**Non-deterministic Finite State Acceptor Automata (NFA)**

Defined by a 5-tuple

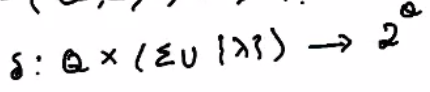
M = (Q, Sigma, Delta, q0, F) (same as DFA)

Differences b/w DFA and NFA:

For a **DFA**, we had



However, the difference is in the definition of delta for an NFA



Where 2^Q is the power set of Q

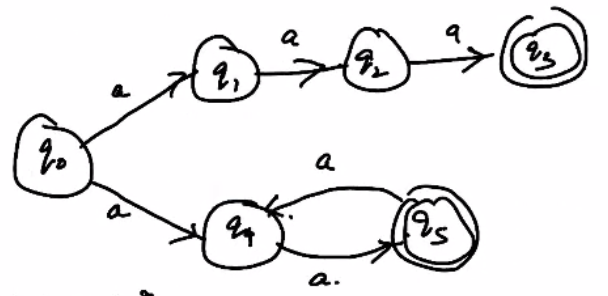
Further, we allow Lambda as a second arg for Delta (i.e. we can transition without consuming any input)

We also allow Delta(qi, a) to be empty or not defined.

Similar to any acceptor automaton, an input that causes the automaton to transition to the final state, the input must be a part of the language of the automaton.

Eg

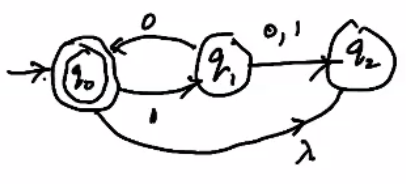
1)



This is an NFA

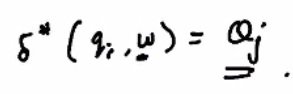
Language, L = {aa, aaa, aaaa ...}

2)



Language, L = {lambda, 10, 1010, (10)^n}

For an NFA

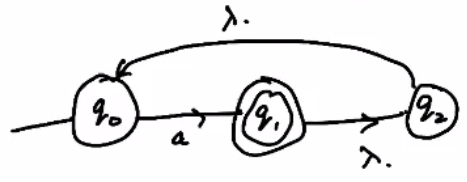


Where Delta\* is the extended transition function. Qj is the set of all possible states the automaton may be in starting from qi and having read input w.

If there are no transitions possible after reading string w, it is said to be a dead configuration.

A state qn will be in the extended transition function Delta\* iff there is a walk in the transition graph from qi to qn labelled w.

Eg



Here

Delta\*(q1, a) = {q2, q0, q1} because we can move to q2 and q0 without any input string.

Delta\*(q2, lambda) = {q0}

Delta\*(q1, lambda) = {q2, q0}

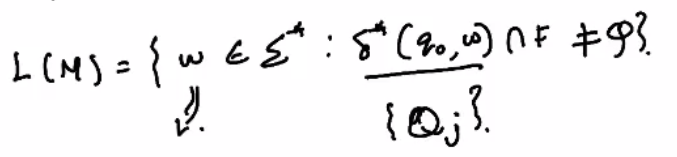
Delta\*(q2, aa) = {q1, q2, q0}

**Language of an NFA**

Let F be set of final states. Let R be the set of reachable states after reading a string w. If

F (intersection) R is non-empty then the string w is in the language of the NFA.

For an NFA defined as M = (Q, Sigma, Delta, q0, F), the Language L(M) is defined as the set of all accepted strings w such that:



**Uses of NFA**

Used to model any kind of non-mechanical device, because no mechanical device can exhibit non-determinism by design.

There may be certain programs, AI systems, random number generators, etc are some of the examples of when we may have non-deterministic behaviour.

Some optimization techniques such as backtracking can be modeled by NFAs.

NFAs can also be used to model some complex languages which have rules that make some choices.

There are some theoretical results that are easily established for NFA than DFA.

**NOTE:**

Any language that can be defined by an NFA can be defined by a DFA. There exists an equivalent DFA for every NFA.