58093 String Processing Algorithms (Autumn 2010)

Course Exam, 16 December 2010 at 9–12 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. [*3+3+3+3 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) Morris–Pratt algorithm and Aho–Corasick algorithm.

   (b) Horspool algorithm and BNDM algorithm.

   (c) Suffix array and Burrows–Wheeler transform.

   (d) Compact trie and suffix tree.

   A few lines for each part is sufficient.

2. [*6+6 points*] Consider a variant of the edit distance that allows an unlimited number of *insertions* at the end of the string without a cost. In other words, the variant edit distance is

   $$ed'(A, B) = \min\{ed(A, C) \mid C \text{ is a prefix of } B\} ,$$

   where $ed(\cdot, \cdot)$ is the standard edit distance.

   (a) Describe an algorithm that, given strings A and B, computes $ed'(A, B)$.

   (b) Describe an algorithm that, given strings A and B and an integer k, finds out whether B has a *suffix* B' such that $ed'(A, B') \leq k$.

   The time complexity should be $\mathcal{O}(|A||B|)$ in both cases. You may assume that any algorithms described on the lectures are known but any modifications to them should be described precisely.

3. [*5+8 points*]

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

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

4. [*13 points*] Let $T[0..n)$ be a string over an integer alphabet $\Sigma = [0..\sigma)$. Describe an algorithm that finds the *shortest* string over the alphabet Σ that does *not* occur in T. The time complexity should be $\mathcal{O}(n)$.