

Homework 3 (CptS 471/571)

Spring 2013

Due Date: April 1, 2013

Total Points: 53

- Submit the cover sheet (course website has it) and submit it along with your homework, regardless of whether you collaborate or not on the designated questions.
 - For algorithm design problems, present your algorithm in the form of a pseudocode, just like we have been discussing in class. Try to use figures to help illustrate your solution more clearly. Besides other factors such as algorithmic simplicity and asymptotic complexity, I will be looking out for your main approach idea and critical details required to show that your solution works. (For the latter, if you think it is necessary to augment a solution with a proof of correctness then do so.)
 - Also state the runtime complexity of your solution for all algorithm design problems.
 - Try to reuse the result of any algorithm we discussed in class. For example, if your solution uses LCA queries (using the Bender-Farach algorithm) as a subroutine, then simply state that in your answer without going over the internal details of the Bender-Farach method.
-

1. (5 points)

Using suffix trees, give an algorithm to find a longest common substring shared among three input strings: s_1 of length n_1 , s_2 of length n_2 and s_3 of length n_3 .

2. (5 points)

A non-empty string α is called a *minimal unique substring* of s if and only if it satisfies:

- (i) α occurs exactly once in s (uniqueness),
- (ii) all proper prefixes of α occur at least twice in s (minimality), and
- (iii) $|\alpha| \geq l$ for some constant l .

Give an optimal algorithm to enumerate all minimal unique substrings of s .

3. (5 points)

Redundant sequence identification: Given a set of k DNA sequences, $S = \{s_1, s_2, \dots, s_k\}$, give an optimal algorithm to identify all sequences that are completely contained in (i.e., substrings of) at least one other sequence in S .

4. (8 points)— collaborative

Let $S = \{s_1, s_2, \dots, s_k\}$ denote a set of k genomes. The problem of *fingerprinting* is the task of identifying a shortest possible substring α_i from each string s_i such that α_i is unique to s_i — i.e., no other genome in the set S has α_i . Such an α_i will be called a *fingerprint* of s_i . (Note that it is OK for α_i to be present more than once within s_i .)

Give an algorithm to enumerate a fingerprint for each input genome, if one exists. Assume that no two input genomes are identical.

5. (10 points) — collaborative

A string s is said to be *periodic* with a *period* α , if s is α^k for some $k \geq 2$. (Note that α^k is the string formed by concatenating α k times.) A DNA sequence s is called a *tandem repeat* if it is periodic. Given a DNA sequence s , determine if it is periodic, and if so, the values for α and k . Note that there could be more than one period for a periodic string. In such a case, you need to report the shortest period.

6. (10 points) — collaborative

A non-empty string β is called a *repeat prefix* of a string s if $\beta\beta$ is a prefix of s . Give a linear time algorithm to find the longest repeat prefix of s .

Hint: Think of using lca queries.

7. (10 points) — collaborative

Given strings s_1 and s_2 of lengths m and n respectively, a *minimum cover* of s_1 by s_2 is a decomposition $s_1 = w_1w_2 \dots w_k$, where each w_i is a non-empty substring of s_2 and k is minimized. Eg., given $s_1 = accgtatct$ and $s_2 = cgtactcatc$, there are several covers of s_1 by s_2 possible, two of which are: (i) *cover*₁: $s_1 = w_1w_2w_3w_4$ (where $w_1 = ac$, $w_2 = cgt$, $w_3 = atc$, $w_4 = t$), and (ii) *cover*₂: $s_1 = w_1w_2w_3w_4w_5$ (where $w_1 = ac$, $w_2 = c$, $w_3 = gt$, $w_4 = atc$, $w_5 = t$). However, only *cover*₁ is a minimal cover. Give an algorithm to compute a minimum cover (if one exists) in $O(m + n)$ time and space. If a minimum cover does not exist your algorithm should state so and terminate within the same time and space bounds. Give a brief justification of why you think your algorithm is correct — meaning, how it guarantees finding the minimal cover (if one exists).