Project 3: Genome Scale algorithms

Maria, Mateo and Jacob

2/20/2018

Instruction of the project: https://github.com/mailund/gsa-exercises/tree/master/Project03

Project 3 - Tandem repeats

Program description

The program is implemented in Python and we had no unsolved issues implementing the algorithms. Execution of the program on command line takes the form:

```
python tandem.py [text-filename]
```

The program outputs the number of branching and non-branching tandem repeats to the console, and the full list in a file with same basename as the input file, but with the suffix out.

Test of correctness

We used the test data provided in testdata.zip to test the correctness of the program.

Time complexity of the Tandem repeat search

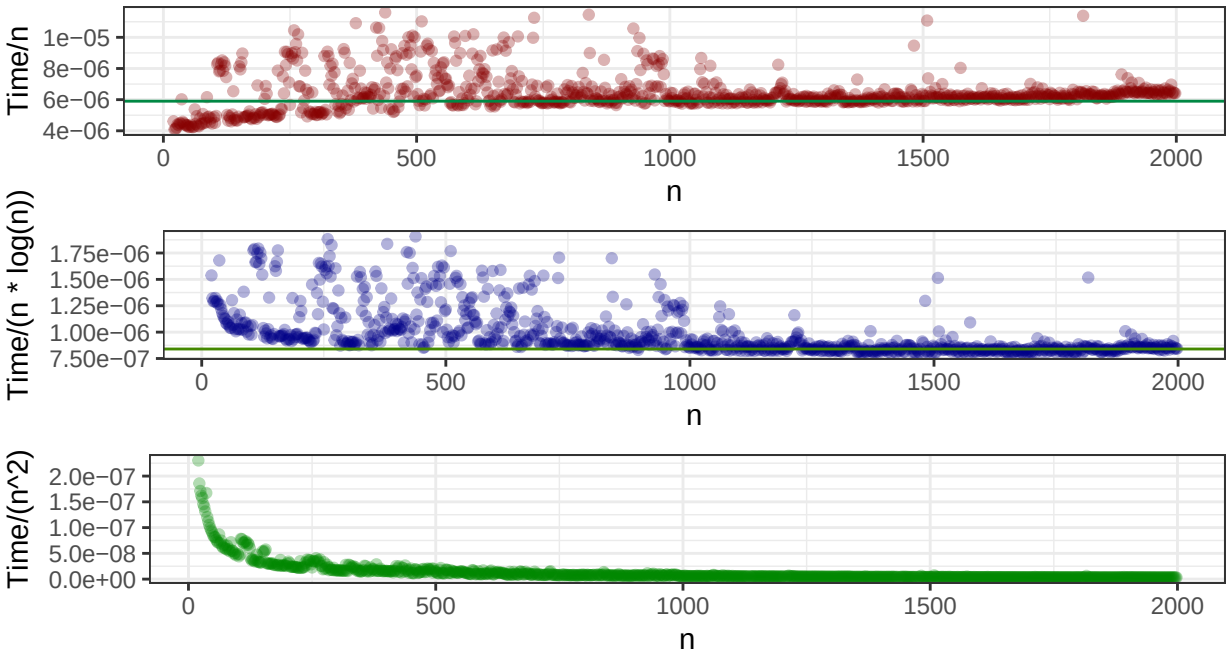
The implementation is done in two parts. First the suffix tree construction from the last project which runs in $O(n^2)$ time. Time measurements for this will not be featured here. The second part is the tandem search algorithm, which the following measurements revolve around, and is measured in code/python.

The implementation of the algorithm was modeled on Stoye-Gusfield approach, and supposedly runs in $O(log(n) * n + z)$ time as a worst case, where n is the input string length and z the number of tandem repeats.

The worst case scenario was created using fibonacci strings of varying length, while regular usage was simulated with random strings over the alphabet A,T,G,C.

Random strings

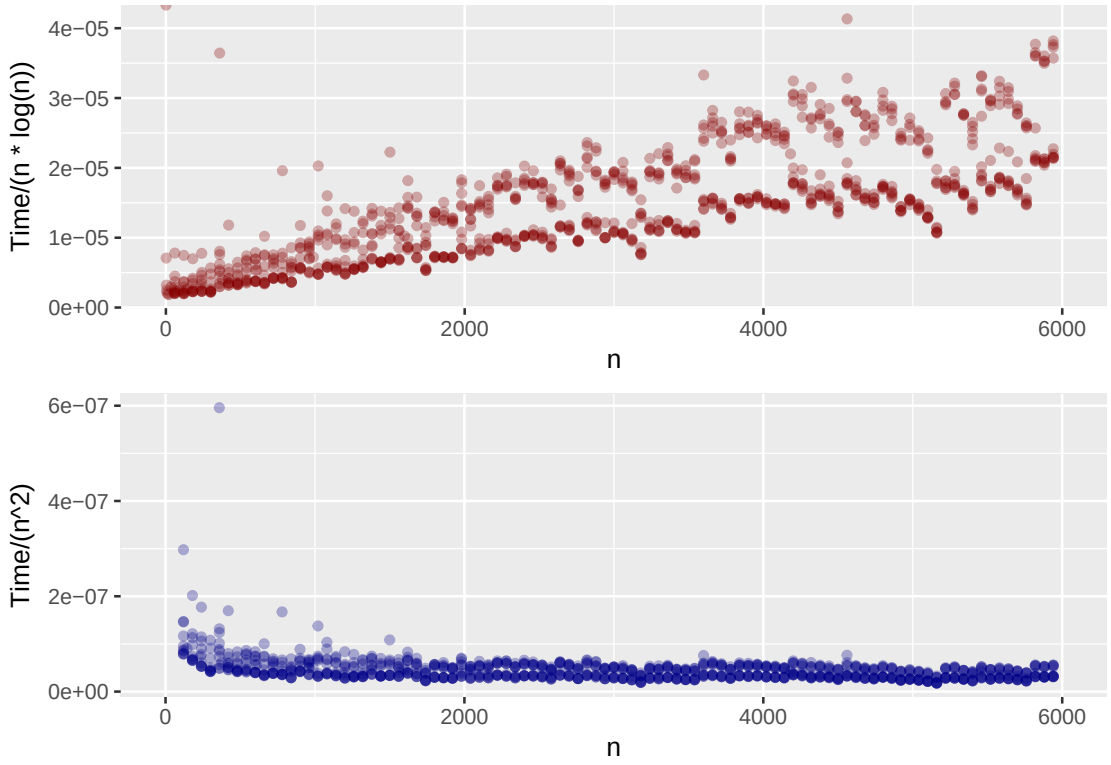
The first plot shows the search *time* over n, and yet still rising. The second is search *time* over $nlog(n)$, where an asymptote seems to appear. The third plot is search *time* over n^2, and this is too much as it forces a asymptote at zero, supporting the $nlog(n)$ runtime.



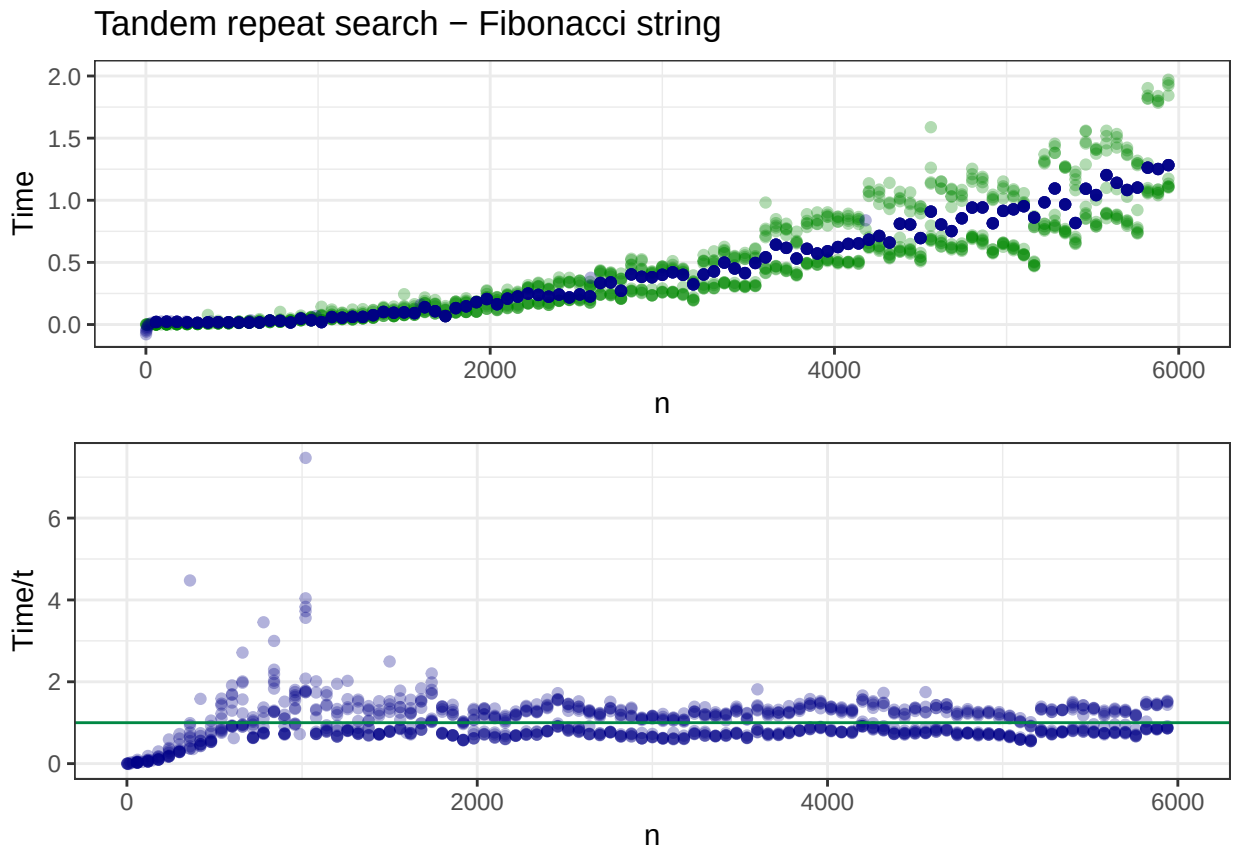
Fibonacci strings

Fibonacci string contains $0.7962 * n * log(n) + n$ tandem repeats, and are therefore suitable for a worst case scenario.

The first plot shows the search *time* over $nlog(n)$, and yet still rising. The second is search *time* over n^2, where an asymptote seems to appear, this is however not in line with the theory, but z has yet to be added to the analysis.



The following graph shows the fit of $time = n * log(n) + z$, in an attempt to let z explain the seemingly squared runtime from before. The first plot shows the time used in green, and the precidtions of the fit in blue. The second is search *time* over the prediction, here the asymptote around 1 (green) supports the theoretical runtime.


Tandem repeat search − Fibonacci string
Time
2.0
1.5
1.0
0.5
0.0
0
2000
4000
6000
n
Time/t
6
4
2
0
0
2000
4000
6000
n