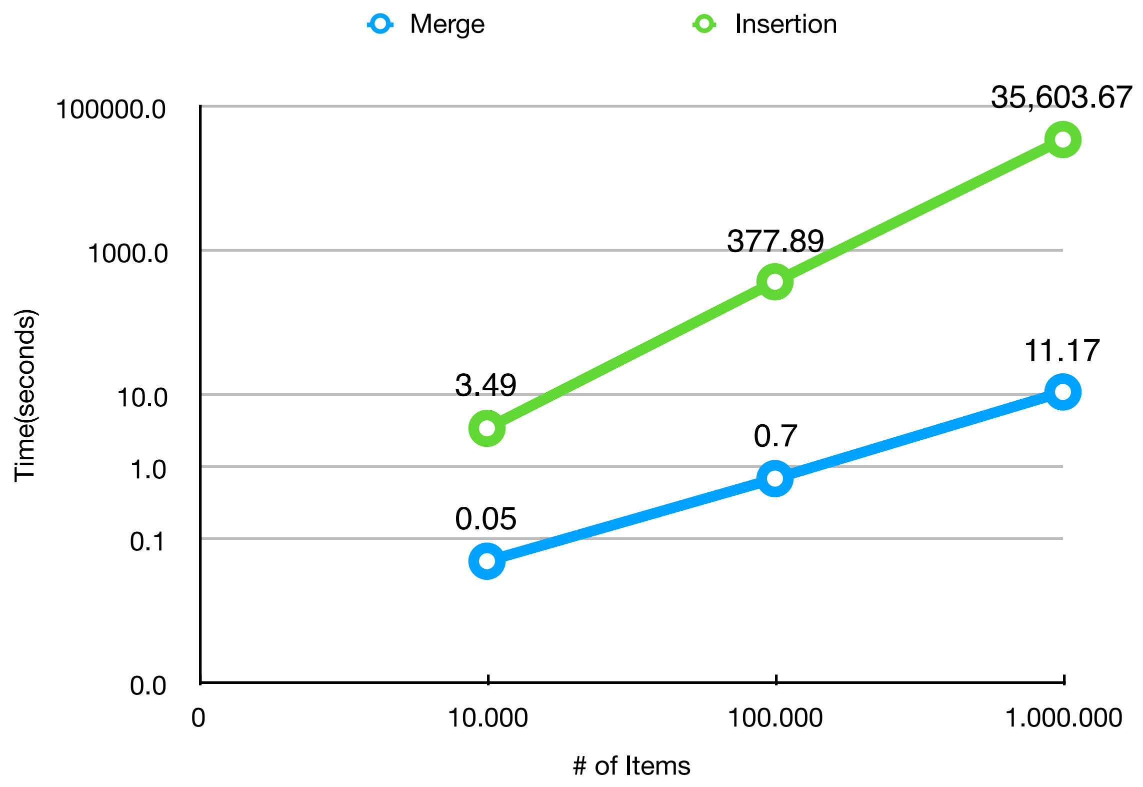
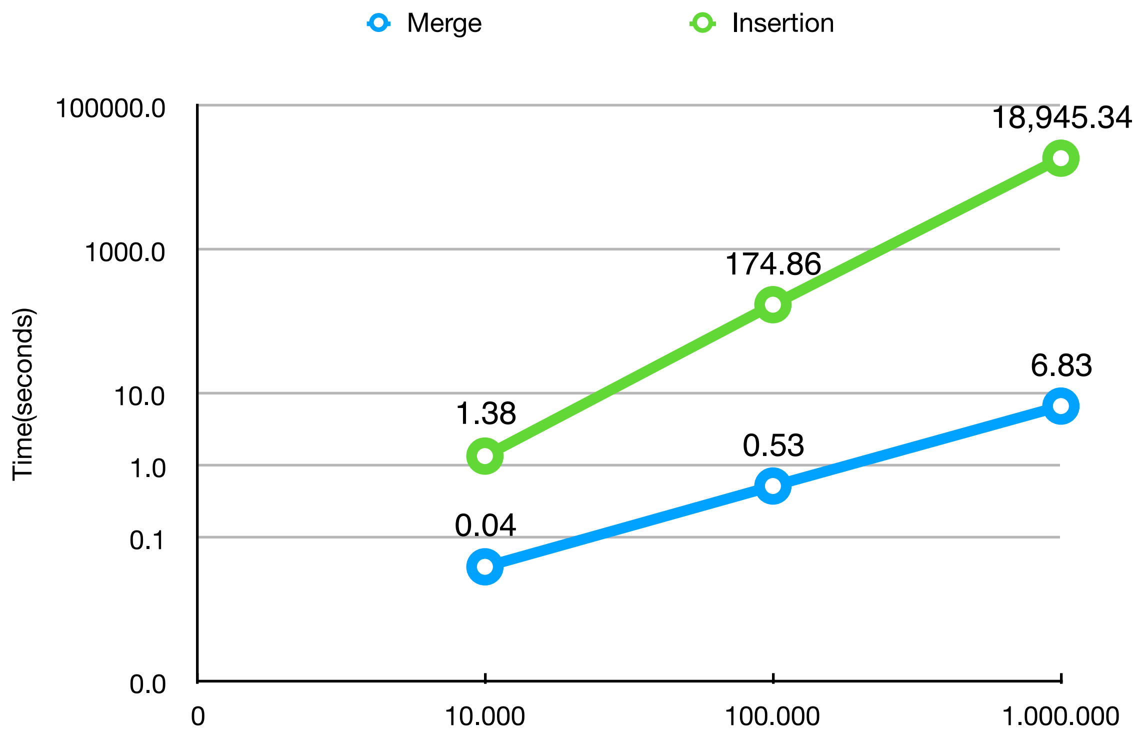
FULL SHORT



FILTER SHORT



Insertion is a progressively sorting algorithm. Its' worst case is n^2, best case is n and average case is n^2. So, complexity of insertion sort is $O(n^2)$.

Merge sort is dive and conquer type algorithm. It divides an array to binary tree and merge the pieces of tree accordingly. Its' worst case is $n * lg(n)$, best case is still $n * lg(n)$. So, average case is also $n * lg(n)$. Complexity of merge sort is $O(n * lg(n))$.

Merge sort usually works faster than insertion sort. Especially with bigger data sets. Insertion sorts complexity is linear and this makes insertion really bad algorithm for bigger data sets. For example if insertion sort sorts an array with 10 elements in 10 seconds, it will sort 100 element array in 1000 seconds. However, merge sort is not always faster than insertion sort. Since, best running time of insertion sort is n. If array is already sorted, insertion sort works faster than merge sort.

Name variety is larger than rarity or set an type. Is much harder to have equal value at name. If we have more equal values on a data set. Insertion sort will work slightly faster. Merge sort does not effect from having more equal data set. Merge sort is divide and conquer type of algorithm. So, it will do its job regardless from values at array and values order at array.

In stable algorithm order of equal values stay same. For example, in stable algorithm (1,3(1), 3(2),2) this dataset always sort like this (1,2,3(1),3(2)).However in unstable algorithm, same dataset might be sorted like this (1,2,3(2),3(1)).Merge sort and insertion sort are stable sorting algorithms. They never change order of same values.