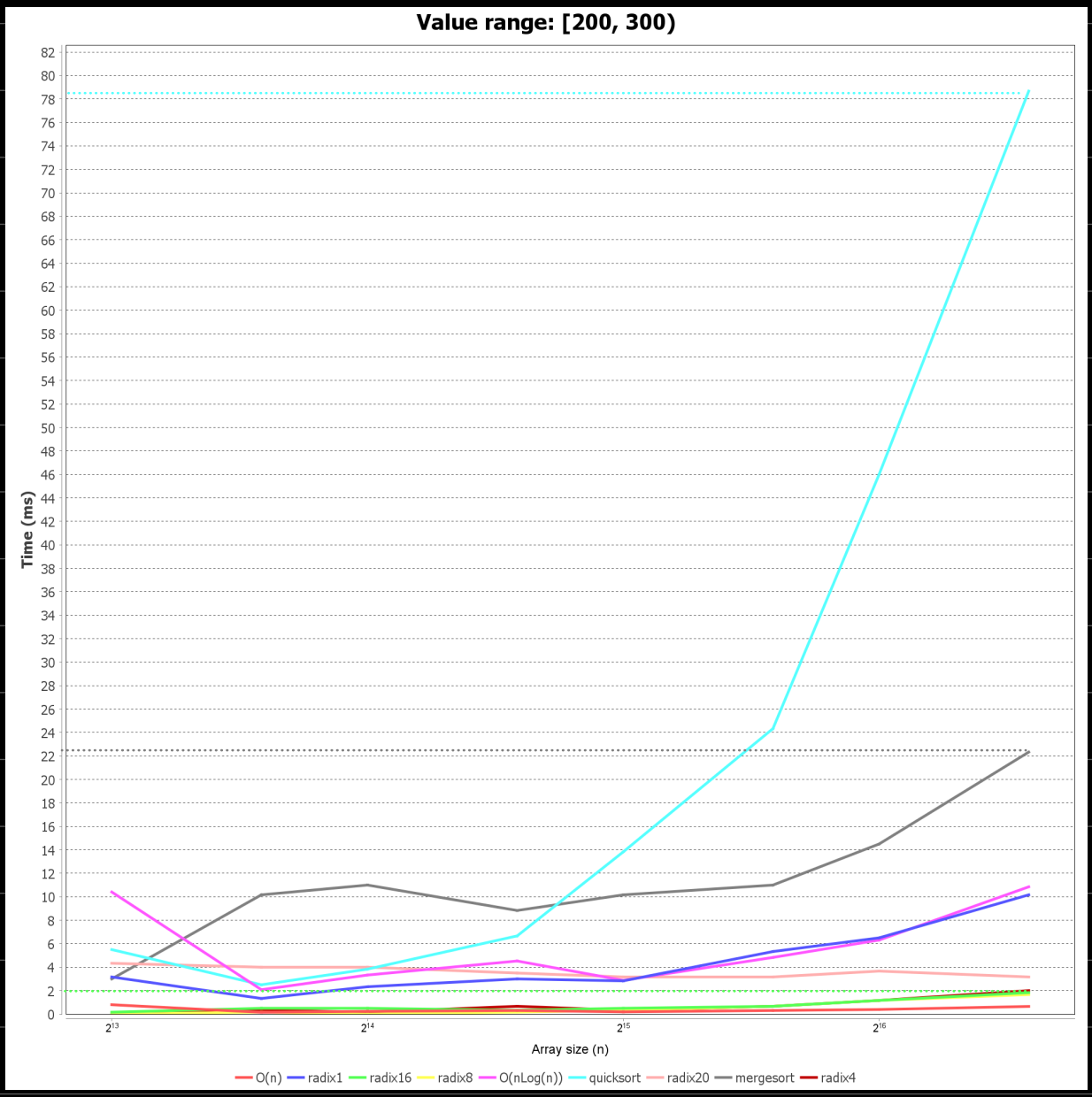
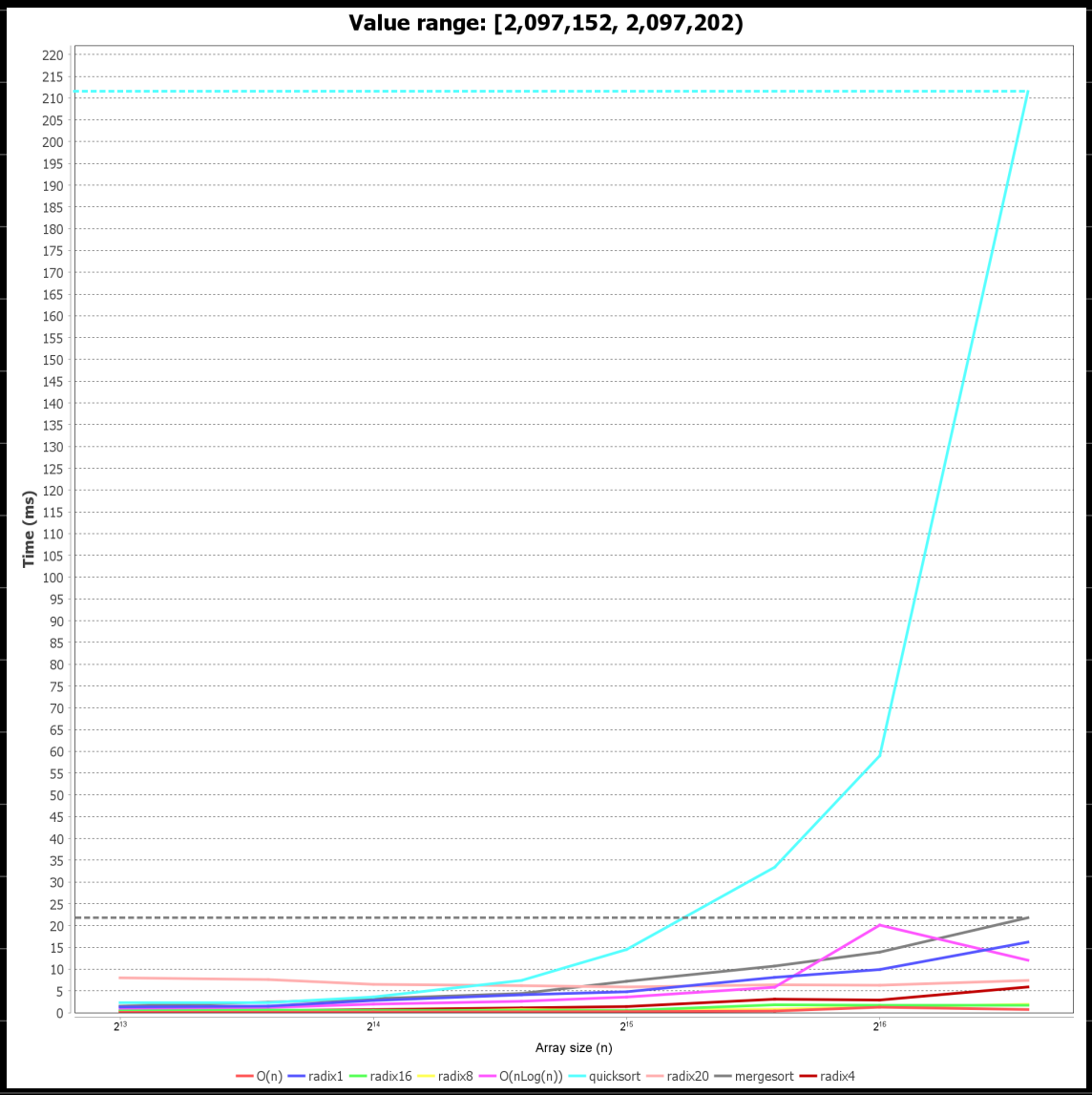
g1.



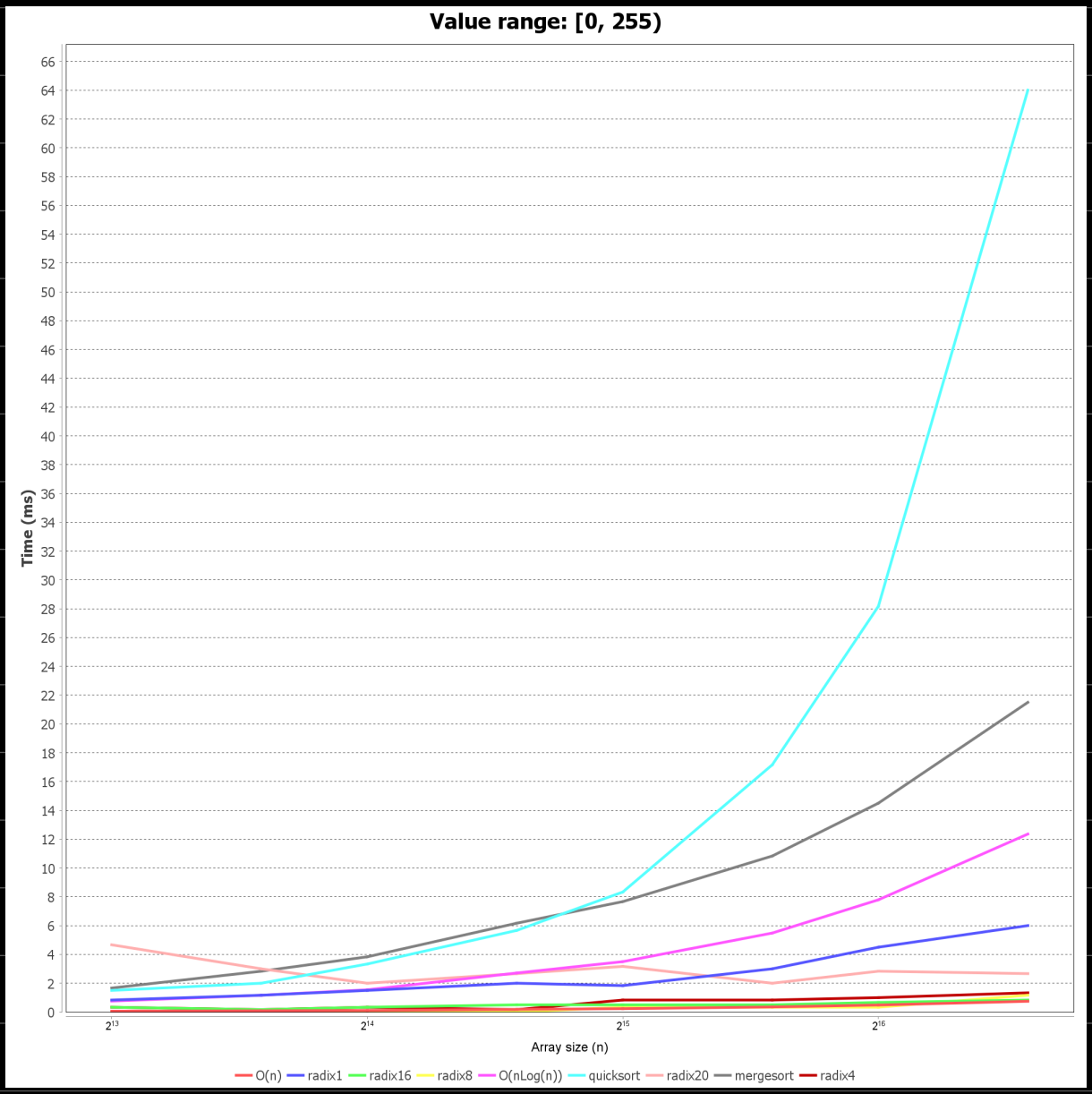
First experement was with narrow range small numbers. We got random array in range [200,300) and as we can see on graph g1. fastest sort to work with is Radix Sort with digit width 16.
And slowest one is Quick Sort apperently this sort went in to worst case time complexity of $O(n^2)$, because Quick Sort uses simple sorts algorithms such as Insertion sort $T_w(n^2)$ to small range arrays
And Radix Sort $O(n \cdot k)$. Also we can see, that Merge Sort is $O(n \log n)$ as if its time complexity T_w, T_A, T_b are $O(n \log n)$.

g2.



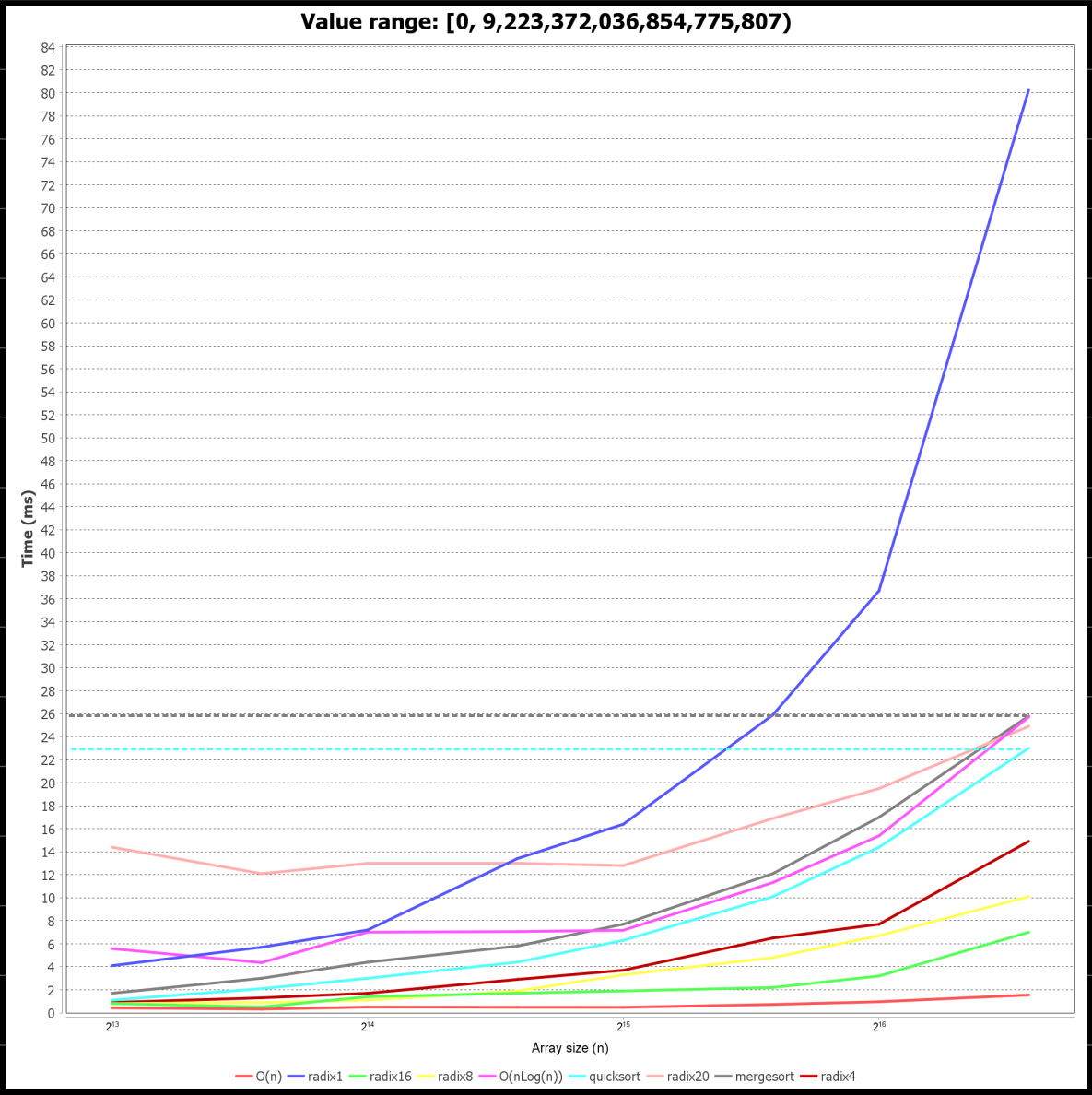
On graph g2. we can see an experement with narrow range of large numbers.
As we can see on g2. fastest sort is Radix Sort with 16 digits width and slowest as in exp.1 Quick Sort that went in to worst case time complexity $O(n^2)$, but this time it was way longer(≈210ms)than exp1 (with ≈78ms) to complete sort.
Every other sort was executing sort as expected Merge Sort with time complexity of O(nlogn) finished sort around same time as in exp 1. with ≈ 22 ms.

g3.



An exp 3 was with narrow range of small numbers. As we can see on graph g3. we have almost the same outcome as in exp 1. and exp 2. Radix Sort (16) is again sorted faster than every other sort and Quick Sort slowest an in previous experements, but made it faster in time ≈ 64ms to sort array. Merge Sort did sorting in same time as in previous experements ≈ 22 ms.

g4.



Experement 4, was with random arrays of Large numbers.

As we can see on graph g4., this time we got diffirent outcome. Radix Sort (16) as in previous experements did sort an arrays faster than every other Sort. But this time we can see that Quick Sort and Merge Sort algorithms did sort an arrays with O(nlogn) time complexity and Quick Sort did it faster as we can see on g4. Quick Sort (≈ 22 ms) Merge Sort (≈ 26 ms).

Another diffirense, that Radix Sort (4) went in to

time complexity of $O(n \log n)$ so we can say that $K = \log n$. Same with Radix Sort (20), Radix Sort (8). Exeption is Radix Sort (1), that went in to time complexity of $O(n^2)$. So we used digit width of 1, so we considered only 1 digit at a time during Sorting process.
In this case it became similar to a Counting Sort with time complexity of $O(n^2)$.

Those experements did show us that best Sorting algorithm to sort an arrray of numbers will be Radix Sort with 16 digit width. Also we saw that Quick Sort is not really useful in sort of small arrrays, and can be helpful with an arrray of large numbers.
We saw, that Merge Sort is very effective in every range of array and stages at the same time in every experement ≈ 22 ms.