Problem Set 6

Problem 1 Show how to modify the KPM pattern matching algorithm so as to find every occurrence of a pattern string P that appears as a substring in T, while still running in O(n+m) time. (Be sure to catch even those matches that overlap.)

Problem 2 A pattern P of length m is said to be a circular substring of a text T of length n if P is equal to the concatenation of a suffix of T and a prefix of T, where neither the suffix and nor the prefix is an empty string. For example, if T=aacabca, all the circular substrings of length 3 of T are caa and aaa. Give an O(n+m)-time algorithm for determining whether P is a circular substring of T.

Problem 3 Draw a standard trie and a compressed trie for the following set of strings:

{abab, baba,ccccc, bbaaaa, caa, bbaacc, cbcc, cbca}.

Problem 4 Draw the frequency array and Huffman tree for the following string:

“dogs do not spot hot pots or cats”.

Problem 5 Give an efficient algorithm for deleting a string from a compressed trie and analyse its running time.

Problem 6 Give a sequence $S=(x_0, x_1, x_2, \ldots, x_{n-1})$ of numbers, describe an $O(n^2)$-time algorithm for finding a longest subsequence $T=(x_{i0}, x_{i1}, x_{i2}, \ldots, x_{ik-1})$ of the numbers, such that $i_j<i_{j+1}$ and $x_{ij}>x_{ij+1}$. That is, T is a longest decreasing subsequence of S.

Problem 7 Given a string s with repeated characters, design an efficient algorithm for rearranging the characters in s so that no two adjacent characters are identical, or determine that no such permutation exists. Analyse the time complexity of your algorithm.