Tutorial Week 7

1 Week 7

Definition 1. *The suffix table* SUF *of a string y of length n gives us the order of the suffixes of y, such that* $\text{SUF}[i] < \text{SUF}[i+1]$. *The* LCP *array corresponding to y gives us the lengths of the longest common prefixes of consecutive elements in the suffix table:*

$$\text{LCP}[i] = |\text{lcp}(y[\text{SUF}[i-1]..n-1], y[\text{SUF}[i]..n-1])|.$$

Together the two form the suffix array. If by $\text{First}_k(u)$ we denote the prefix of length k of u, when $|u| \geq k$, and u, otherwise, then by $R_k[i]$ we denote the rank of $\text{First}_k(y[i..n-1])$ inside the ordered list of all $\text{First}_k(u)$, where u is a non-empty suffix of y.

Exercise 1. *Consider the following list of sequences: ababa, abcacabb, abcacababc, and abacabacab. For each of them, using the prefix doubling technique find out the corresponding ranks. Next, find out their corresponding suffix array consisting of the suffix table* SUF *and the corresponding* LCP *table, respectively.*
    What is the complexity of the algorithm?

Exercise 2. *Consider now the same list of sequences: ababa, abcacabb, abcacababc, and abacabacab. For each of them, using the skew algorithm find out the corresponding ranks, considering the t and s sequences. Next, find out their corresponding suffix array consisting of the suffix table* SUF *and the corresponding* LCP *table (obviously, these are the same as for the previous exercise).*
    What is the complexity of the algorithm?