CS21003 - Tutorial 10

November 9th, 2018

1. 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.

2. Let S and T be strings each of length n. Your task is to determine whether T can be obtained by cyclically rotating S. For example, the string *star* can be obtained by cyclically rotating the string *tars*, whereas the string *arts* cannot be obtained by cyclically rotating the string *tars*. Supply an O(n)-time algorithm to solve this problem.

3. 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.

4. Let S and T be two strings, and you plan to find all matches of T in S. Let # be a symbol outside the alphabet of S and T. You first compute the table of longest proper border values $F()$ for the string $T\#S$. Determine how you can solve the string matching problem in linear time using this table.

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.