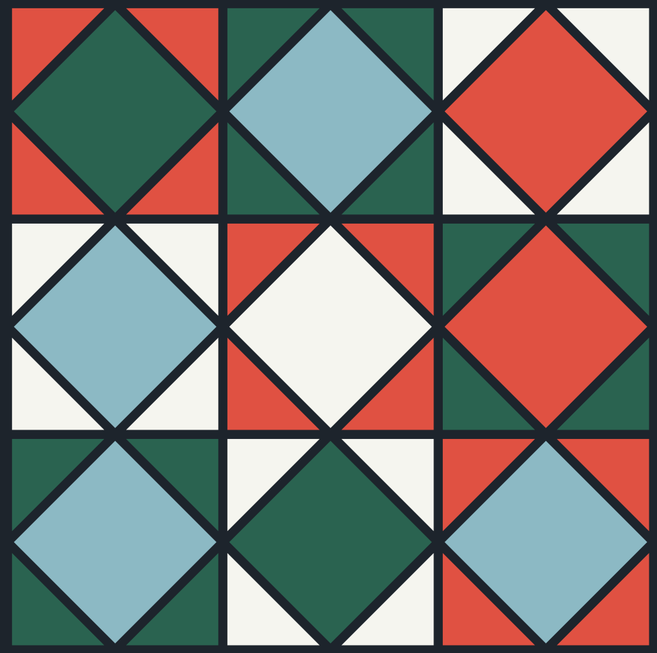
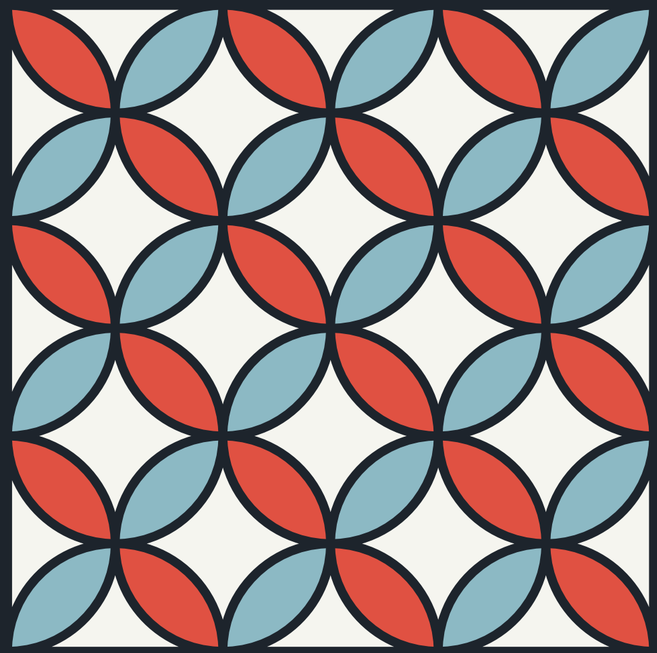
ALGORITMI DE SORTARE





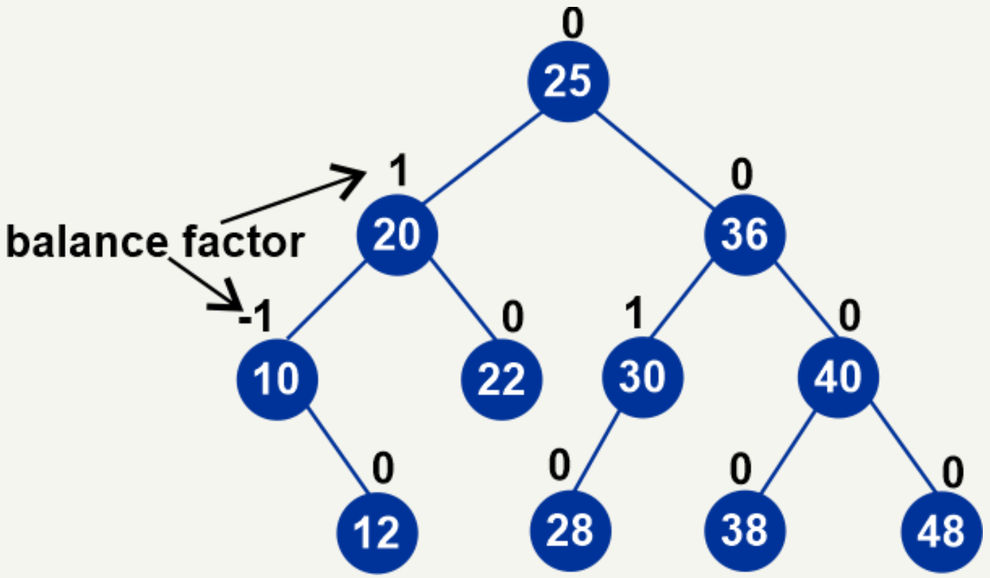
ALGORITMI DE SORTARE

ALGORITMI ALESI:

1. AVL Sort
2. Counting Sort
3. Merge Sort
4. Quick Sort
5. Radix Sort
6. Shell sort
7. STL sort

AVL SORT

Sortarea AVL este un algoritm de sortare care folosește structura de date a arborelui AVL. Acesta construiește un arbore AVL din elementele de intrare și apoi efectuează o traversare în ordine a arborelui rezultat pentru a obține elementele sortate. In implementarea noastra retinem si frecventa fiecarui numar.

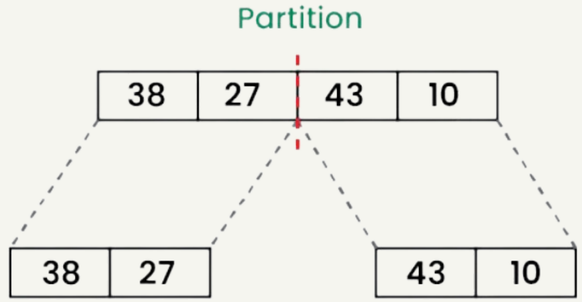


COUNTING SORT

Sortarea prin numărare (Counting Sort) este un algoritm eficient de sortare care sortează elementele în funcție de numărul lor de apariții într-o secvență. Algoritmul presupune că elementele sunt întregi și apar într-un anumit interval finit.

MERGE SORT

Merge sort este un algoritm recursiv care împarte continuu matricea în jumătate până când nu poate fi împărțită în continuare, adică vectorul mai are un singur element (un vector cu un element este întotdeauna sortat). Apoi sub-vectorii sortati sunt îmbinati într-o singură matrice sortată.



QUICK SORT

Quick sort este un algoritm de sortare **"împarte și cucerește"**. Alegând un **pivot** și rearanjând lista astfel încât elementele mai mici să fie înaintea pivotului, iar cele mai mari după acesta, algoritmul sortează recursiv subliste până când lista este complet sortată.

Ca implementare acest algoritm imparte lista in 3 subliste, mai mici, egale si mai mari decat pivotul care este mediana din 5!

RADIX SORT

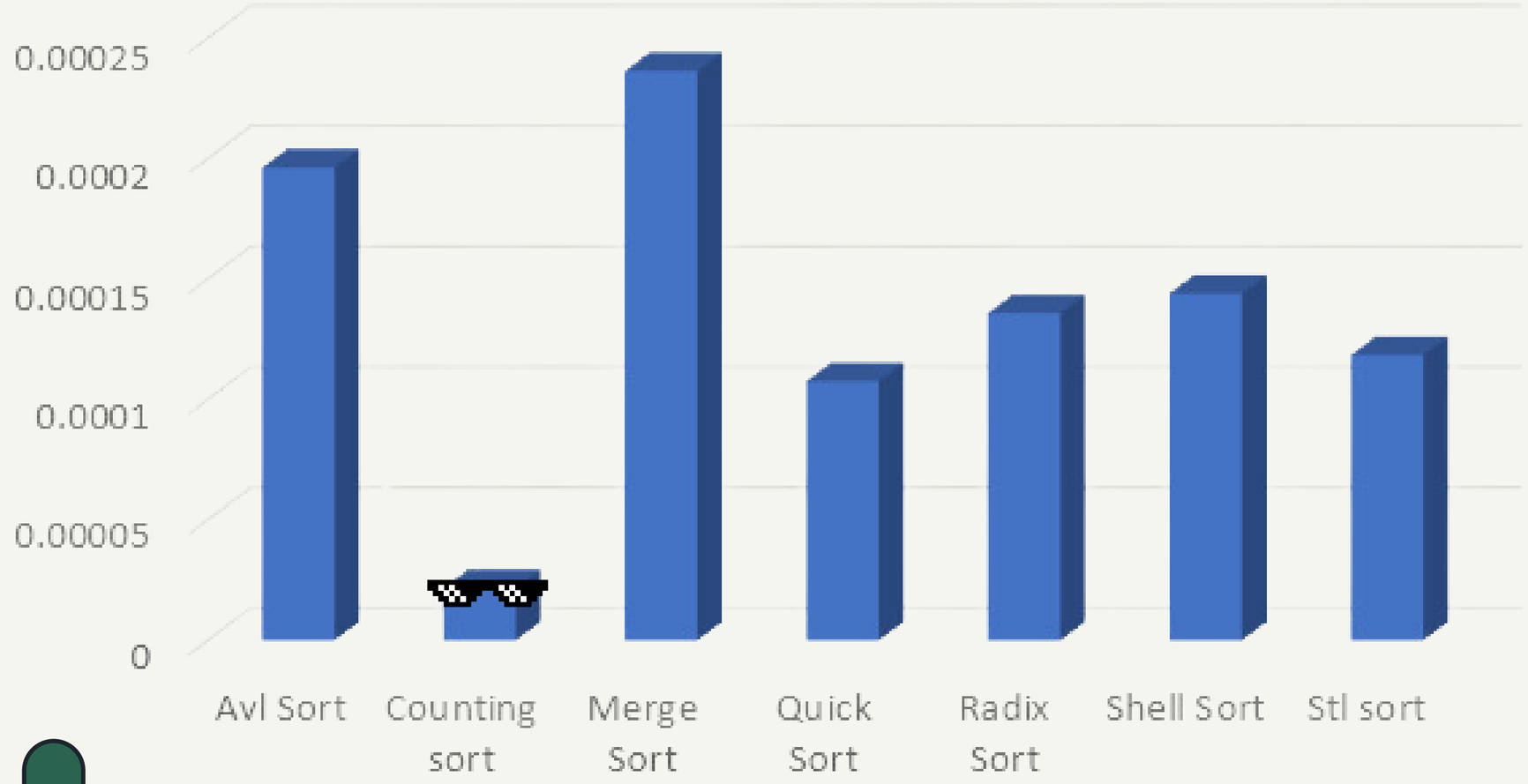
Sortarea radix (Radix sort) este un algoritm de sortare care sortează elementele pe baza cifrelor acestora. Algoritmul funcționează prin efectuarea unui număr fix de treceri prin lista de elemente, împărțind elementele în grupuri pe baza cifrelor lor de la cifra cea mai puțin semnificativă la cea mai semnificativă. La fiecare trecere, elementele sunt rearanjate în funcție de cifra curentă, astfel încât la final lista este sortată în întregime.

In implementarea noastră acesta foloseste Counting Sort si baza 10, respectiv baza 2^16!

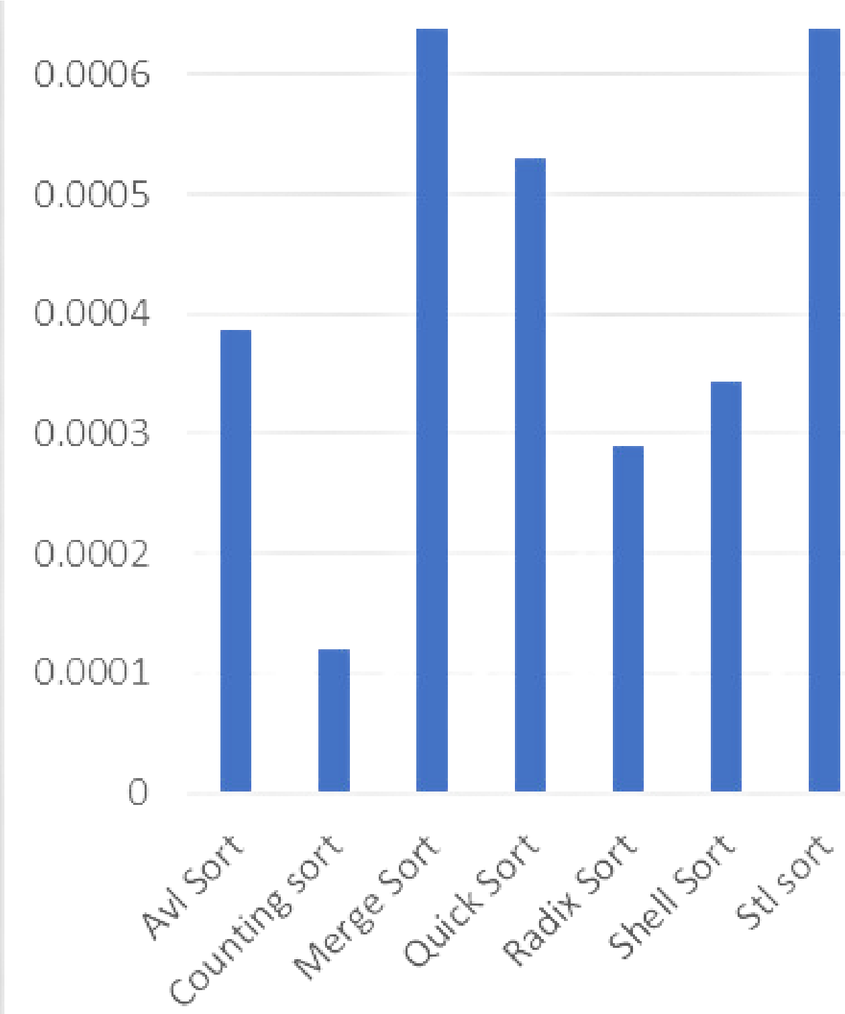
SHELL SORT

Shell sort este un algoritm de sortare care extinde metoda de **sortare prin inserție**. Acesta funcționează prin împărțirea listei în subliste mai mici, care sunt sortate folosind sortarea prin inserție. Aceste subliste sunt apoi "împinse" împreună printr-o serie de pasări, folosind un pas mic de comparare, până când întreaga listă devine aproape sortată. În cele din urmă, sortarea prin inserție este aplicată din nou pentru a finaliza sortarea listei.

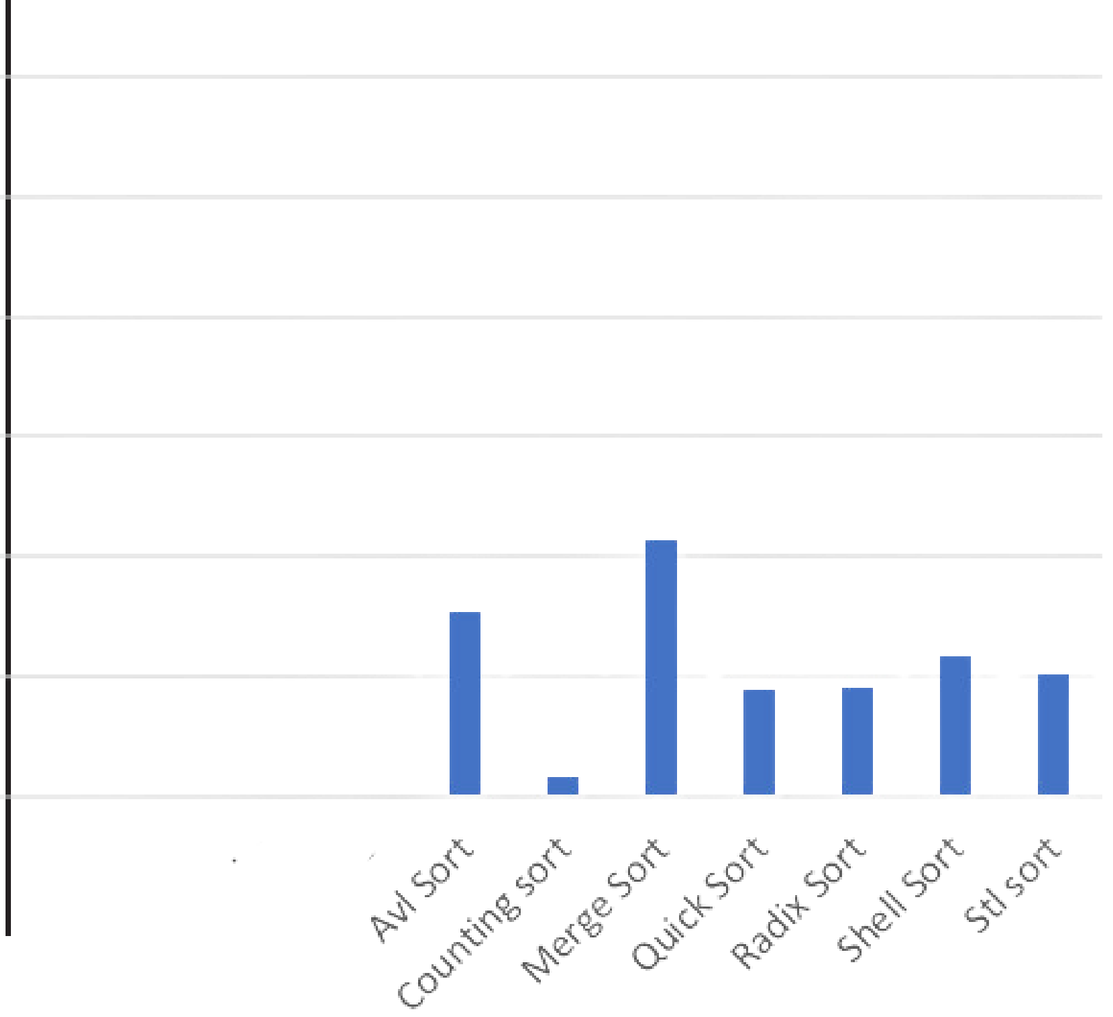
In implementare gapul porneste de la n/2 si se injumatateste succesiv!

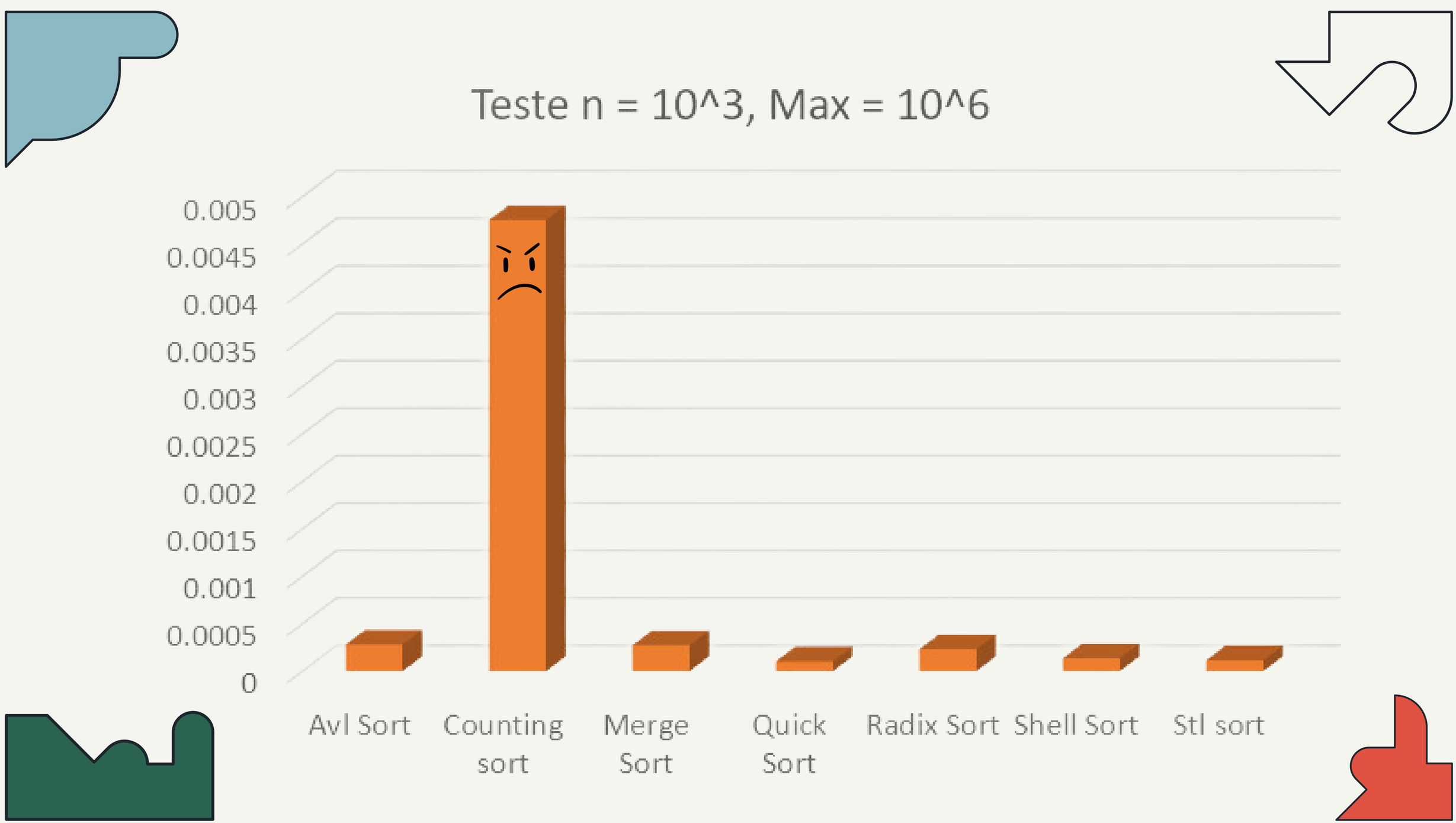

Teste n = 10^3, Max = 10^3
0.00025
0.0002
0.00015
0.0001
0.00005
0
Avl Sort
Counting sort
Merge Sort
Quick Sort
Radix Sort
Shell Sort
Stl sort

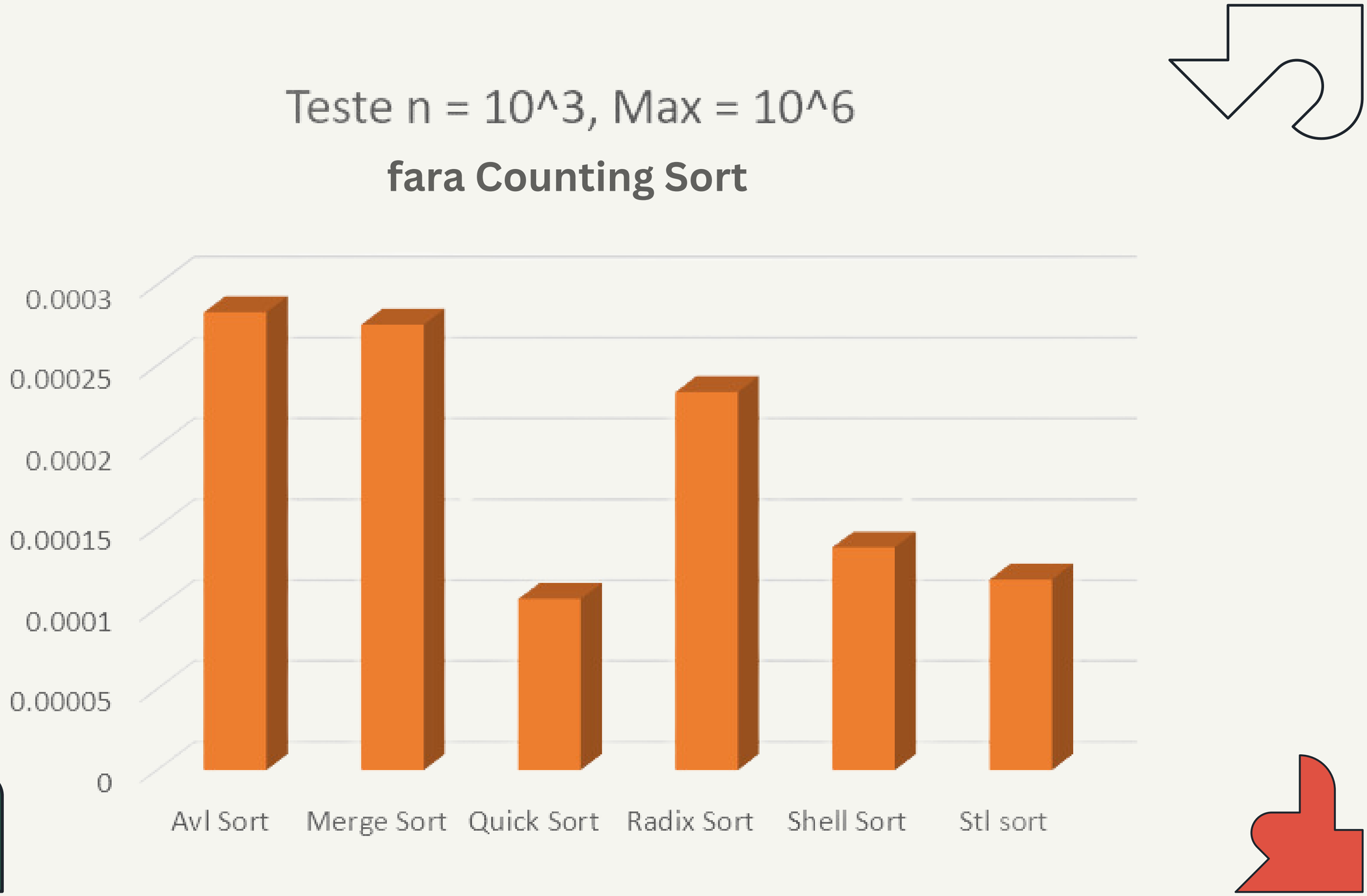
Worst case



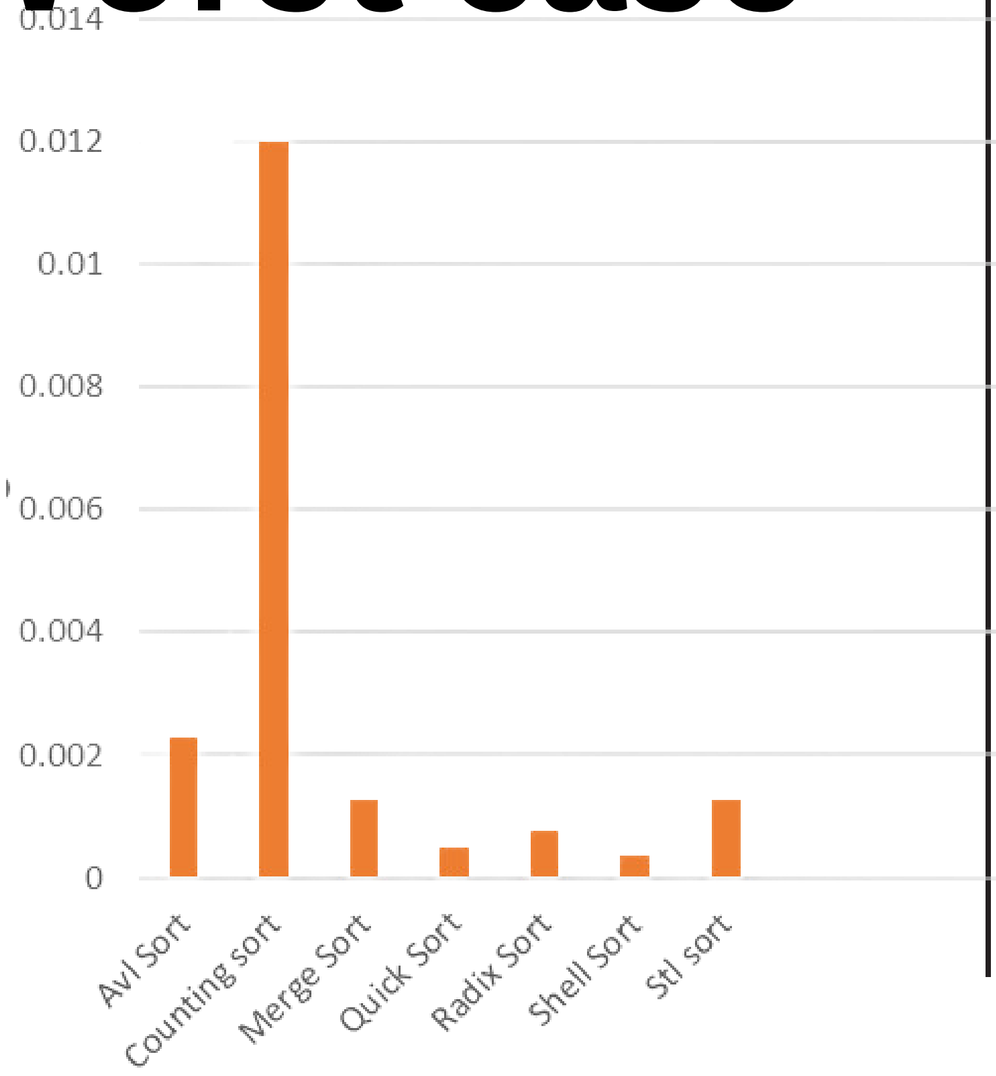
Best Case



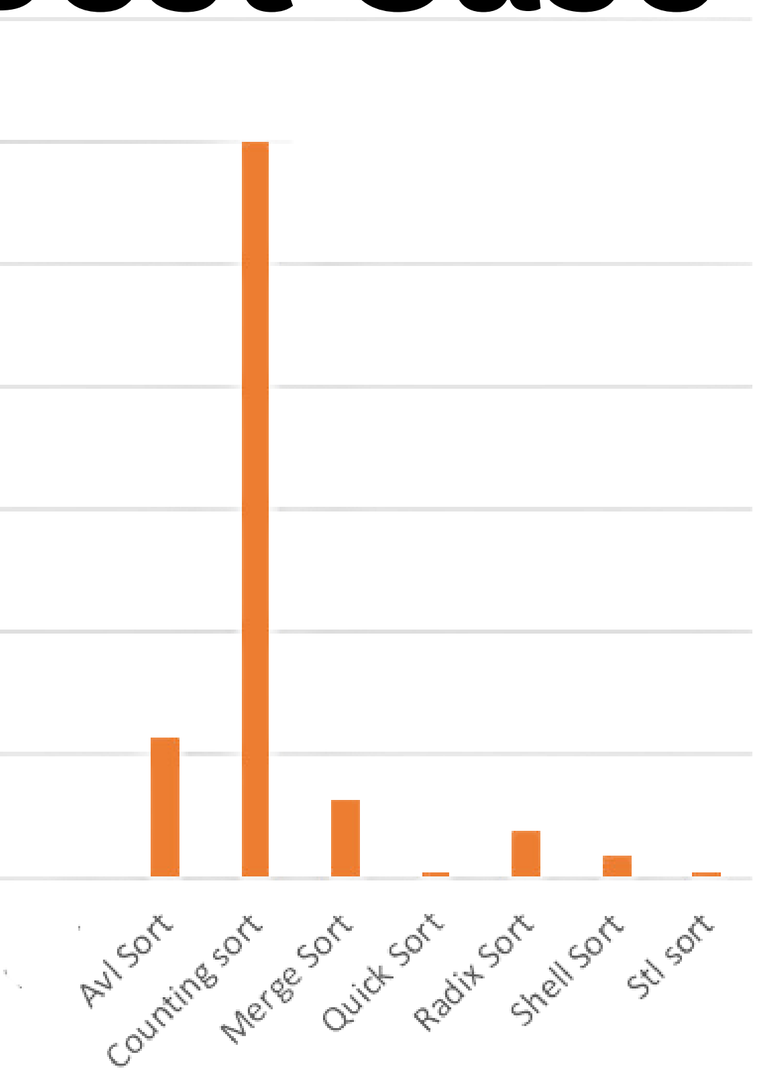

Teste n = 10^3, Max = 10^6
0.005
0.0045
0.004
0.0035
0.003
0.0025
0.002
0.0015
0.001
0.0005
0
Avl Sort
Counting sort
Merge Sort
Quick Sort
Radix Sort
Shell Sort
Stl sort


Teste n = 10^3, Max = 10^6
fara Counting Sort
0.0003
0.00025
0.0002
0.00015
0.0001
0.00005
0
Avl Sort
Merge Sort
Quick Sort
Radix Sort
Shell Sort
Stl sort

Worst case

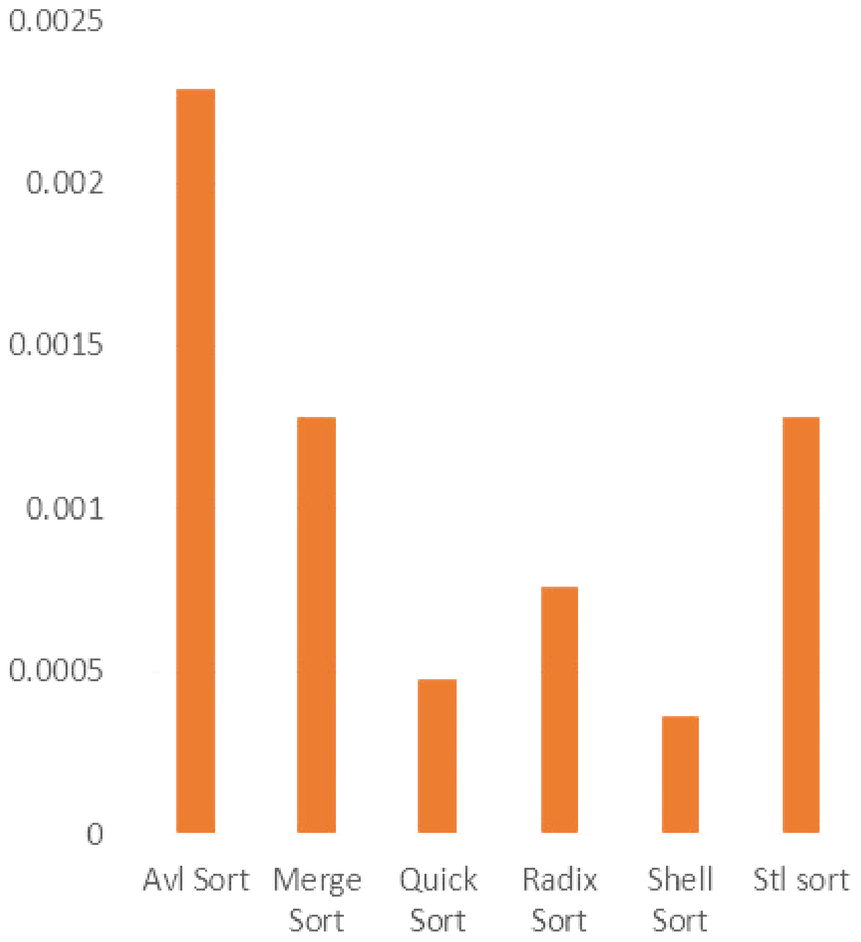


Best Case

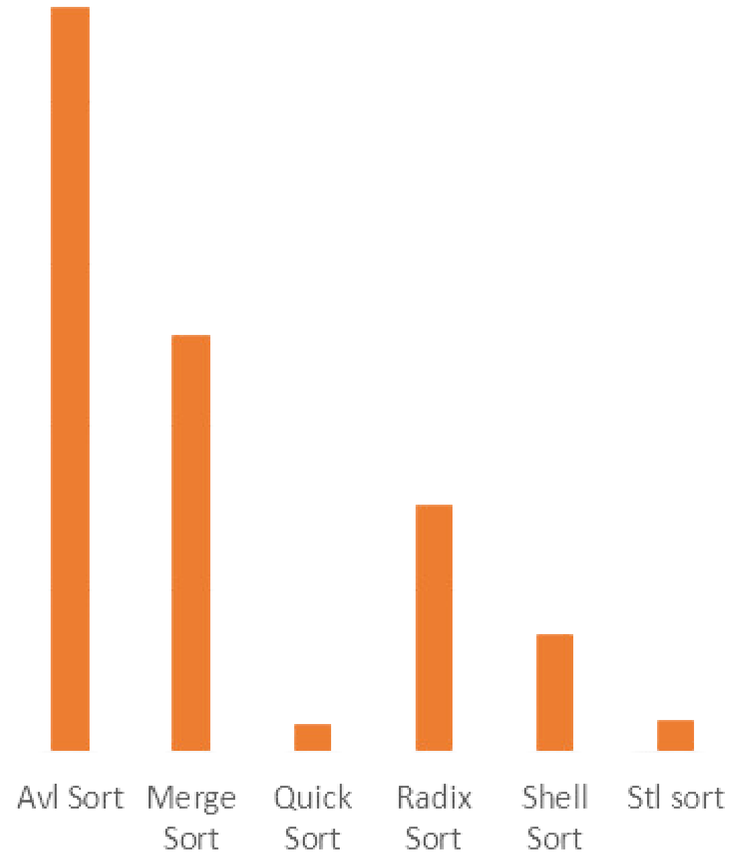


fara Counting Sort

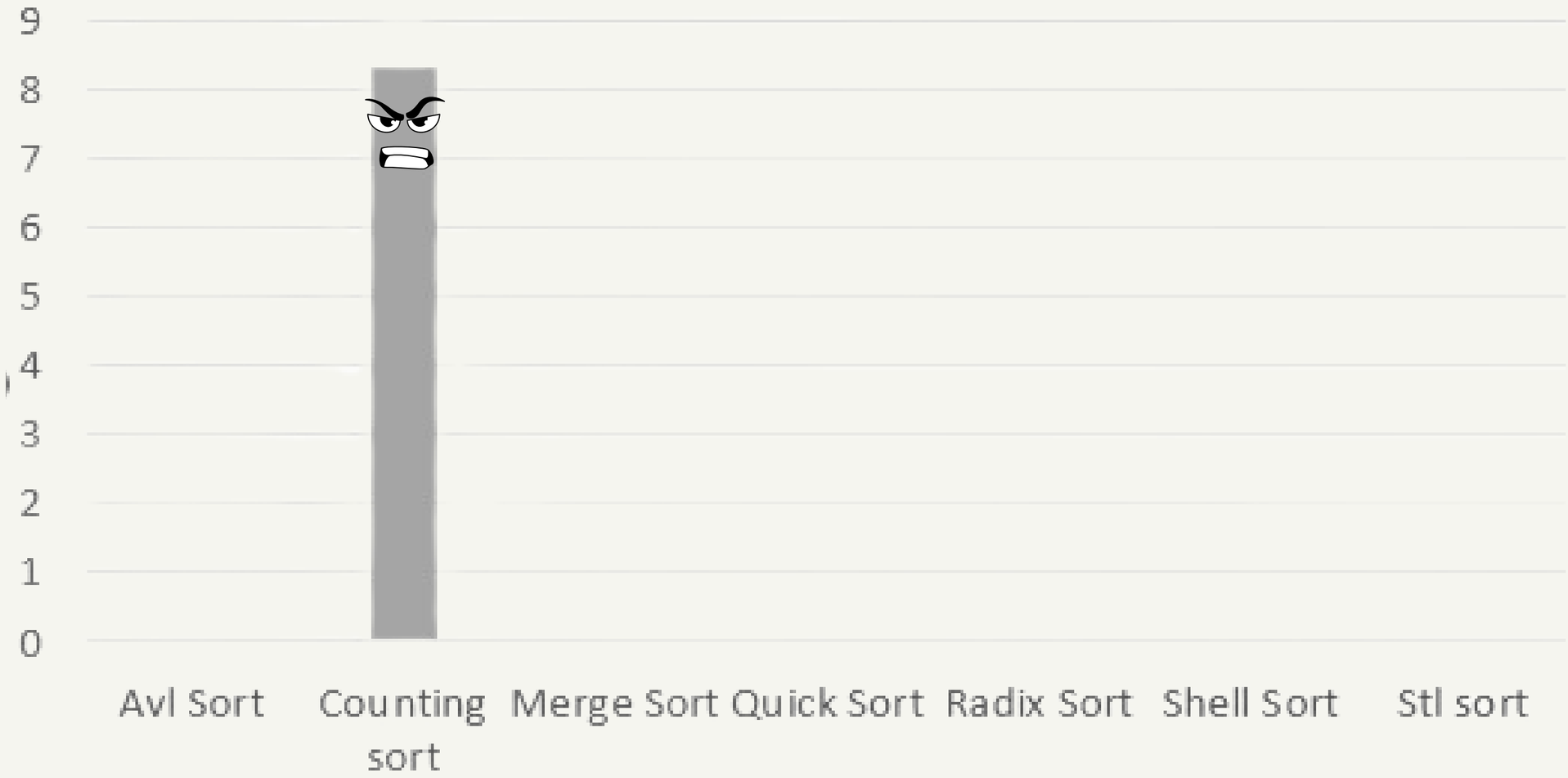
Worst case

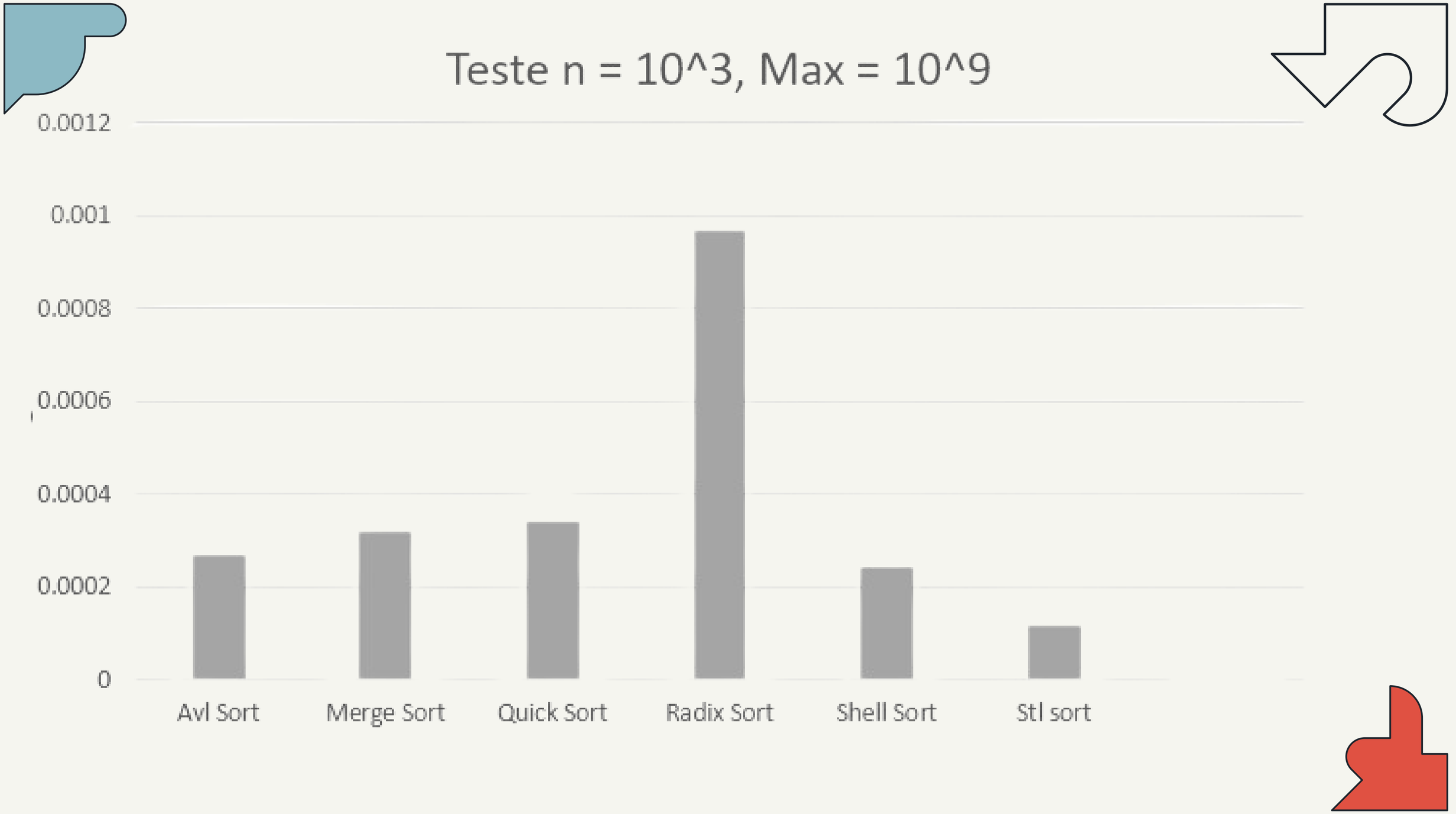


Best Case



Teste n = 10^3, Max = 10^9




Teste n = 10^3, Max = 10^9
0.0012
0.001
0.0008
0.0006
0.0004
0.0002
0
Avl Sort
Merge Sort
Quick Sort
Radix Sort
Shell Sort
Stl sort

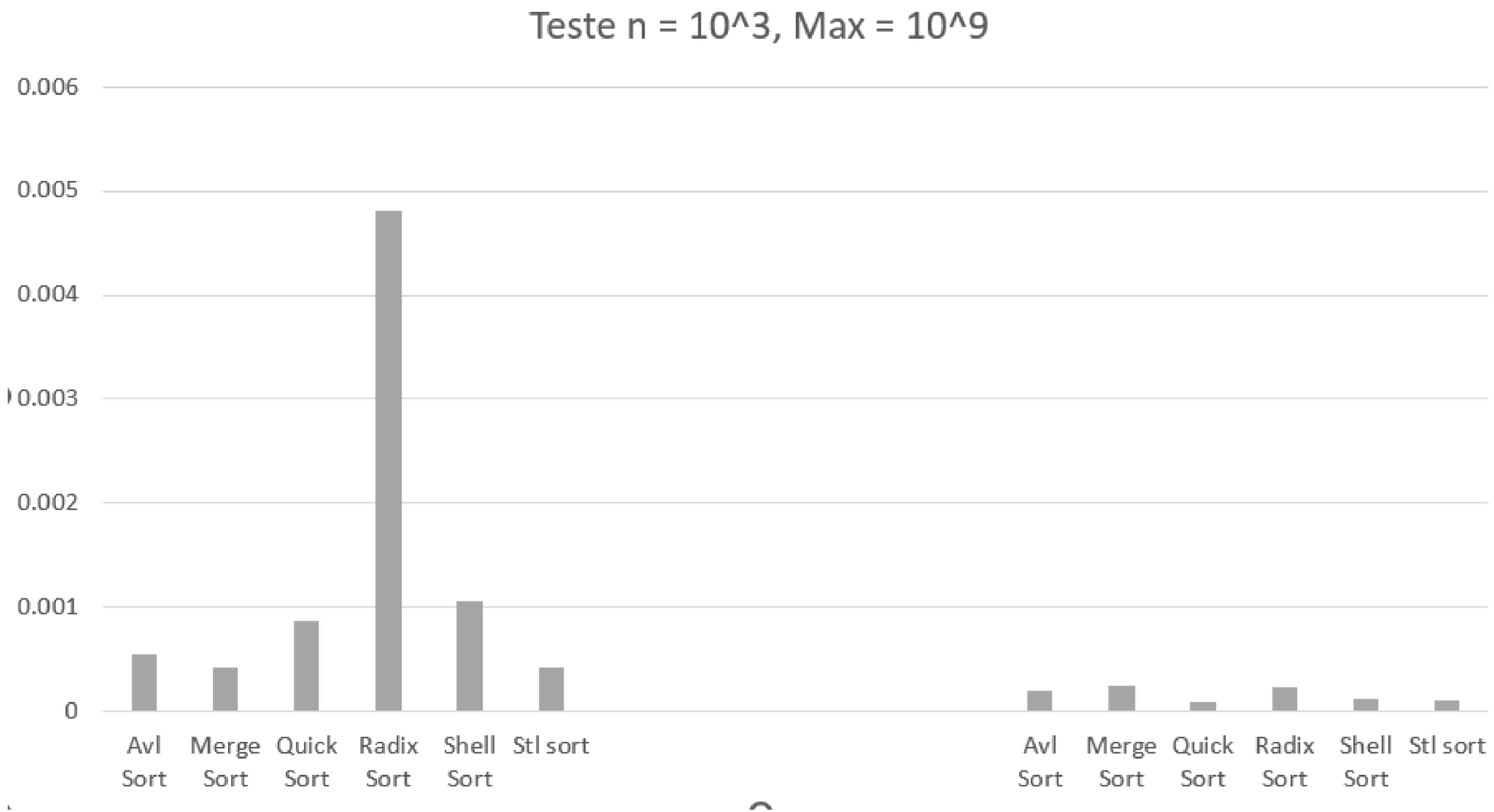
Worst case

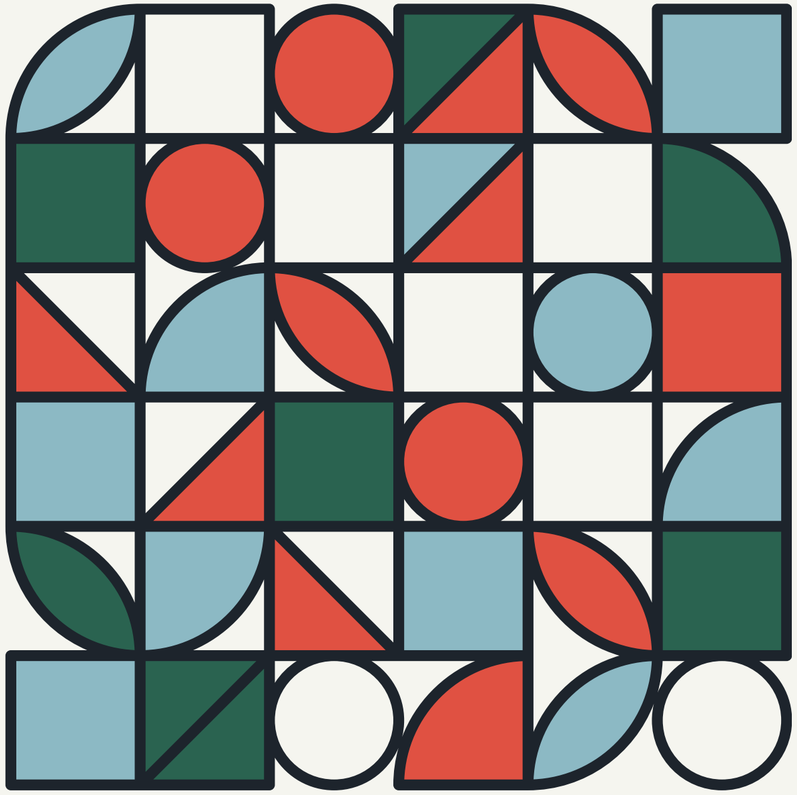
Best Case

Teste n = 10^3, Max = 10^9

Worst case

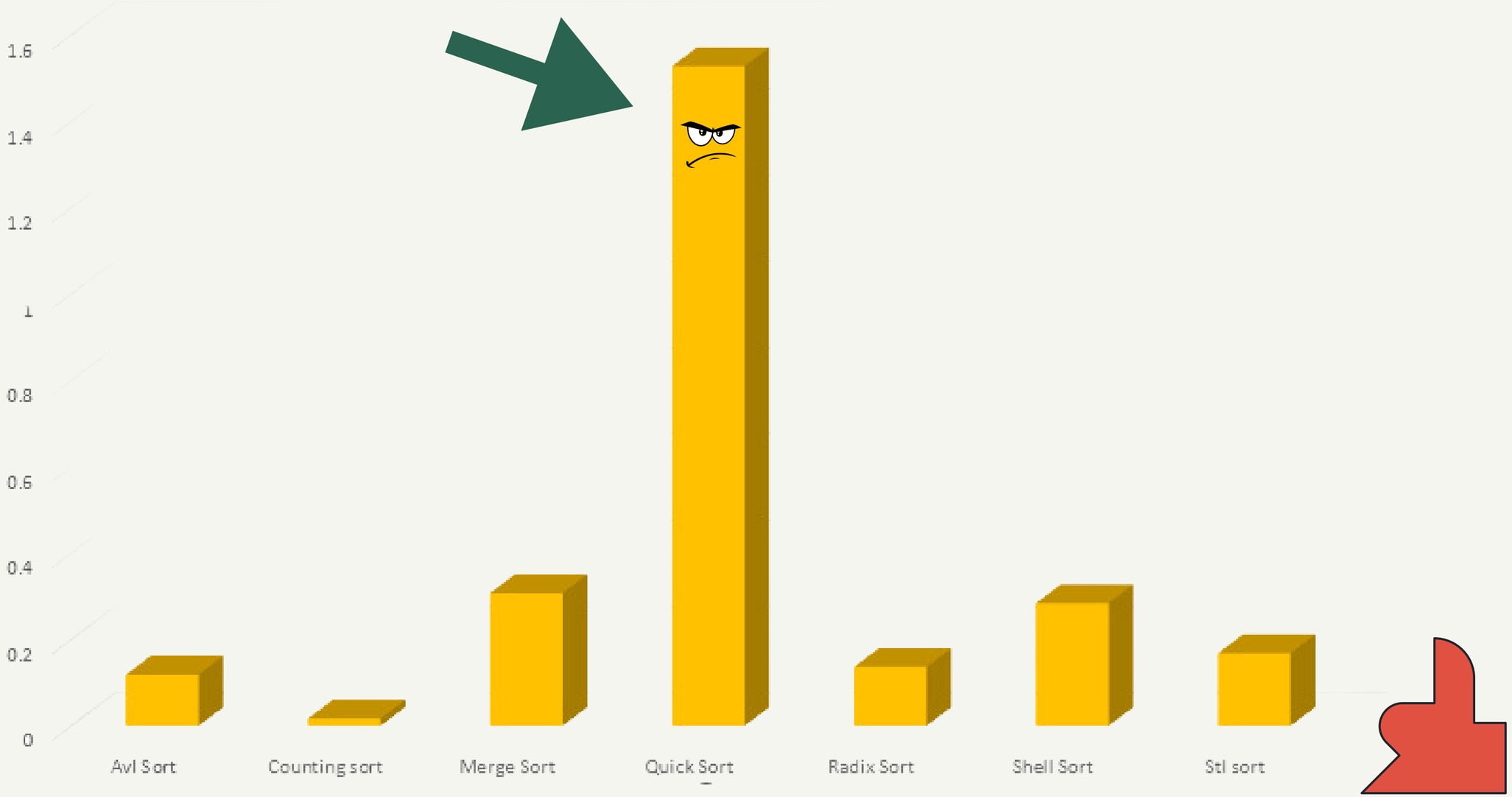
Best Case





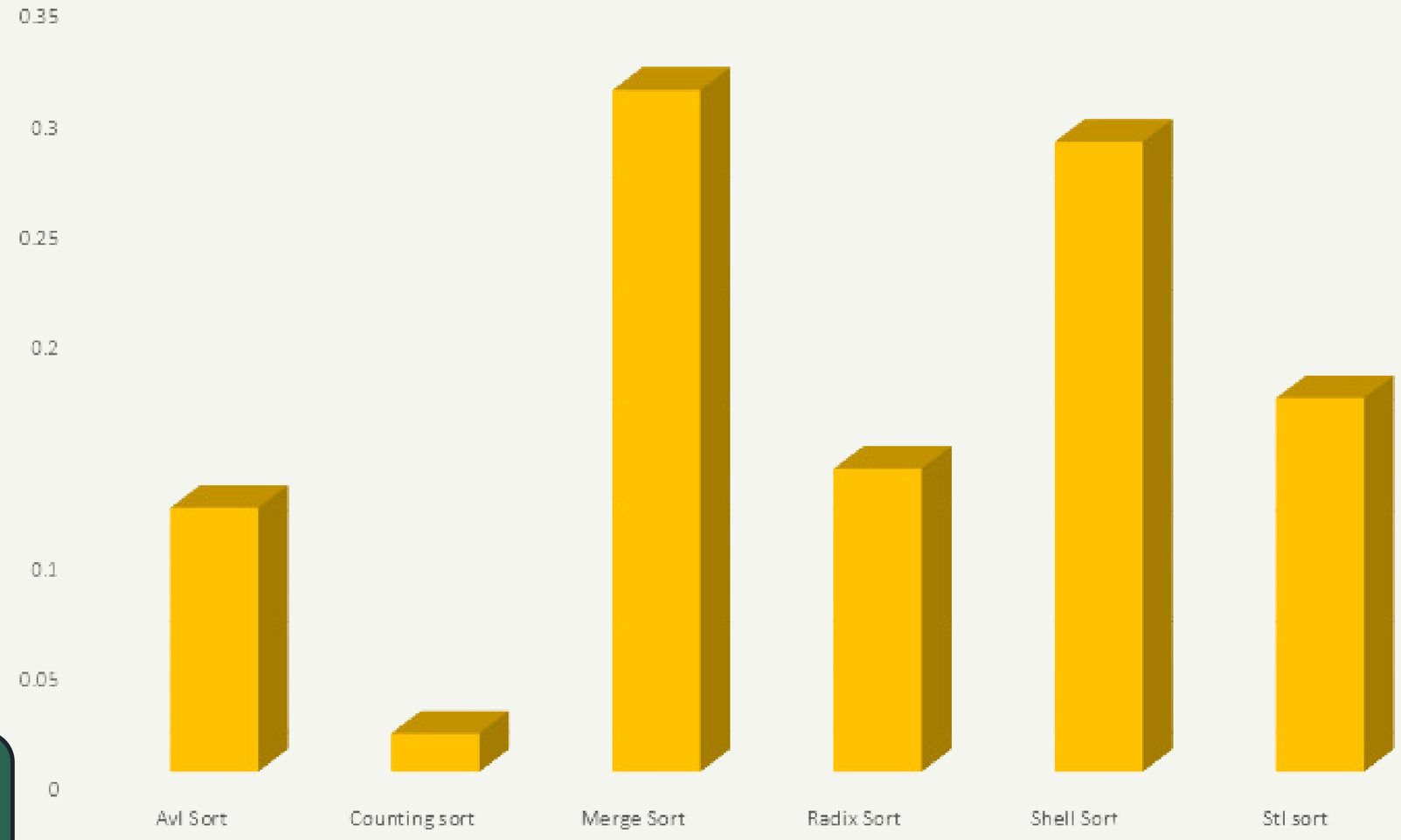
MAI MULTE NUMERE?

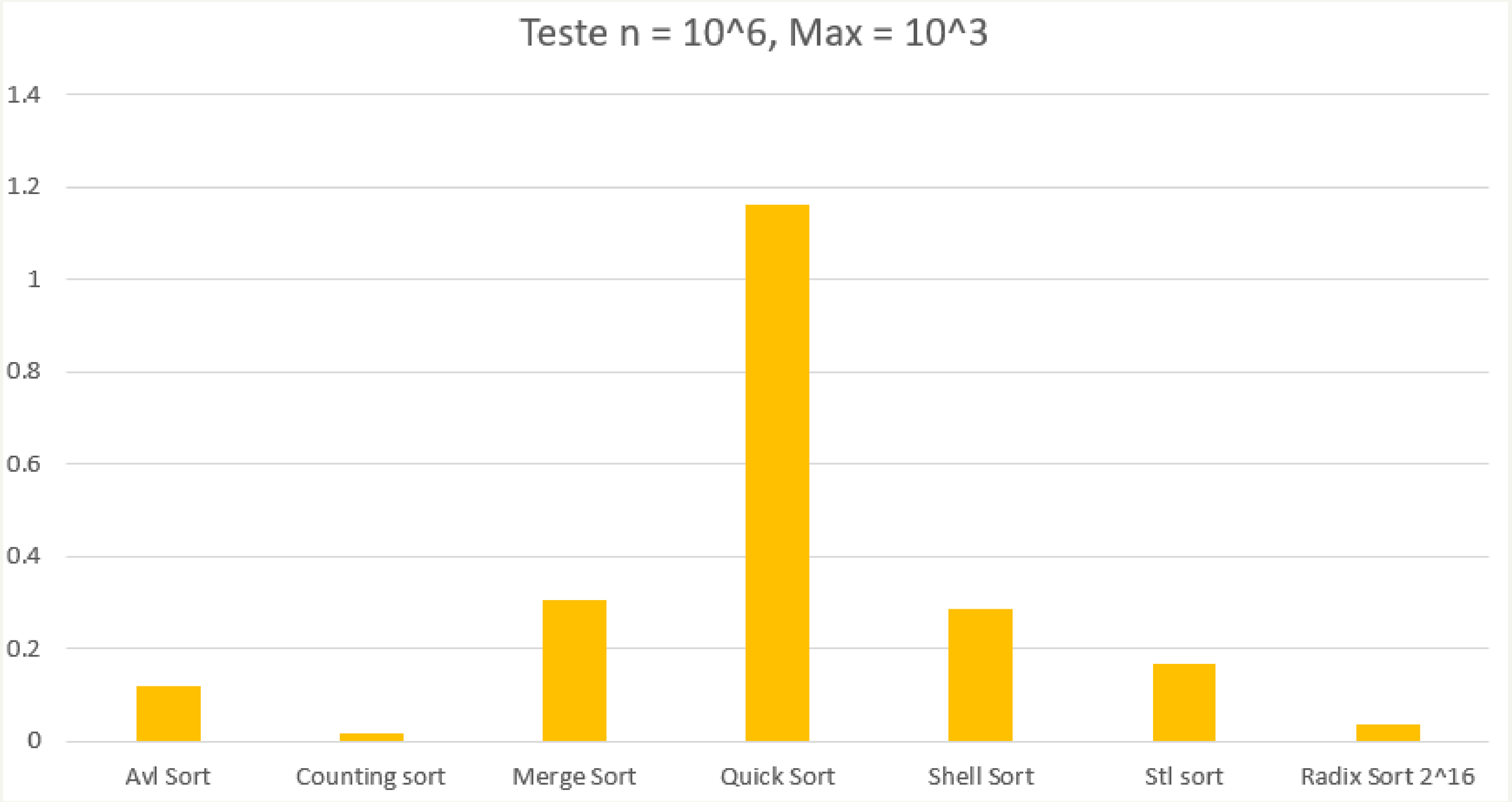
Teste n = 10^6, Max = 10^3



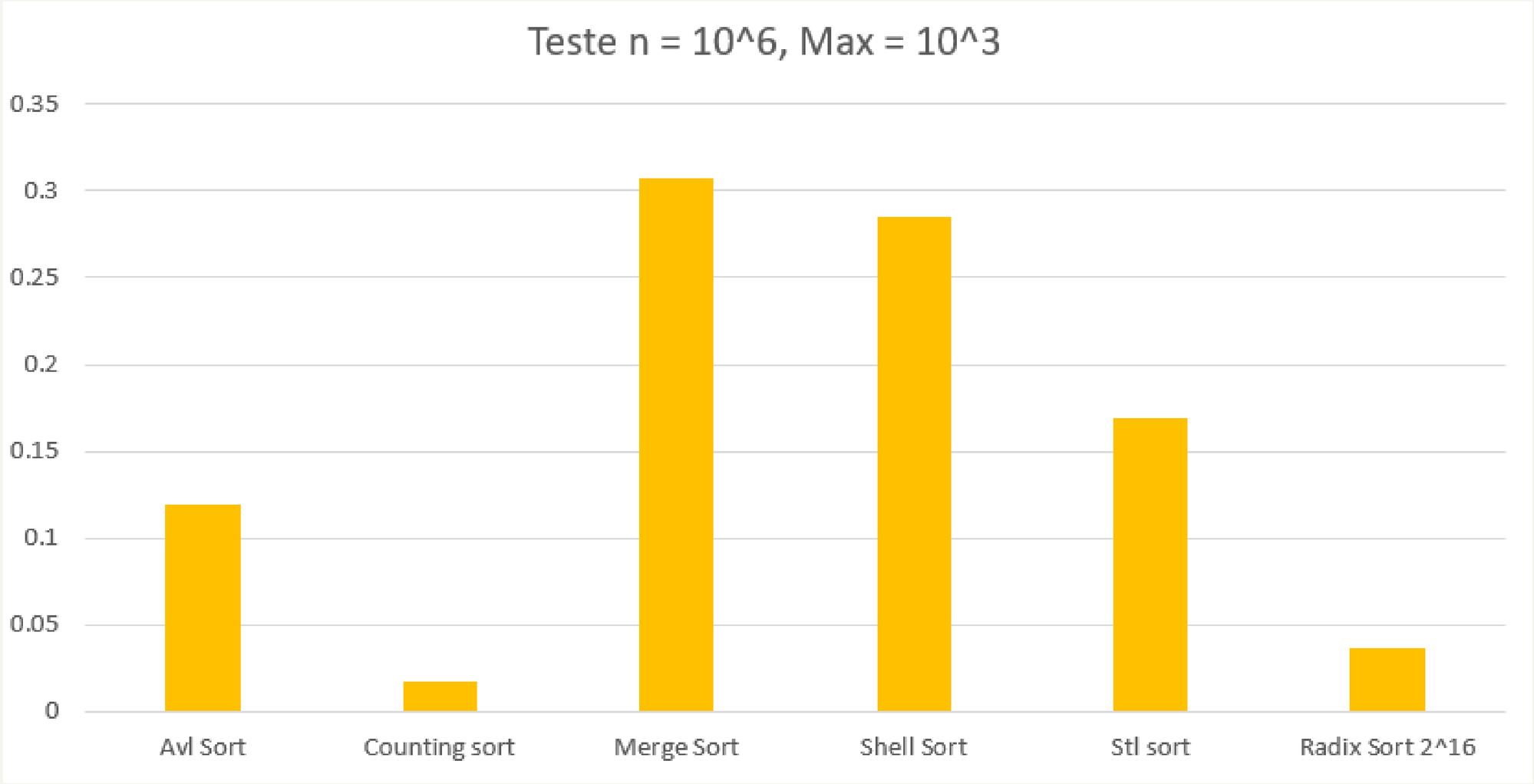
Teste n = 10^6, Max = 10^3

fără Quick Sort




Teste n = 10^6, Max = 10^3
1.4
1.2
1
0.8
0.6
0.4
0.2
0
Avl Sort
Counting sort
Merge Sort
Quick Sort
Shell Sort
Stl sort
Radix Sort 2^16

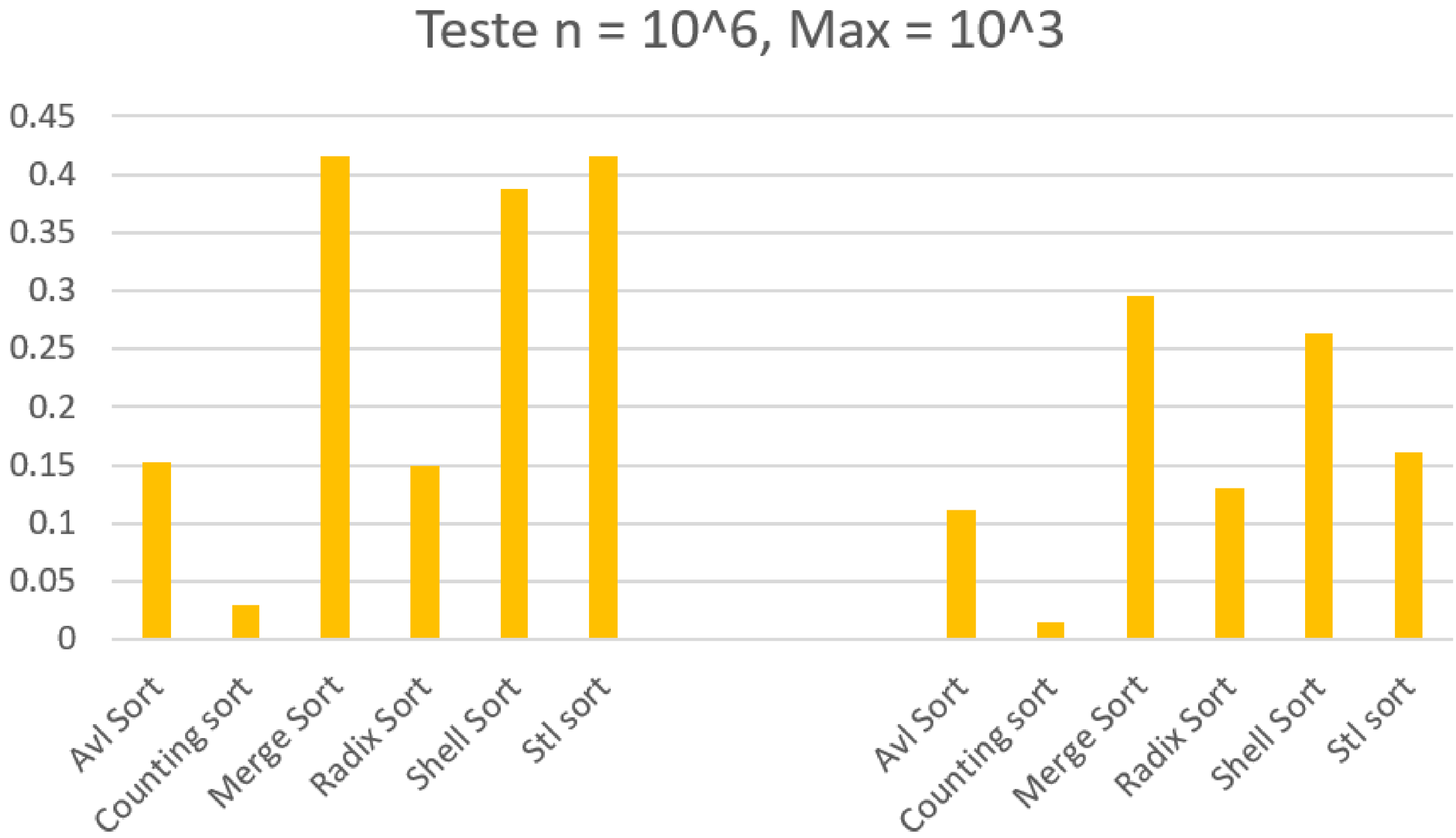
fara quick sort

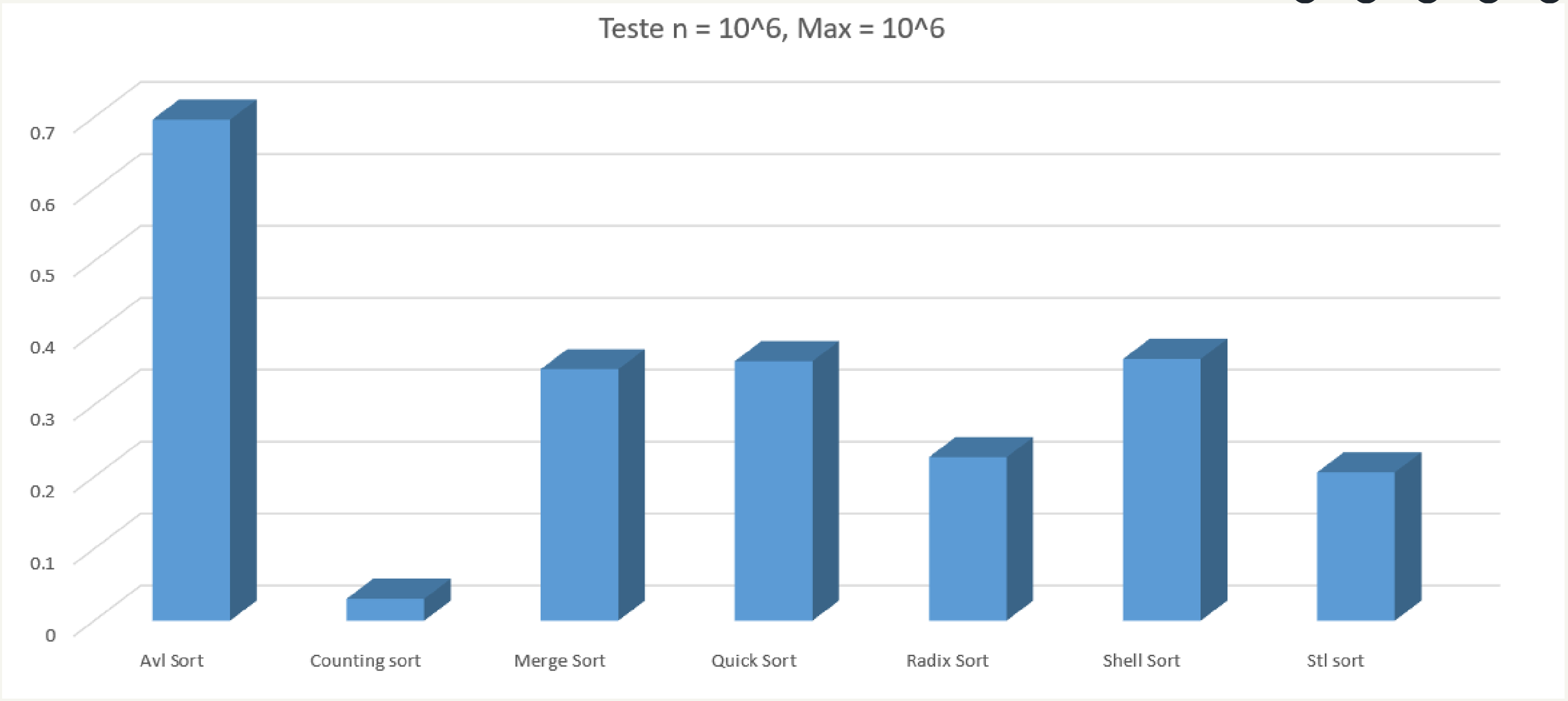


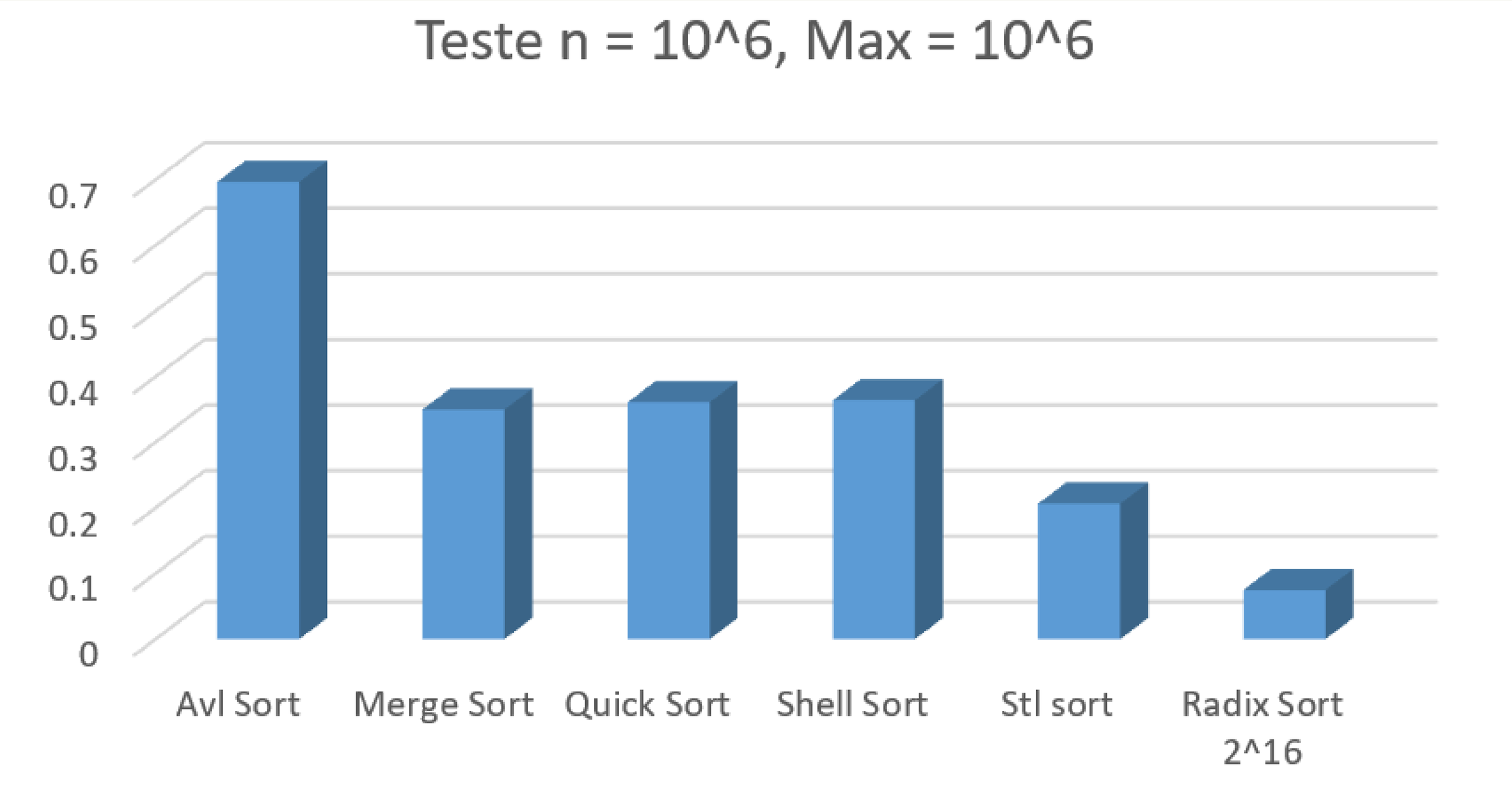
fara quick sort

Worst case

Best Case



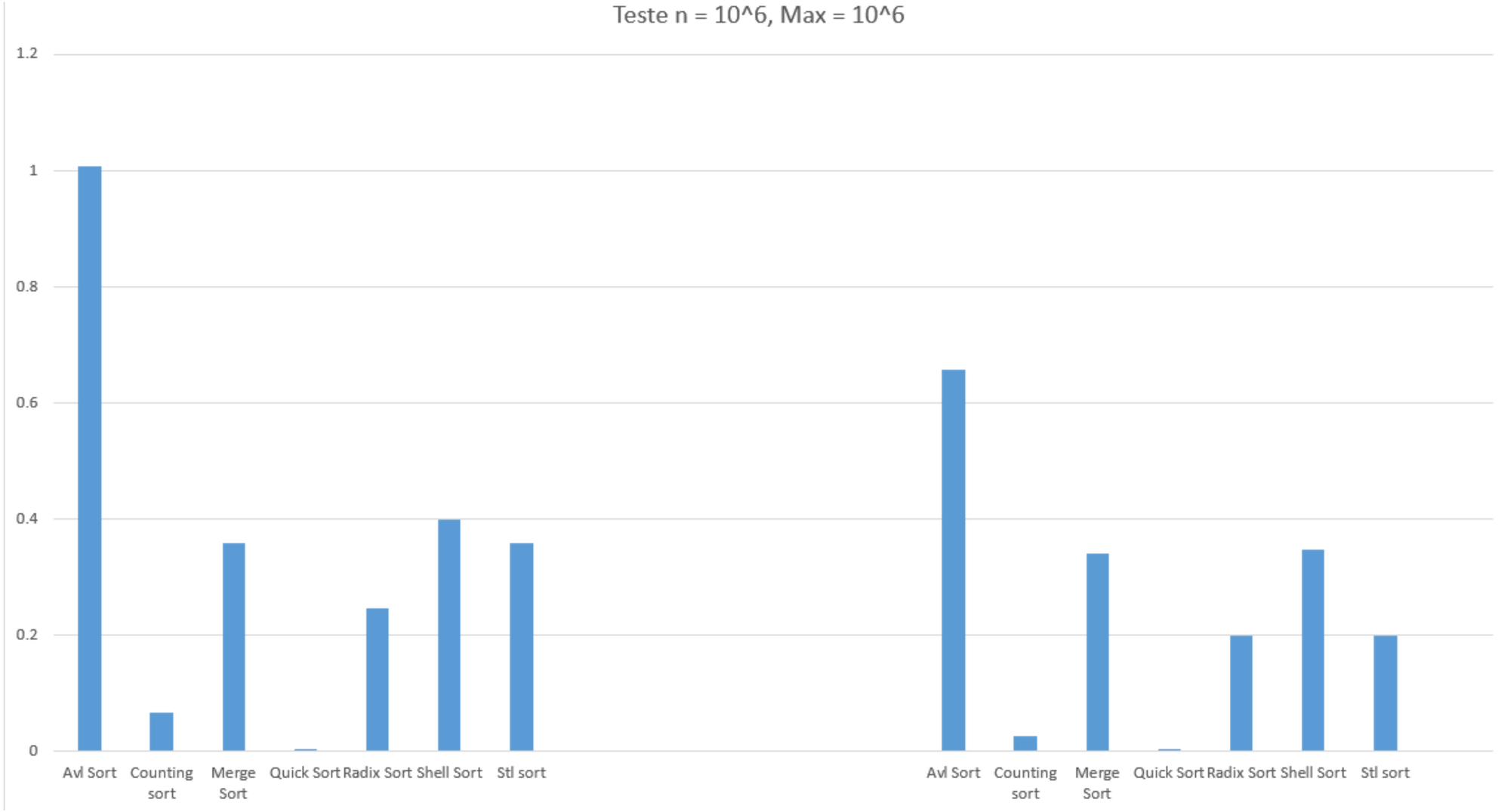

Teste n = 10^6, Max = 10^6
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Avl Sort
Counting sort
Merge Sort
Quick Sort
Radix Sort
Shell Sort
Stl sort


Teste n = 10^6, Max = 10^6
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Avl Sort
Merge Sort
Quick Sort
Shell Sort
Stl sort
Radix Sort 2^16

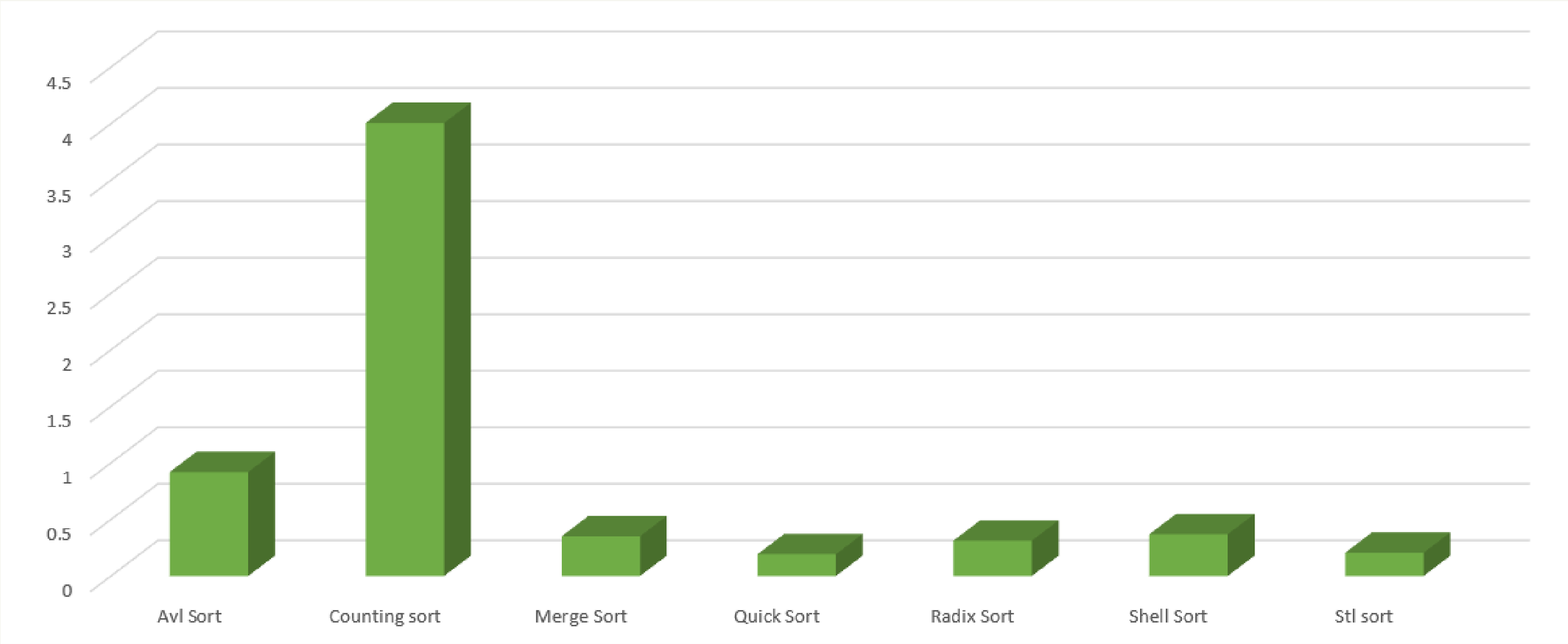
radix sort

Worst case

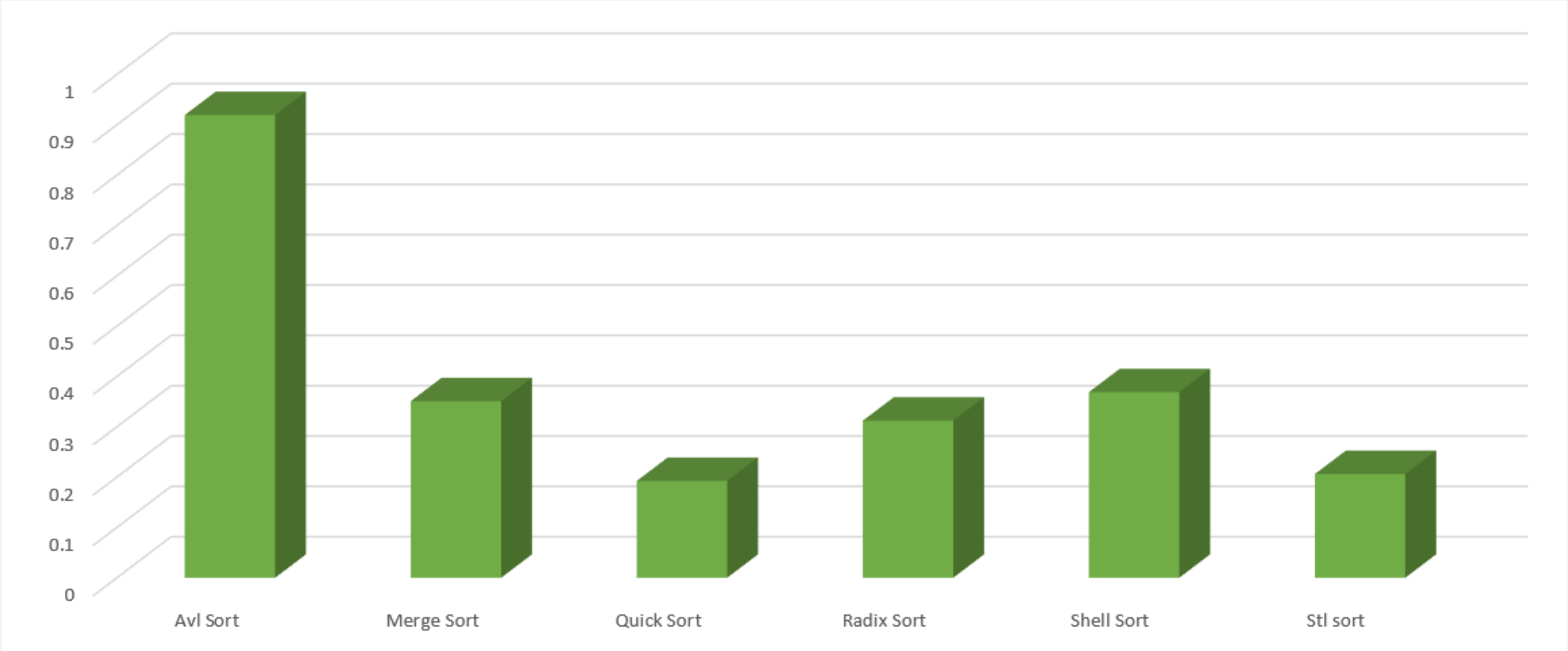
Best Case

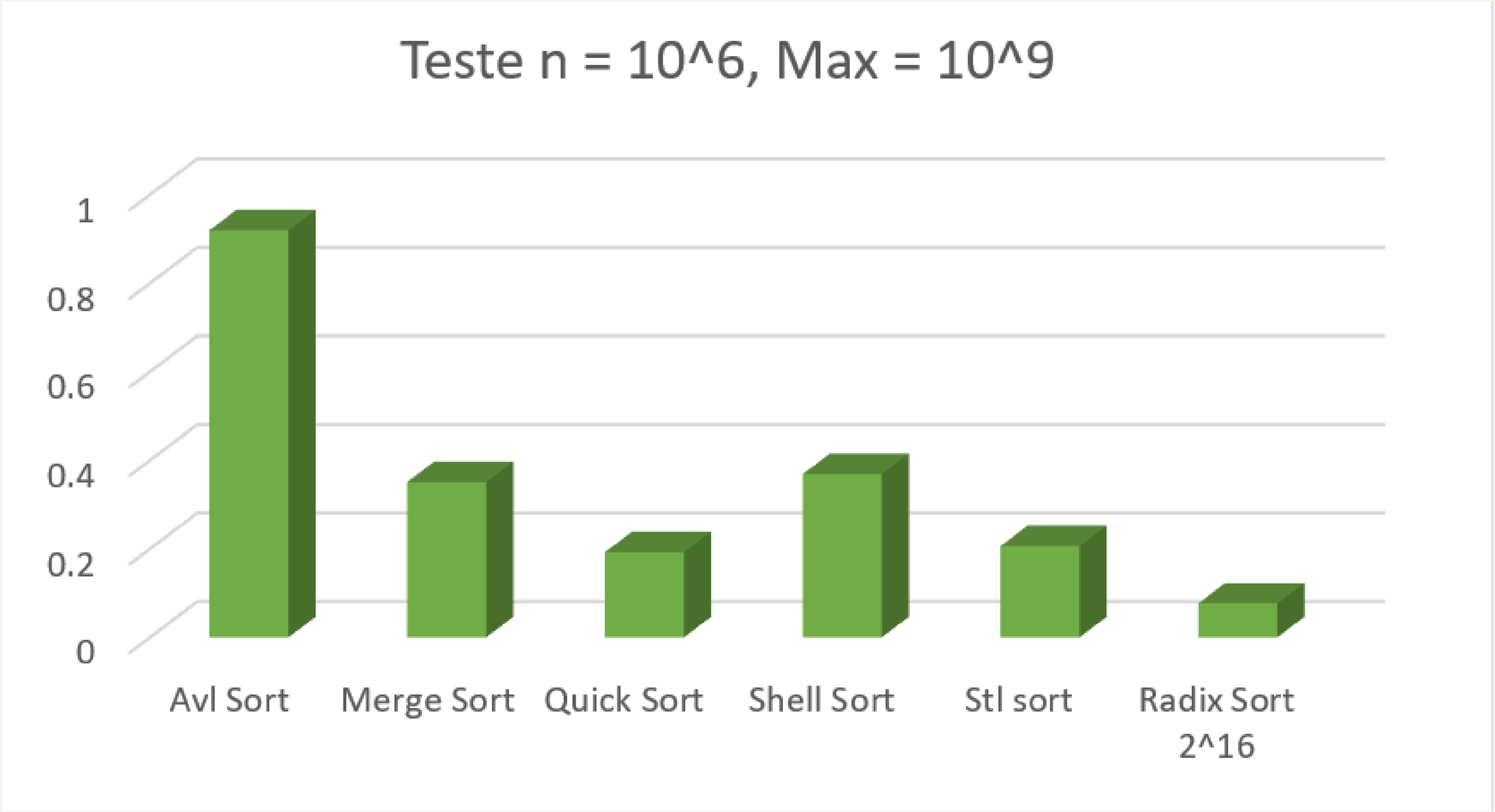


Teste n = 10^6, Max = 10^9



fără counting Sort

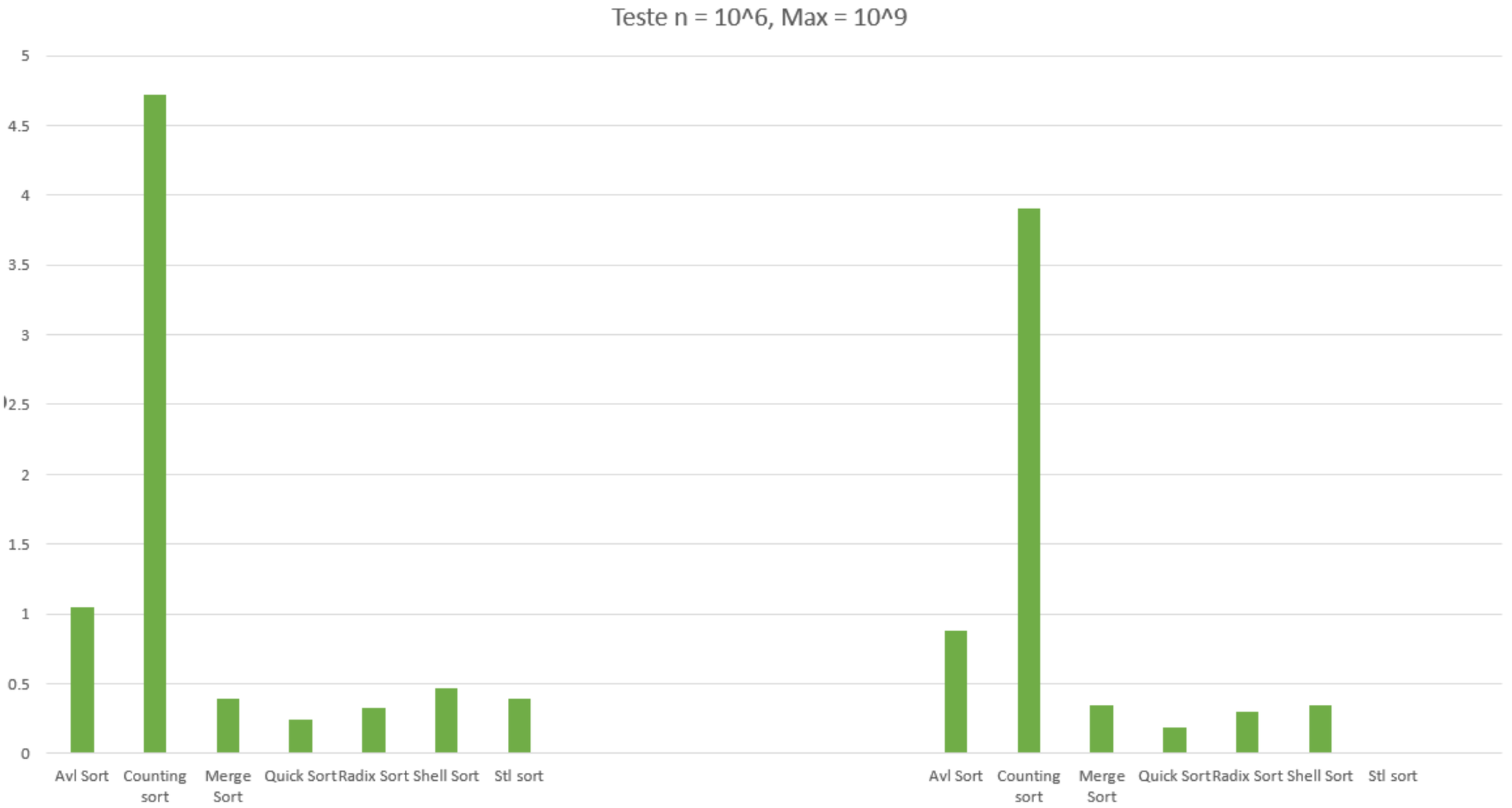



Teste n = 10^6, Max = 10^9
1
0.8
0.6
0.4
0.2
0
Avl Sort
Merge Sort
Quick Sort
Shell Sort
Stl sort
Radix Sort 2^16

radix sort

Worst case

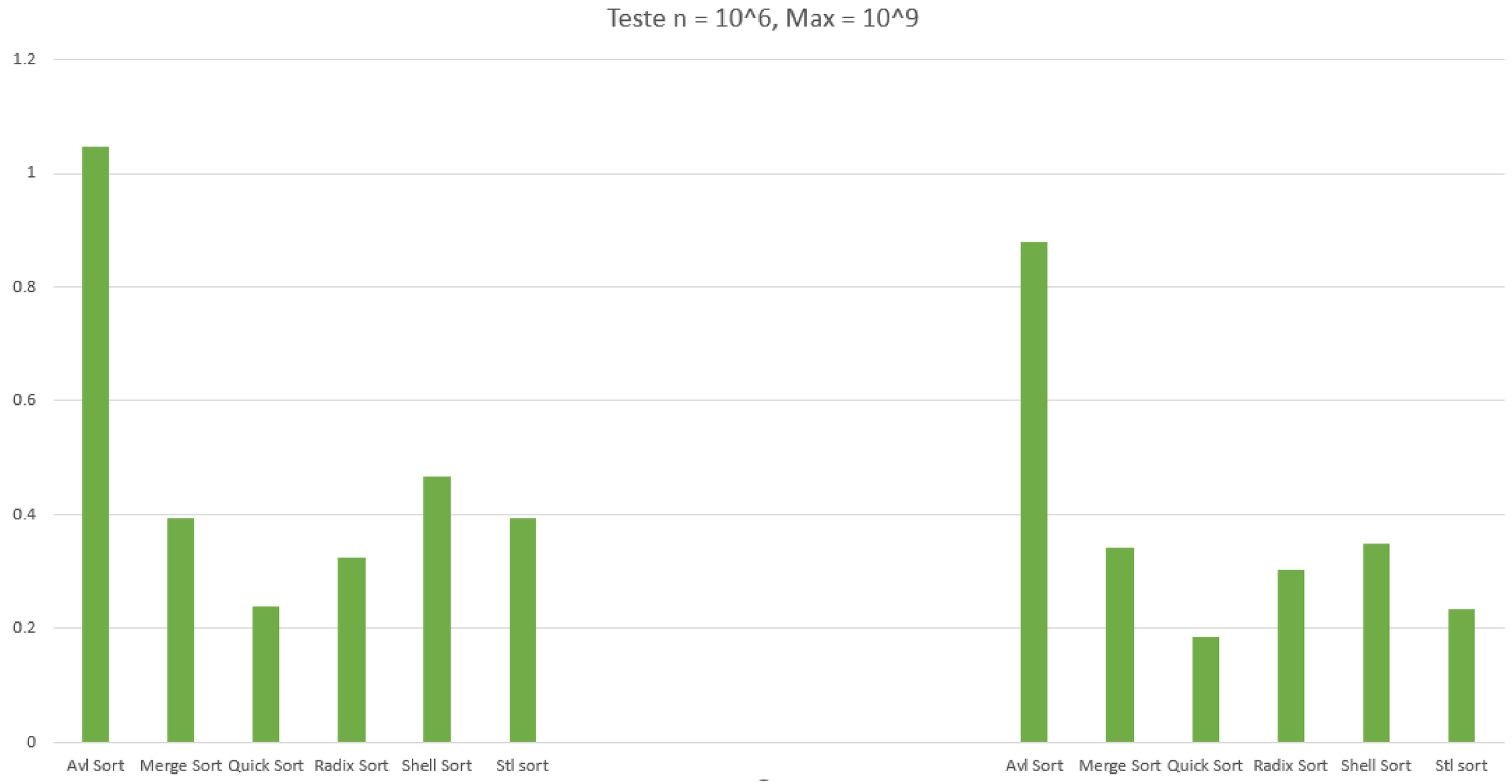
Best Case



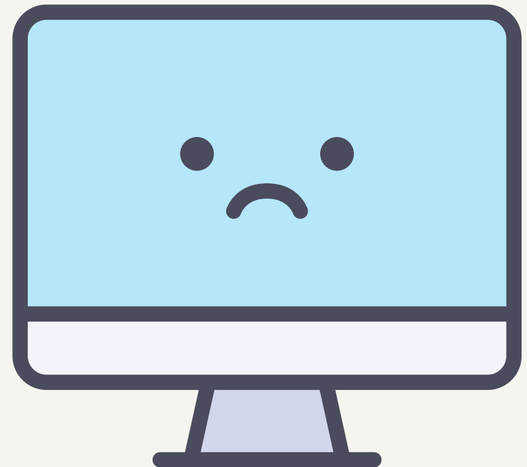
fara Counting Sort

Worst case

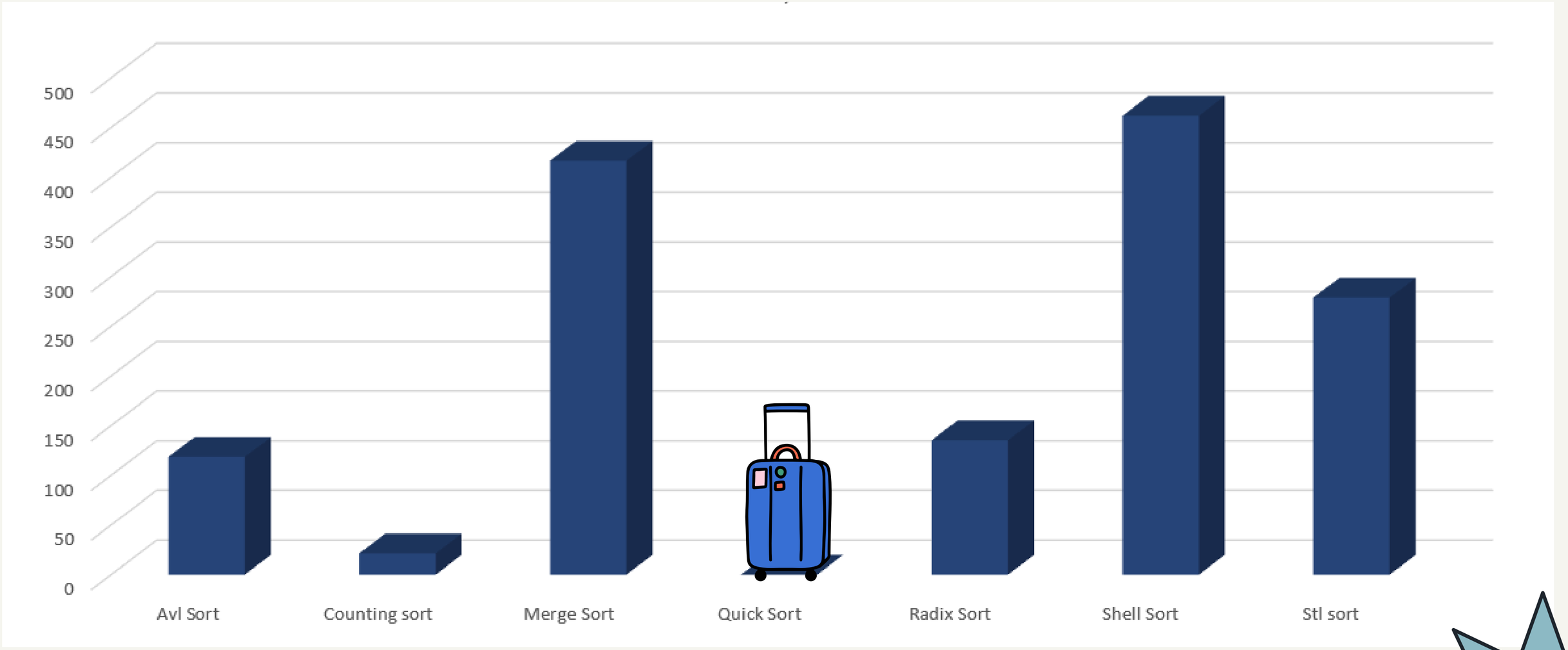
Best Case



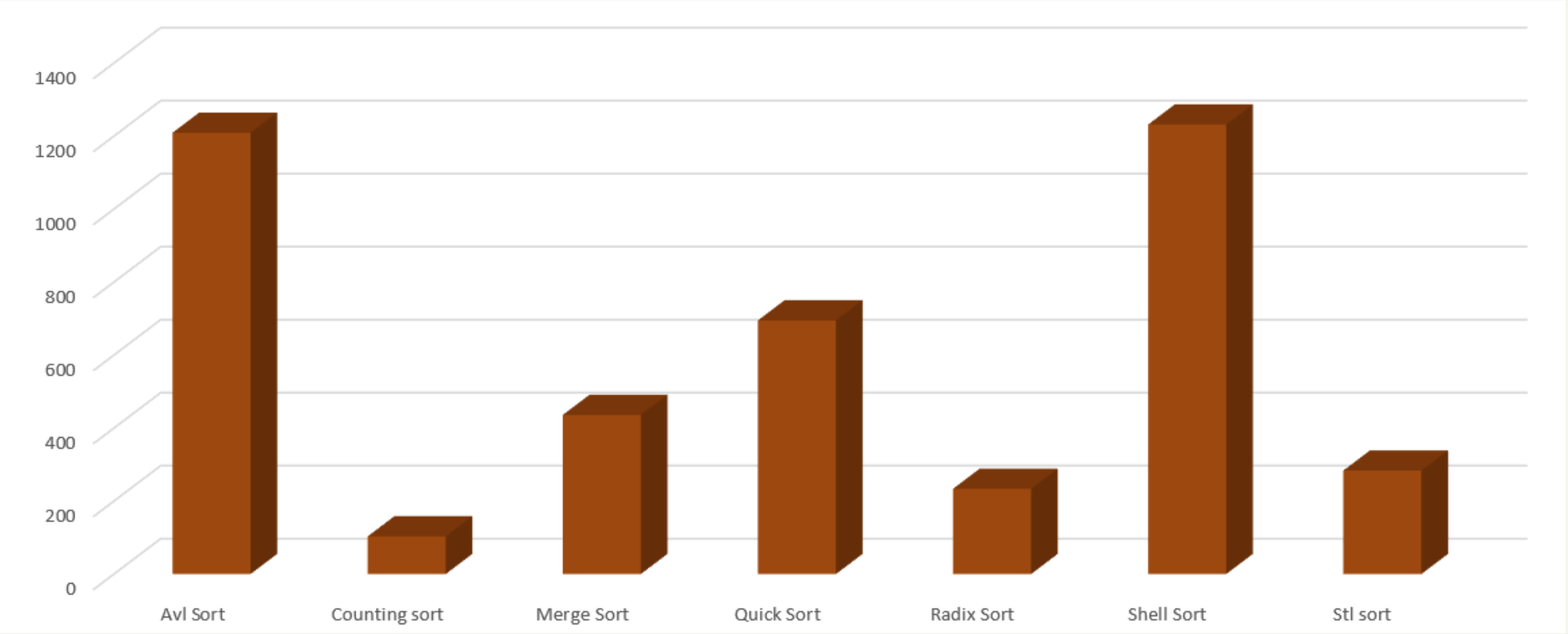
SI MAI MULTE NUMERE..



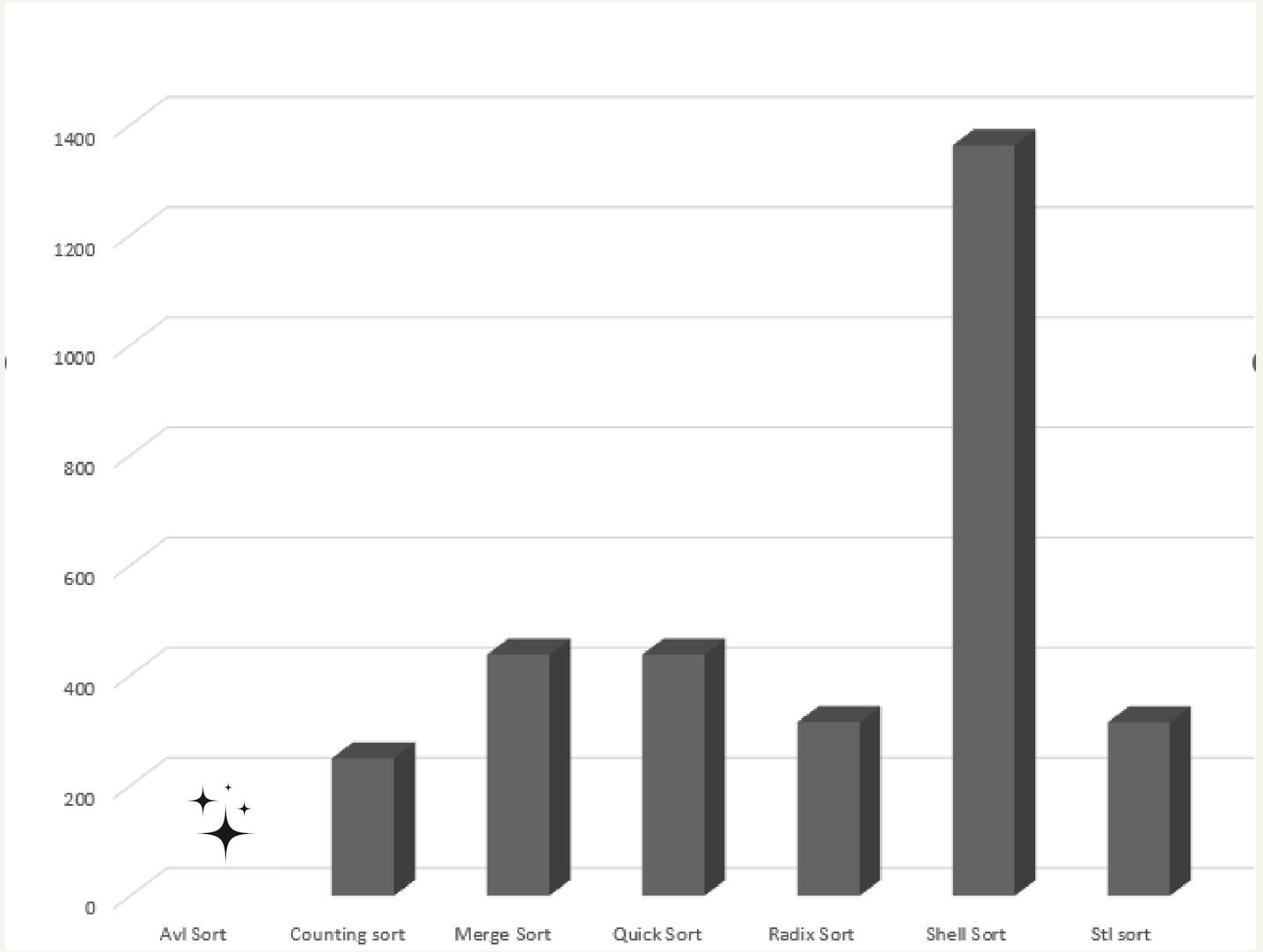
Teste n = 10^9, Max = 10^3



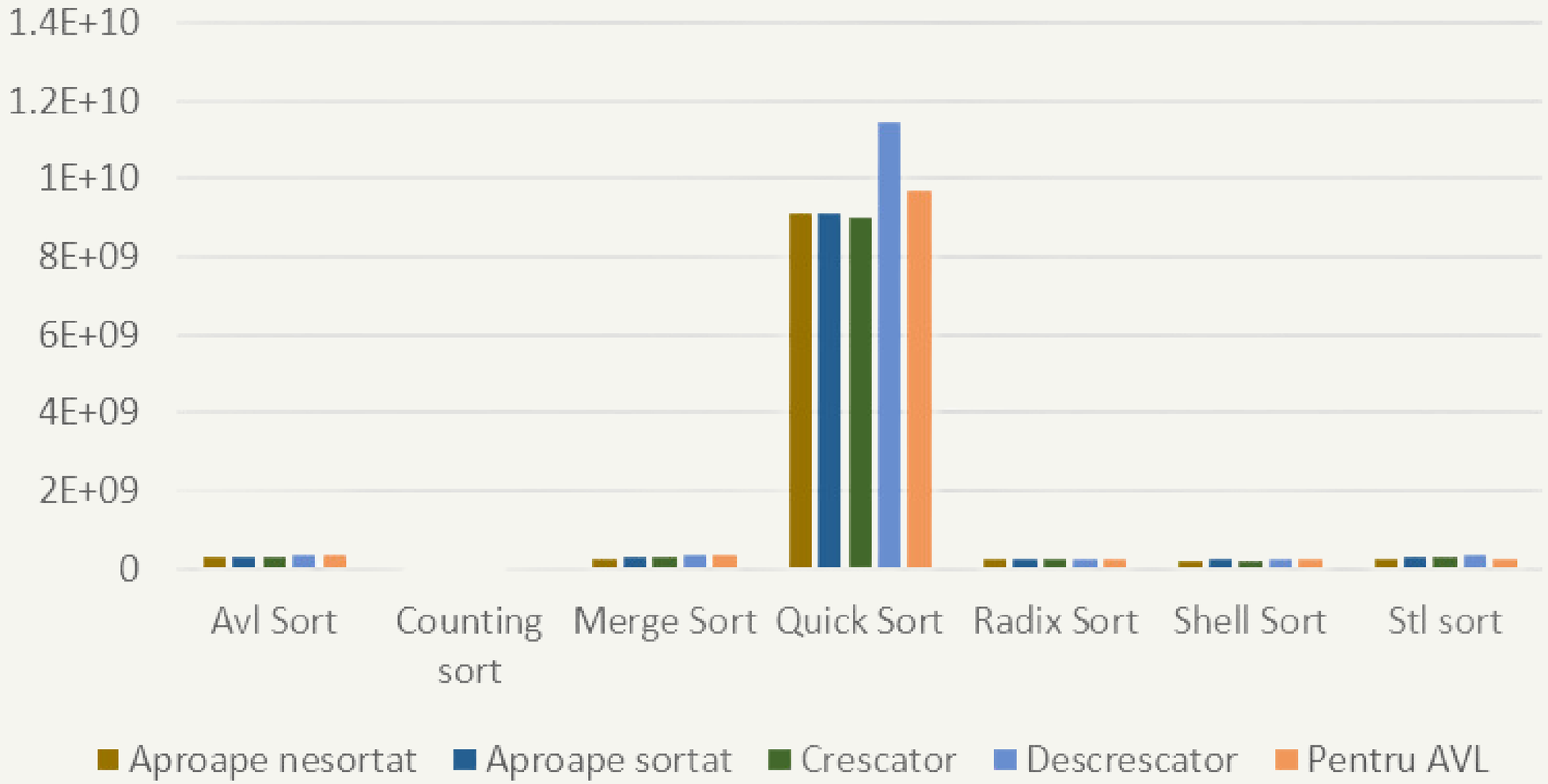
Teste n = 10^9, Max = 10^6



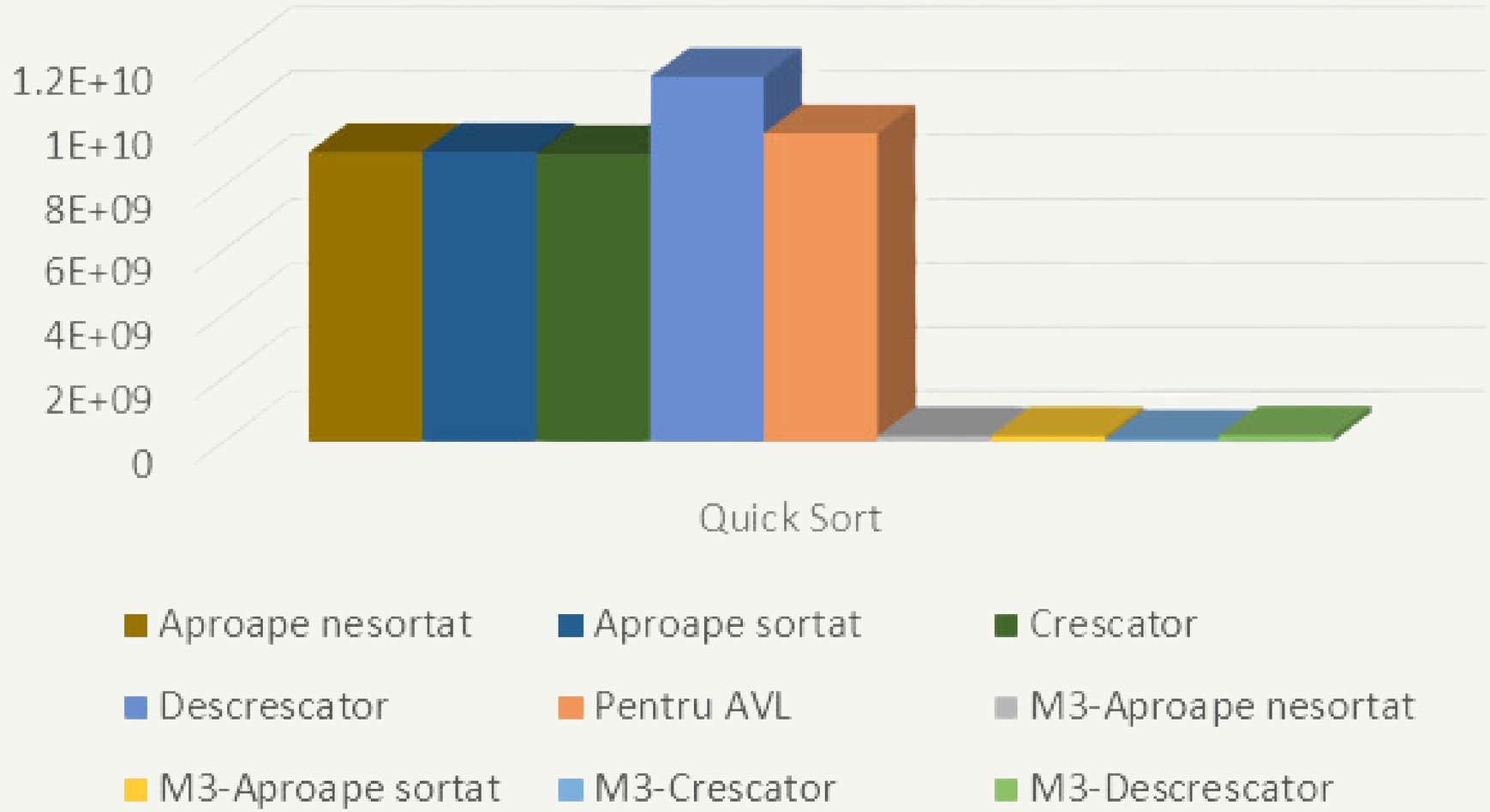
Teste n = 10^9, Max = 10^9



CAZURI SPECIALE



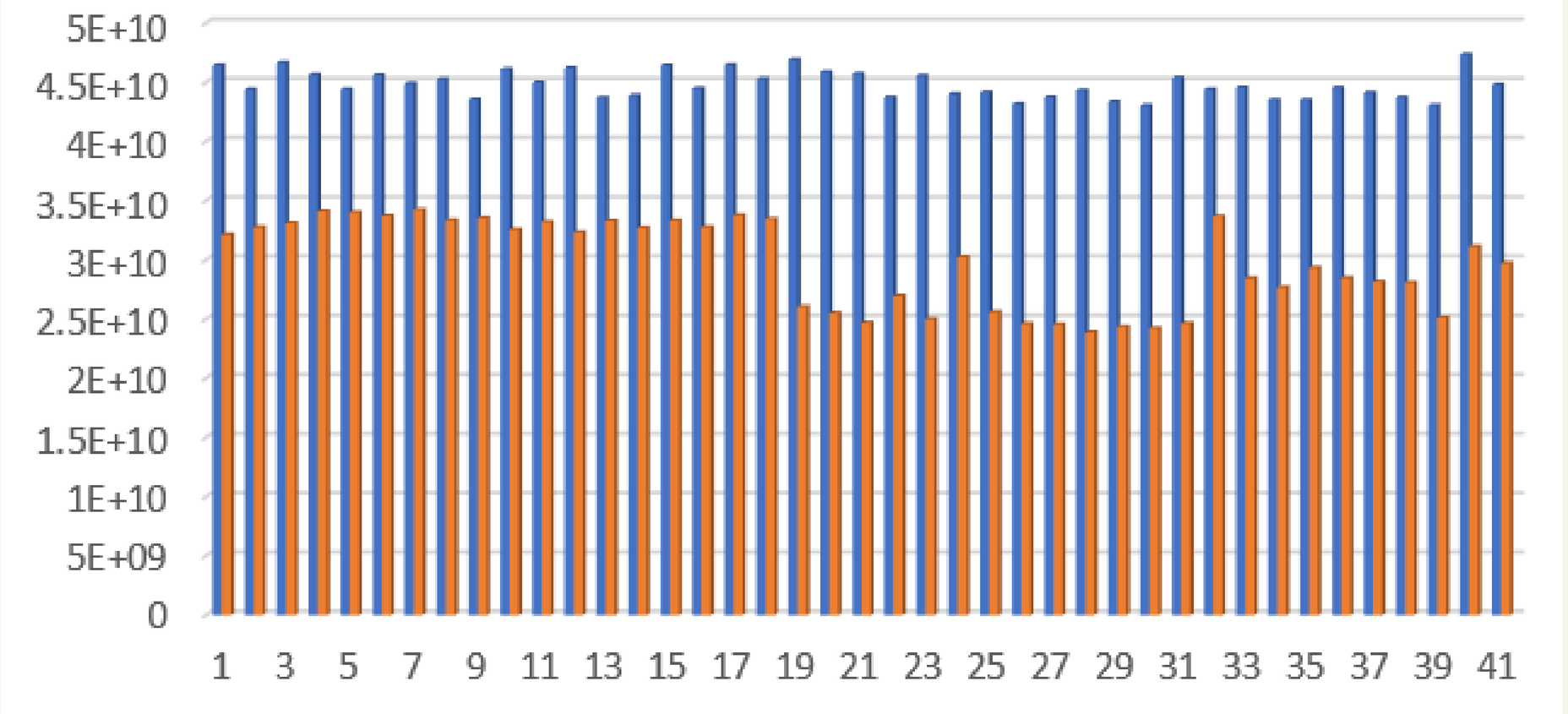
QUICK SORT PIVOT CAPĂT VS M3



TESTE N=10^6, MAX=10^6

Quick Sort

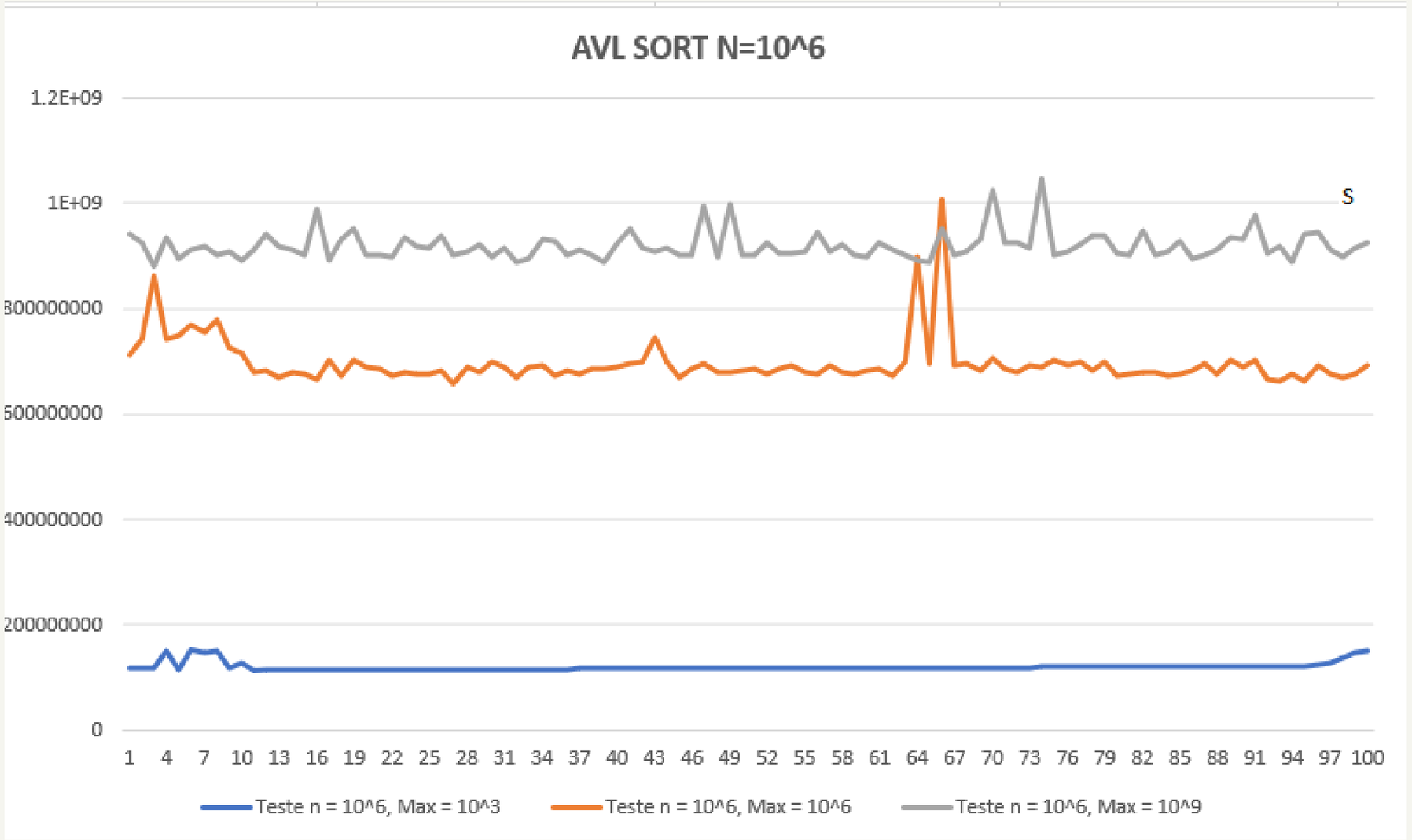
n=10^8 max=10^9



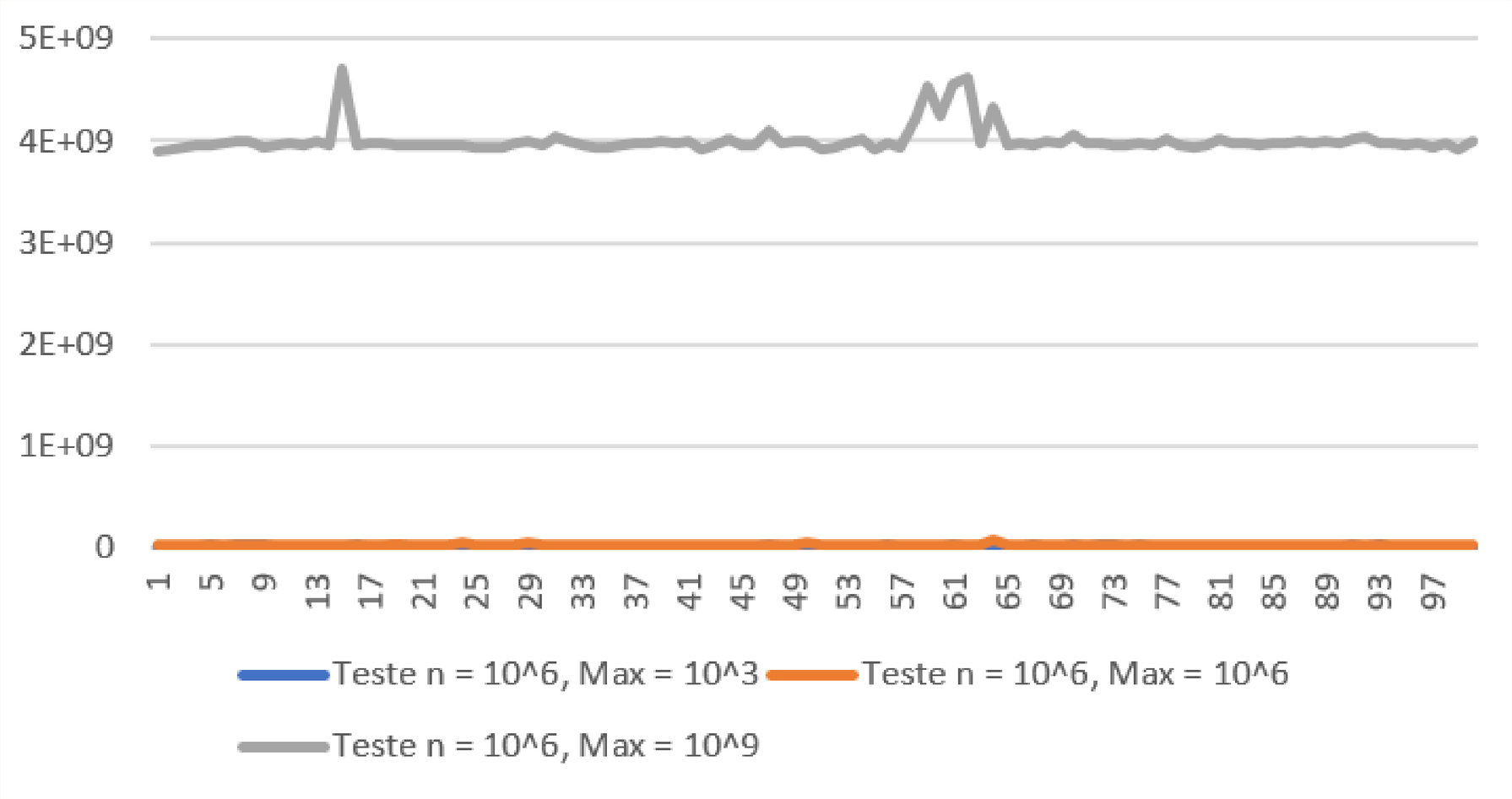
Merge Sort

Quick Sort

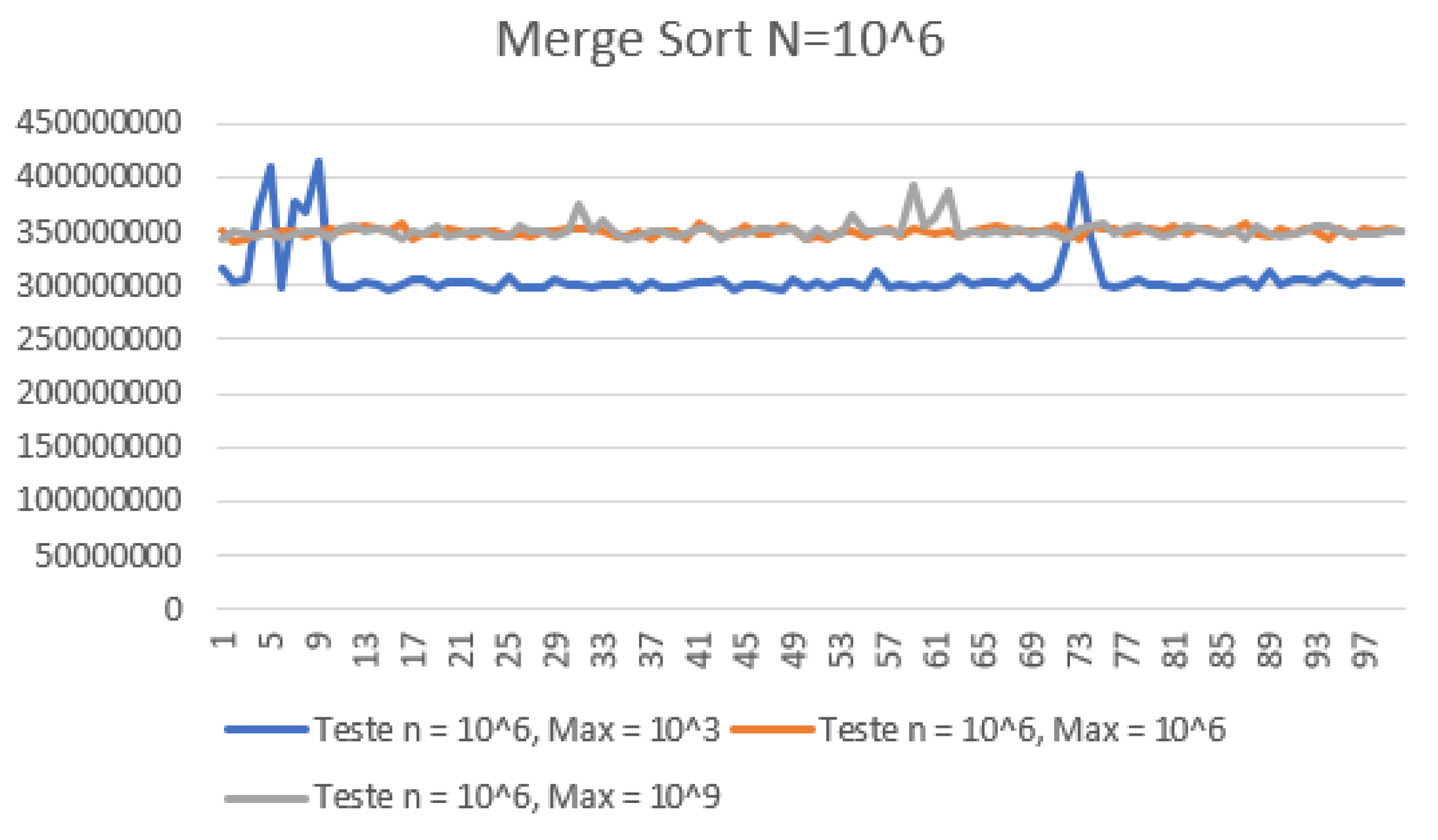
AVL SORT


AVL SORT N=10^6
1.2E+09
1E+09
800000000
600000000
400000000
200000000
0
1 4 7 10 13 16 19 22 25 28 31 34 37 40 43 46 49 52 55 58 61 64 67 70 73 76 79 82 85 88 91 94 97 100
Teste n = 10^6, Max = 10^3
Teste n = 10^6, Max = 10^6
Teste n = 10^6, Max = 10^9

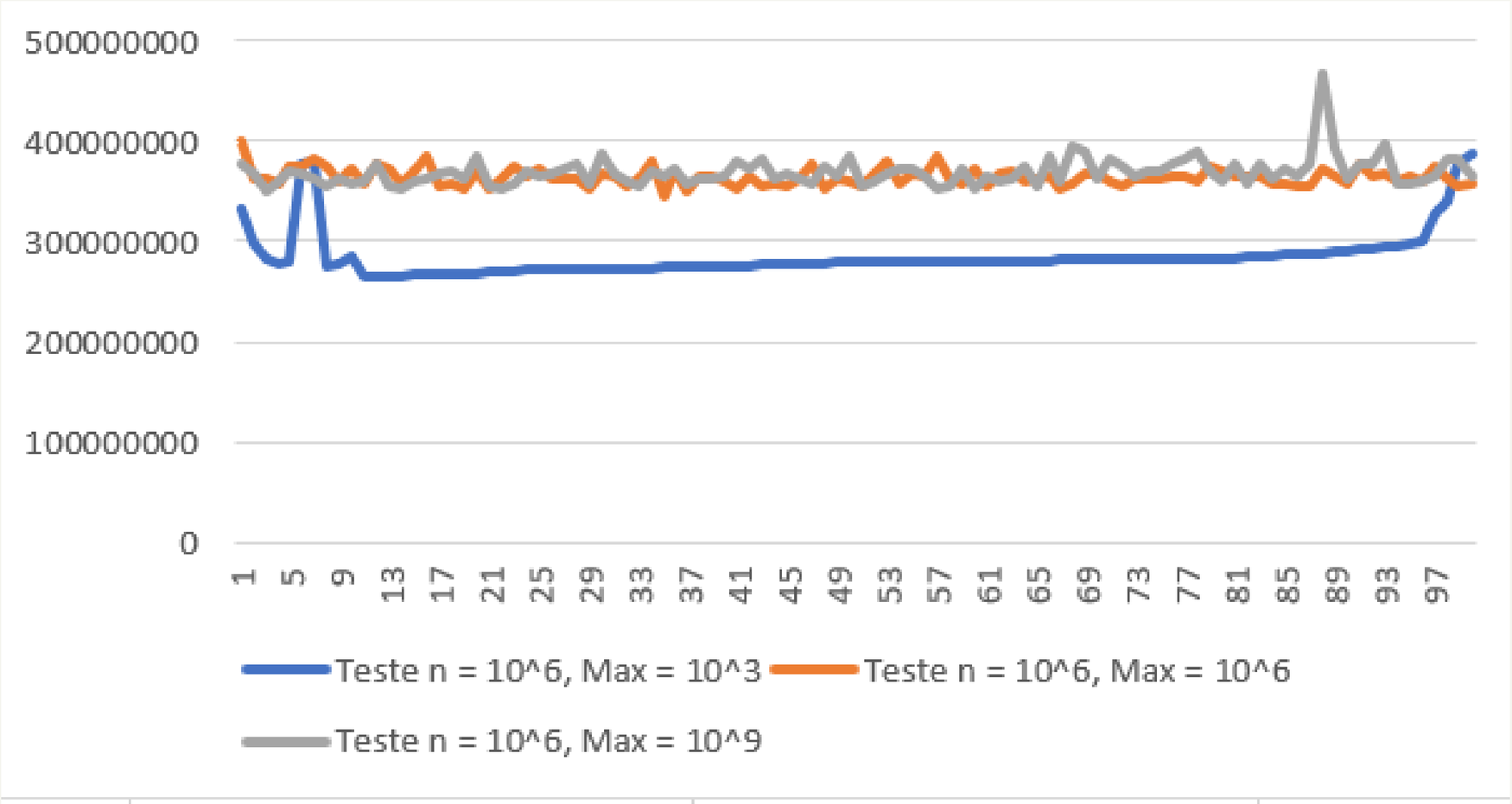
COUNTING SORT



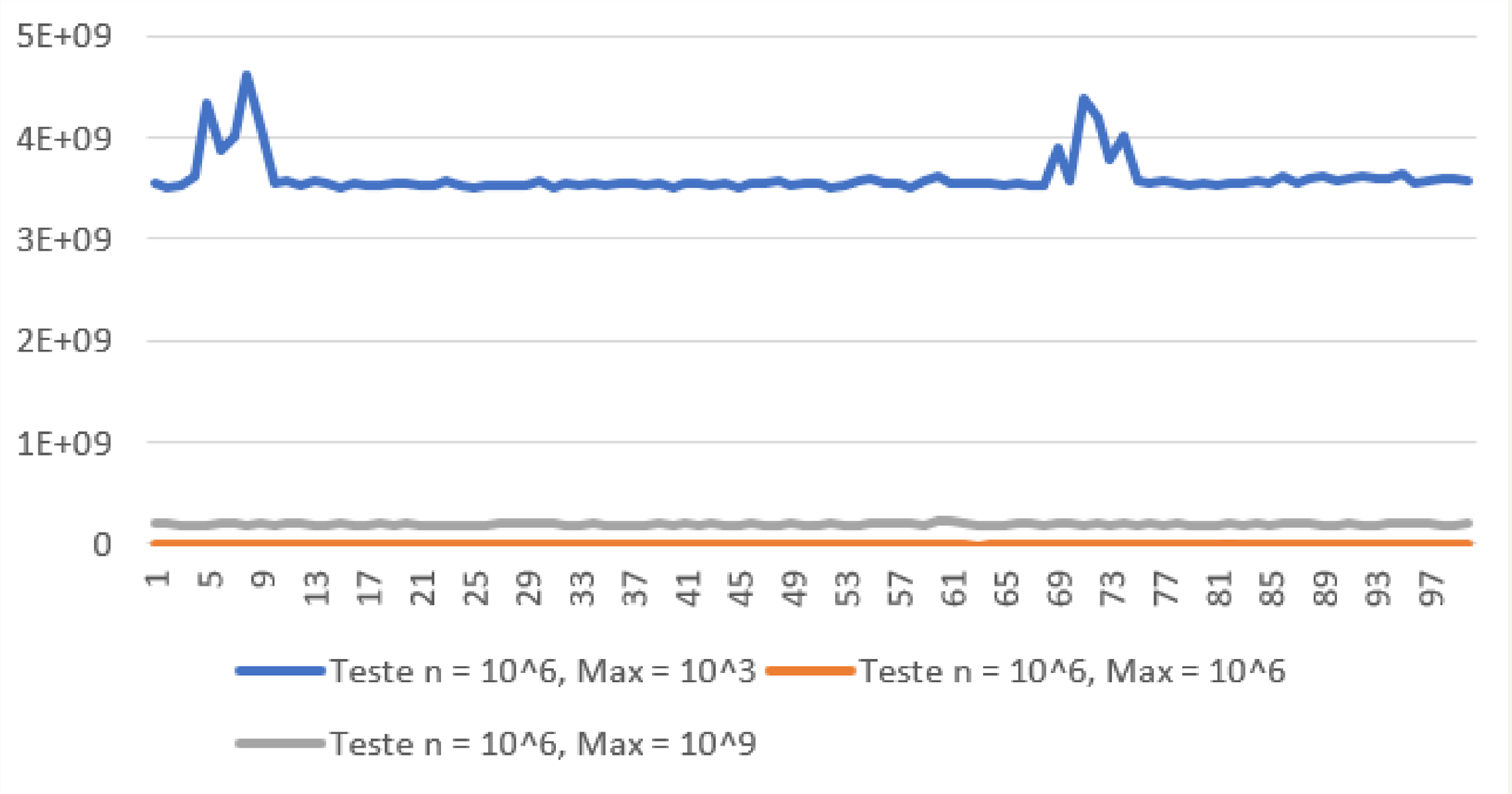
MERGE SORT


Merge Sort N=10^6
450000000
400000000
350000000
300000000
250000000
200000000
150000000
100000000
50000000
0
1
5
9
13
17
21
25
29
33
37
41
45
49
53
57
61
65
69
73
77
81
85
89
93
97
Teste n = 10^6, Max = 10^3
Teste n = 10^6, Max = 10^6
Teste n = 10^6, Max = 10^9

SHELL SORT

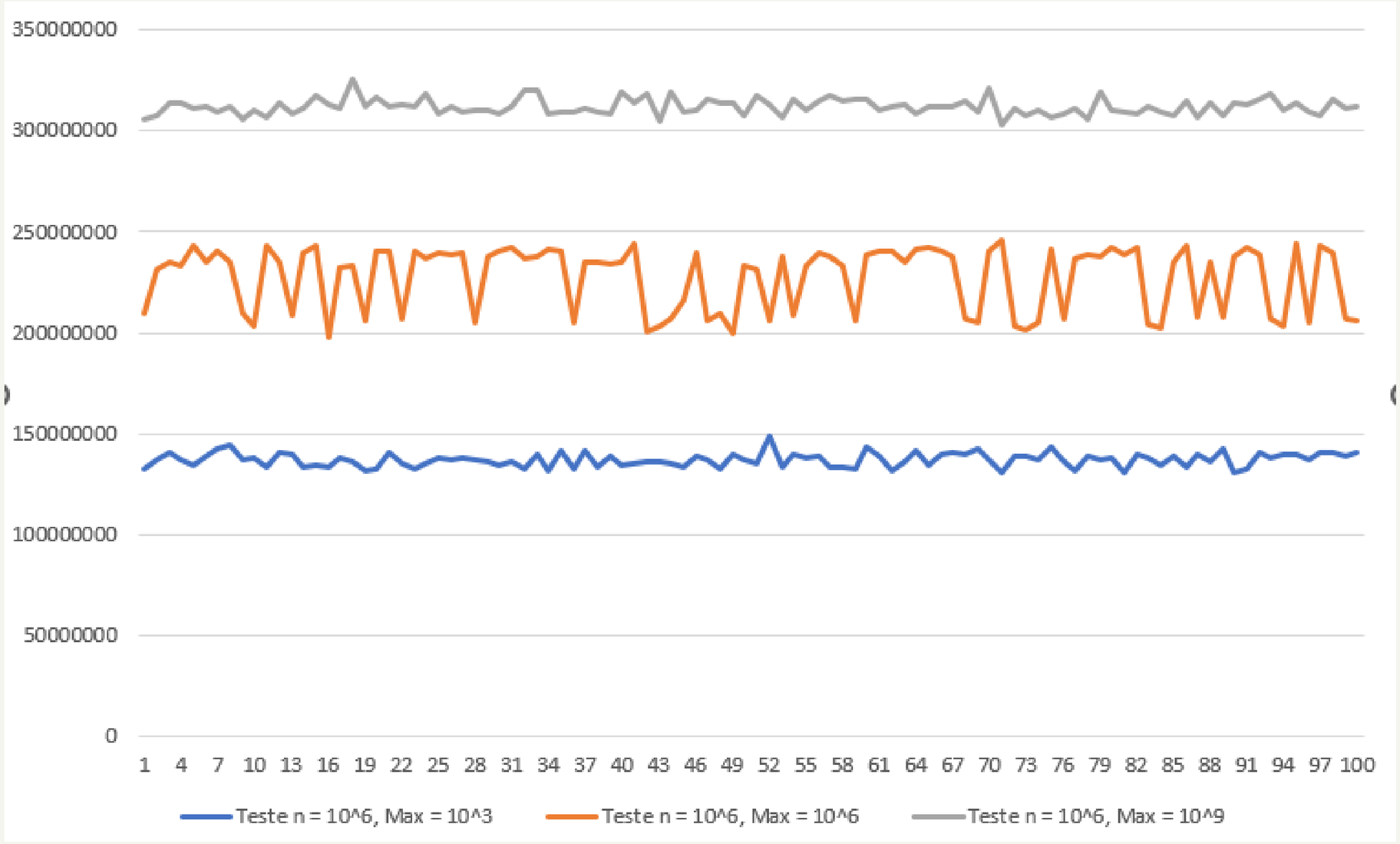

500000000
400000000
300000000
200000000
100000000
0
1
5
9
13
17
21
25
29
33
37
41
45
49
53
57
61
65
69
73
77
81
85
89
93
97
Teste n = 10^6, Max = 10^3
Teste n = 10^6, Max = 10^6
Teste n = 10^6, Max = 10^9

QUICK SORT


5E+09
4E+09
3E+09
2E+09
1E+09
0
1
5
9
13
17
21
25
29
33
37
41
45
49
53
57
61
65
69
73
77
81
85
89
93
97
Teste n = 10^6, Max = 10^3
Teste n = 10^6, Max = 10^6
Teste n = 10^6, Max = 10^9

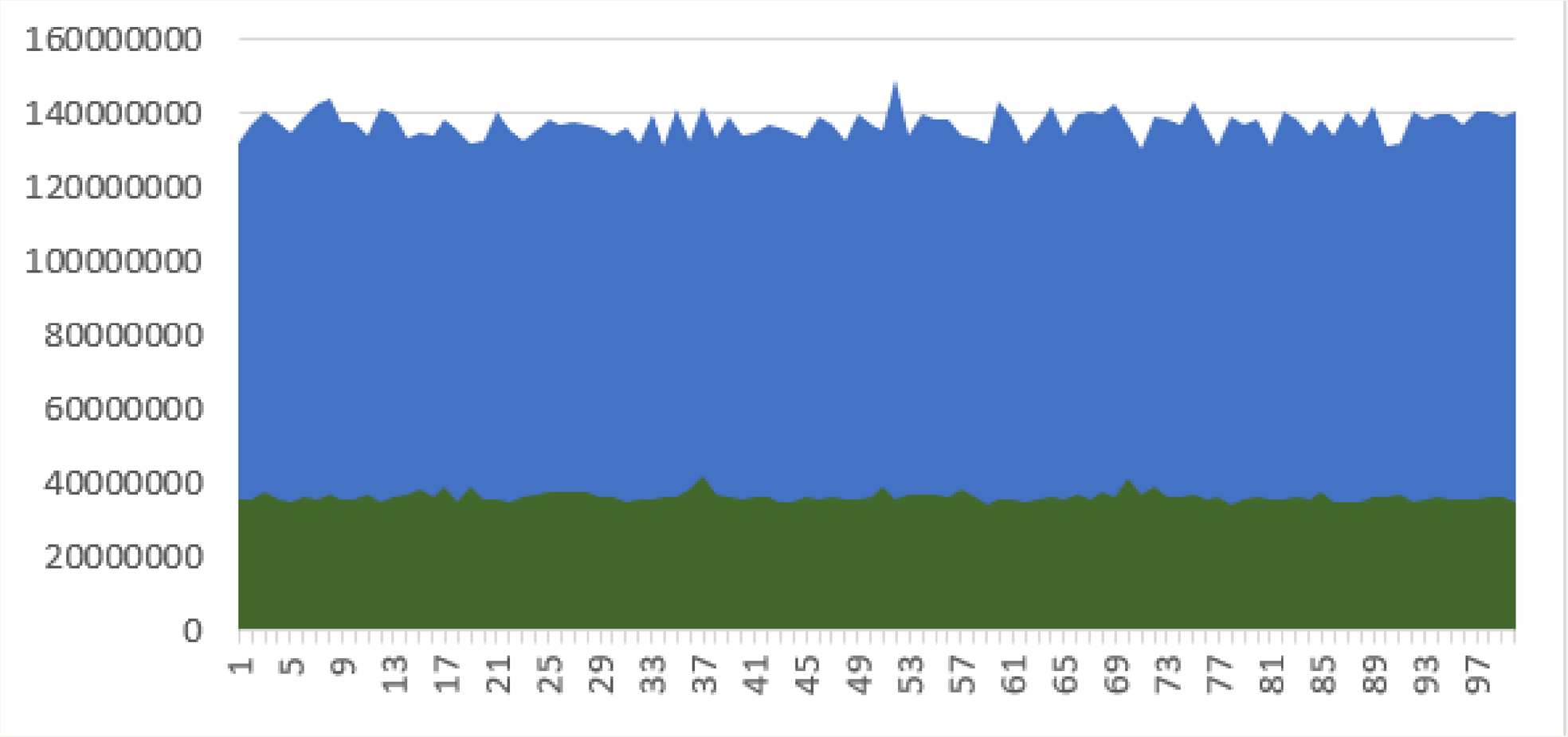
RADIX SORT

RADIX SORT


350000000
300000000
250000000
200000000
150000000
100000000
50000000
0
1 4 7 10 13 16 19 22 25 28 31 34 37 40 43 46 49 52 55 58 61 64 67 70 73 76 79 82 85 88 91 94 97 100
Teste n = 10^6, Max = 10^3
Teste n = 10^6, Max = 10^6
Teste n = 10^6, Max = 10^9

Radix sort

n=10^6 max=10^3

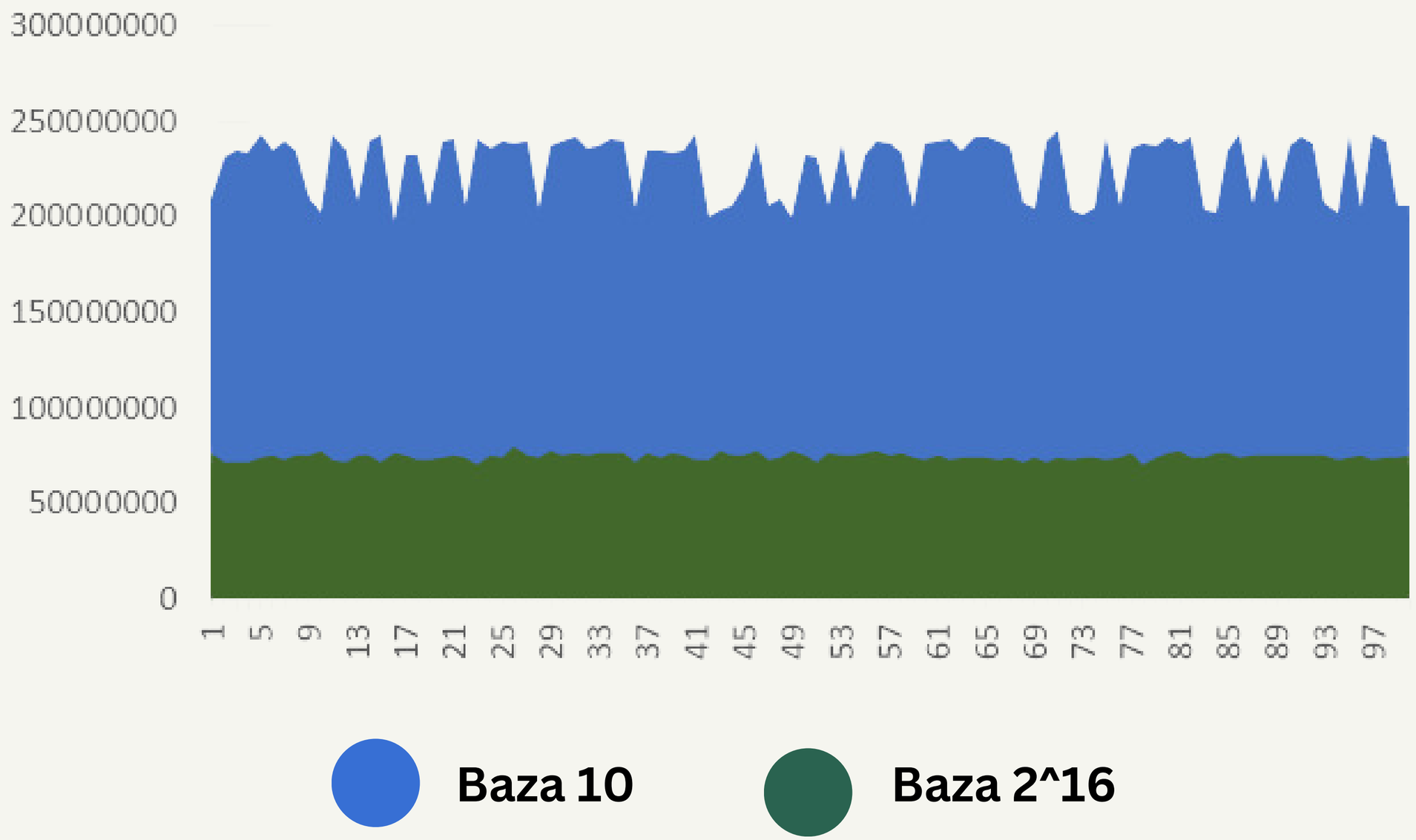


Baza 10

Baza 2^16

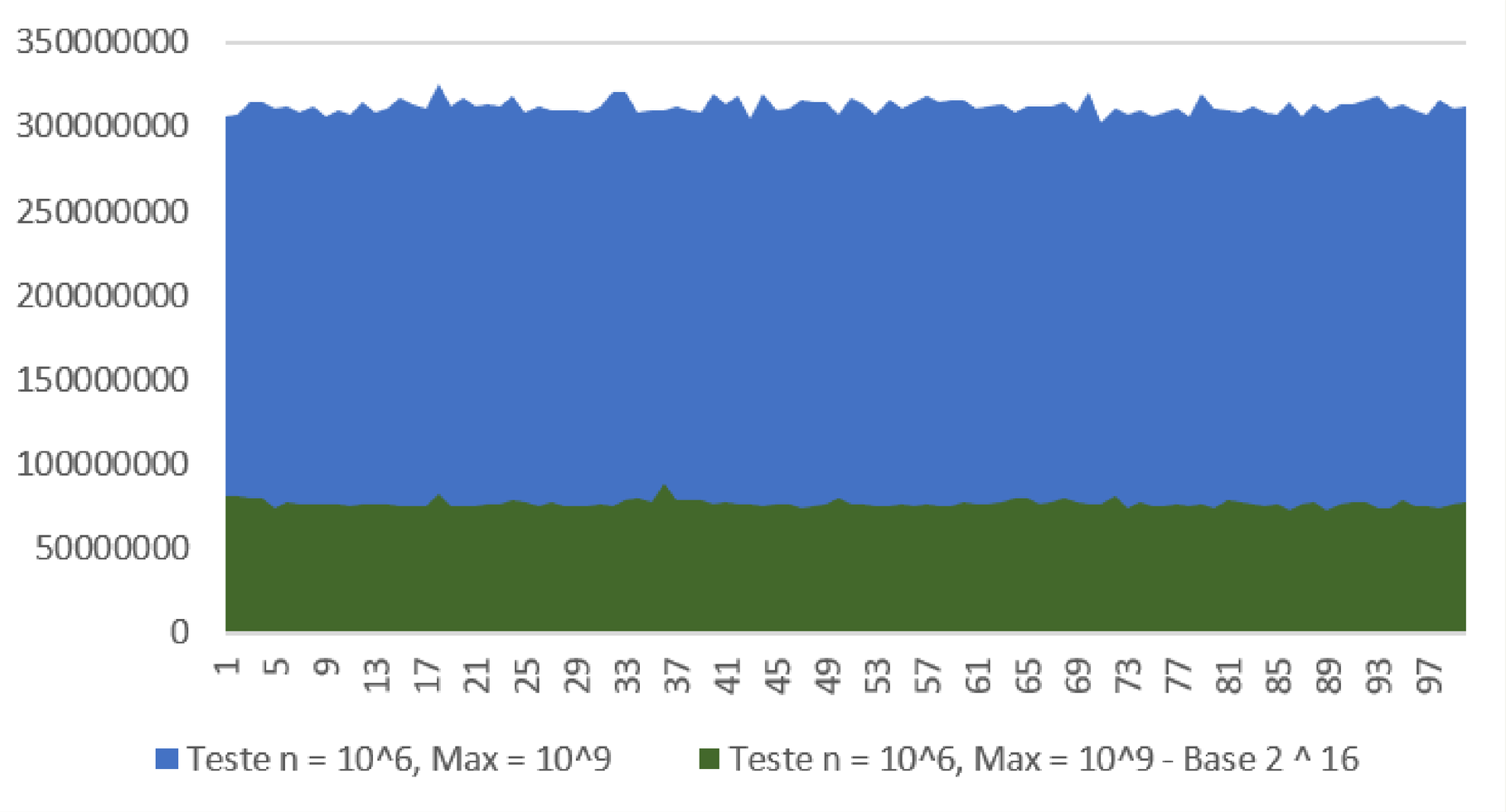
Radix sort

n=10^6 max=10^6



Radix sort

n=10^6 max=10^9



Baza 10

Baza 2^16