Foi escolhido implementação de 4 algoritmos de ordenação sendo eles:

quicksort

mergesort

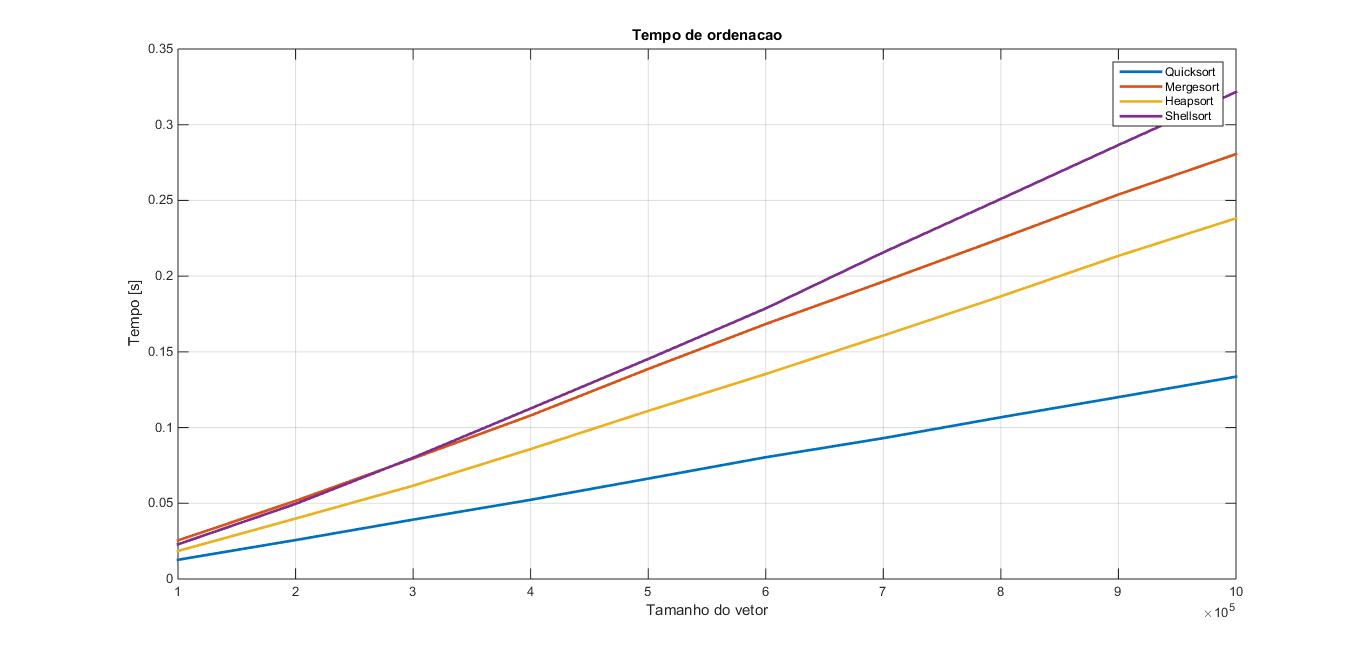
shellsort

heapsort

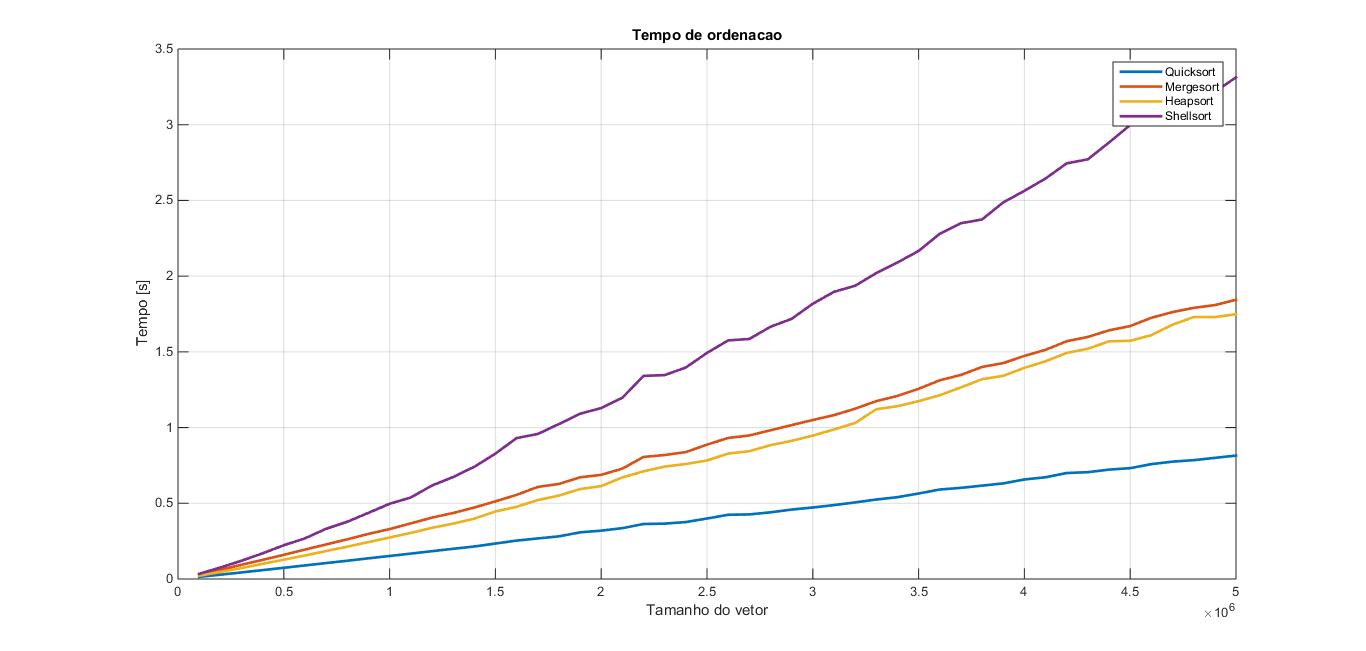
Foi executado estes 4 algoritmos 10x para cada tamanho especifico de vetor e aumentando de 100k em 100k até 1milhao e 5 milhões.

Na primeira parte... foi testado em 4 processadores diferentes a ordenação de cada algoritmo.

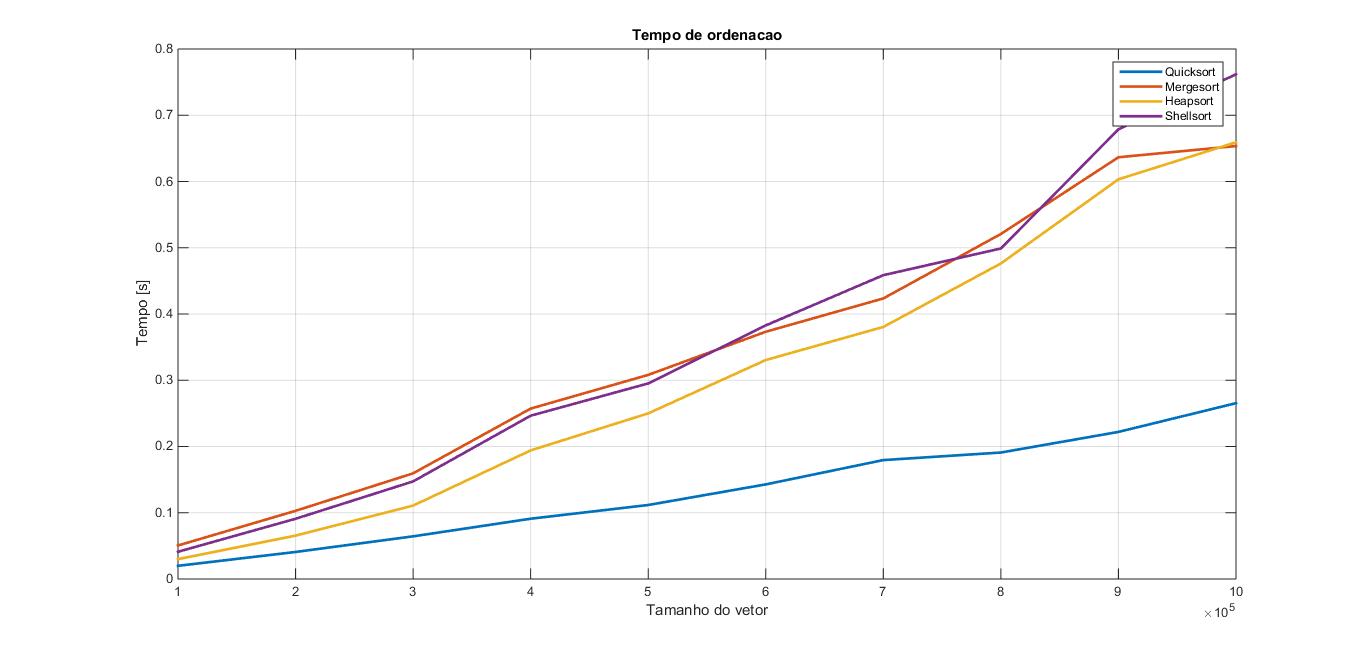
Primeiro gráfico, foi utilizado um i5\_4670\_3.4ghz



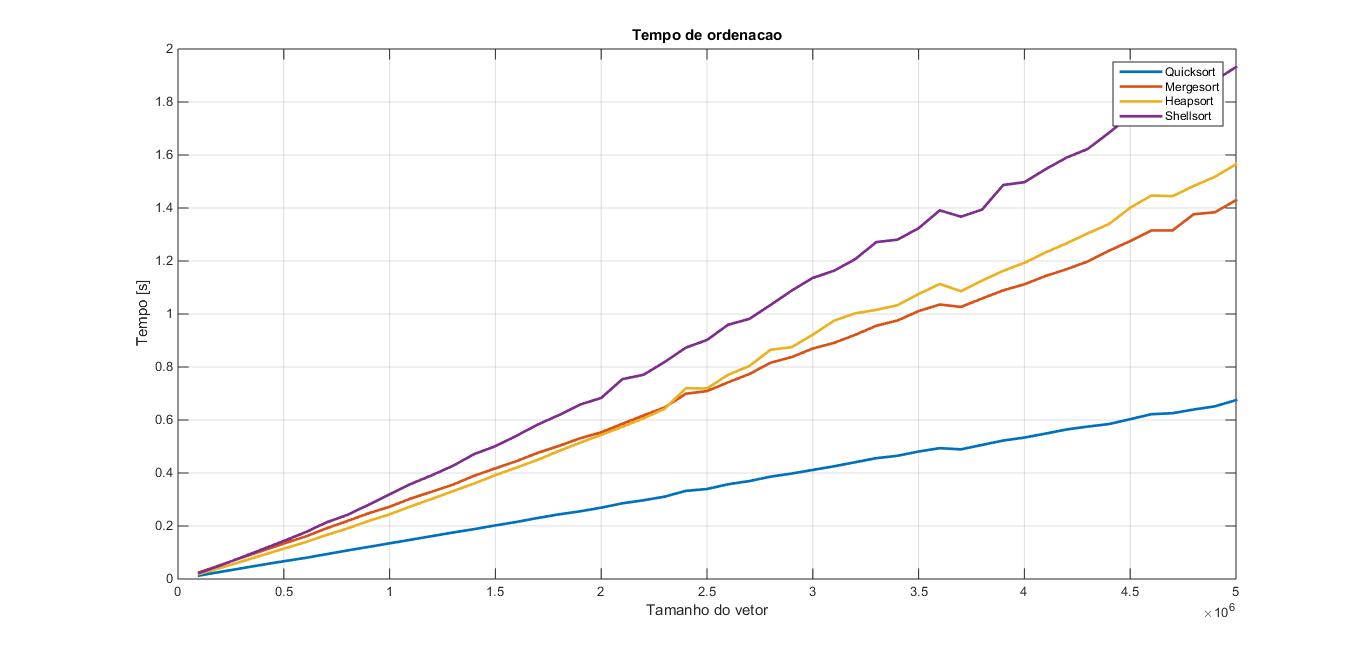
O segundo gráfico foi um i7\_860, com overclock de 3.8ghz



O terceiro gráfico, um intel\_dualcore\_T4-500 2.3ghz



E o último gráfico sobre ordenação de vetores aleatórios, foi utilizado um i7\_3610qm\_2.3ghz



Em todos eles quicksort foi o mais eficiente, e muito mais eficiente, por ir dividindo o vetor recursivamente pela metade, sendo menor o tempo necessário para percorrer todo o vetor,

Mergesort funciona criando uma arvore estrutural com os vetores. Ordenando esta arvore, eficiente mas não tanto.

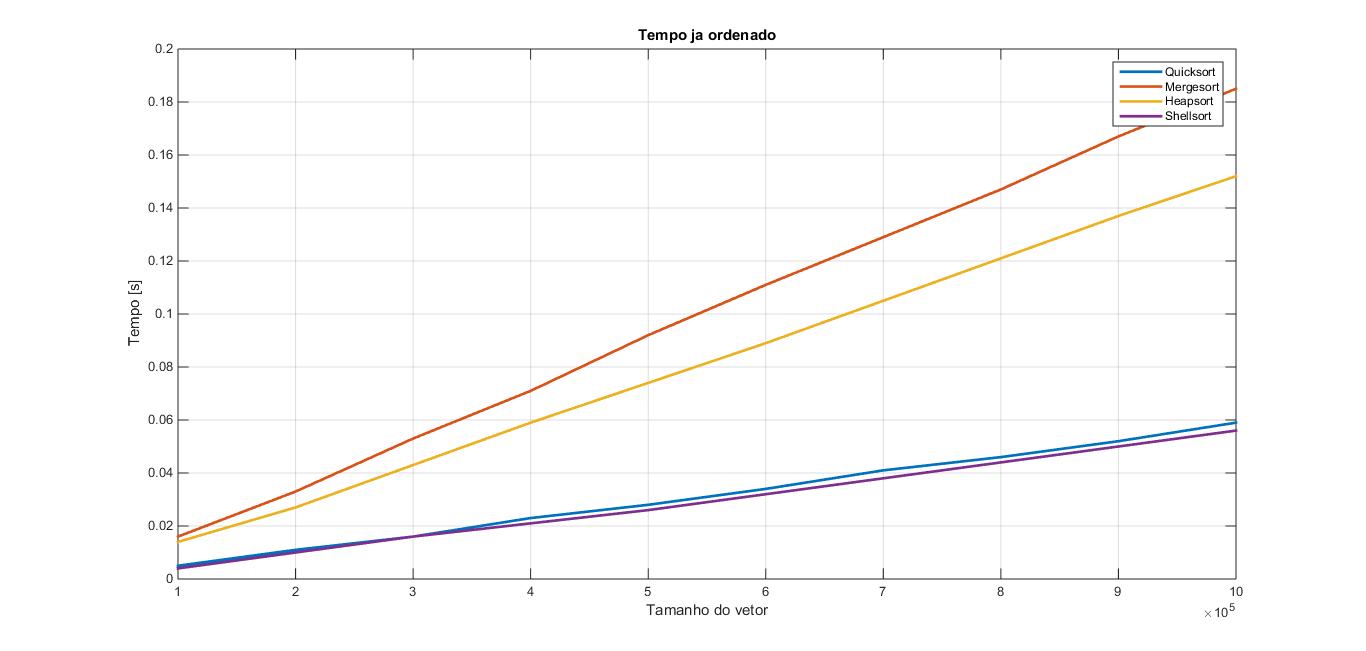
Heapsort trabalha dividindo vetor em 2 partes chamadas peneira, ordenando estas duas separadamente e devolvendo-as ordenadas.

Shellsort percorre o vetor em pulos definidos pelo programador, e com isso percorre o vetor inteiro ordenando.

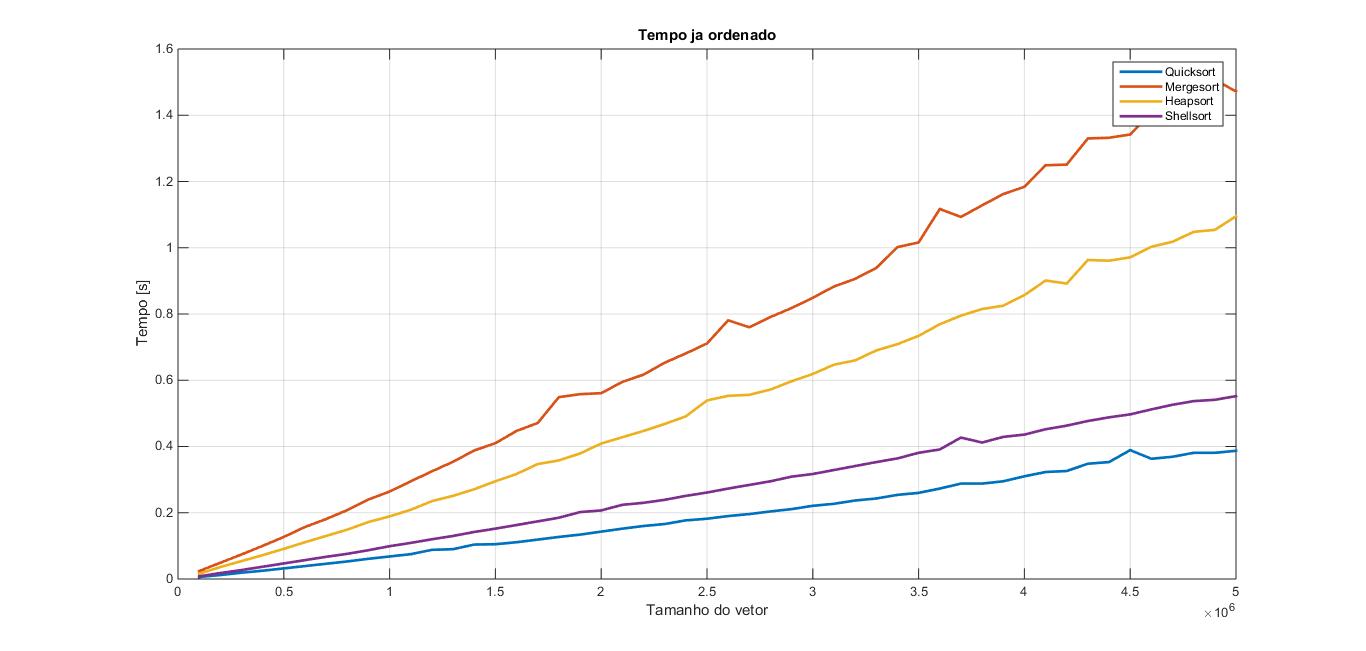
Vetor já ordenado.

Também foi medido tempo médio de ordenação de um vetor já ordenado- ou basicamente o tempo que demora pro algoritmo perceber que este vetor já está em ordem...

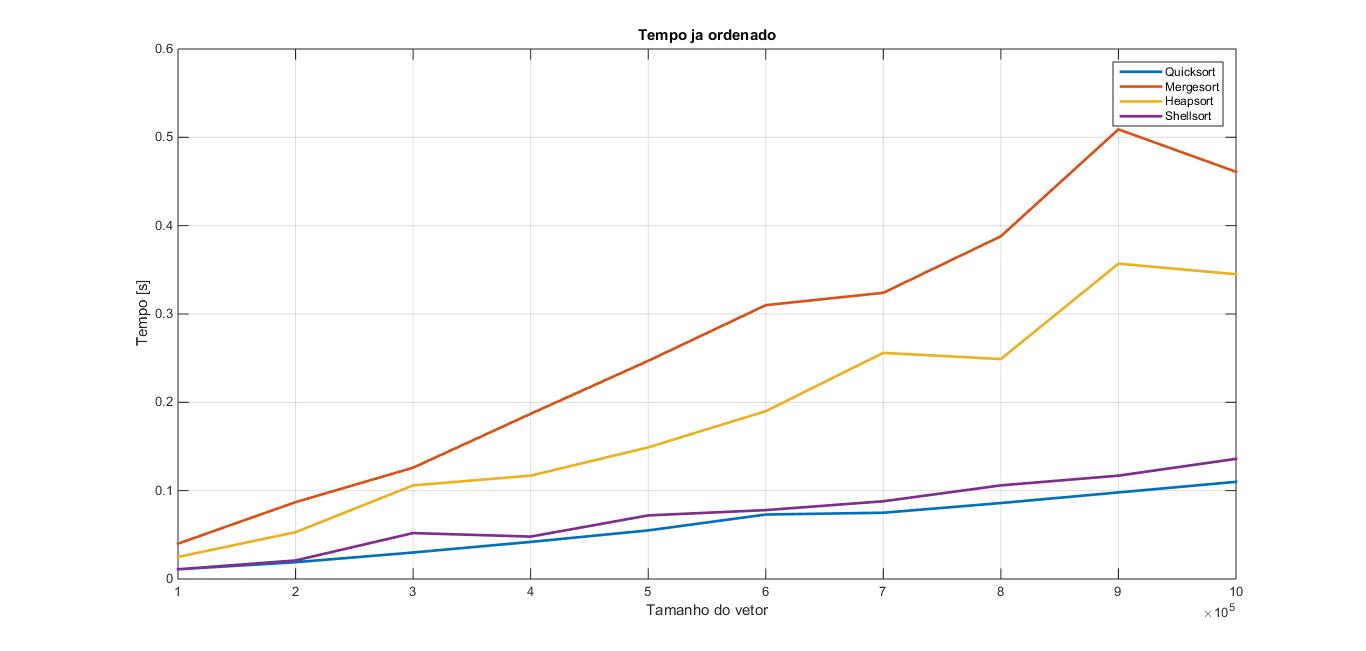
Primeiro gráfico i5\_4670 3.4ghz



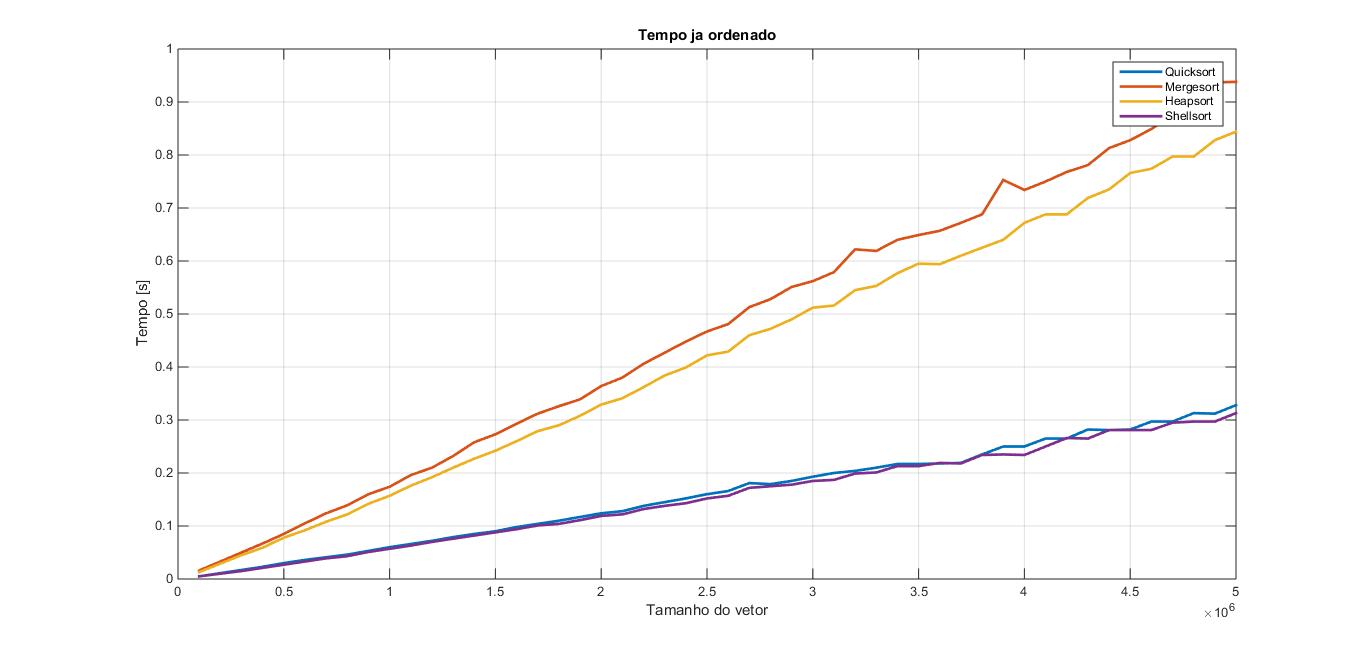
I7\_860 overclock 3.8ghz



Intel dual core T4\_500 2.3ghz



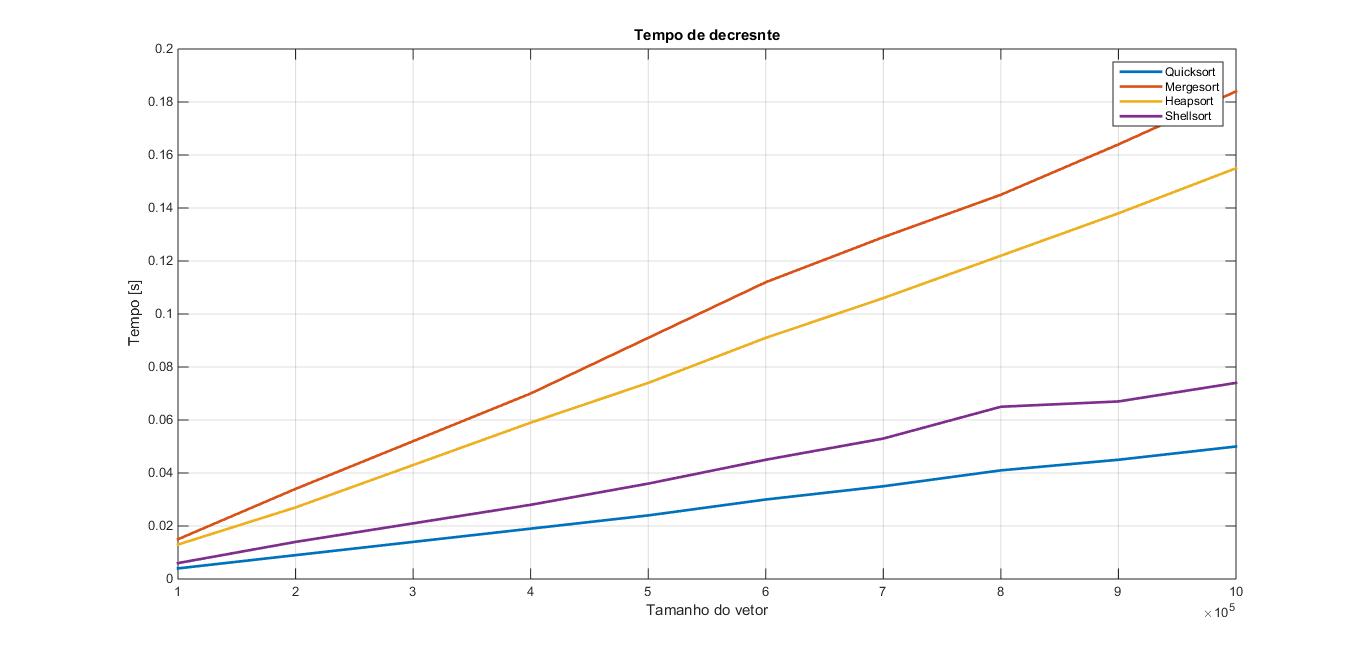
I7\_3610qm 2.3ghz



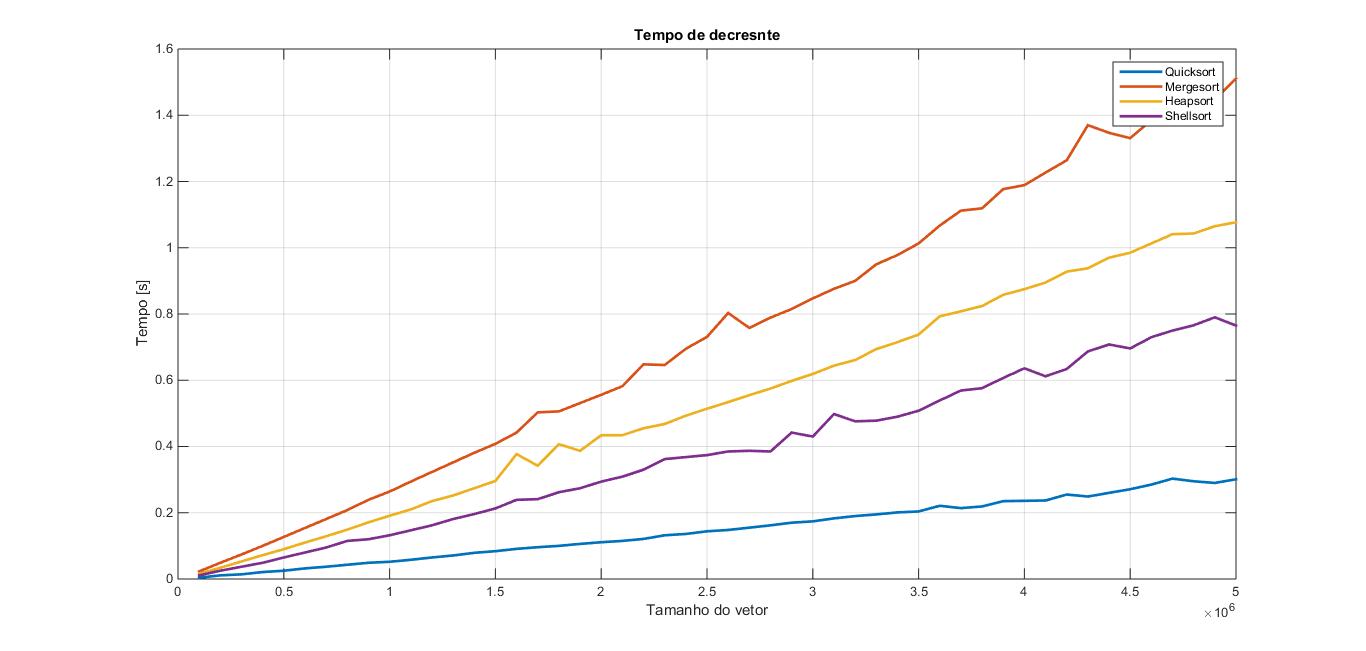
Em todos os gráficos, quicksort e shellsort partilharam quase o mesmo tempo, devido a forma do programa ser de “saltos” quicksort indo de metade em metade, e shell a definido pelo usuário. Eles são mais rápidos nesse processo!

Também foram feitos testes com o vetor ordenado em ordem decrescente ou, todos os valores necessitando serem ordenados.

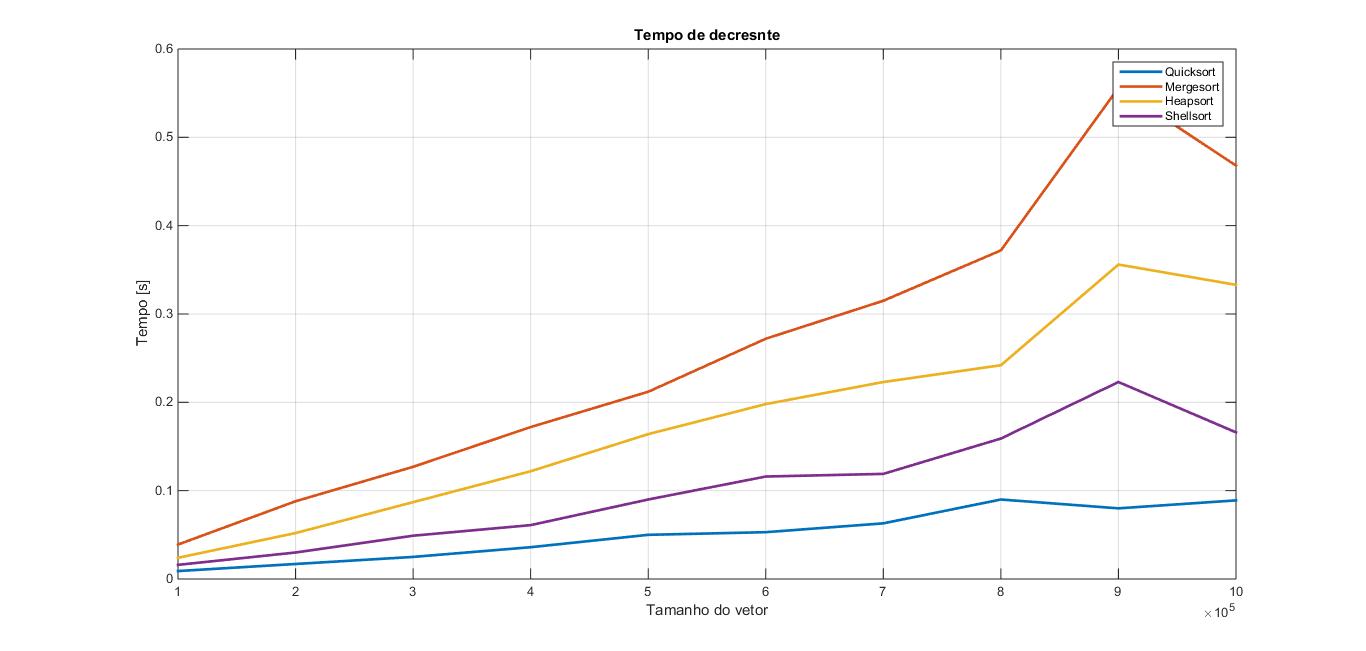
I5\_4670\_3.4ghz



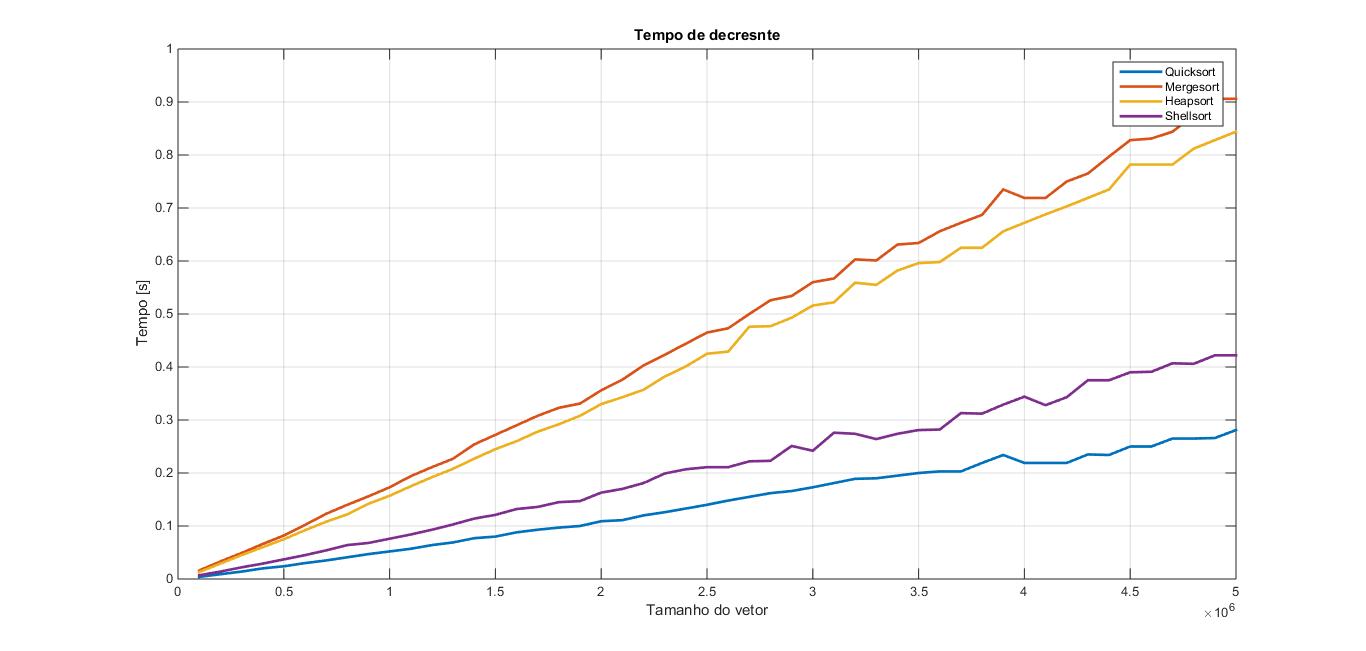
I7\_860 overclock 3.8ghz



Intel dual core T4-500 2.3ghz



Intel i7-3670qm 2.3ghz



Quicksort e shellsort foram os melhores, o quicksort por ele utilizar apenas 1 das chamadas recursivas. O caso de j > esquerda, e o shellsort por causa da temática de ordenar o vetor em saltos.

O quicksort foi. Dos vetores testados o melhor, em todos os casos testados, extremamente mais otimizado, em um sentido computacional, o algoritmo entende melhor como o processador trabalha e soube utilizar isso para ordenar mais rapidamente um vetor, pagando o preço de não ser estável.