tphols-2011

By xingyuan

February 5, 2011

Contents

theory *Myhill*
  imports *Myhill-2*
begin

1 Direction *regular language ⇒finite partition*

The intuition behind our treatment is still automata. Taking the automaton in Fig.1(a) as an example, like any automaton, it represents the vehicle used to recognize a certain regular language. For any given string, the process starts from the left most and proceed character by character to the right, driving the state transtion of automaton starting from the initial state (the one marked by an short arrow). There could be three outcomes:

1. The process finally reaches the end of the string and the automaton is at an accepting state, in which case the string is considered a member of the language. For the automaton in Fig.1(a), a, ab, abb, abc, $abbcc$, b, baa are such kind of strings.

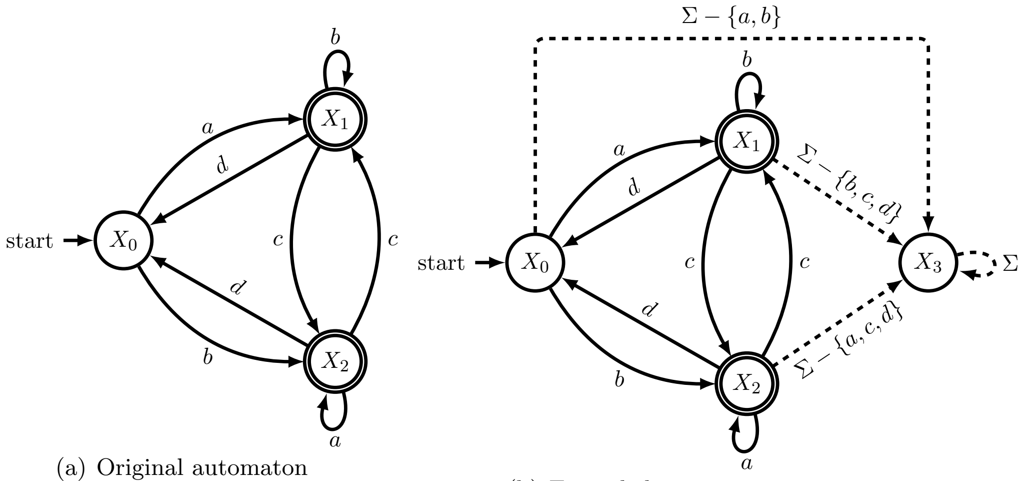
2. The process finally reaches the end of the string but the automaton is at an non-accepting state, in which case, the string is considered a non-member of the language. For the automaton in Fig.1(a), ad, $abbd$, $adbd$, $bdabbccd$ are such kind of strings.

3. The process get stuck at the middle of the string, in which case, the string is considered a non-member of the lauguage. For the automaton in Fig.1(a), c, acb, $bbacd$, $aaccd$ are such kind of strings.

To avoid the situation 3 above, we can augment a normal automaton with a "absorbing state", as the state X_3 in Fig.1(b). In an auguments automaton, the process of strings never get stuck: whenever a string is recognized as not belonging to the language, the augmented automaton is transfered into the “absorbing state” and kept there until the process reaches the end of the string, in which case, the string is rejected by situation 2 above.

Given a language *Lang* and a string x, the equivalent class $\approx$*Lang* ‘‘ $\{x\}$ corresponds to the state reached by processing x with the augmented automaton. Since $\approx$*Lang* ‘‘ $\{x\}$ is defined for every x, it corresponds to the fact that the processing of x will never get stuck.

The most acquainted way to define a regular language *Lang* is by giving an automaton which recorgizes every string in *Lang*. Fig.1(a) gives such a automaton, which is esstially a graph where every finite path leading from initial state to one finial state corresponds to a string in *Lang*.



(a) Original automaton

(b) Extended automaton

Figure 1: The relationship between automata and finite partition

end