

The Inverse Suffix Array

Suffix Array: $SA[i]$ is the starting pos of the i -th smallest suffix of input S

Inverse Suffix Array \overline{SA} : $SA[i] = j \Rightarrow \overline{SA}[j] = i$,

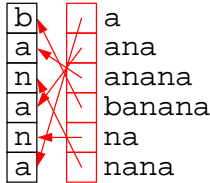
i.e., $\overline{SA}[j]$ is the **rank** of the j -th suffix $S[j : n]$

Example:

$S = [b, a, n, a, n, a]$

$SA = [5, 3, 1, 0, 4, 2]$

$\overline{SA} = [3, 2, 5, 1, 4, 0]$



Longest Common Prefix Length

$lcp(S, S') := \max \{i : S[0 : i] = S'[0 : i]\}$ for any two strings S and S' .

We focus on

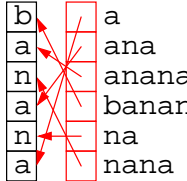
$$lcp[i] := \begin{cases} 0 & \text{if } i = 0 \\ lcp(S[SA[i-1] : n], S[SA[i] : n]) & \text{if } i > 0 \end{cases}$$

Example:

$S = [b, a, n, a, n, a]$

$SA = [5, 3, 1, 0, 4, 2]$

$lcp = [0, 1, 3, 0, 0, 2]$



A Continuity Lemma

Lemma 1. $lcp[\overline{SA}[i]] \geq lcp[\overline{SA}[i-1]] - 1$

Proof. Obvious if $lcp[\overline{SA}[i-1]] \leq 1$.

Otherwise, suppose $lcp[\overline{SA}[i-1]] = \ell > 1$,

i.e., $\exists S[j : n) : S[j : j + \ell) = S[i-1 : i + \ell - 1)$. Hence

$S[j+1 : j + \ell) = S[i : i + \ell - 1)$

...

□

Computing lcp-Information

$lcp[0] := [0, \dots, 0]$

$\ell := 0$

// lower bound for lcp

foreach $i \in [0 : n)$ **do**

// find $lcp[\overline{SA}[i]]$

if $\overline{SA}[i] \geq 1$ **then**

$j := SA[\overline{SA}[i] - 1]$ // $S[j : n)$ precedes $S[i : n)$ in the suffix array

while $S[i + \ell) = S[j + \ell)$ **do** $\ell++$

$lcp[\overline{SA}[i]] := \ell$

$\ell := \max \{ \ell - 1, 0 \}$

$\ell \leq n$

n iterations of main loop

total decrease of ℓ is $\leq n$

time $\mathcal{O}(n)$

Pattern Matching Using Suffix Trees

Function $match(P[0 : m) : String; T = (V, E) : SuffixTree) :$

$i := 0$ // first unmatched character

$v := T.root$

while $i < m$ **do**

if $\exists e = (v, u) \in E : e.label = P[i : i + |e.label|)$ **then**

$i := i + |e.label|$

$v := u$

else return \emptyset

report all leaves rooted at v as occurrences

Time $\mathcal{O}(m + \#occurrences)$