Programming Languages: Lecture 4 String Matching using a Finite-State Automaton

Rishabh Dhiman

11 January 2022

1 String Matching Automaton

Assume an alphabet A with A^* being the set of all finite strings from A. We use $y \succeq z$ to denote that y is a suffix of z.

Definition 1 (Suffix Function)**.** For a string pattern $P[1 \dots p]$, $\sigma : A^* \to [0 \dots p]$. $\sigma(x)$ is the length of the longest prefix of P which is a suffix of x, ie, $\sigma(x) = \max\{k \mid P_k \succeq\}$.

Definition 2 (String Matching Automaton)**.** There is a string matching deterministic automaton $D(P)$ for every pattern P,

The string matching automaton $D(P) = (Q, A, 0, p, \delta)$ corresponding to $P[1 \dots p]$, with

1. $Q = \{0, \dots, p\}$,
2. alphabet A,
3. 0 is the start state,
4. p is the lone accepting state and
5. $\delta : Q \times A \to A$ is the transition function defined by $\delta(q, a) = \sigma(P_q a)$ for each state $q \in Q$ and $a \in A$.

Is this called suffix automaton? It's not.

Lemma 1 (Suffix-Function inequality)**.** *For any $x \in A^*$ and $a \in A$, $\sigma(xa) \leq \sigma(x) + 1$.*

Lemma 2 (Suffix-Function Recursion)**.** *For any $x \in A^*$ and $a \in A$, if $q = \sigma(x)$ then $\sigma(xa) = \sigma(P[1 \dots q]a)$.*

Theorem 1. *For any text $T[1 \dots t]$, $\varphi(T[1 \dots i]) = \sigma(T[1 \dots i])$ for all $0 \leq i \leq t$.*

2 Modules in SML

- signature $\equiv$ type
- structure $\equiv$ value
- functor $\equiv$ function