May 24, 2021

Assignment 6
Algorithms for Sequence Analysis, Summer 2021

Algorithmic Bioinformatics · Prof. Dr. Sven Rahmann

Hand in date: Monday, May 31, before 20:00

Exercise 1: Lemma RMQ ≡ LCA (4 Theory)
Prove: Let T be the Cartesian tree of array A. We have $\mathrm{RMQ}_A(i,j) = \mathrm{LCA}_T(i,j)$ for all i, j, where $\mathrm{LCA}_T(i,j)$ is the *lowest common ancestor* of nodes i, and j in T.

Exercise 2: BWT example (4 Theory)
For the string $s =$ `motorrotorrom$`, compute the BWT, the C array, the LF mapping, and the Occ table. You do not need to implement any of the algorithms; it is sufficient to present the final tables and arrays.

Exercise 3: BWT without the sentinel (4 Theory)
If we omit the sentinel character (\$), the Burrows Wheeler Transform can no longer be inverted as multiple input strings lead to the same BWT. Explain why this is the case. List all input strings s (without sentinel) for which the BWT results in bwt(s)=PSSMIPISSII. (This could be rectified by storing the index of the last character of s within the BWT.)

Exercise 4: ±1RMQ (8 Programming)
An RMQ on an array A with $A[i+1] - A[i] \in \{-1, 1\}$ is called ±1RMQ. Write a program to preprocess the input array A in $O(|A|)$ time and is then able to answer ±1RMQ queries in $O(1)$ time. It should take two command line arguments:

(a) a file containing an integer array, where the integer elements are separated by whitespace and the absolute difference between neighboring elements in the input array is always one; i.e., $A[i+1] - A[i] \in \{-1, 1\}$. It should be checked that this condition holds.

(b) a file containing an arbitrary number of lines, each line containing a pair of integers $i < j$ specifying an interval $[i, j]$ for the query $\mathrm{RMQ}(i, j)$. The integers should be separated by a space.

Your program should first read the array, convert it to a numeric representation, and preprocess it for fast RMQ queries. It should then compute $\mathrm{RMQ}(i, j)$ for each interval in the second file and report the minimum element and its smallest index in the range $[i, j]$ separated by a space on one line per query. A small framework (in Python) is provided.

Example: The following input files generate the following output.

```
$ cat array.txt
2 3 4 3 2 3 2 1 0
$ cat queries.txt
2 6
0 8
$ ./myprogram array.txt queries.txt
2 4
0 8
```

Here $A[4] = 2$ is the minimum in index range $[2, 6]$ (endpoints included),
and $A[8] = 0$ is the minimum in index range $[0, 8]$.

You should apply your program to (very long) randomly generated arrays and random queries of different lengths. You should always verify correctness with a naive implementation (scanning the array).

Note: This is a complex programming exercise worth 8 points. Please be sure to allocate sufficient time for the implementation. (There will be easier programming assignments when we get to sequence alignment later during the course.) In particular, review the lecture material before the implementation and think about the details: How to partition the array into blocks, how to determine the block type, how to reduce the array, how to encode the (quadratic number) of intervals in each block into integers and how to store the full block tables, how to generate the sparse lookup table for the reduced array, etc.