Lecture #3: DFA Design and Verification — Part One

Key Concepts

When designing a deterministic finite automaton for a given language one is given the following:

- An ***alphabet*** Σ.
- The ***language*** $L \subseteq \Sigma^\star$ of the DFA that is to be designed.

The objective is to produce the following:

- A deterministic finite automaton

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

  whose alphabet is the alphabet, Σ, that has been given.

- A ***proof*** that $L = L(M)$ — or enough information so that it is clear that this proof could be written.

DFA design begins by answering the following question: What does a DFA, with the given language, need to remember about the part of the string that has been seen so far?

- This is used to describe a set of *cases* that might apply, for the part of the string that has been seen so far.
- This is used to develop a collection of subsets $S_0, S_1, S_2, \ldots, S_{n-1}$ of subsets of $\Sigma^\star$ — with each subset corresponding to one of the cases that has been identified.[1]

Additional steps (and "sanity checks") include the following.

1. Confirm that only *finitely* many cases — with corresponding sets $S_0, S_1, S_2, \ldots, S_{n-1}$ (for a fixed positive integer n) — have been identified. These will correspond to states $q_0, q_1, q_2, \ldots, q_{n-1}$ in the deterministic finite automaton that is, eventually, designed.

[1] These subsets should be *nonempty*: Any of the sets that are empty can be removed from this collection, to simplify the process and the deterministic finite automaton that is designed using it.

2. Confirm that every string $\omega \in \Sigma^\star$ belongs to *exactly one* of the subsets $S_0, S_1, S_2, \ldots, S_{n-1}$ of $\Sigma^\star$.

   Re-order these states, if necessary, so that $\lambda \in S_0$. Then q_0 can be used as the *start state*.

   The deterministic finite automaton, to be designed, will then include states $q_0, q_1, \ldots, q_{n-1}$.

   The ***desired relationship*** between sets $S_0, S_1, S_2, \ldots, S_{n-1}$ and states $q_0, q_1, q_2, \ldots, q_{n-1}$ can now be described as follows: For every integer i such that $0 \leq i \leq n-1$,

   $$S_i = \{\omega \in \Sigma^\star \mid \delta^\star(q_0, \omega) = q_i\}. \tag{1}$$

3. Confirm that, for every integer i such that $0 \leq i \leq n-1$, either $S_i \subseteq L$ or $S_i \cap L = \emptyset$.

   The set, F, of accepting states can now be defined as follows. For every integer i such that $0 \leq i \leq n-1$, include q_i in F if $S_i \subseteq L$, and do not include q_i otherwise. That is, define F so that

   $$F = \{q_i \mid 0 \leq i \leq n-1 \text{ and } S_i \subseteq L\}.$$

4. Confirm that, for every integer i such that $0 \leq i \leq n-1$, and for every symbol $\sigma \in \Sigma$, there exists an integer j such that $0 \leq j \leq n-1$ such that

   $$\{\omega \cdot \sigma \mid \omega \in S_i\} \subseteq S_j \tag{2}$$

   — in which case, one can set $\delta(q_i, \sigma)$ to be q_j.

Note that, if all of the above conditions have been verified, and q_0, F and the transition function $\delta : Q \times \Sigma \to Q$ have been defined as described above, then a deterministic finite automaton

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

has been completely described. The next lecture will concern what to do when these steps *cannot* be completed (generally because it is not possible to establish one or more of the conditions at line (2)) — and how a proof of the correctness of the deterministic finite automaton can be completed, if they can be.