Towards a DFA minimization algorithm

This notes assumes the material in the notes on Myhill-Nerode theorem.

- Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA with k states. Let $L(M)$ be the language recognized by M.
- Define an equivalence relation on Q as follows:

$$p \equiv_Q q \Longleftrightarrow \forall z \in \Sigma^*, \ (\hat{\delta}(p, z) \in F \Leftrightarrow \hat{\delta}(q, z) \in F).$$

- The two equivalence relations $\equiv_Q$ and $\equiv_{L(M)}$ are related as follows:
- **Claim 4:** Let $x, y \in \Sigma^*$ such that $\hat{\delta}(q_0, x) = p$ and $\hat{\delta}(q_0, y) = q$. Then, $x \equiv_{L(M)} y$ if and only if $p \equiv_Q q$.

  Proof:

  – Consider

  $$\forall z \in \Sigma^*, \hat{\delta}(q_0, xz) = \hat{\delta}(\hat{\delta}(q_0, x), z) = \hat{\delta}(p, z).$$

  Similarly,

  $$\forall z \in \Sigma^*, \hat{\delta}(q_0, yz) = \hat{\delta}(q, z).$$

  – Let $p \equiv_Q q$. Therefore,

  $$\forall z \in \Sigma^*, (\hat{\delta}(p, z) \in F \Leftrightarrow \hat{\delta}(q, z) \in F).$$

  It follows that

  $$\forall z \in \Sigma^*, \ (xz \in L(M) \Leftrightarrow yz \in L(M)).$$

  That is, $x \equiv_{L(M)} y$.

  – Let $x \equiv_{L(M)} y$. Therefore,

  $$\forall z \in \Sigma^*, \ (xz \in L(M) \Leftrightarrow yz \in L(M)).$$

  Then,

  $$\forall z \in \Sigma^*, \ (\hat{\delta}(q_0, xz) \in F \Leftrightarrow \hat{\delta}(q_0, yz) \in F).$$

  Since $\hat{\delta}(q_0, x) = p$ and $\hat{\delta}(q_0, y) = q$, we have:

  $$\forall z \in \Sigma^*, (\hat{\delta}(p, z) \in F \Leftrightarrow \hat{\delta}(q, z) \in F).$$

  That is, $p \equiv_Q q$.

- **Corollaries:**

  1. $p \not\equiv_Q q \Longleftrightarrow \exists \ z \in \Sigma^*, \$ one of $\{\hat{\delta}(p, z), \ \hat{\delta}(q, z)\} \in F$ and the other $\notin F$.
  2. The number of equivalence classes with respect to $\equiv_Q$ is i$ndex(L(M))$.

Minimum equivalent DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA. Let $[p]_Q$ denote the equivalence class $\{q \in Q \mid q \equiv_M p\}$. Then, a minimum DFA $M' = (Q', \Sigma, \delta', q_0{}', F')$ that is equivalent to M is defined as follows:

- Q' is the set of equivalence classes that partition Q.
- The start state $q_0{}' = [q_0]_Q$.
- $\delta'([p]_Q, a) = [q]_Q$ if $\delta(p, a) = q$.
- $F' = \{[q]_Q \mid q \in F\}$.

A DFA minimization algorithm

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA that recognizes L. A DFA $M' = (Q', \Sigma, \delta', q_0{}', F')$ with minimum number of states that is equivalent to M can be constructed if all the equivalence classes with respect to $\equiv_Q$ are constructed.

The following algorithm constructs the equivalence classes with respect to $\equiv_Q$ using an upper-triangular matrix T whose rows and columns are indexed by the states Q of M.

Let $Q = \{q_0, q_1, \cdots, q_{r-1}\}$. Then, T has r rows and r columns. The goal is to identify pairs of states (q_i, q_j) for which there is a string x such that $\hat{\delta}(q_i, x)$ is in F and $\hat{\delta}(q_i, x)$ is not in F or vice-versa. Such a string x is said to *distinguish* the pair (q_i, q_j). Initially, all pairs (q_i, q_j) such that one of $\{q_i, q_j\}$ is in F and the other is not in F get marked with 0 since the empty string distinguishes the two states. Having identified all pairs of states that are distinguishable by strings of length k, we consider pairs of states that are still unmarked to see if they can be distinguished by strings of length $k + 1$. The process stops when no more new pairs of states are marked. At the end of the marking proces, the pairs of states (q_i, q_j) that do not get marked belong to the same equivalence class.

- Remove all states that are not accessible from the start state:
  - Let $A_0 = \{q_0\}$.
  - Let $j = 0$.
  - Repeat Until $A_{j+1} = A_j$ for some j:
    * Construct

$$A_{j+1} = A_j \cup \{q \mid \text{for some } p \in A_j \text{ and } a \in \Sigma,\ \delta(p, a) = q\}$$

  - A_j is the set of all states accessible from the start state.

- Initialize the matrix T whose rows and columns are indexed by the states Q of M.
  - For all pairs (q_i, q_j):
    * Unmark the entry $T(q_i, q_j)$.
    * Mark the entry with 0 if one of q_i, q_j is in F and the other is not.

- $k = 0$.

- Repeat until no more new entries are marked:
  - For each unmarked entry $T(q_i, q_j)$, mark it with $k + 1$ if for some $a \in \Sigma$, the entry $T(\delta(q_i, a), \delta(q_j, a))$ is marked with k.
  - $k = k + 1$.