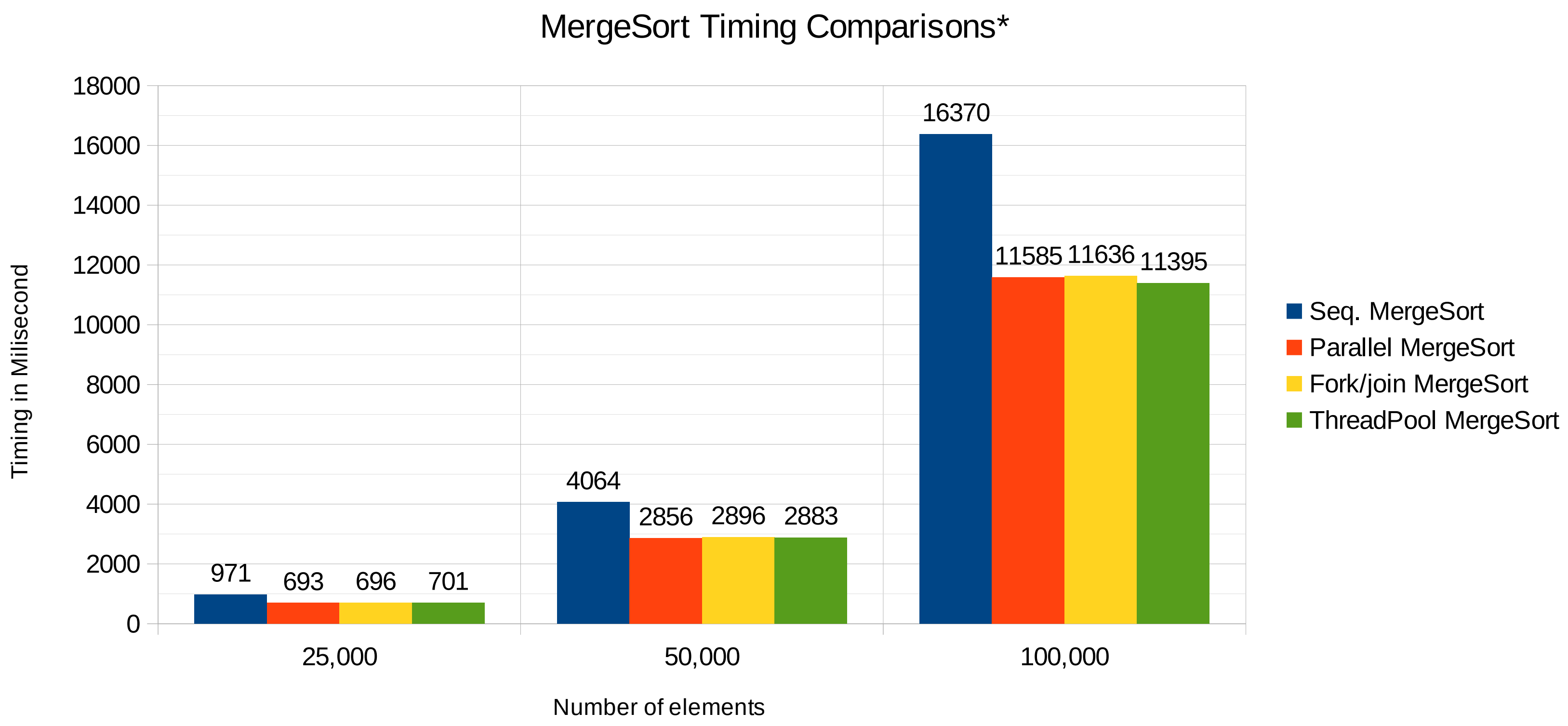
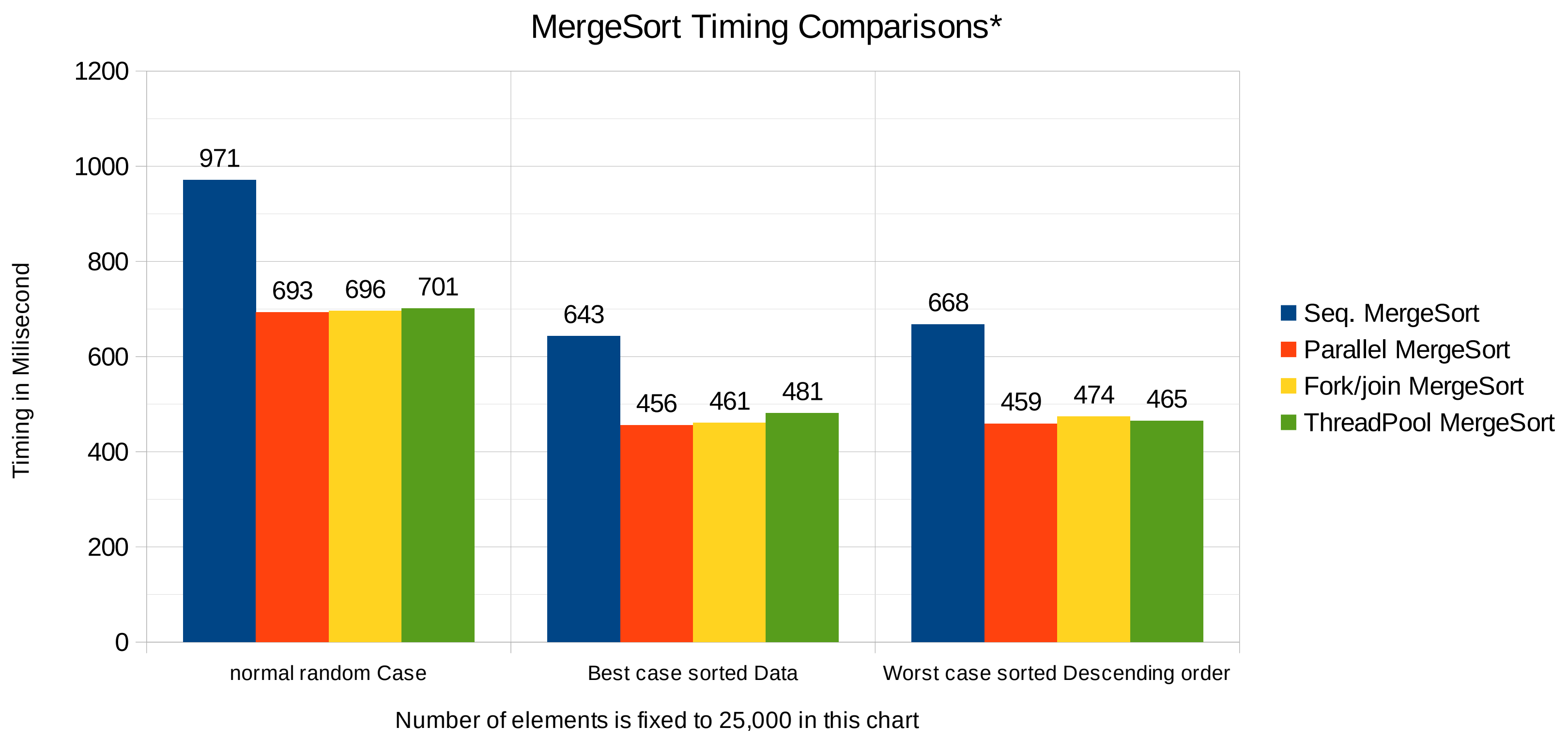


*to increase precision the timing is calculated as follows: (i^1+ i^2+...+i^n)/10. Where i^n is the timing at n. n= {0,…., iterations-1}



*to increase precision the timing is calculated as follows: (i^1+ i^2+...+i^n)/10. Where i^n is the timing at n. n= {0,….,
iterations-1}

Notes About the MergeSort classes

- Please note that due to the fact that im working on a Raspberry pi 4 with very limited Ram (4Gb). I preferred to implement it in place without preserving the Original List/ data.
- An Optimization could be done to make the merge sort faster is using “threshold” after which Insertion sort would be called because as we already learned in the Pr2 that insertion sort is faster for the pre-sorted elements.
- As well I could’ve preserved the original list and created an “unsorted list” to save the sorted elements in it and with the aid of a non-Void mergeSort methods I would be able to improve the sorting speed.
  knowing the fact that you didn’t specified any approach in your Assignment I assumed that you needed a working solution as long as its valid mergeSort approach :) .
- I tried to optimize stuff as much as I could further info could be seen at my Github repo.
- So, Please kindly accept my project and I hope you like it.