Example NFA and Equivalent DFA

Example

$L = \{w \text{ over } \{a_1,a_2,a_3\} \mid w \text{ does not contain } a_i \text{ for some } i\}$
Th. The class of reg. lang. are closed under $\bigcup$

Let $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$, let $N_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ be NFA’s.

Construct NFA $N$ such that $L(N) = L(N_1) \bigcup L(N_2)$

Proof Idea:

Proof: We define NFA $N$ as follows....

Regular languages are closed under Union

$N = (Q, \Sigma, \delta, q_0, F)$

1. $Q = \{q_0\} \bigcup Q_1 \bigcup Q_2$ (q0 new, Q1 and Q2 disjoint)

2. $q_0$ is start

3. $F = F_1 \bigcup F_2$

4.

$$\delta(q, a) = \begin{cases} 
\delta_1(q, a) & \text{if } q \text{ in } Q_1 \\
\delta_2(q, a) & \text{if } q \text{ in } Q_2 \\
\{q_1, q_2\} & \text{if } q = q_0 \text{ and } a = \varepsilon \\
\phi & \text{if } q = q_0 \text{ and } a = \varepsilon 
\end{cases}$$
Regular languages are closed under Star

Proof Idea:

Proof: Define $N = (Q, \Sigma, \delta, q_0, F)$ from $N_1$. 
1. $Q = \{q_0\} \cup Q_1$  $q_0$ is new
2. $q_0$ is start
3. $F = \{q_0\} \cup F_1$
4. 

$\delta(q, a) = \begin{cases} 
\delta_1(q, a) & \text{if } q \text{ in } Q_1, q \text{ not in } F_1 \\
\delta_1(q, a) & \text{if } q \text{ in } F_1, a \text{ not } = \epsilon \\
\delta_1(q, a) \cup \{q_1\} & \text{if } q \text{ in } F_1 \text{ and } a = \epsilon \\
\{q_1\} & \text{if } q = q_0, a = \epsilon \\
\phi & \text{if } q = q_0, a \text{ not } = \epsilon 
\end{cases}$
NFAs that accept (apple)* or (orange)*

NFA that accepts (apple, orange)*
Regular languages are closed under Concatenation

Proof Idea: Start out in $N_1$, and in any final state of $N_1$, can switch to $N_2$; accept if wind up in final state in $N_2$.

Proof: Define $N = (Q, \Sigma, \delta, q_1, F)$ from $N_1$ and $N_2$
1. $Q = Q_1 \cup Q_2$
2. $q_1$ is start (same as start of $N_1$)
3. $F = F_2$
4. 
   \[
   \delta(q, a) = \begin{cases} 
   \delta_1(q, a) & \text{if } q \text{ in } Q_1, q \not\text{ in } F_1 \\
   \delta_1(q, a) & \text{if } q \text{ in } F_1, a \not\text{ not } = \varepsilon \\
   \delta_1(q, a) \cup \{q_2\} & \text{if } q \text{ in } F_1 \text{ and } a = \varepsilon \\
   \delta_2(q, a) & \text{if } q \text{ in } Q_2 
   \end{cases}
   \]

NFA that accepts strings (apple)*orange
Regular Expressions & Languages

Regular expressions are a formalism for describing patterns in strings *(used in tools like Emacs, Perl,...)*

Language of a regular expression is the set of strings described

\[
\begin{align*}
\text{a} & \quad \text{a in } \Sigma & \{\text{a}\} \\
\varepsilon & & \{\varepsilon\} \\
\phi & & \phi \\
(R_1 \cup R_2) & (R_1, R_2 \text{ reg. exp}) & \text{L}(R_1) \cup \text{L}(R_2) \\
(R_1 \circ R_2) & & \text{L}(R_1) \circ \text{L}(R_2) \\
(R_1)^* & & (\text{L}(R_1))^*
\end{align*}
\]

What kind of definition is this?

---

Regular Expressions, examples over \{a,b\}

\[
\begin{align*}
(a \cup b) & \quad (a)^* & \quad (a \cup b)^* \\
\text{All these parentheses are clumsy--can omit but} & \\
\text{must know what order to carry out operations} & \\
\text{a} & \cup \quad \text{b}^* & \quad ??? \\
\text{Precedence of operations:} & \quad * \\
& \circ \\
& \text{(associate to left)} \quad \cup \\
\text{but ( ) must always be followed if included!} & \\
\text{Shorthand: } \Sigma \text{ for alphabet, ex a } \cup \quad \text{b not ab} & \\
\text{Omit } \circ \text{ where understood} & \\
(\Sigma \Sigma \Sigma)^* & \Sigma^* \quad \text{bb} \Sigma^* \quad a^*b & \quad a^* \varepsilon
\end{align*}
\]
Constructing Regular Expressions

EX. Write a regular expression to describe IP addresses:
    a sequence of 4 numbers, in the range 0-255, separated by .’s

Let D = [0-9] (same as
        [0,...,9] {0-9} {0,...,9} )

1. Write a regular expression for numbers in the range 0-255 using D.
2. Use that expression 4 times, separated by dots, to get the answer.