Technische Universität Kaiserslautern

Department of Computer Science
Algorithms & Complexity Group

Issue Date: 28.10.2013
Version: 2013-10-22 12:34

Exercise Sheet 1 for Computational Biology (Part 1), WS 13/14

Hand In: *Until Monday, 11.11.2013, 10:00 am, email to* `wild@cs...` *or in lecture.*

Problem 1

2 points

A *(simple) suffix tree*[1] $B_{T'}$ for $T' = T \cdot \$$ with text $T \in \Sigma^{n-1}$ and $\$ \notin \Sigma$ is a rooted directed tree with the following properties:

(1) $B_{T'}$ has n leaves labeled 1 to n.

(2) Every edge in $B_{T'}$ is labeled with a symbol in $\Sigma \cup \{\$\}$.

(3) All edges leaving one node are labeled differently.

(4) The path from the root r of $B_{T'}$ to leaf 'i' is labeled with $T_{i,n}$.

Show that the method for constructing a (simple) suffix tree given in lecture[2] is correct, i. e., it outputs the unique (simple) suffix tree for T'.

Problem 2

2 + 3 points

a) Give an infinite family (T_n) of texts with $T_n \in \{a, b\}^{n-1}\$$ such that the number of nodes t_n of the corresponding simple suffix trees B_{T_n} is quadratic in n, i. e., $t_n = \Theta(n^2)$.

b) Give a second infinite family (T_n) of texts, for which the compact suffix trees IB_{T_n} have worst case size, i. e., the number of nodes of IB_{T_n} is maximal among all compact suffix trees for texts of the same size $|T_n| = n$. What is the worst case number of nodes?

---

[1] as defined on page 49 of the German lecture notes

[2] see page 49 of the German lecture notes

Problem 3

3 points

Design a linear time algorithm to compute the set of all *maximal repeats* of a text T along the lines given on pages 61ff of the German lecture notes.

More precisely, for every maximal repeat P of $T \in \Sigma^n$, your algorithm is supposed to output one pair of indices (i, j) and its length $m = |P|$, such that P is found at positions i and j in T:

$$T_{i,i+m-1} = T_{j,j+m-1} = P \quad \wedge \quad T_{i-1} \neq T_{j-1} \quad \wedge \quad T_{i+m} \neq T_{j+m}$$

where we set $T_0 := \$ =: T_{n+1}$ for $\$ \notin \Sigma$. The running time should be in $\mathcal{O}(n)$.

Problem 4

4 points

For two strings S and T over alphabet Σ, we define the *overlap* of S and T as

$$ov(S,T) \; := \; \max\Big\{|y| \;\Big|\; y \in \Sigma^\star \;\wedge\; \exists\, x, z \in \Sigma^+ : S = xy \;\wedge\; T = yz\Big\} \tag{1}$$

Design an algorithm to compute *all* pairwise overlaps of a given set of strings $\mathcal{T} = \{T^{(1)}, \dots, T^{(m)}\}$ over Σ, i. e. for all $i, j \in [m]$, compute $ov(T^{(i)}, T^{(j)})$. The running time of your algorithm should be in $\mathcal{O}(n \cdot m)$, where $n := \sum_{i=1}^{m} |T_i|$ is the total length of all strings in $\mathcal{T}$.