6.851 Advanced Data Structures (Spring'07)

Prof. Erik Demaine   TA: Oren Weimann

Problem 4 – Solution

Pattern matching via suffix arrays.

(a) Its easy to show that $lcp(i,j) = \min\{LCP[i], LCP[i+1], \ldots, LCP[j-1]\}$

(b) To find a pattern p in SA, we find the largest interval $[L,R]$ such that p is the prefix of all elements in $SA[L], \ldots, SA[R]$. We start with $L = 1$ and $R = n$, and in our binary search, we keep track of the number of characters k of p that we have matched so far. Suppose we have matched k characters and the binary search is at position L (position R is symmetric). Let M be the midpoint between L and R. We compare k to $lcp(L,M)$.

- if $k < lcp(L,M)$ we narrow the search to the interval $[M,R]$.
- if $k > lcp(L,M)$ we narrow the search to the interval $[L,M]$.
- if $k = lcp(L,M)$, only then do we have to read additional characters from p and compare with $SA[M]$ until a mismatch is found (which determined the next direction of the binary search).

We read each character of p only once, so the total time is $O(m + \lg n)$.

(c) For non constant alphabets, the $O(n)$ space required by the suffix array is better than the $O(n|\Sigma|)$ space required for the suffix tree. Also, the additional $O(\lg n)$ time used in searching a suffix array is negligible if $m = \Omega(\lg n)$.