Assignment 12

Post Date: 31 Jan 2014  Due Date: 07 Feb 2014, 08:00 (AM)
You are permitted and encouraged to work in groups of two.

Problem 1: Boyer-Moore  5 Points

Compute the bad character function and the good suffix function for the alphabet $\Sigma = \{0, 1, 2, 3\}$ and the pattern $P = 0101101201$.

Problem 2: Transition Function  5 Points

Let $\delta$ be the transition function of a pattern $P[1, \ldots, m]$.

Show that $\delta(q, a) = \delta(\pi(q), a)$ for any $a \in \Sigma$ and $0 < q \leq m$ with $q = m$ or $P[q+1] \neq a$.

Problem 3: Wildcards  6 Points

Now, a pattern can contain also wildcards *. A wildcard * can stand for arbitrarily many (also zero) characters.

(a) Modify Algorithm Naive-Transition-Function such that it computes a string-matching automaton of a pattern that may contain wildcards.

Consider the pattern $P = aba * bab$ and the input alphabet $\Sigma = \{a, b, c\}$.

(b) Give the string-matching automaton for the pattern $P$ computed by your algorithm from (a). Does this automaton find all occurrences in a text of pattern $P$?

Problem 4: Weak Good Suffix Rule  4 Points

Consider the algorithm of Boyer and Moore with the weak good suffix rule in which we do not require that the mismatched character should differ, i.e. for a pattern $P[1, \ldots, m]$ and for $0 \leq j < m$ we define $S_P(j)$ to be the length of the largest prefix of $P$ such that

$P[j + 1, \ldots, m]$ is a suffix of $P[1, \ldots, S_P(j)]$ or $P[1, \ldots, S_P(j)]$ is a suffix of $P[j + 1, \ldots, m]$.

If the rightmost mismatch occurs at position $j$ of $P$ we augment the shift by $m - S_P(j)$.

Show that now the worst-case run-time of the algorithm of Boyer and Moore is not linear even if the pattern does not occur in the text. Don’t consider the bad character rule.