University of Waterloo
CS240E, Winter 2023
Assignment 5

Due Date: Wednesday, April 5, 2023, at 5pm

Be sure to read the assignment guidelines (`http://www.student.cs.uwaterloo.ca/~cs240e/w23/guidelines/guidelines.pdf`). Submit your solutions electronically as individual PDF files named a5q1.pdf, a5q2.pdf, . . . (one per question).

Question 1 [4+3+2+3=12 marks]

Let P be a set of n points in general position. A *2-dimensional partial match query* specifies a value a, and asks whether there are any points in P that have either x-coordinate a or y-coordinate a (or both).

a) Assume P is stored in a 2-dimensional kd-tree. Design an algorithm that can answer a partial match query in $O(\sqrt{n})$ time.

b) Argue that any comparison-based algorithm to do partial matches must use $\Omega(\log n)$ comparisons on some instance of size n.

c) Assume P is stored in a 2-dimensional range-tree. Design an algorithm to answer a partial match query. Make it as efficient as you can. It suffices to describe the idea and analyze the run-time.

d) Design a data structure to store P that uses $O(n)$ space and permits to insert points, delete points, and answer 2-dimensional partial match queries in $O(\log n)$ worst-case time. Briefly say how these operations are implemented.

Question 2 [5+5=10 marks]

A *range-counting-query* is like a range search, except that you only need to report *how many* items fall into the range, you do not need to list which items they are.

a) Describe how any balanced binary search tree can be modified such that a range counting query can be performed in $O(\log n)$ time (independent of s, the number of points in the query-interval). Briefly state the changes needed, then describe the algorithm for the range counting query.

b) Now consider the 2-dimensional-case: Describe an appropriate range-tree based data structure such that you can answer range-counting-queries among 2-dimensional points in time $O((\log n)^2)$. Then describe the algorithm for the range counting query.

Question 3 [7 marks]

Let T be a 01-string of length n. Describe an algorithm that has run-time $O(n)$ and finds the longest pattern P that occurs *at least twice* in T.

The two occurrences of P may overlap each other (e.g. for $T = 011010100$, we have $P = 1010$) and they need not be adjacent (e.g. for $T = 0111010111$ we have $P = 0111$). If there are multiple such patterns P, then an arbitrary one should be returned.

Question 4 [3+3+3=9 marks]

We are searching for pattern P in text T where $|T| = n$, $|P| = m$, and $n \geq m \geq 1$.

a) Show that any pattern matching algorithm must look at at least $\lfloor n/m \rfloor$ characters of T in the worst case.

b) Consider pattern $P = 0^m$ and let text T be a string of $n \geq m$ bits that were randomly chosen to be 0 or 1 with equal probability. Let X be the number of checks done by Boyer-Moore until it mismatches for the first time or returns with success. (The check that leads to a mismatch is included in this count.) Show that $E[X] \leq 2$.

c) Consider the same setup as in the previous part. Assume you just had a mismatch. Show that the expected amount by which you shift the guess forward is at least $m - 1$.

Motivation: For the special string $P = 0^m$, the expected number of checks is hence $\approx 2\frac{n}{m-1}$ (i.e., roughly within a factor 2 of the lower bound) because you expect to do 2 checks until a mismatch and then shift forward by $m - 1$ characters.

Question 5 [2+4+7=13 marks]

a) Consider the text $S =$ `ARECEDEDDEER`. Show a Huffman-trie for this text (using $\Sigma_S = \{A, C, D, E, R\}$). Also indicate with every node (including interior nodes) the frequency that this node had when building the Huffman-trie.

b) Assume we have characters $x_1, \ldots, x_n$ where x_i has frequency $F(i)$. Here $F(i)$ is the *Fibonacci-sequence*: $F(1) = 1, F(2) = 1$, $F(i) = F(i-1) + F(i-2)$ for $i \geq 3$. Argue that any Huffman tree of these characters has height $n-1$.

Hint: For $i \geq 2$, what is the frequency associated with the parent p_i of x_i?

c) Assume we have characters $x_1, \ldots, x_n$ where x_i has frequency f_i and $\min_i\{f_i\} = 1$. Assume further that some Huffman-tree T for these characters has height $n-1$. Argue that $\max_i\{f_i\} \geq F(n-1)$, where $F(\cdot)$ is again the Fibonacci-sequence.

Hint: Use the structure of a binary tree of height $n - 1$ to enumerate your characters suitably, and then argue a lower bound on f_i and on the frequency associated with the parent p_i of x_i.