FEDERAL STATE AUTONOMOUS EDUCATIONAL INSTITUTION OF HIGHER EDUCATION ITMO UNIVERSITY

Report
on the practical task No. 1
“Experimental time complexity analysis”

Performed by
Kozlov Alexey
Dmitriy Koryakov
J4133c
Accepted by
Dr Petr Chunaev

St.Petersburg
2022

TABLE OF CONTENTS

Goal

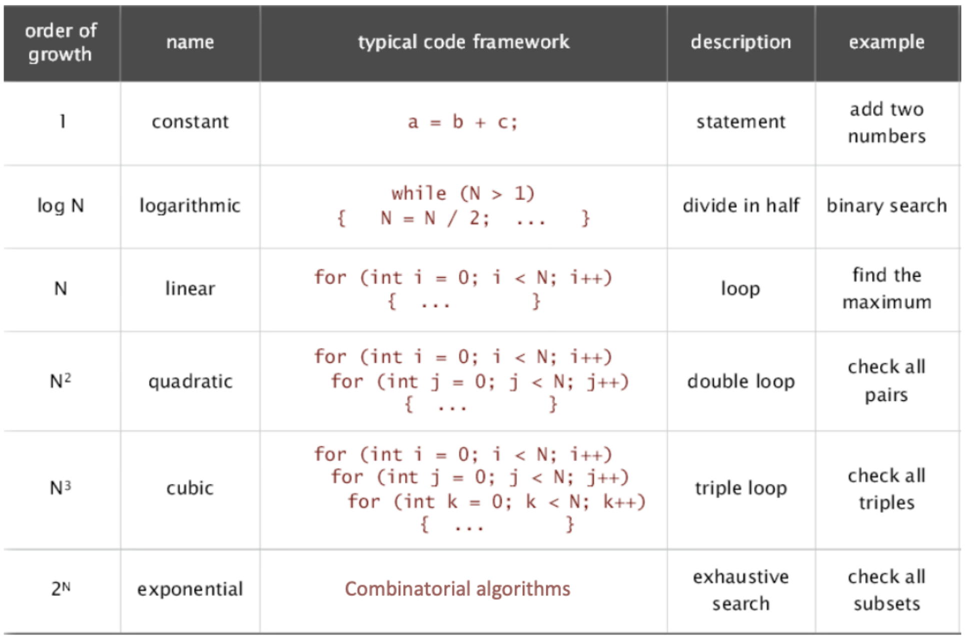
Experimental study of the time complexity of different algorithms

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.

Brief theoretical part

The (Big) O Notation, table with examples:



order of growth	name	typical code framework	description	example
1	constant	`a = b + c;`	statement	add two numbers

log N	logarithmic	`while (N > 1)`
		`{ N = N / 2; ... }`
			divide in half	binary search

N	linear	`for (int i = 0; i < N; i++)`
		`{ ... }`
			loop	find the maximum

N^2	quadratic	`for (int i = 0; i < N; i++)`
		`for (int j = 0; j < N; j++)`
		`{ ... }`
			double loop	check all pairs

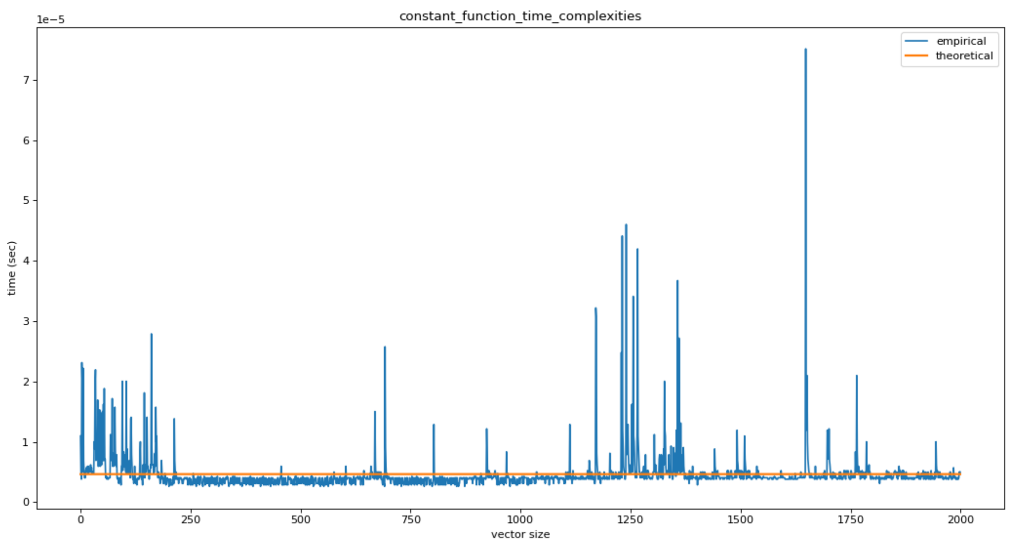
N^3	cubic	`for (int i = 0; i < N; i++)`
		`for (int j = 0; j < N; j++)`
		`for (int k = 0; k < N; k++)`
		`{ ... }`
			triple loop	check all triples

2^N	exponential	Combinatorial algorithms	exhaustive search	check all subsets

I. Generate an n-dimensional random vector $\boldsymbol{v} = [v_1, v_2, \ldots , v_n]$ with non-negative elements. For $\boldsymbol{v}$, implement the following calculations and algorithms:

1) f(v) = const (constant function);

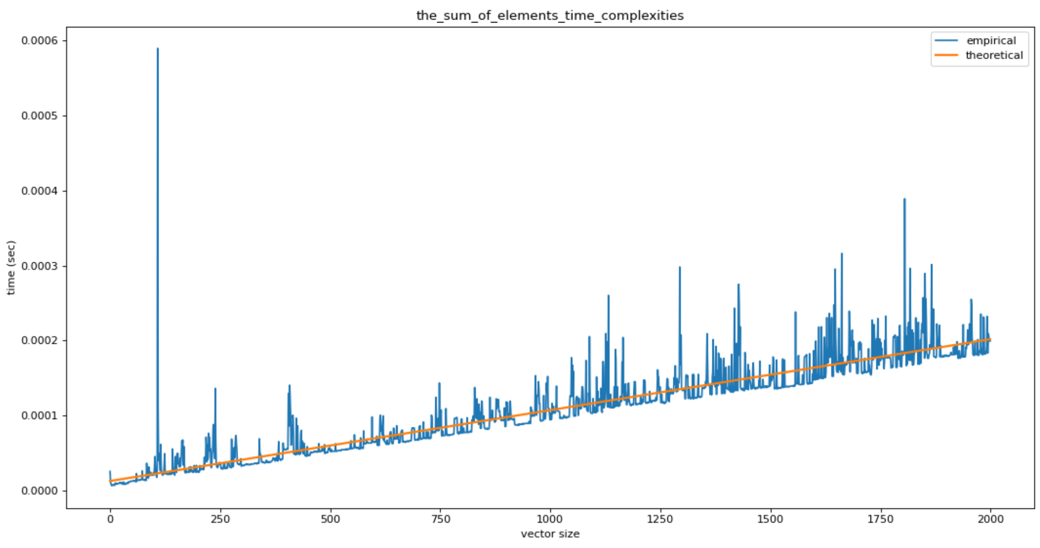
Theoretical time complexity of the constant function: *O(1)*.



Empirical and theoretical time complexity are the same.

2) $f(v) = \sum_{k=1}^{n} v_k$ (the sum of elements);

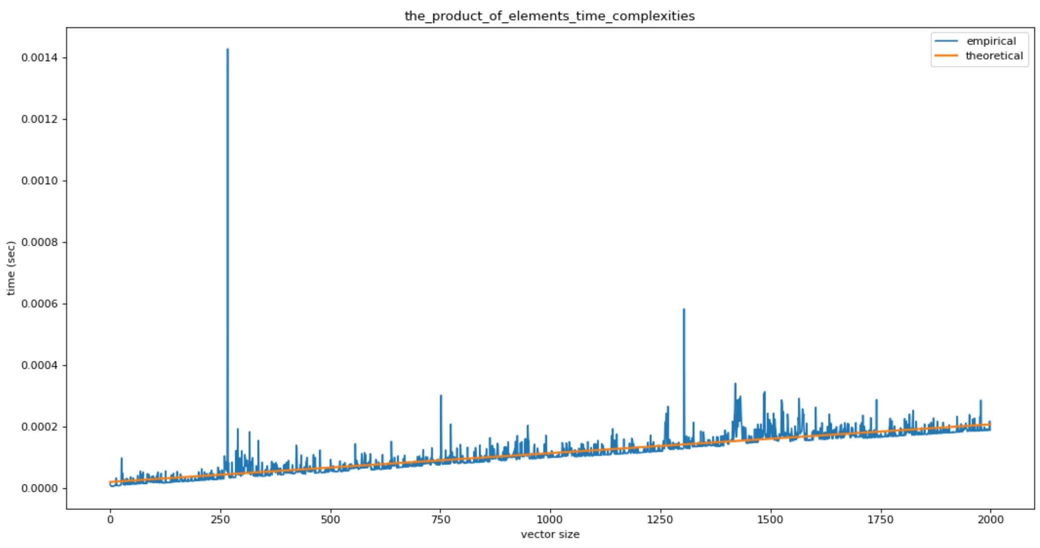
Theoretical time complexity of the constant function: *O(n)*.



Empirical and theoretical time complexity are the same.

3) $f(v) = \prod_{k=1}^{n} v_k$ (the product of elements);

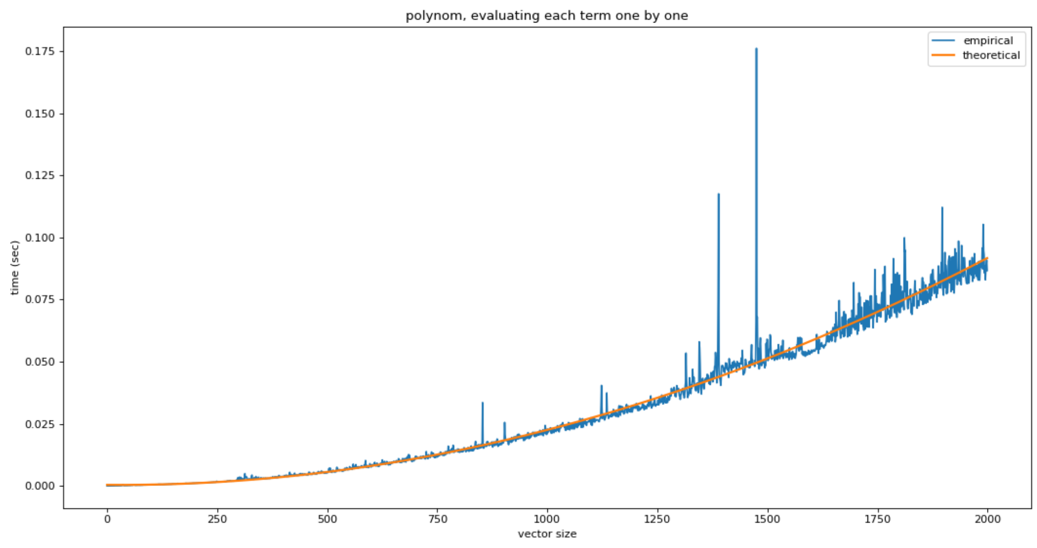
Theoretical time complexity of the constant function: *O(n)*.



Empirical and theoretical time complexity are the same.

4) The elements of $\boldsymbol{v}$ are the coefficients of a polynomial P of degree $(n-1)$, calculate the value $P\,(1.5)$ by a direct calculation of $P(x) = \sum_{k=1}^{n} v_k x^{k-1}$ *(i.e. evaluating each term one by one);*

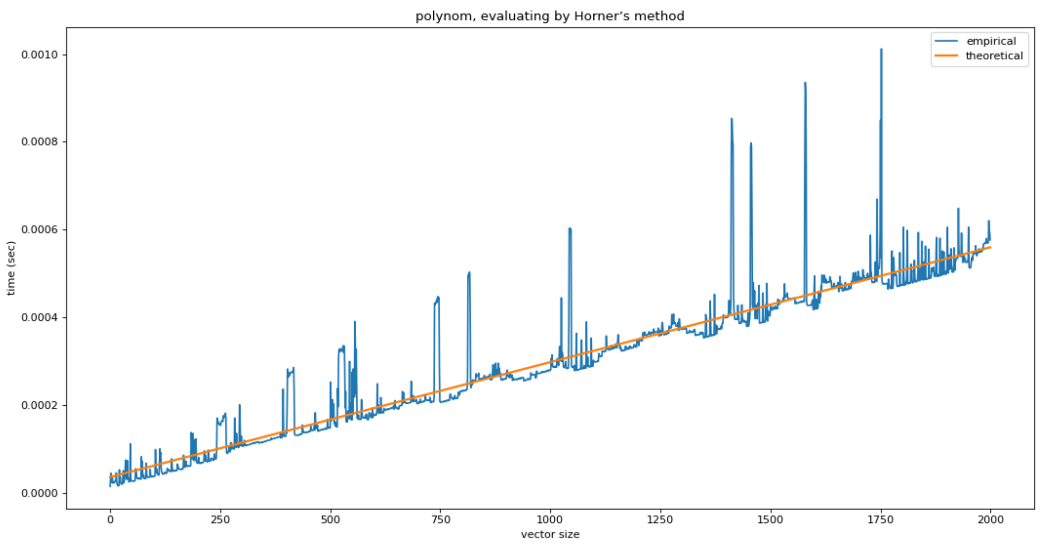
Theoretical time complexity of the constant function: *$O(n^2)$*.



Empirical and theoretical time complexity are the same.

Calculation of the value $P\,(1.5)$ by Horner's method by representing the polynomial as $P(x) = v_1 + x(v_2 + x(v_3 + \ldots))$;

Theoretical time complexity of the constant function: *$O(n)$*.



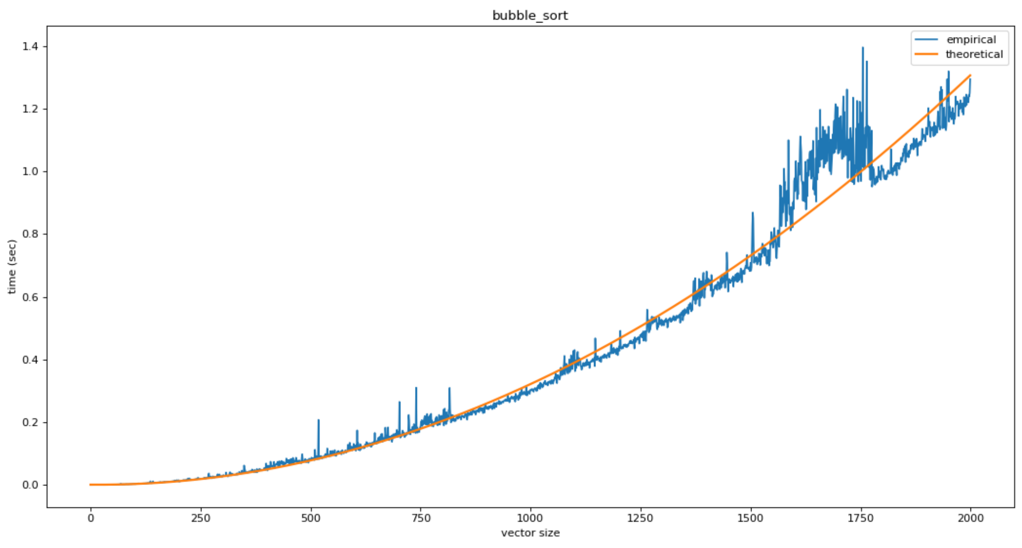
Empirical and theoretical time complexity are the same.

5) Bubble Sort of the elements of ***v***;

Description:
At each iteration, adjacent elements are sequentially compared, and if the order in the pair is incorrect, then the elements are swapped. This action is performed until the entire array has been sorted. With each pass through the array, at least one element falls into place, so it takes at most (n − 1) passes to sort the array, where n is the size of the array.

Theoretical time complexity of bubble sort:

- worst $O(n^2)$;
- average $O(n^2)$;
- best $O(n)$.



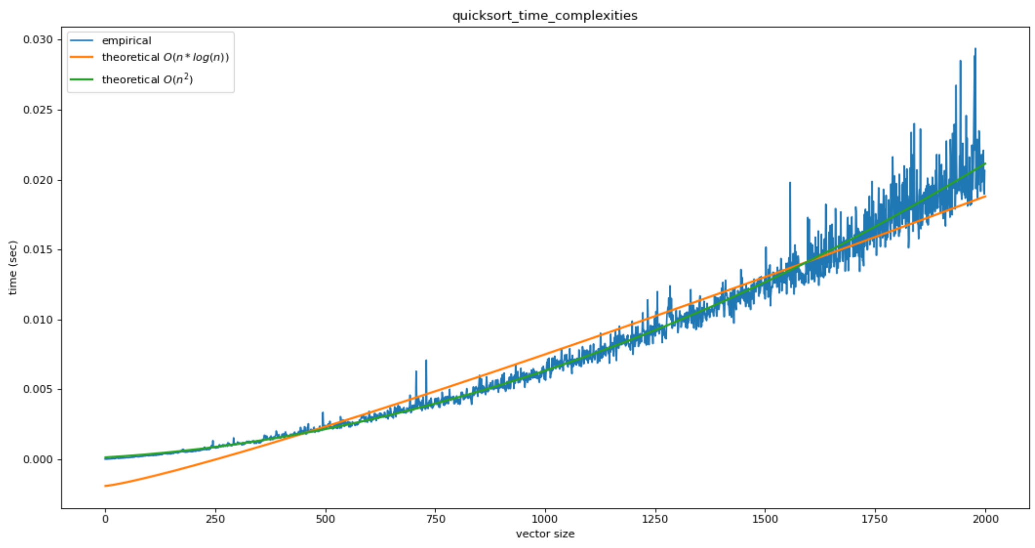
Empirical and theoretical time complexity are the same.

6) Quick Sort of the elements of ***v***;

Description:
The array a[l….r] is divided into two subarrays a[l…q] and a[q+1…r], so that each element is less than or equal to a[q]. The element a[q] is less than each element of the subarray a[q+1...r]. The index is calculated during the split. The subarrays a[l…q] and a[q+1…r] are sorted using quicksort recursion.

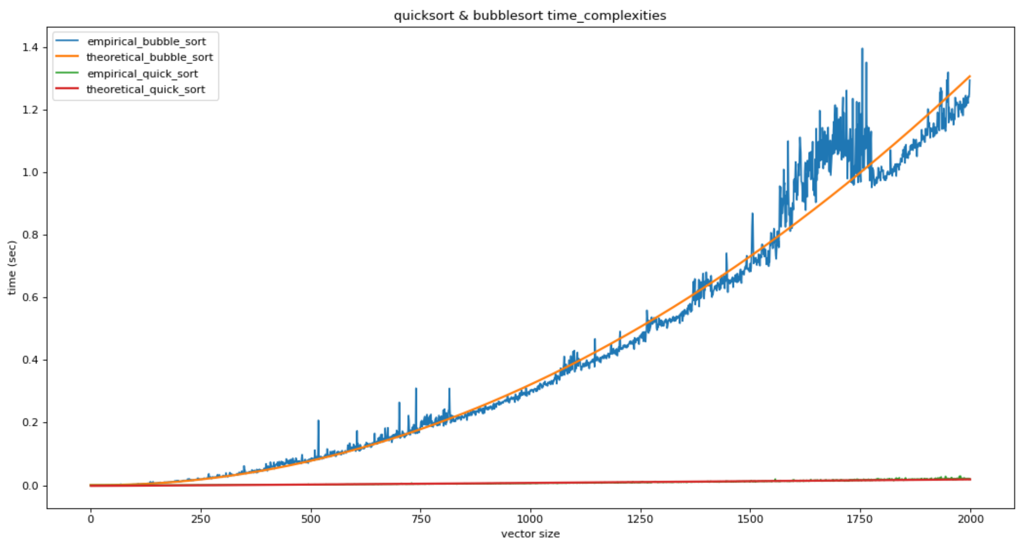
Theoretical time complexity of quick sort:

- worst *$O(n^2)$*;
- average *$O(n \log(n))$*;
- best *$O(n \log(n))$*.



Empirical and theoretical time complexity are the same.

Comparison of Bubble Sort and Quick Sort.



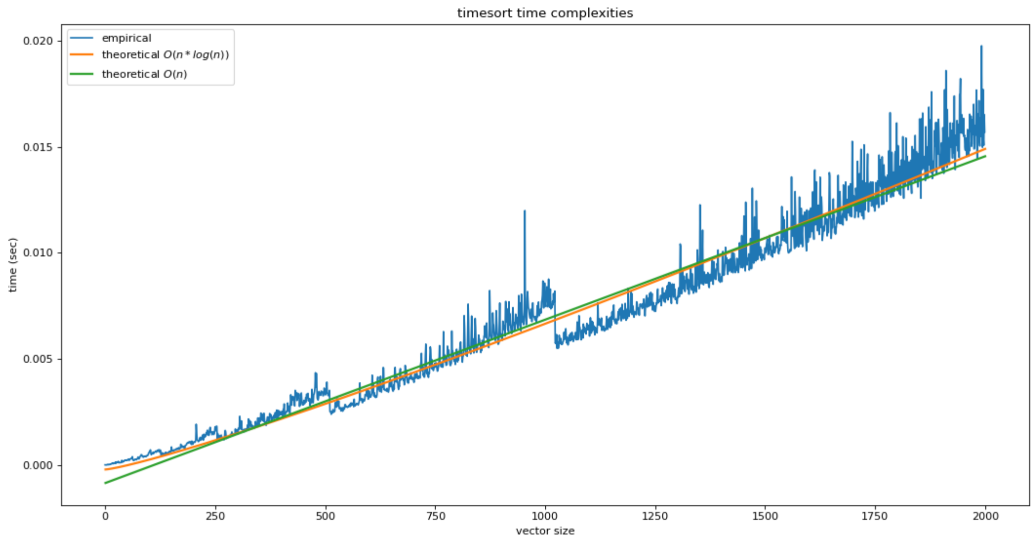
7) Timsort of the elements of ***v***;

Description:
Step 1. The input array is divided into sub-arrays of a fixed length, calculated in a

certain way.
Step 2. Each subarray is sorted by insertion sort.
Step 3. The sorted subarrays are merged into one array using a modified merge sort.

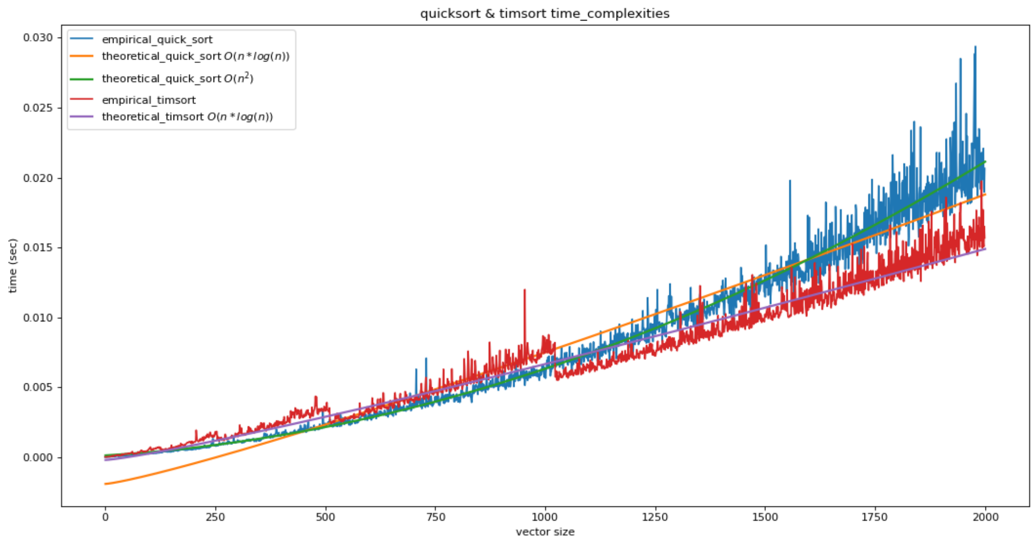
Theoretical time complexity of tim sort:

- worst *O(n log(n))*;
- average *O(n log(n))*;
- best *O(n)*.



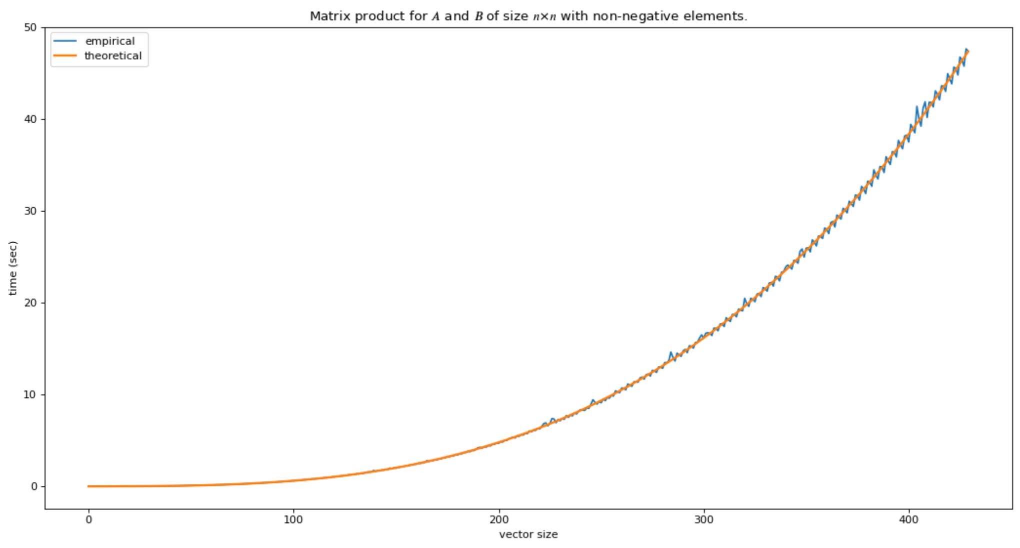
Empirical and theoretical time complexity are the same.

Comparison of Quick Sort and Tim Sort.



II. Generate random matrices A and B of size $n \times n$ with non-negative elements. Find the usual matrix product for A and B.

Theoretical time complexity of the constant function: *$O(n^3)$*.



Empirical and theoretical time complexity are the same.

Conclusions

The empirical and theoretical time complexity of the algorithms are the same. Bubble sort has the worst score time complexity.