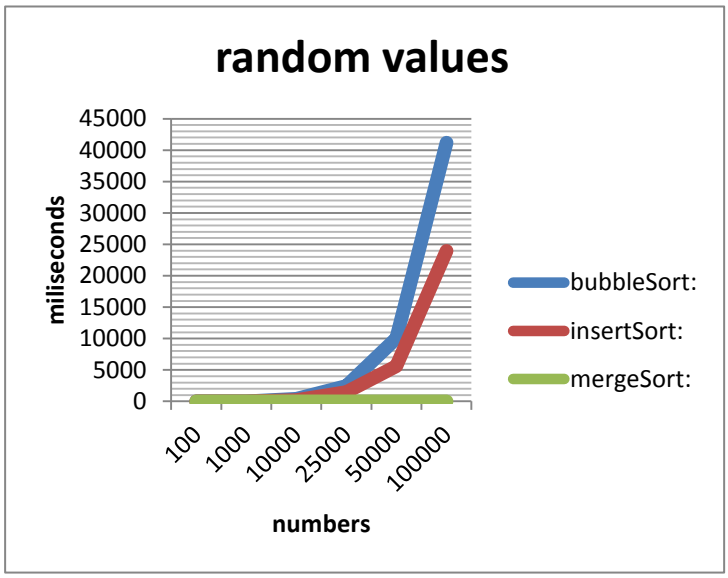
Author:
Michał Foryś, 203449

Complexity of sorting algorithms

In my report I'm going to compare complexity of three sorting algorithms: bubble sort, insert sort and merge sort. Each one has a different time of execution and I've run the on different data sets: sorted, random, reverse sorted, equal values. The row at the top presents **how many** numbers were in the array and the execution times are in **milliseconds**.

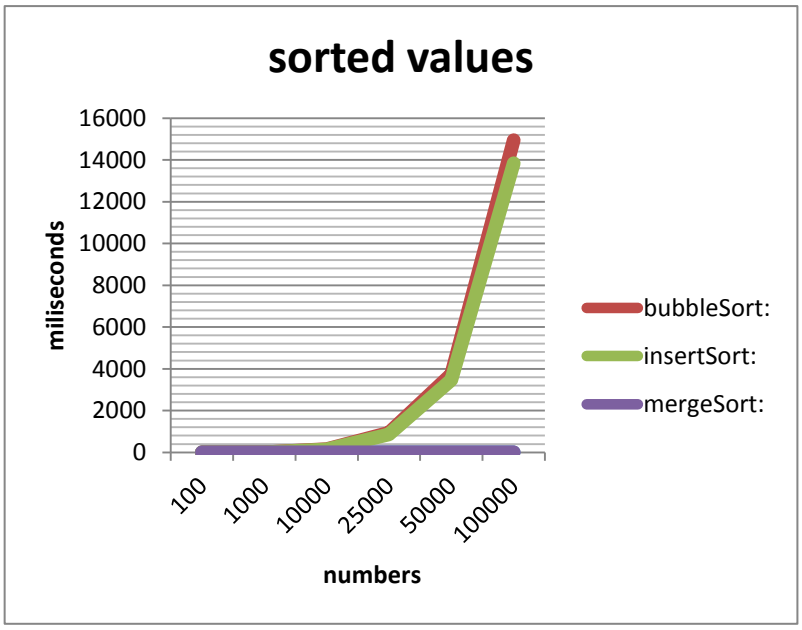
1. Random values:

	100	1000	10000	25000	50000	100000
bubbleSort:	0	3	378	2437	10031	41164
insertSort:	0	2	215	1451	5622	23959
mergeSort:	0	0	1	4	10	18



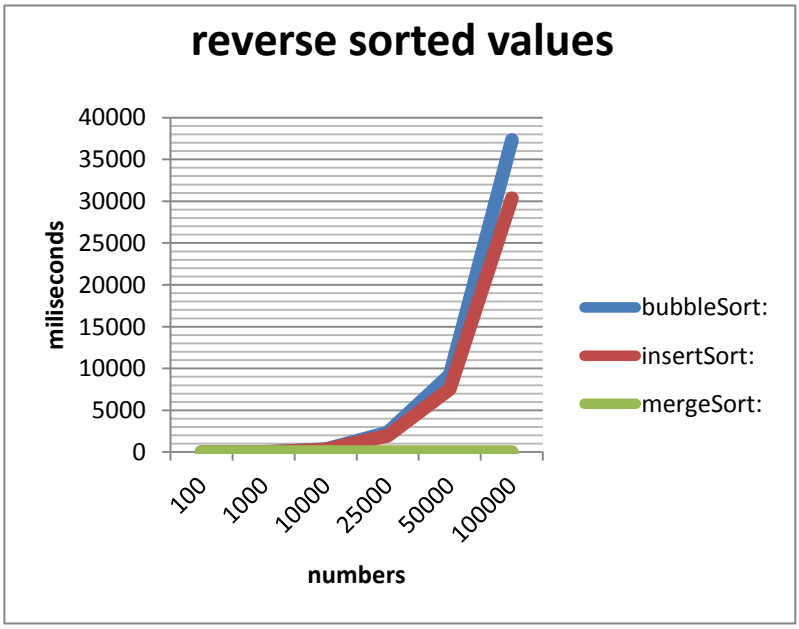
2. Sorted values

	100	1000	10000	25000	50000	100000
bubbleSort:	0	4	154	935	3740	14952
insertSort:	0	1	137	857	3461	13848
mergeSort:	0	0	1	2	5	8



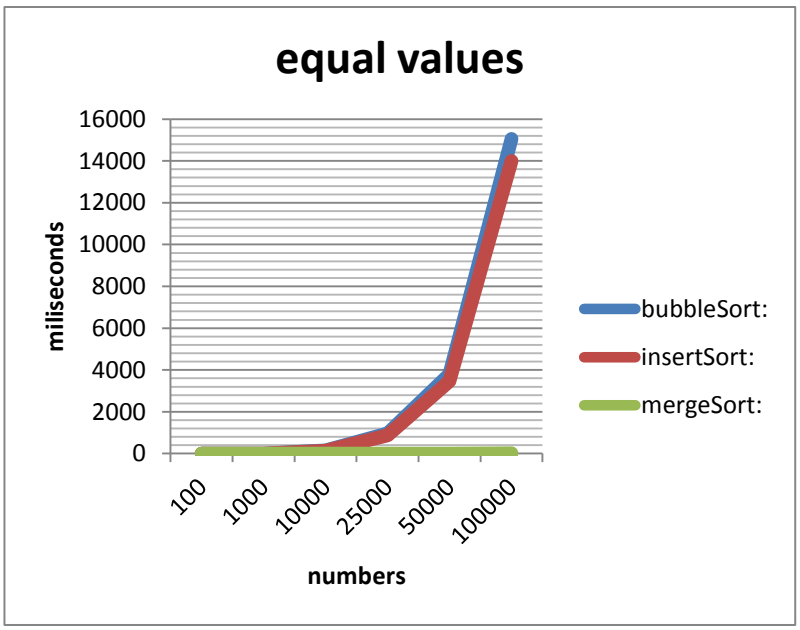
3. Reverse sorted values

	100	1000	10000	25000	50000	100000
bubbleSort:	0	4	363	2350	9235	37324
insertSort:	0	4	318	1959	7547	30353
mergeSort:	0	0	1	3	5	12



4. Equal values

	100	1000	10000	25000	50000	100000
bubbleSort:	0	2	149	957	3792	15047
insertSort:	0	2	138	868	3468	13995
mergeSort:	0	0	1	3	5	11



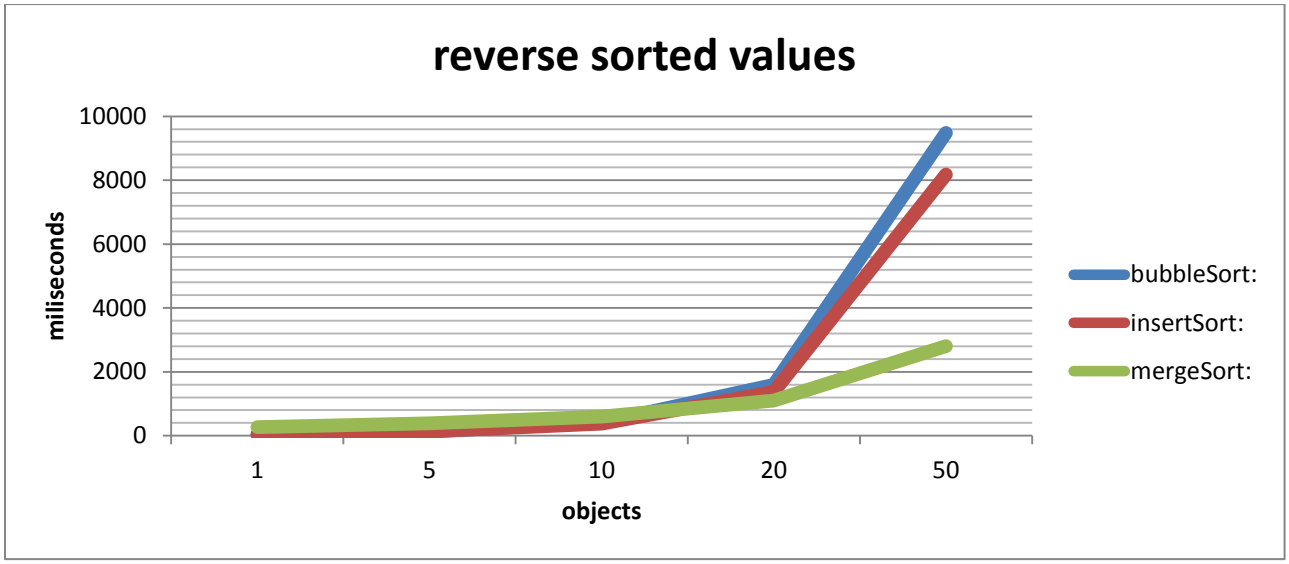
Unfortunetly to notice difference for small numbers we have to execute sorting many more times and eventually divide it by numbers of executions and total time. I'll skip dividing it by numbers of executions because the numbers might me smaller than 1 ms. In sorting times presented below filling array will be also included but it's a constant time so does not have impact on the measurements. The sorting was executed 1 000 000 times for each number of numbers.

Random values:

	1	5	10	20	50
bubbleSort:	51	171	427	1457	7514
insertSort:	54	183	494	1358	7962
mergeSort:	281	483	777	1616	3704

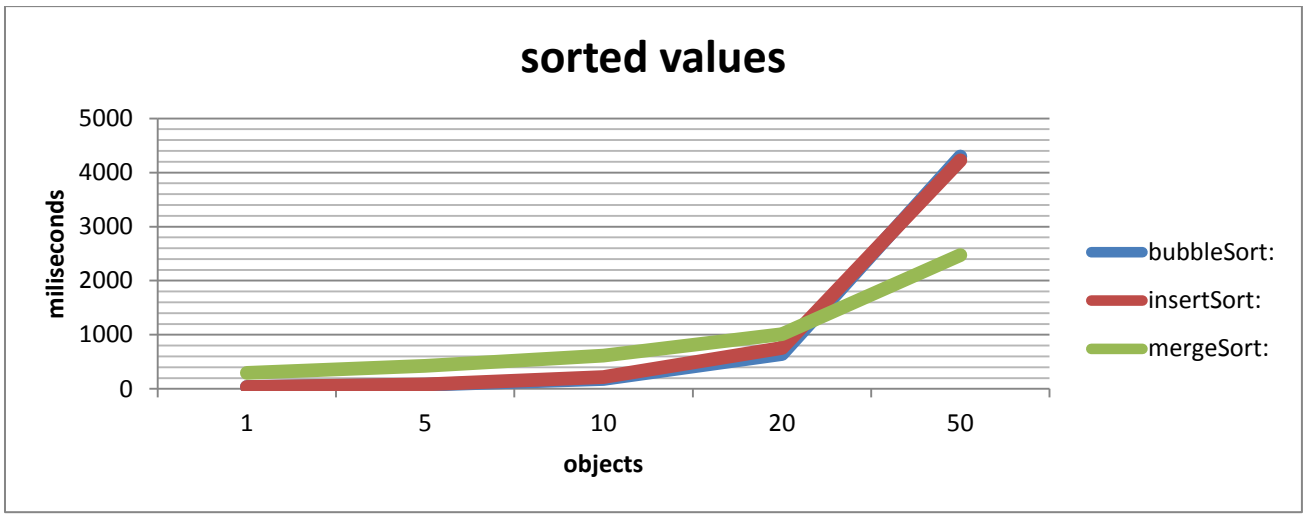
Reverse sorted values

	1	5	10	20	50
bubbleSort:	35	132	422	1589	9477
insertSort:	37	135	376	1376	8173
mergeSort:	265	387	603	1105	2801



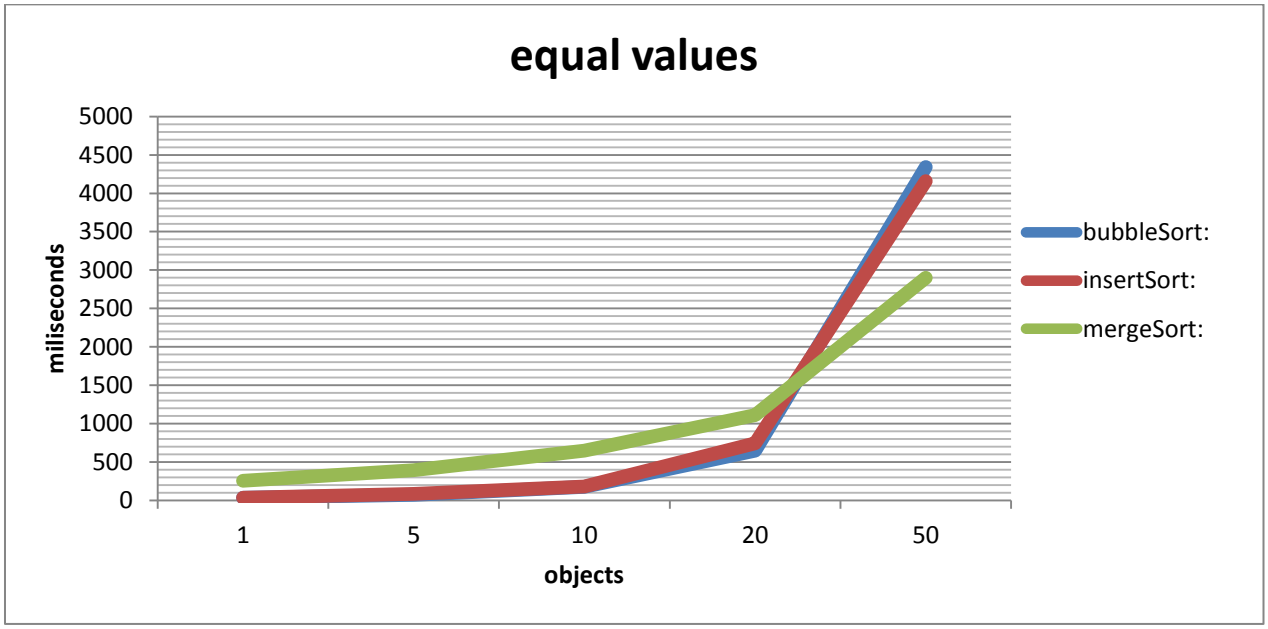
Sorted values

	1	5	10	20	50
bubbleSort:	34	74	182	644	4294
insertSort:	35	83	218	754	4227
mergeSort:	295	427	612	1011	2471



Equal values

	1	5	10	20	50
bubbleSort:	37	72	175	646	4340
insertSort:	36	86	183	746	4160
mergeSort:	256	392	650	1110	2898



Conclusion

The graphs show us clearly that bubble sort and insert sort can't even compare to merge sort. While insert sort complexity in *O* notation ranges between $O(n^2)$ for worst case and O(n) case the merge sort complexity is almost constant and is equal O(nlogn). However, merge sort has few disadvantages, the most crucial one is that it needs extra memory for coping tables. The graphs also show as if we wanted to sort lots of times small number of objects we should use bubble sort or insert sort not merge sort. Their complexity is suitable for such case. We can observe that to around 20 objects merge sort loses in execution time.