58093 String Processing Algorithms (Autumn 2012)

Separate Exam, 20 September 2013 at 16-20 Lecturer: Juha Kärkkäinen

Please write on each sheet: your name, student number or identity number, signature, course name, exam date and sheet number. You can answer in English, Finnish or Swedish.

1. [*4+4+4 points*] Each of the following pairs of concepts are somehow connected. Describe the main connecting factors or commonalities as well as the main separating factors or differences.

   (a) Horspool algorithm and BNDM algorithm.

   (b) Edit distance computation and approximate string matching.

   (c) LCA (Lowest Common Ancestor) preprocessing ja RMQ (Range Minimum Query) preprocessing.

   A few lines for each part is sufficient.

2. [*12 points*] A string A is a *subsequence* of a string B if A can be obtained by deleting characters from B. For example, `abc` is a subsequence of `abadc` but it is not a subsequence of `acadb`.

   Let P be a pattern and T a text. Describe an efficient algorithm for finding the length of the shortest factor of T that contains P as a subsequence. For example, if $P =$ `abc` and $T =$ `cabadcabbddc`, then the answer is 5 as `abc` is a subsequence of $X =$ `abadc`, and X is shortest of such substrings of T. What is the time complexity of your algorithm in terms of the lengths of P and T?

3. [*4+8 points*]

   (a) What is the lcp-comparison technique? Describe the main principles.

   (b) Give two examples of algorithms or data structures that use the lcp-comparison technique. Describe the role of the lcp-comparison technique in the algorithms.

4. [*6+6 points*] Let $\{\mathtt{a}, \mathtt{b}\}$ be the alphabet. For any integers $k \geq 1$ and $m \geq k$, describe a set of 2^k strings of length m such that the number of nodes in the (uncompact) trie for the set is

   (a) as large as possible

   (b) as small as possible.

   What is the number of nodes in each case? Note that all the strings in the set must be different.

5. [*12 points*] Let S and T be strings over the integer alphabet $[0..\sigma)$. Describe an algorithm that finds the shortest string that occurs in S but does not occur in T. The time complexity should be $\mathcal{O}(|S| + |T| + \sigma)$.