tphols-2011

By xingyuan

February 8, 2011

Contents

theory *Myhill*
imports *Myhill-2*
begin

1 Preliminaries

1.1 Finite automata and Myhill-Nerode theorem

A *determinisitc finite automata (DFA)* M is a 5-tuple $(Q, \Sigma, \delta, s, F)$, where:

1. Q is a finite set of *states*, also denoted Q_M.
2. Σ is a finite set of *alphabets*, also denoted Σ_M.
3. δ is a *transition function* of type $Q \times \Sigma \Rightarrow Q$ (a total function), also denoted δ_M.
4. $s \in Q$ is a state called *initial state*, also denoted s_M.
5. $F \subseteq Q$ is a set of states named *accepting states*, also denoted F_M.

Therefore, we have $M = (Q_M, \Sigma_M, \delta_M, s_M, F_M)$. Every DFA M can be interpreted as a function assigning states to strings, denoted $\hat{\delta}_M$, the definition of which is as the following:

$$\begin{aligned} \hat{\delta}_M([]) &\equiv s_M \\ \hat{\delta}_M(xa) &\equiv \delta_M(\hat{\delta}_M(x), a) \end{aligned} \tag{1}$$

A string x is said to be *accepted* (or *recognized*) by a DFA M if $\hat{\delta}_M(x) \in F_M$. The language recoginzed by DFA M, denoted $L(M)$, is defined as:

$$L(M) \equiv \{x \mid \hat{\delta}_M(x) \in F_M\} \tag{2}$$

The standard way of specifying a laugage $\mathcal{L}$ as *regular* is by stipulating that: $\mathcal{L} = L(M)$ for some DFA M.

For any DFA M, the DFA obtained by changing initial state to another $p \in Q_M$ is denoted M_p, which is defined as:

$$M_p \equiv (Q_M, \Sigma_M, \delta_M, p, F_M) \tag{3}$$

Two states $p, q \in Q_M$ are said to be *equivalent*, denoted $p \approx_M q$, iff.

$$L(M_p) = L(M_q) \tag{4}$$

It is obvious that $\approx_M$ is an equivalent relation over Q_M. and the partition induced by $\approx_M$ has $|Q_M|$ equivalent classes. By overloading $\approx_M$, an equivalent relation over strings can be defined:

$$x \approx_M y \equiv \hat{\delta}_M(x) \approx_M \hat{\delta}_M(y) \tag{5}$$

It can be proved that the the partition induced by $\approx_M$ also has $|Q_M|$ equivalent classes. It is also easy to show that: if $x \approx_M y$, then $x \approx_{L(M)} y$, and this means $\approx_M$ is a more refined equivalent relation than $\approx_{L(M)}$. Since partition induced by $\approx_M$ is finite, the one induced by $\approx_{L(M)}$ must also be finite, and this is one of the two directions of Myhill-Nerode theorem:

Lemma 1 (Myhill-Nerode theorem, Direction two)**.** *If a language $\mathcal{L}$ is regular (i.e. $\mathcal{L} = L(M)$ for some DFA M), then the partition induced by $\approx_{\mathcal{L}}$ is finite.*

The other direction is:

Lemma 2 (Myhill-Nerode theorem, Direction one)**.** *If the partition induced by $\approx_{\mathcal{L}}$ is finite, then $\mathcal{L}$ is regular (i.e. $\mathcal{L} = L(M)$ for some DFA M).*

The M we are seeking when prove lemma **??** can be constructed out of $\approx_{\mathcal{L}}$, denoted $M_{\mathcal{L}}$ and defined as the following:

$$Q_{M_{\mathcal{L}}} \equiv \{[\![x]\!]_{\approx_{\mathcal{L}}} \mid x \in \Sigma^*\} \tag{6a}$$
$$\Sigma_{M_{\mathcal{L}}} \equiv \Sigma_M \tag{6b}$$
$$\delta_{M_{\mathcal{L}}} \equiv (\lambda([\![x]\!]_{\approx_{\mathcal{L}}}, a).[\![xa]\!]_{\approx_{\mathcal{L}}}) \tag{6c}$$
$$s_{M_{\mathcal{L}}} \equiv [\![[]]\!]_{\approx_{\mathcal{L}}} \tag{6d}$$
$$F_{M_{\mathcal{L}}} \equiv \{[\![x]\!]_{\approx_{\mathcal{L}}} \mid x \in \mathcal{L}\} \tag{6e}$$

It can be proved that $Q_{M_{\mathcal{L}}}$ is indeed finite and $\mathcal{L} = L(M_{\mathcal{L}})$, so lemma 2 holds. It can also be proved that $M_{\mathcal{L}}$ is the minimal DFA (therefore unique) which recoginzes $\mathcal{L}$.

1.2 The objective and the underlying intuition

It is now obvious from section 1.1 that Myhill-Nerode theorem can be established easily when *reglar languages* are defined as ones recognized by finite automata. Under the context where the use of finite automata is forbiden, the situation is quite different. The theorem now has to be expressed as:

Theorem 1 (Myhill-Nerode theorem, Regular expression version)**.** *A language $\mathcal{L}$ is regular (i.e. $\mathcal{L} = L(e)$ for some* regular expression *e) iff. the partition induced by $\approx_{\mathcal{L}}$ is finite.*

The proof of this version consists of two directions (if the use of automata are not allowed):

Direction one: generating a regular expression e out of the finite partition induced by $\approx_{\mathcal{L}}$, such that $\mathcal{L} = L(e)$.

Direction two: showing the finiteness of the partition induced by $\approx_{\mathcal{L}}$, under the assmption that $\mathcal{L}$ is recognized by some regular expression e (i.e. $\mathcal{L} = L(e)$).

The development of these two directions consititutes the body of this paper.

2 Direction *regular language* $\Rightarrow$*finite partition*

Although not used explicitly, the notion of finite autotmata and its relationship with language partition, as outlined in section 1.1, still servers as important intuitive guides in the development of this paper. For example, *Direction one* follows the *Brzozowski algebraic method* used to convert finite autotmata to regular expressions, under the intuition that every partition member $[\![x]\!]_{\approx_{\mathcal{L}}}$ is a state in the DFA $M_{\mathcal{L}}$ constructed to prove lemma 2 of section 1.1.

The basic idea of Brzozowski method is to set aside an unknown for every DFA state and describe the state-trasition relationship by charateristic equations. By solving the equational system such obtained, regular expressions characterizing DFA states are obtained. There are choices of how DFA states can be characterized. The first is to characterize a DFA state by the set of striings leading from the state in question into accepting states. The other choice is to characterize a DFA state by the set of strings leading from initial state into the state in question. For the first choice, the lauguage recognized by a DFA can be characterized by the regular expression characterizing initial state, while in the second choice, the languaged of the DFA can be characterized by the summation of regular expressions of all accepting states.

end

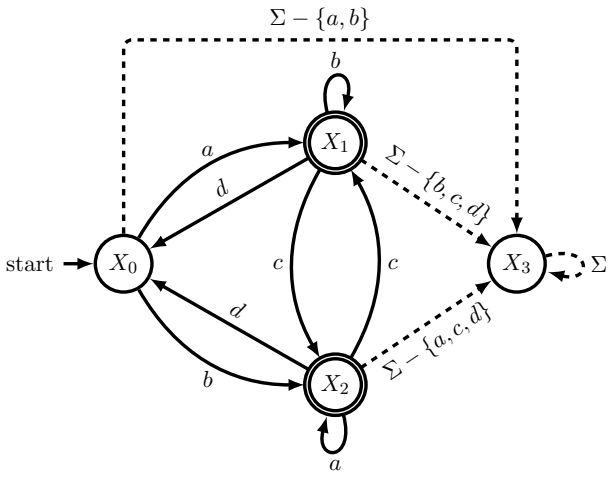

Figure 1: The relationship between automata and finite partition