Mandatory Exercise: Range Reporting and Suffix Trees

Philip Bille

1 Sorted Subarrays and Substrings Consider two following two problems.

Sorted Subarrays Let A be an array of n numbers from $\mathscr{R}$. Given indices i and j, the *subsort query* is defined as follows.

- subsort(i, j): return the sorted sequence of the numbers in $A[i, j]$.

Given an A, the *sorted subarray problem* is to preprocess A into a compact data structure that supports efficient subsort queries.

Sorted Substrings Let S be a string of length n over an alphabet Σ. Given a string P, the *sortsearch* query is defined as follows.

- sortsearch(P): return the starting positions of all occurrences of P in S in sorted order.

Given S, the *sorted substring problem* is to preprocess S into a compact data structure that supports efficient sortsearch queries.

Solve the following exercises.

1.1 Give a fast data structure for the sorted subarray problem that uses $O(n \log n)$ space and answer queries fast.

1.2 Give a data structure for the sorted substring problem that uses $O(n \log n)$ space and answer queries fast.

Your query times should be output sensitive, that is, achieve a bound that efficiently depends on the size of the output (occ). Ignore preprocessing in the exercises.