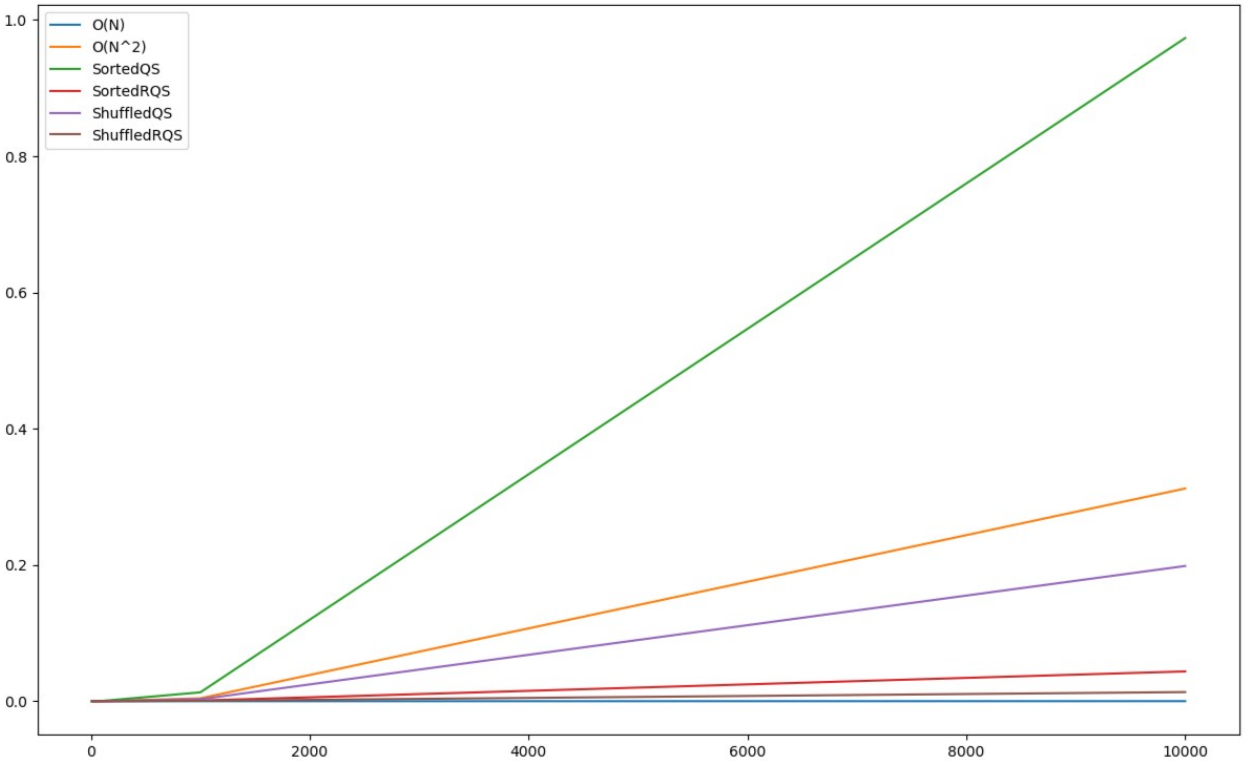
Review of QuickSort algorithm implementations and efficiency

There are a number of sorting algorithms some of them are quite slow some really fast but the most commonly used is QuickSort with running time of O(nlog(n)), where n - is size of array to be sorted. There is also the proof of that is the fastest asymptotical time we can get from sorting algorithms based on comparison. But there are some widely used implementations of quick sort - they are QuickSort - choosing pivots with some previously determined criteria, and Randomized QuickSort - choosing pivots randomly with the equal amount of probability for every element in the range.

There are cases when one beats the other with speed. Below is overviewed the cases which could be critical and key ones.



As you can see from the diagram QS's running time is very critical for the case of sorted array even quadratic sorting algorithms beat it with their speed. I assume that's the result of recursion for every call of the function memory is allocated and that takes serious time when you sort sorted array with QS because it makes N calls consecutively and iterates through whole array, whereas RQS makes less recursive calls due to its indeterminism. Indeed RQS is faster in all cases even when sequence is random I suppose the reason is same. Although there is slight difference in runtime on random sequence when the data is small.

To sum up, RQS in small cases the difference between QS and RQS is hardly noticeable, but RQS takes the lead when size of data is raising.