CS21003 - Tutorial 9

October 27th, 2017

1. Let S and T be strings of lengths n and m respectively, with $m \leq n$. T is called a cover of S if every position in S belongs to some match of T in S. For example, $T = aba$ is a cover of $S = ababaaba$. The three matches of T in S cover all the positions in S. On the other hand, $T = ab$ is not a cover of $S = ababaaba$ as demonstrated here: abab**a**ab**a** (the uncovered positions are shown in bold face). Propose an $O(n + m)$-time algorithm to determine whether T is a cover of S.

2. Let T be a string of length m. Propose an $O(m)$-time algorithm to determine whether T can be represented as $T = \alpha\beta = \beta\alpha$ for two non-empty strings α and β.

3. You are given an array $F[0...m-1]$ which is known to be the failure-function table of some (unknown) string. Propose an efficient algorithm to construct a string T of length m such that F is the failure-function table of T. Your algorithm may use any number of symbols to construct T.

4. Let T be a string of length m. The prefix table of T is an array $P[0...m-1]$ such that $P[k]$ stores the length of the longest common prefix of $T[k...m-1]$ and T (for each k in the range $0 \leq k \leq m-1$). Propose an algorithm to compute the prefix table P of T, given only the failure function table $F[0...m-1]$ for T. Notice that T itself is not provided as an input to your algorithm, only F and m are supplied. What is the running time of your algorithm?

5. Let $S, T_1, T_2, \ldots, T_k$ be strings of lengths $|S| = n$ and $|T_j| = m$ for all $j = 1, 2, ..., k$. Assume that $n > m$. Your task is to locate all the positions i in S at which one of the patterns $T_1, T_2, \ldots, T_k$. Describe an algorithm to solve this problem in $O(n + mk)$ expected time.