CSMTSP - Text Searching and Processing

1. We consider the alphabet $\Sigma = \{a, b\}$. For $x \in \Sigma^*$, the string-matching automaton of x, $SMA(x)$, is the minimal deterministic automaton accepting the language $\Sigma^* x$.

(a) Design the following string-matching automata:

$$SMA(\texttt{a}),\ SMA(\texttt{aa}),\ SMA(\texttt{aab}),\ SMA(\texttt{aaba}).$$

[10 marks]

(b) Describe the main step of the on-line algorithm that builds $SMA(x\sigma)$ from $SMA(x)$ ($x \in \Sigma^*, \sigma \in \Sigma$). *[15 marks]*

(c) How many backward arcs are there in $SMA(\texttt{aaba})$? List these arcs. *[10 marks]*

(d) Prove that $SMA(x)$ has no more than $|x|$ backward arcs. *[15 marks]*

2. (a) Give all the periods and borders of the string

`aaabaaaabaaaabaaaa`.

[10 marks]

(b) Prove that the border of a border of a string x is also a border of x. Let $border(x)$ be the longest (proper) border of x, prove that a border of x is either $border(x)$ or a border of $border(x)$. *[15 marks]*

(c) Design an algorithm that computes the lengths of borders of all non-empty prefixes of a string x. *[20 marks]*

(d) Give the output of your algorithm for `aaabaaabaaaa`. *[5 marks]*

3. Consider a list of strings $L = (y_1, y_2, \ldots, y_k)$, in lexicographic order: $y_1 \leq y_2 \leq \ldots \leq y_k$. Let x be another string that is to be found in the list. All strings x and y's have the same length n.

(a) What is the asymptotic cost of a binary search for x in the list L if no extra information on the strings y's is known? Give a "worst-case" example to your answer. *[15 marks]*

(b) For two strings u and v, $lcp(u, v)$ is the length of their longest common prefix. Let $\ell = lcp(x, y_1)$, $r = lcp(x, y_k)$, and $i = \lfloor (k+1)/2 \rfloor$. Assume that

$y_1 \leq x \leq y_k$ and $\ell > r$. How does x compare with y_i when $\ell < lcp(y_1, y_i)$ and $\ell > lcp(y_1, y_i)$ respectively? *[15 marks]*

(c) State the cost of the binary search algorithm based on the use of longest common prefixes of y's? *[10 marks]*

(d) How many longest common prefixes of y's need to be preprocessed to run the binary search of the previous part (c)? *[10 marks]*

4. (a) Draw the expanded and compact suffix trees associated with the string

$$abaababa\$$$

[5 marks]

(b) Given a string x of length n, how many internal nodes can the expanded suffix tree of x contain in the "worst case" as a function of n? How many internal nodes can one have in the most favourable case? Give two examples to support your answers. *[10 marks]*

(c) How many internal nodes are there at most in the compact suffix tree of x? Prove your answer. *[15 marks]*

(d) Recall that a string w is called *primitive* if and only if it is not the power of a substring of w, i.e., $w = v^k,\ k \geq 1 \Rightarrow k = 1$, where v is a substring of w. For example `aba` is primitive but `ababab` = $(\texttt{ab})^3$ is not. Prove that a *square* string ww is a string consisting of two consecutive occurrences of a primitive string w. (Hint: use the periodicity lemma) *[10 marks]*

(e) Using the above result prove that there is a square ww beginning at position i in a string x if and only if the following condition is met: there exists a node α in the compact suffix tree T_x of x such that i and $i + |w|$ are *consecutive* leaves in the subtree of T_x rooted at α. *[10 marks]*

5. (a) Define the Hamming and Levenstein distances of two strings x and y. *[5 marks]*

(b) Write an algorithm that computes the edit distance matrix C for two strings x and y, where $|x| = n$, $|y| = m$, and $C[i, j]$ is the cost of a cheapest edit script that transforms the first i characters of x into the first j characters of y. *[10 marks]*

(c) Expand the algorithm of part (b) so that it produces and outputs one optimum edit script. I.e., it should recover a sequence of edit operations (insert, delete, substitute) that results in a minimum total cost. *[15 marks]*

(d) Prove that when insertion and deletion have unit cost and the cost of a nontrivial substitution (i.e., a substitution in which a character is replaced by a different one) is at least 2, then the minimum edit distance e between two strings of length m and n is $e = n + m - 2s$, where s is the length of a longest common subsequence between x and y. *[20 marks]*