon \in L(r_1^*)$.

(d) (r_1)

$L((r_1)) = L(r_1)$.

Precedence

$*$ has highest precedence

$\cdot$ has next highest precedence

$+$ has least precedence

Association

DFA to Regular Expressions

Let the DFA $A = (Q, \Sigma, \delta, q_{start}, F)$.
We assume $Q = \{1, 2, \ldots, n\}$ for some n, and $q_{start} = 1$.
$R^k_{i,j}$ denote the regular expression for the set of strings which can be formed by going from state i to state j using intermediate states numbered $\leq k$.
Base Case: Definition of $R^0_{i,j}$.
If $i \neq j$: $R^0_{i,j} = a_1 + a_2 \ldots + a_m$, where $a_1, a_2, \ldots, a_m$ are all the symbols such that $\delta(i, a_r) = j$ (If no such symbols, then $R^0_{i,j} = \emptyset$).
If $i = j$: $R^0_{i,i} = \epsilon + a_1 + a_2 \ldots + a_m$, where $a_1, a_2, \ldots, a_m$ are all the symbols such that $\delta(i, a_r) = i$
Induction Case:
$R^{k+1}_{i,j} = R^k_{i,j} + R^k_{i,k+1}(R^k_{k+1,k+1})^* R^k_{k+1,j}$.
Regular Expression for language $L(A)$ is given by: $\Sigma_{j \in F} R^n_{1,j}$.

Regular Expressions to ϵ-NFA

We will show how to inductively construct a ϵ-NFA for every regular expression which additionally satisfies:
a) It has only one final state.
b) There is no transition into the starting state.
c) There is no transition out of the final state.
d) The starting and final states are different.
Base Cases:
(i) $\emptyset$:
(ii) ϵ:
(iii) a:
Induction Case:
(iv) $r_1 + r_2$
(v) $r_1 \cdot r_2$
(vi) r_1^*

Some Properties of Regular Expressions

- $M + N = N + M$
- $L(M + N) = LM + LN$
- $L + L = L$
- $(L^*)^* = L^*$.
- $\emptyset^* = \epsilon$
- $\epsilon^* = \epsilon$
- $L^+ = LL^* = L^*L$
- $L^* = \epsilon + L^+$
- $(L + M)^* = (L^*M^*)^*$