58093 String Processing Algorithms (Autumn 2013)
Exercises 6 (3 December)

1. A string R is a *repeat* in a text T if R occurs at least twice in T. A repeat R is a *right-maximal* if any extension of R to the right has fewer occurrences than R, i.e., for all $c \in \Sigma$, the number of occurrences of Rc in T is less than the number of occurrences of R in T.

   Show that R is a right-maximal repeat in T if and only if the suffix tree of T has an internal node representing R.

2. Write a pseudocode algorithm for finding all occurrences of a pattern P in a text T using the suffix tree of T.

3. The relative Lempel-Ziv (RLZ) factorization of S with respect to T is the smallest partitioning $S_1 S_2 \ldots S_z = S$ of S such that each factor S_i is a factor of T too. Describe a fast algorithm for computing the RLZ factorization.

4. Hamming distance is the edit distance with substitution as the only allowed edit operation. Let $ed_H(A, B)$ denote the Hamming distance of two strings A and B of the same length.

   (a) Suppose we have preprocessed the strings A and B so that the longest common extension for any pair of suffixes can be computed in constant time. Show how the Hamming distance $ed_H(A, B)$ can be computed in $\mathcal{O}(ed_H(A, B))$ time.

   (b) Design an $\mathcal{O}(kn)$ worst case time algorithm for approximate string matching with Hamming distance.

5. What is the number of distinct factors in the string `abracadabra`?

6. Give a linear time algorithm for computing the matching statistics of S with respect to T from the generalized suffix *array* of S and T and the associated LCP array (without constructing the suffix tree).