Saarland University
Jun.-Prof. Dr. Tobias Marschall
Dr. Marcel Schulz

3.Assignment - Algorithms for Sequence Analysis, SS 2018

Exercise 1: Enhanced suffix array (1+1+1+1=4 Theory)
For text $T =$ `ABABAAABBABA` construct:

(a) a suffix array (`pos`),

(b) an inverse suffix array (`rank`),

(c) a *LCP* table (`lcp`),

(d) a *sparse RMQ* table (`rmq`) for the *LCP* table from the previous subtask (note that we will discuss RMQ tables in lecture 6).

Do not forget to include the sentinel $.

Exercise 2: Induced sorting / pen and paper (2 Theory)
For text $T =$ `ABACABACABACABBACABACABAC$`, illustrate how the induced sorting algorithm recurses. That is, compute the type array, LMS substrings, and the string's representation based on the reduced alphabet. Repeat this process recursively until it terminates.

Exercise 3: Induced Sorting (6 Programming)
Induced sorting is used to compute suffix arrays in linear time. Implement a program which, when given an input string, constructs a suffix array using induced sorting. To simplify this task, you are allowed to assume that, for the given input string, all LMS substrings are unique. Therefore, you do not need to implement the recursion step. In case the LMS substring are not unique for the provided input, your program should output a corresponding error message.

Besides the final suffix array, your program should also output the type array, the suffix array of the sorted LMS positions and the suffix array obtained after sorting the L-positions i.e the suffix array with all (and only) the L-positions correctly inserted. You may denote the unknown positions in the array with -1 (in the lecture this was denoted by a dot).

Note that the program should take the input string from the command line and produce the output exactly like the example shown below. Also the string positions should start with 0 for the output.You may assume that the alphabet consists only of the letters from the DNA alphabet $\Sigma = A, C, G, T$. If not done during input, prepend a $ character to the input string.

[Example on the next page.]

```
$ ./program ABACABA$
type array:
SLSLSLLS
suffix array of sorted LMS positions:
[7,4,2]
pos after sorting of L-positions:
[-1,6,-1,-1,-1,5,1,3]
final pos:
[7,6,4,0,2,5,1,3]
```

Remarks:

- There are 12 points to be earned on this assignment sheet.
- 50% of programing points (18/36) and 50% of theoretical exercises (18/36) are necessary to take the exam.
- You are allowed to work in groups of two.
- Hand in your solutions on paper (except for source code) by putting it in the letter box of Tobias Marschall in E2.1 (ground floor).
- Programming code is to be send as a *tar.gz* package by mail to the email address:
  - aryan.3264@gmail.com
- Source code is only considered if
  - it is in one of the languages Python, Perl, C, C++, or Java,
  - it is reasonably documented, commented, and readable,
  - command-lines for compiling and calling are provided,
  - compilation does not fail, and
  - executables are named according to:

    `lastname1_lastname2_assignment3_exercise3`
- For the programing task, there will be more than one input sequences. One is for you for testing and the others are for us for grading. The latter two are kept secret and points are awarded based on whether your program computes the right answer for these two input files.
- Do not forget to mention your names and matriculation numbers on your solutions!
- Copying between groups will result in zero points for all involved groups!

Hand in date: Tuesday, June 5, 2018, before 16:00.