FEDERAL STATE AUTONOMOUS EDUCATIONAL INSTITUTION

OF HIGHER EDUCATION

ITMO UNIVERSITY

Report

on the practical task No. 1

“Experimental time complexity analysis”

Performed by

Alexandra Matveeva

J4134c

Accepted by

Dr Petr Chunaev

St. Petersburg

2021

**Goal**

Experimental study of the time complexity of different algorithms

**Problems**

For each n from 1 to 2000, measure the average computer execution time (using timestamps) of programs implementing the algorithms and functions below for five runs. Plot the data obtained showing the average execution time as a function of n. Conduct the theoretical analysis of the time complexity of the algorithms in question and compare the empirical and theoretical time complexities.

**I.** Generate an n-dimensional random vector with non-negative elements. For , implement the following calculations and algorithms:

1. (constant function);
2. (the sum of elements);
3. (the product of elements);
4. supposing that the elements of are the coefficients of a polynomial 𝑃 of degree *n − 1*, calculate the value 𝑃(1.5) by a direct calculation of   
    (i.e. evaluating each term one by one) and by Horner’s method by representing the polynomial as ;
5. Bubble Sort of the elements of ;
6. Quick Sort of the elements of ;
7. Timsort of the elements of .

**II.** Generate random matrices 𝐴 and 𝐵 of size 𝑛 × 𝑛 with non-negative elements.

Find the usual matrix product for 𝐴 and 𝐵.

**III.** Describe the data structures and design techniques used within the algorithms.

**Brief theoretical part**

An **algorithm** is any well-defined computational procedure that takes some value, or set of values, as input and produces some value, or set of values, as output. An algorithm is thus a sequence of computational steps that transform the input into the output. The main characteristic of an algorithm is **the** **time complexity**. Time complexity signifies the total time required by the program to run till its completion. It is most commonly expressed using the Big-O notation .

**Constant function** , where is a function that has the same output value (no matter what input value is). Algorithm of calculation of constant function has constant time complexity and is called constant time algorithm. An algorithm is said to be a constant time algorithm if ( is the function that demonstrate the dependence of the execution time on the size of the input data) is limited to a value that does not depend on the size of the input data.

The **sum of elements** algorithm and the **product of elements** algorithm are sequential algorithms and have linear time complexity . It means that execution time proportional to their input. Algorithms that require iterating through all elements of an array/list in a *for*/*while* loop have such complexity (i.e. Horner’s method).

**Polynomial function**

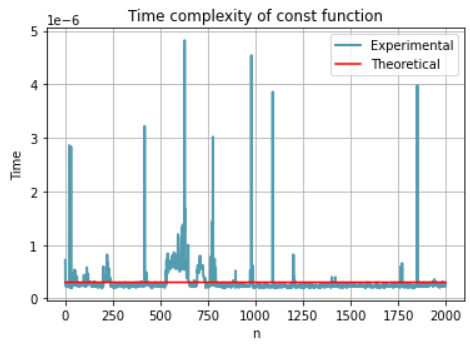
**Horner method**

**Bubble sort**

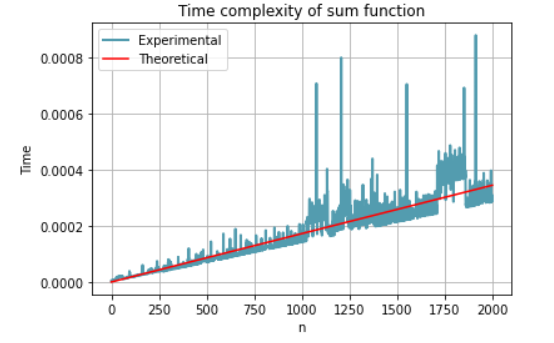
**Quick sort**

**Timsort** is a stable sorting algorithm that works in time. This algorithm based on dividing the array into blocks known as Run. Each Run is sorted by insertion sort and after that merged by combine function used in merge sort. Depending on the size of array, the run size can range from 32 to 64. Merge function performs well when the size subarrays are powers of 2. The idea is based on the fact that for small arrays, insertion sort works well. In best-case time complexity of Timsort is .

**Results**

****

Pic. 1 – Time complexity of the constant function



Pic. 2 - Time complexity of the sum elements function

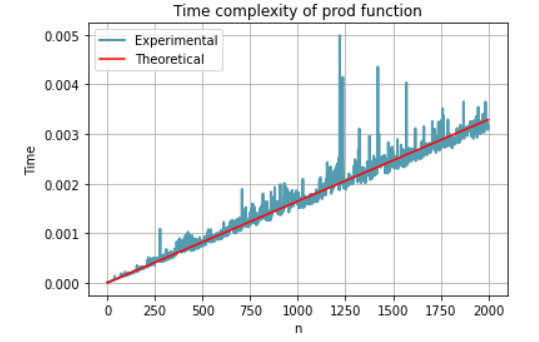


Fig. 3 – Time complexity of the product of the elements

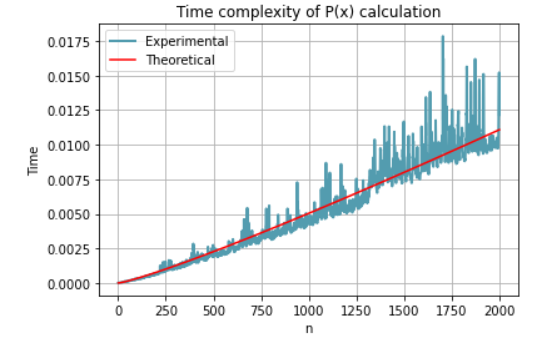


Fig. 4 – Time complexity of the Native method

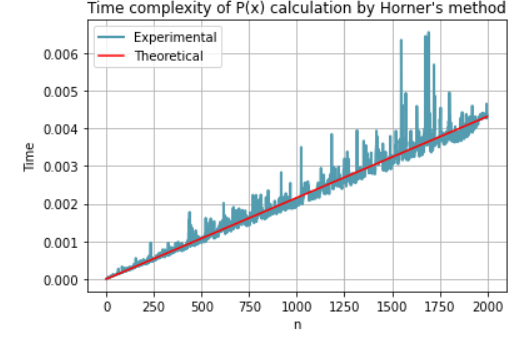


Fig. 5 – Time complexity of Horner’s method

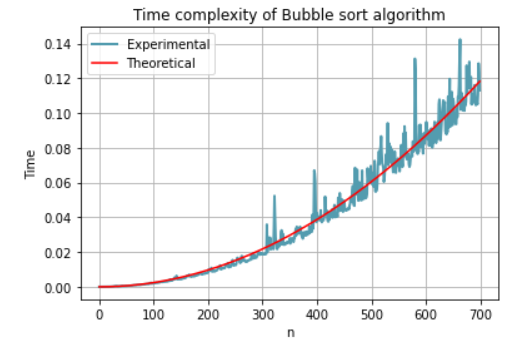


Fig. 6 – Time complexity of Bubble Sort algorithm

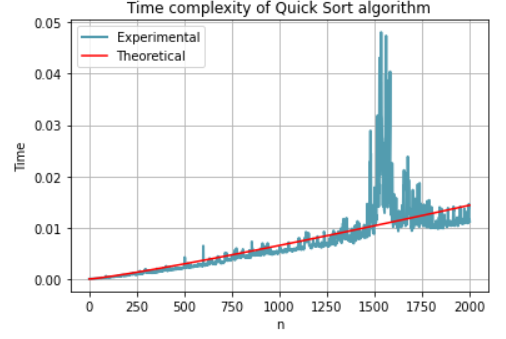


Fig. 7 – Time complexity of Quick Sort algorithm

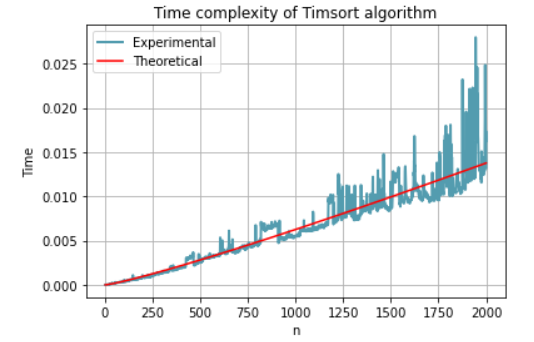


Fig.6 – Time complexity of Timsort algorithm

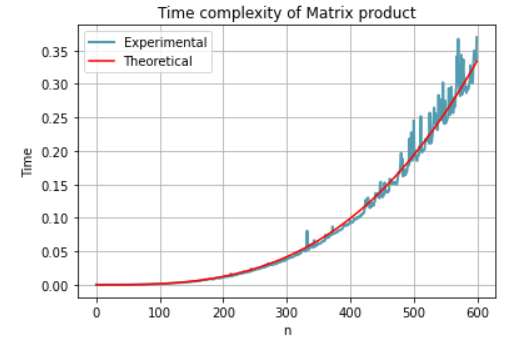


Fig. 9 – Time complexity of product of matrices algorithm

**Conclusion**

During laboratory work performance, a theoretical analysis of the time complexity of nine algorithms was fulfilled. The obtained empirical data deviate from the theoretical, but their similarity is obvious and demonstrated in the graphs. As a result we demonstrate the dependence of the execution time of various algorithms on the amount of input data.

**Appendix**

GitHub Link: <https://github.com/alex-mat-s/Algorithms/blob/main/Lab1.ipynb>