58093 String Processing Algorithms (Autumn 2012)
Practice problems

Please give feedback by filling the feedback form at `https://ilmo.cs.helsinki.fi/kurssit/servlet/Valinta?kieli=en`

The first two problems below are related to the last week of lectures and the other four represent typical exam questions. You are not expected to return answers to the problems in any form. Solutions will be posted to the course home page on Tuesday, December 11.

1. Let $L = \texttt{rttrraa\$ii}$ be the Burrows–Wheeler transform of a text T.
   (a) What is T?
   (b) Simulate backward search on T for the pattern $P = \texttt{ari}$.
2. Let $T = \texttt{senselessness\$}$.
   (a) Give the $C = C_1 \cup C_2$ suffixes of the DC3 algorithm for T.
   (b) Give the $C = C^*$ suffixes of the SAIS algorithm for T.
3. 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) Shift–Or algorithm and Myers' bitparallel algorithm.
   (b) LSD radix sort and MSD radix sort.
   (c) Karp–Rabin algorithm and Karp–Miller–Rosenberg naming technique.

   A few lines for each part is sufficient.
4. Let T be a string and let R be a multiset of symbols. In *jumbled string matching*, a factor S of T is an occurrence of R if S consists of exactly the symbols of R. For example, if $T = \texttt{abahgcabah}$ and $R = \{\texttt{a}, \texttt{a}, \texttt{b}, \texttt{c}\}$, the only occurrence of R in T is the factor $S = \{caba\}$. Describe an algorithm for finding all occurrences of R in T. The time complexity should be $\mathcal{O}(|T| + |R|)$ on an alphabet of constant size.
5. Construct the Aho–Corasick automaton for the pattern set $\{\texttt{string}, \texttt{ring}, \texttt{trie}, \texttt{log}, \texttt{ecology}\}$. Simulate the scanning of the text `stringology` with the automaton.
6. Let $\mathcal{R} = \{S_1, S_2, \ldots, S_k\}$ be a set of strings over a constant size alphabet such that no string in $\mathcal{R}$ is a factor of another string in $\mathcal{R}$. The *shortest distinguishing factor* of S_i is the shortest string that occurs in S_i but not in any other string in $\mathcal{R}$. Describe an algorithm for finding the shortest distinguishing factor for all strings in $\mathcal{R}$. The time complexity should be $\mathcal{O}(||\mathcal{R}||)$, where $||\mathcal{R}||$ is the total length of the strings in $\mathcal{R}$.