Nondeterministic Finite Automata

- A "nondeterministic" finite automaton (NFA) has the power to be in several states at once.
- Each NFA accepts a language that is also accepted by some DFA. That is, NFA's accept exactly the regular languages, just as DFA's do.
- However, NFA's are often more succinct (compact/condensed) and easier to design than DFA's.
- Like the DFA, an NFA has a finite set of states, a finite set of input symbols, one start/initial state, a transition function and a set of accepting states.
- The difference between the DFA and the NFA is, for the NFA, δ is a function that takes a state and input symbol as arguments (like the DFA's transition function), but returns a set of zero, one or more states (rather than exactly one state, as the DFA must.)
- While a DFA has exactly one arc out of each state for each input symbol, an NFA has no such constraint.

Example: Design a nondeterministic finite automaton that accepts all and only the strings of 0's and 1's that end in 01 and show the states the NFA is in during the processing of input sequence 00101.

- *[Book] Section 2.3.1, Example 2.6*

❖ **Formal Definition of NFA:**

An NFA is a quintuple (5-tuple), that is, a system which consists of 5 elements. We write,

$$A = (Q, \Sigma, \delta, q_0, F), \text{ where}$$

- **Q**: finite nonempty set of states;
- **Σ** (capital sigma): finite nonempty set of input symbols, input alphabet;
- **q_0**: initial/start state, $q_0 \in Q$;
- **F**: set of final (or accepting) states, $F \subseteq Q$.
- **δ** (small delta): transition function that takes as arguments a state and an input symbol and returns a subset of Q.

Now, the *example 1* above can be specified formally as **$(\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_2\})$** where the transition function δ is given by the transition table below:

	0	**1**
$\rightarrow$ **q_0**	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\emptyset$	$\{q_2\}$
***q_2**	$\emptyset$	$\emptyset$

- Notice that, the only difference between the transition table for DFA and NFA is that each entry in the table for NFA is a set, even if the set is a *Singleton*.

- When there is no transition at all from a given state on a given input symbol, the entry in the table is $\emptyset$, the empty set.

❖ **Extended Transition Function:**

As for DFA's, we need to extend the transition function δ of an NFA to a function $\hat{\delta}$ that takes a state ***q*** and a string of input symbols ***w***, and returns the set of states that the NFA is in if it starts in state ***q*** and processes the string ***w***. Formally, we define $\hat{\delta}$ for an NFA's transition function δ by:

Basis: $\hat{\delta}(q, \varepsilon) = \{q\}$. That is, without reading any input symbols, we are only in the state we began in.

Induction: Suppose ***w*** is of the form ***w = xa***, where ***a*** is the final symbol of ***w*** and ***x*** is the rest of ***w***. Also suppose that $\hat{\delta}$ $(q, x) = \{p_1, p_2, \ldots, p_k\}$. Let

$$\bigcup_{i=1}^{k} \delta(p_i, a) = \{r_1, r_2, \ldots, r_m\}$$

Then, $\hat{\delta}$ $(q, w) = \{r_1, r_2, \ldots, r_m\}$.

Example: Let's use $\hat{\delta}$ to describe the processing of input **00101** by the NFA in the example above. A summary of the steps is:

1. $\hat{\delta}(q_0, \epsilon) = \{q_0\}$.
2. $\hat{\delta}(q_0, 0) = \delta(q_0, 0) = \{q_0, q_1\}$.
3. $\hat{\delta}(q_0, 00) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$.
4. $\hat{\delta}(q_0, 001) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$.
5. $\hat{\delta}(q_0, 0010) = \delta(q_0, 0) \cup \delta(q_2, 0) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$.
6. $\hat{\delta}(q_0, 00101) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$.

❖ **The Language of an NFA:**

An NFA accepts a string ***w*** if it is possible to make any sequence of choices of next state, while reading the characters of ***w***, and go from the start state to any accepting state. The fact that other choices using the input symbols of w lead to nonaccepting state, or do not lead to any state at all (i.e., the sequence of states "dies"), does not prevent ***w*** from being accepted by the NFA as a whole. Formally, if **$A = (Q, \Sigma, \delta, q_0, F)$** is an NFA, then

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

That is, **L(A)** is the set of strings ***w*** in **Σ^*** such that **$\hat{\delta}$ (q, w)** contains at least one accepting state.