Q1. Prove that:

"Let L be a language accepted by a Non-deterministic Finite Automata (NFA). Then there exists a Deterministic Finite Automata (DFA) that accepts L."

OR

Prove equivalence of NFA and DFA.

Ans: Let $L(D)$ be the language accepted by DFA and $L(N)$ be that of NFA. If $D = (Q_D, \Sigma, \delta_D, \{q_0\}, f_D)$ is the DFA constructed from NFA $N = (Q_N, \Sigma, \delta_N, q_0, f_N)$ by subset construction, then $L(D) = L(N)$. This is what we have to prove. i.e, $\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$.

Basis: Let $|w| = 0$; i.e, $w = \epsilon$. By the basis definitions of $\hat{\delta}$ for DFA and NFA, both $\hat{\delta}_D(\{q_0\}, \epsilon)$ and $\hat{\delta}_N(q_0, \epsilon)$ are $\{q_0\}$.

[$\hat{\delta}$ function returns a set of states from Q_N, but $\hat{\delta}_D$ interprets this set as one of these states of Q_D (which is the power set of Q_N), while $\hat{\delta}_N$ interprets this set as a subset of Q_N]

Induction: Let w be of length $n+1$ and assume the statement for length n. Break w up as $w = xa$, where a is the final symbol of w. By the inductive hypothesis $\hat{\delta}_D(\{q_0\}, x) = \hat{\delta}_N(q_0, x)$. Let both these sets of N's state be $\{p_1, p_2, \ldots, p_k\}$

The inductive part of the definition of $\hat{\delta}$ for NFA's tell us

$$\hat{\delta}_N(q_0, w) = \bigcup_{i=1}^{k} \delta_N(p_i, a) \quad \text{——— (1)}$$

The subset construction, on the other hand, tells us that

$$\delta_D(\{p_1, p_2 \ldots p_k\}, a) = \bigcup_{i=1}^{k} \delta_N(p_i, a) \quad \text{——— (2)}$$

Using the fact, $\hat{\delta}_D(\{q_0\}, x) = \{p_1, p_2 \ldots p_k\}$ in the inductive part of the definition of $\hat{\delta}$ of DFA's:

$$\hat{\delta}_D(\{q_0\}, w) = \delta_D(\hat{\delta}_D(\{q_0\}, x), a)$$

$$= \delta_D(\{p_1, p_2 \cdots p_k\}, a)$$

$$\hat{\delta}_D(\{q_0\}, w) = \bigcup_{i=1}^{k} \delta_N(p_i, a) \quad \text{——— (3)}$$

From equation (1) and (3), $\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$

We observe that D and N both accept w iff $\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$

$\Rightarrow L(D) = L(N)$.

Hence proved.

Q2. Prove that: "If L is a language accepted by an ordinary NFA, then there exist an equivalent ε-NFA that also accepts L".

Ans: Let the NFA be $N = (Q, \Sigma, \delta, q_0, F)$. Construct an ε-NFA $E = (Q, \Sigma, \delta_E, q_0, E)$ by defining

- for every $q \in Q$ and every input symbol $a \in \Sigma$:

$$\delta_E(q, a) = \delta(q, a).$$

- for every $q \in Q$: $\delta_E(q, \varepsilon) = \phi$

So E has exactly the same states, start state and accepting states as N; and it has the same transitions on real symbols, while it has no ε-moves.

We show that for every state $q \in Q$ and every string $w \in \Sigma^*$,

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

Prove by induction on $|w|$.

Base: If $w = \epsilon$, then

$$\hat{\delta}_E(q, \varepsilon) = E - \varepsilon\text{-closure}(q).$$

But since $\delta_E(q, \varepsilon) = \phi$ for all q, the ε-closure of q is just q. Thus $\hat{\delta}_E(q, \varepsilon) = q = \hat{\delta}(q, \varepsilon)$.

Induction: Suppose the equality holds for all strings of length n. Let w be a string of length $n+1$, say $w = xa$ with $x \in \Sigma^*$ and $a \in \Sigma$. Then by definition of the extended transition function for NFAs and ε-NFAs.

$$\hat{\delta}_E(q, xa) = \bigcup_{p \in \hat{\delta}_E(q,x)} \delta_E(p, a)$$

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

This completes the induction. Hence

$$\hat{\delta}_E(q_0, w) = \hat{\delta}(q_0, w)$$

$\forall w$, so w is accepted by E iff it is accepted by N

$\therefore L(E) = L(N)$

Hence proved.

Q3. Prove the closure properties of regular languages.

Ans: Let L and M be regular languages.

(a) Closure under Union:

Since L and M are regular, they have regular expressions, say

Let $L = L(E)$ and $M = L(F)$

Then $L \cup M = L(E+F)$ by the definition of + operator.

∴ For any regular languages L and M, $L \cup M$ is regular

(b) Closure under Intersection:

Let L be recognized by the DFA

$$A_L = (Q_L, \Sigma, \delta_L, q_L, F_L)$$

and M by the DFA

$$A_M = (Q_M, \Sigma, \delta_M, q_M, F_M)$$

We assume that the alphabets of both automata are same; i.e., Σ is the union of the alphabets of

L and M if they are different.
We also assume both automata are deterministic.
We shall construct an automaton A that simulates both A_L and A_M. If A_L goes from state p to state s on reading a and A_M goes from state q to state t on reading a, then $A_{L\cap M}$ will go from state (p, q) to state (s, t) on reading a. The start state of A is the pair of start states of A_L and A_M.

∴ We want to accept iff both automata accept, we select as accepting states of A all pairs (p, q) such that p is an accepting state of A_L and q is an accepting state of A_M.

Formally,

$$A_{L\cap M} = (Q_L \times Q_M, \Sigma, \delta((q_L, q_M), F_L \times F_M)$$

where

$$\delta((p,q),a) = (\delta_L(p,a), \delta_M(p,a))$$

By induction on |ω| it is possible to prove that

$$\hat{\delta}_{L\cap M}((q_L, q_M), \omega) = (\hat{\delta}_L(q_L, \omega), \hat{\delta}_M(q_M, \omega))$$

But A accepts ω iff $\hat{\delta}_{L\cap M}((q_L, q_M), \omega)$ is a pair of accepting states. i.e, $\hat{\delta}_L(q_L, \omega)$ must be in F_L and δ_M must be in F_M

∴ ω is accepted by A iff ω is accepted by A_L and by A_M.
⇒ A accepts the intersection of L and M.

∴

> If L and M are regular languages, then so is $L \cap M$.

(c) Closure under complementation:

Let L be recognized by a DFA

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

Then $\bar{L} = L(B)$ where B is the DFA $B = (Q, \Sigma, \delta, q_0, Q \setminus F)$

i.e, B is exactly like A, but the accepting states of A have become the nonaccepting states of B & vice-versa.
Then w is in L(B) iff $\hat{\delta}(q_0, w)$ is in Q|F, which occurs iff w is not in L(A)

∴ If L is a regular language over alphabet Σ then $\bar{L} = \Sigma^* \setminus L$ is also regular

(d) Closure under Difference:

Observe that L\M (Difference) = $L \cap \bar{M}$. We know that regular languages are closed under complementation and intersection.

∴ If L and M are regular languages, then so is L\M.

(e) Closure under reversal:

Assume L is defined by a regular expression E.
We show that there is another regular expression E^R such that $L(E^R) = (L(E))^R$.
i.e. language of E^R is reversal of language of E.

Basis: If E is ε, ϕ of a then $E^R = E$
Induction: There are 3 cases based on the form of E.
1) $E = F + G$. Then $E^R = F^R + G^R$
2) $E = F \cdot G$ then $E^R = G^R \cdot F^R$
3) $E = F^*$ Then $E^R = (F^R)^*$

Justifications for these:
1) The reversal of the union of two languages is obtained by computing the reversals of the two languages & taking the union of those languages. ∴ $E^R = F^R + G^R$.
2) ~~If L(F) = {01, 11} and L(G) = {00, 10}~~
~~then L(FG) = {000, 0110, 1100, 1110}. The reversal of latter is~~

In general, if a word $w \in L(E)$ is the concatenation of $w_1 \in L(F)$ and $w_2 \in L(G)$, then $w^R = w_2^R \cdot w_1^R$

3) Any string $w \in L(E)$ can be written as $w_1 w_2 \ldots w_n$, where each w_i is in $L(F)$. But $w^R = w_n^R \ldots w_2^R w_1^R$ and each w_i^R is in $L(F^R)$ so w^R is in $L((F^R)^*)$

Conversely, any string in $L((F^R)^*)$ is of the form $w_1 w_2 \ldots w_n$ where each w_i is the reversal of a string in $L(F)$. The reversal of this string is in $L(F^*)$ which is in $L(F)$.

We have shown that a string is in $L(E)$ iff its reversal is in $L((F^R)^*)$

∴ If L is a regular language, then so is L^R.

(e) Closure under Concatenation:

Let E_1 and E_2 be the regular expressions accepting L_1 and L_2.

Let $E = E_1 \cdot E_2$ that accepts $L_1 \cdot L_2$ i.e

$$L(E_1 \cdot E_2) = L(E_1) \cdot L(E_2)$$

[Proof of correctness: trivial by definition of regular expressions]

∴ If L_1 and L_2 are regular languages $L_1 \cdot L_2$ is regular since there is a regular expression $E_1 \cdot E_2$ accepting this language.

⇒ If L_1 and L_2 are regular languages, so is $L_1 \cdot L_2$.

(f) The closure (star) of a regular language is regular.

Let L be a regular language represented by regular expression R. R^* is a regular expression whose language is L^*. ∴ L^* can be represented with a regular expression, it is also a regular language.

(g) Closure under Homomorphism:

Let $L = L(R)$ for some regular expression R.

We claim that $h(R)$ defines the language $h(L)$

In general, if E is the regular expression with symbols in Σ, let $h(E)$ be the expression obtained by replacing each symbol a of Σ by $h(a)$.

The proof is a structural induction that says whenever we take a subexpression E of R and apply h to it to get h(E), the language h(E) is the same we get if we apply h to the language L(E).

Formally, $L(h(E)) = h(L(E))$

Basis: If E is ϕ or ε, then $h(E) = E$ and $L(h(E)) = L(E) = h(L(E))$

If E is a then $L(E) = \{a\}$ and

$L(h(E)) = L(h(a)) = \{h(a)\} = h(L(E))$

Induction: There are 3 cases:

Case 1: $E = F + G$

$h(E) = h(F+G) = h(F) + h(G)$

We know that, $L(E) = L(F) \cup L(G)$

$L(h(E)) = L(h(F+G)) = L(h(F) + h(G)) = L(h(F)) \cup L(h(G))$, by the definition of what + means in regular expressions.

$h(L(E)) = h(L(F) \cup L(G)) = h(L(F)) \cup h(L(G))$ because h is applied to a language by application to each of its strings individually.

By the induction hypothesis,

$L(h(F)) = h(L(F))$ and $L(h(G)) = h(L(G))$

Then $L(h(F)) \cup L(h(G)) = h(L(F)) \cup h(L(G))$

Case 2: $E = F \cdot G$

$h(E) = h(F \cdot G) = h(F) \cdot h(G)$. We also know that, $L(E) = L(F) \cdot L(G)$

$L(h(E)) = L(h(F \cdot G)) = L(h(F) \cdot h(G)) = L(h(F)) \cdot L(h(G))$

$h(L(E)) = h(L(F \cdot G)) = h(L(F)) \cdot h(L(G))$

By induction hypothesis, $L(h(F)) = h(L(F))$ and $L(h(G)) = h(L(G))$

Then $L(h(F)) \cdot L(h(G)) = h(L(F)) \cdot h(L(G))$

Case 3: $E = F^*$

$h(E) = h(F^*)$

$L(h(E)) = L(h(F^*)) = L(h(F))^* = L(h(F))^*$

$h(L(E)) = h(L(F^*)) = h(L(F))^*$

By induction hypothesis, $L(h(F)) = h(L(F))$

Then $L(h(F))^* = h(L(F))^*$

⇒ If L is a regular language over alphabet Σ and h is a homomorphism on Σ, then h(L) is also regular.

(4) Closure under Inverse Homomorphisms:

Let L be the language of the DFA

$$A = (Q, \Theta, \delta, q_0, F)$$

We construct from A and homomorphism from alphabet Σ to alphabet $\Theta(h)$, a DFA for $h^{-1}(L)$. This automaton uses the states of A but translates the input symbols according to h before deciding on the next state.

Formally,

$$B = (Q, \Sigma, \gamma, q_0, F)$$

where transition function γ is constructed by the rule

$$\gamma(q, a) = \hat{\delta}(q, h(a))$$

i.e, the transition B makes on input a is the result of the sequence of transitions that A makes on string of symbols h(a).

By an induction on |w| it is possible to show that

$$\hat{\gamma}(q_0, w) = \hat{\delta}(q_0, h(w))$$

∴ the accepting states of A & B are same, B accepts w iff A accepts h(w).

Put another way, B accepts exactly those strings w that are in $h^{-1}(L)$.

⇒ If h is a homomorphism from alphabet Σ to alphabet Θ and L is a regular language over alphabet Θ, then $h^{-1}(L)$ is also a regular language.

Q4. Prove that :

"Every regular language is ultimately periodic."

Ans: To prove that every regular language is ultimately periodic, we can use the pumping lemma for regular languages. An ultimate periodic language is one for which there exist integers N and $p \geq 0$ such that for any string w in the language with length $|w| \geq N$, there is a decomposition $w = xyz$ where $|xy| \leq N$, $|y| \geq 1$, and for all $i \geq 0$ the string $xy^i z$ is also in the language.

Let L be a regular language. By the definition of a regular language, there exists a DFA, say $M = (Q, \Sigma, \delta, q_0, F)$ that accepts L. Let $k = |Q|$ be the no. of states in the DFA.

Now, we can apply the pumping lemma for regular languages to L. The pumping lemma states that there exists a pumping length p such that for any string $w \in L$ with $|w| \geq p$, there is a decomposition $w = xyz$ satisfying the following conditions:

* $|xy| \leq p$
* $|y| \geq 1$
* for all $i \geq 0$, the string $xy^i z$ is also in L.

This is precisely the definition of an ultimately periodic language. We have an integer N (the pumping length) and period p (the length of the pumped substring).

Existence of N: We have chosen $N = k$ (the no. of states in the DFA).

Existence of P: For a given string $w \in L$ with $|w| \geq N$ the pumping lemma guarantees the existence of a substring y with $|y| \geq 1$. The length of this specific y (let's call it $p_w = |y|$) acts as the period for this particular string w. The definition of an ultimately periodic language requires that for every string w with $|w| \geq N$ such a period exists. The pumping lemma guarantees this for every such string.

Hence proved.