foi escolhido implementacao de 4 algoritimos de ordenacao sendo eles:

quicksort

mergesort

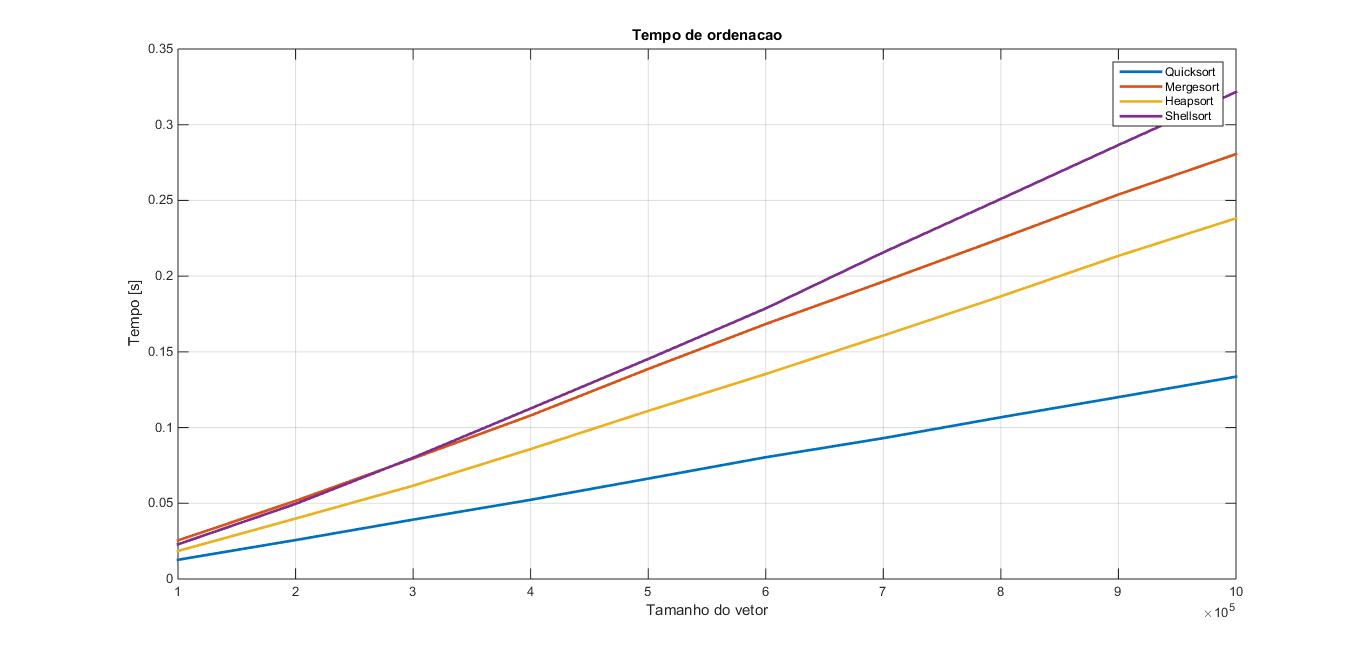
shellsort

heapsort

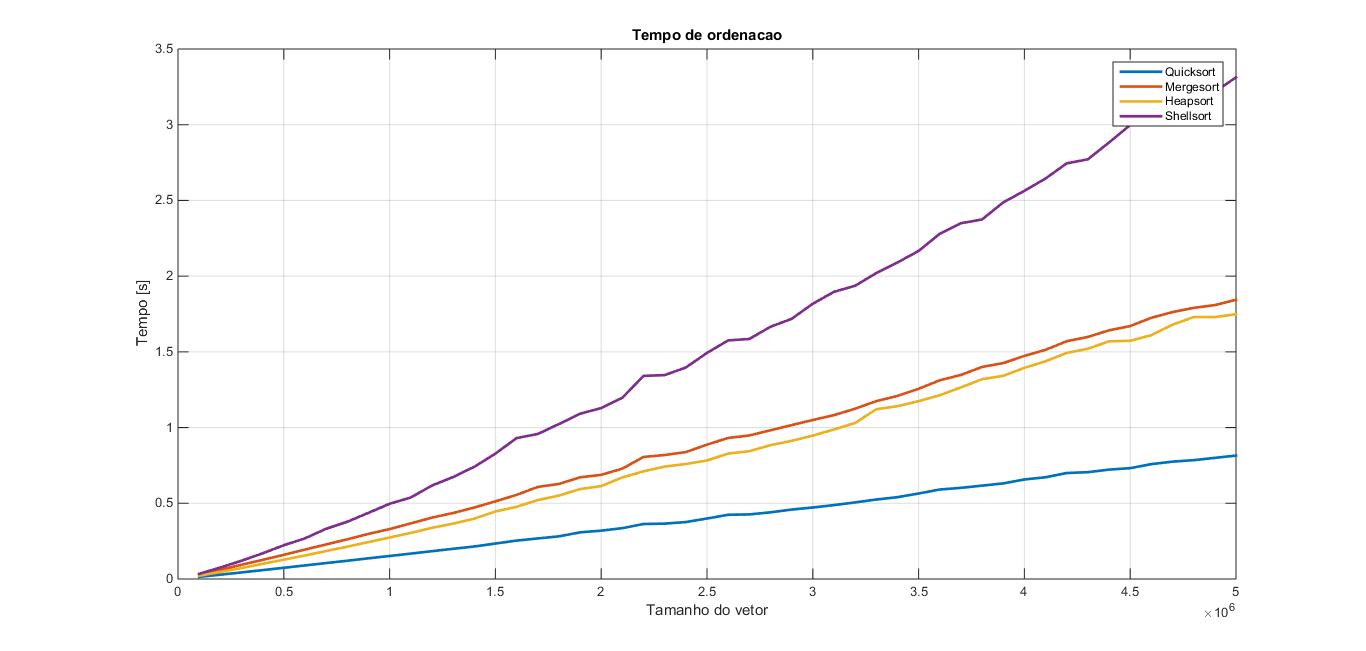
foi executado estes 4 algoritimos 10x para cada tamanho especifico de vetor e aumentando de 100k em 100k ateh 1milhao e 5 milhoes.

Na primeira parte... foi testado em 4 processadores diferentes a ordenacao de cada algoritimo.

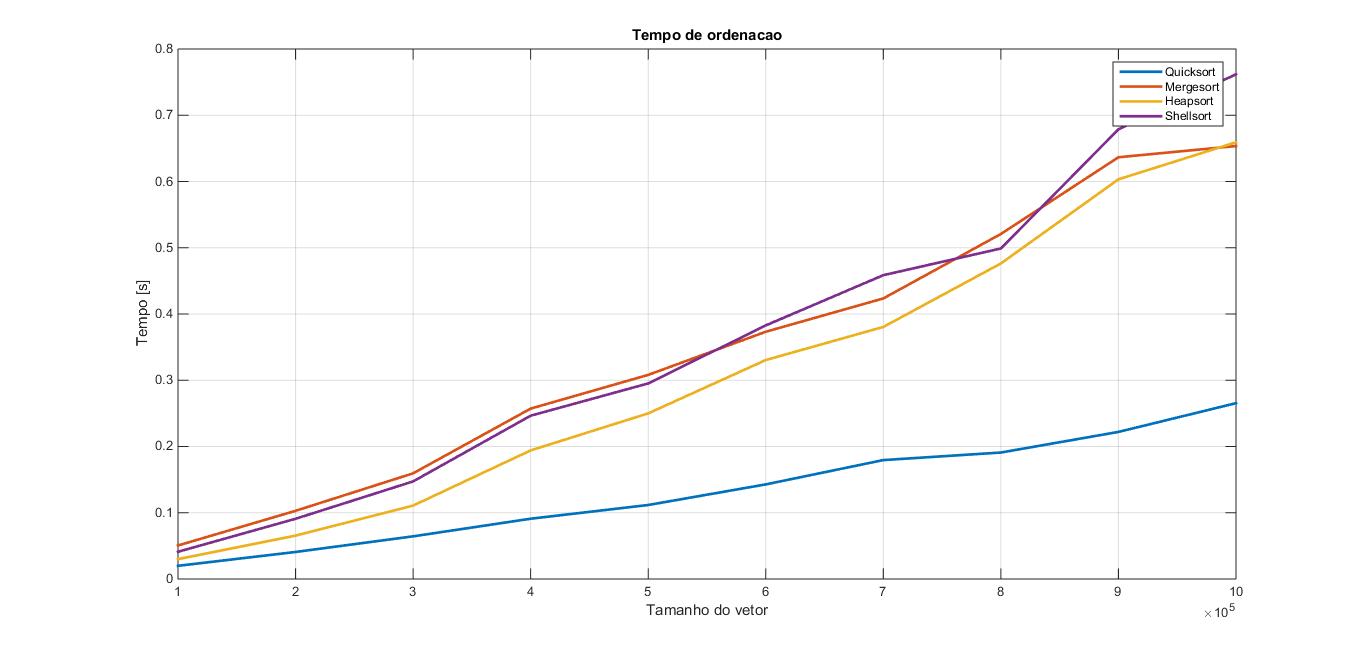
Primeiro grafico, foi utilizado um i5\_4670\_3.4ghz



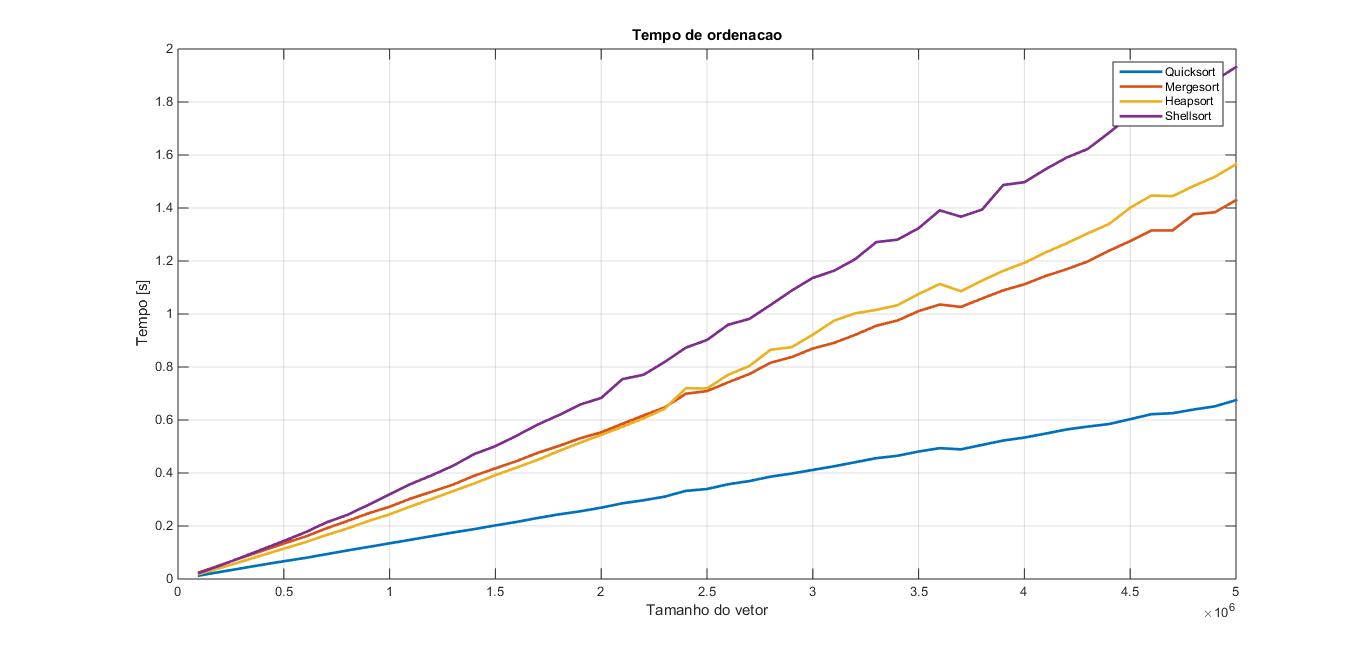
O segundo grafico foi um i7\_860, com overclock de 3.8ghz



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



E o ultimo grafico sobre ordenacao de vetores aleatorios, 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 necessario para percorrer todo o vetor,

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

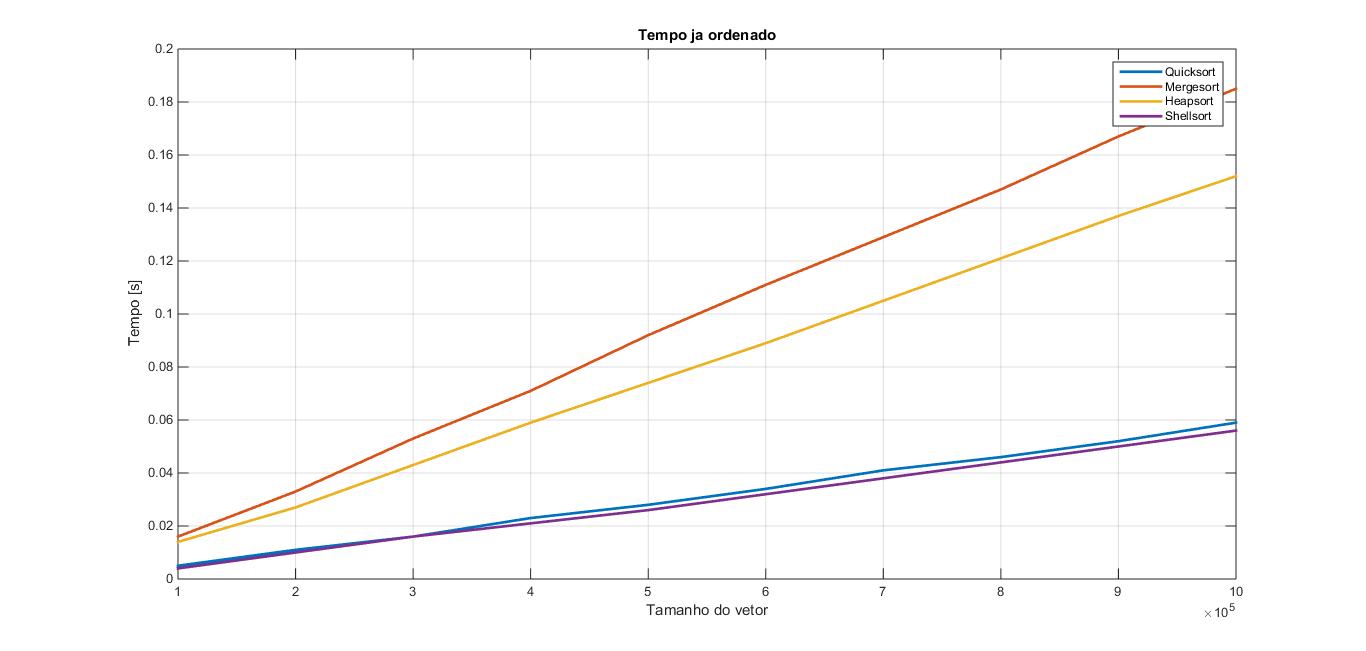
Heapsort trabalha dividin o vetor em 2 partes chamadas peneira, ordenando estas duas separadamente e devolvendo-as ordenadas.

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

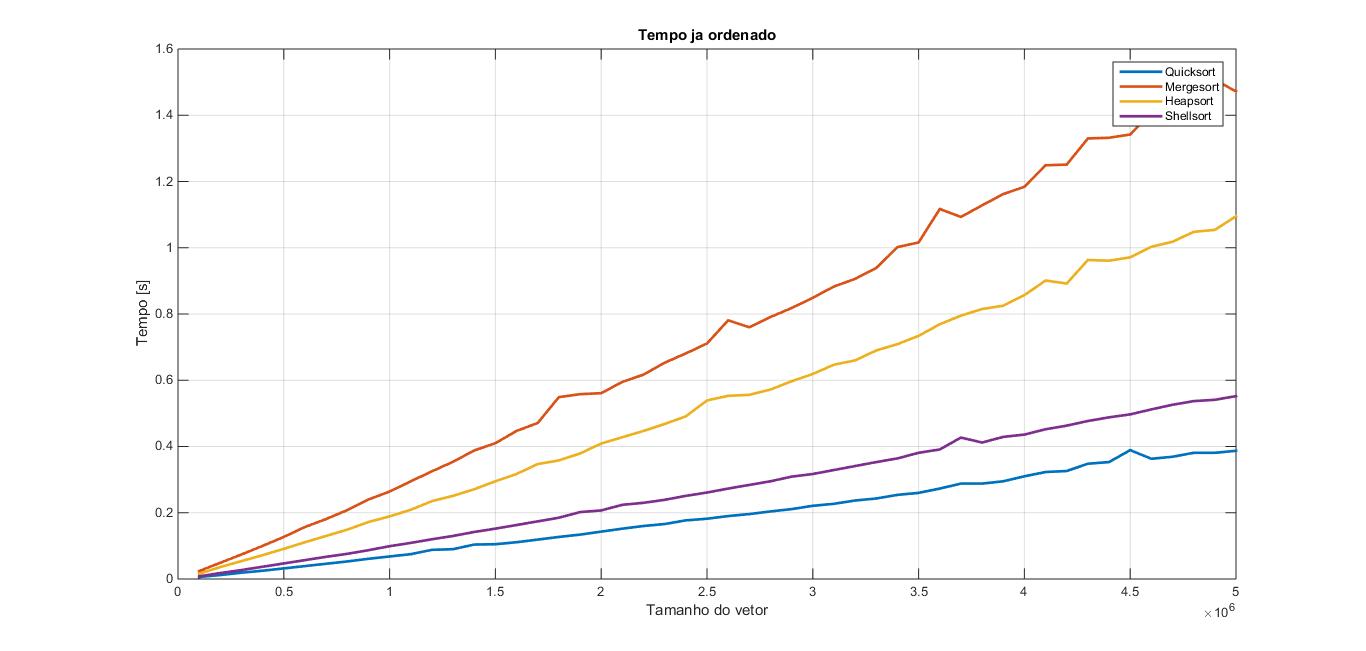
Vetor ja ordenado.

Tambem foi medido tempo medio de ordenacao de um vetor ja ordenado- ou basicamente o tempo que demora pro algoritimo perceber que este vetor ja esta em ordem...

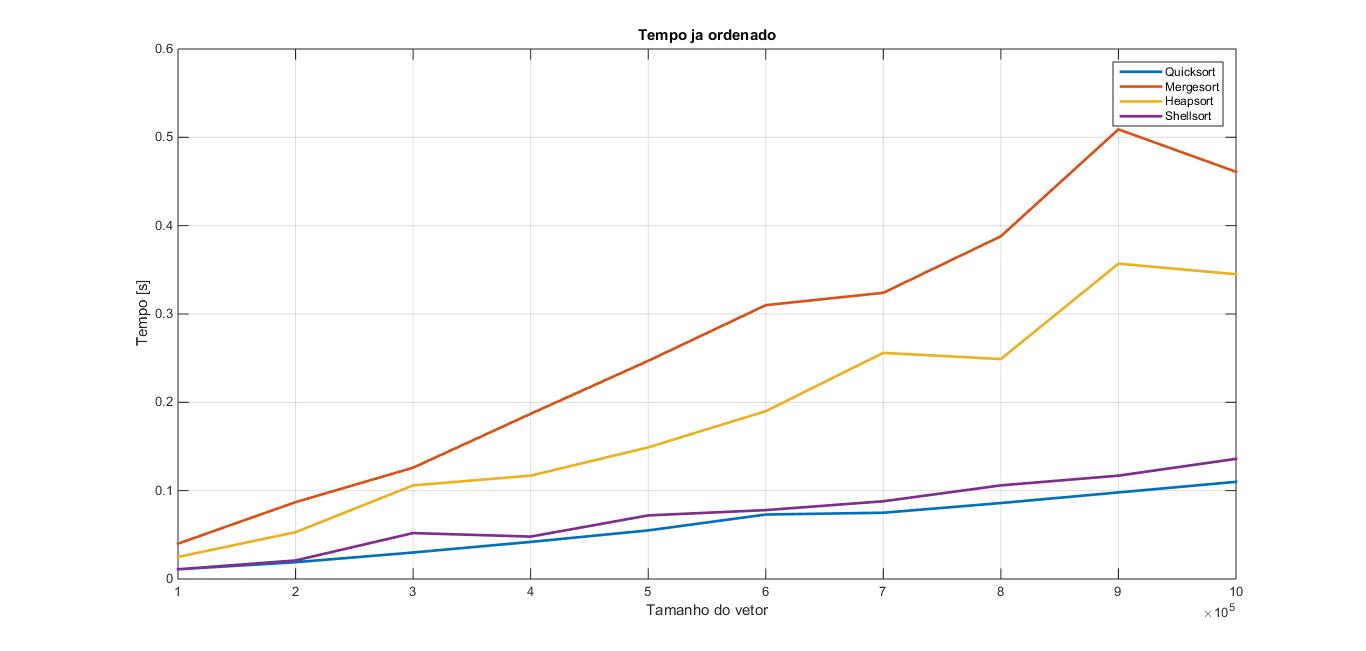
Primeiro grafico i5\_4670 3.4ghz



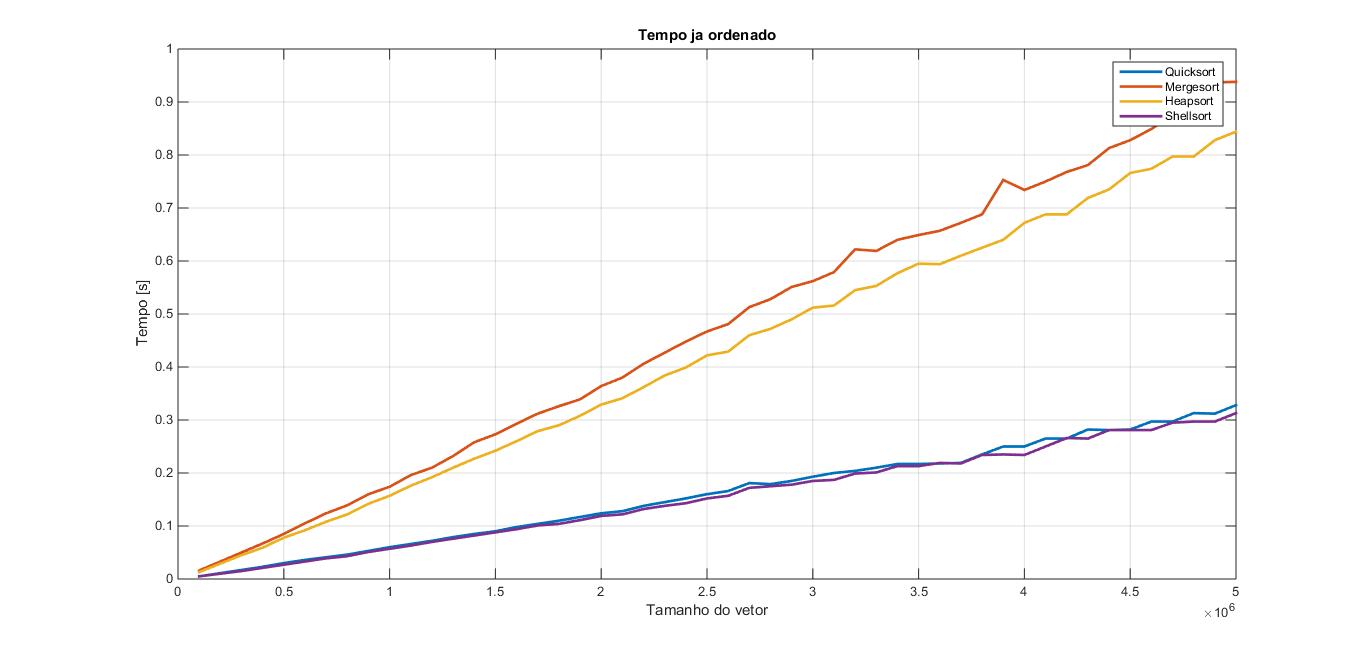
I7\_860 overclock 3.8ghz



Intel dualcore T4\_500 2.3ghz



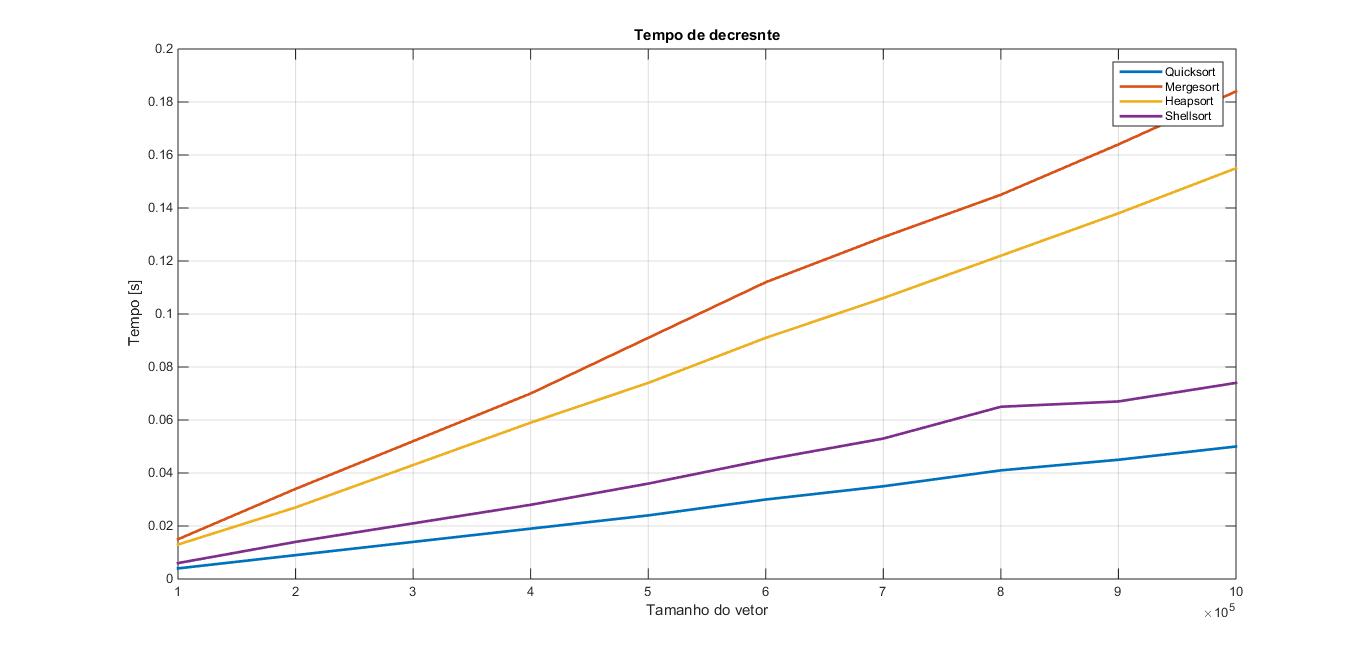
I7\_3610qm 2.3ghz



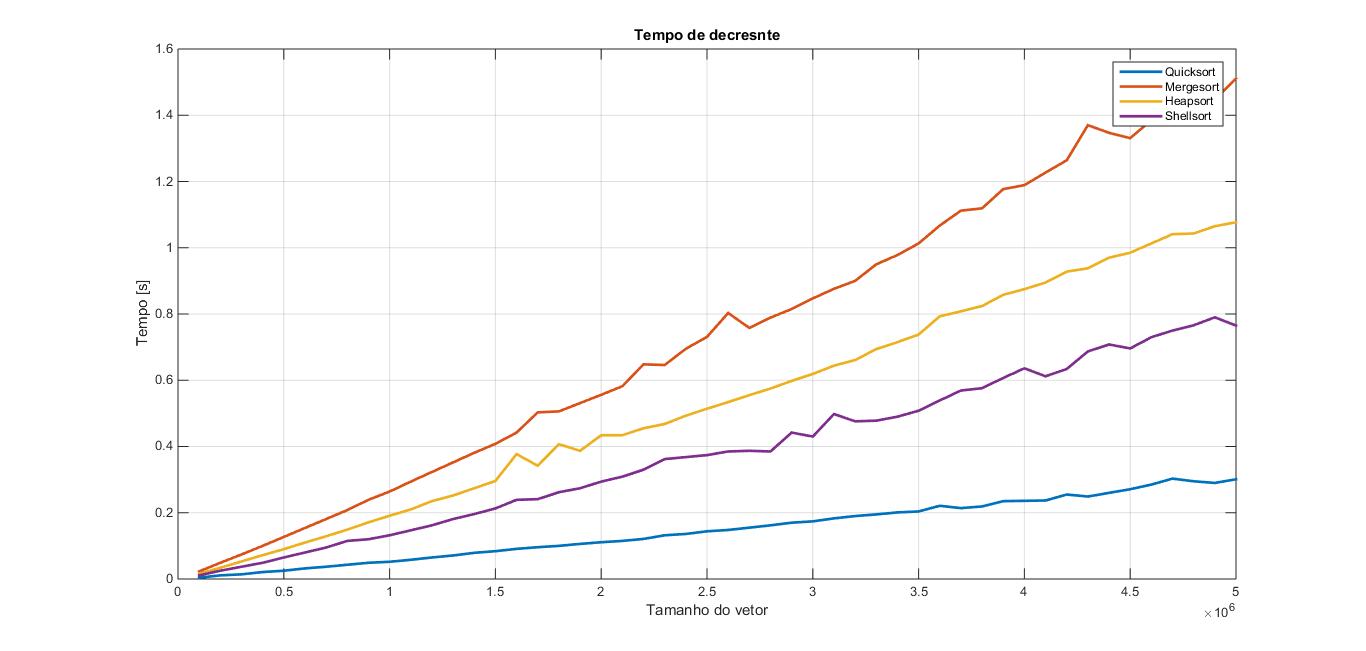
Em todos os graficos, quicksort e shellsort partilharam quase o mesmo tempo, devido a forma do programa ser de “saltos” quicksort indo de metade em metade, e shell a definido pelo usuario. Eles sao mais rapidos nesse processo!

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

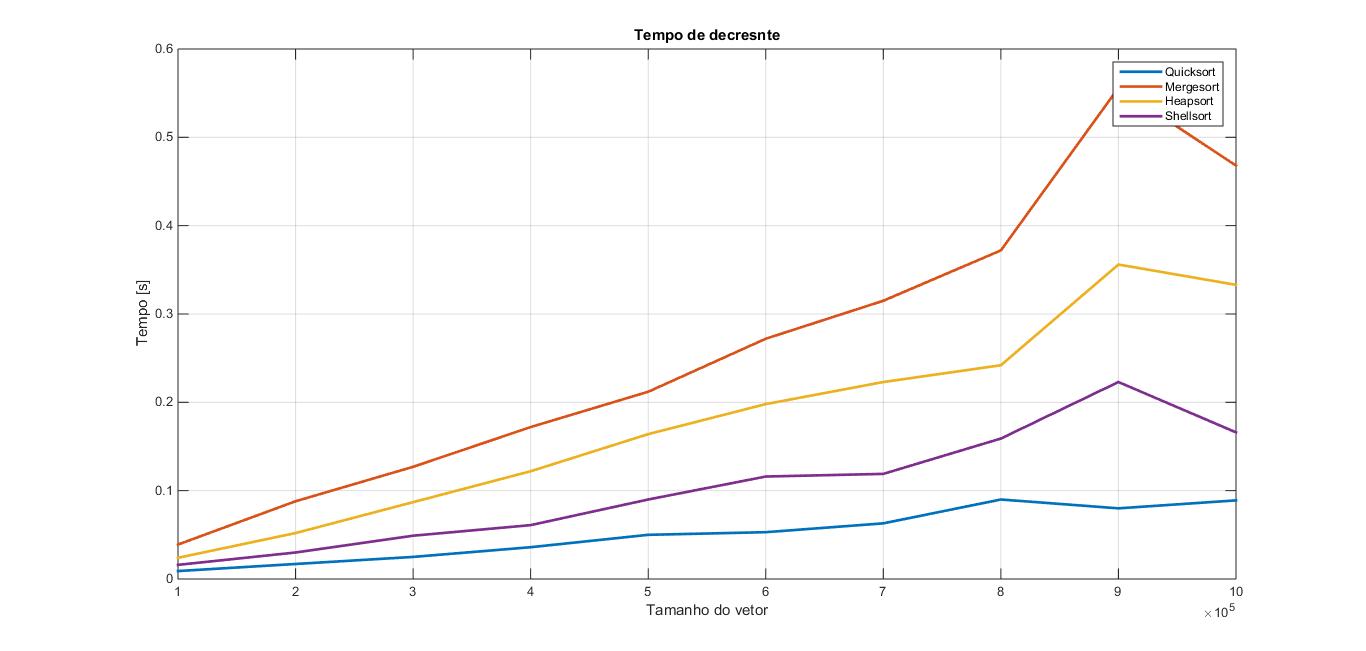
I5\_4670\_3.4ghz



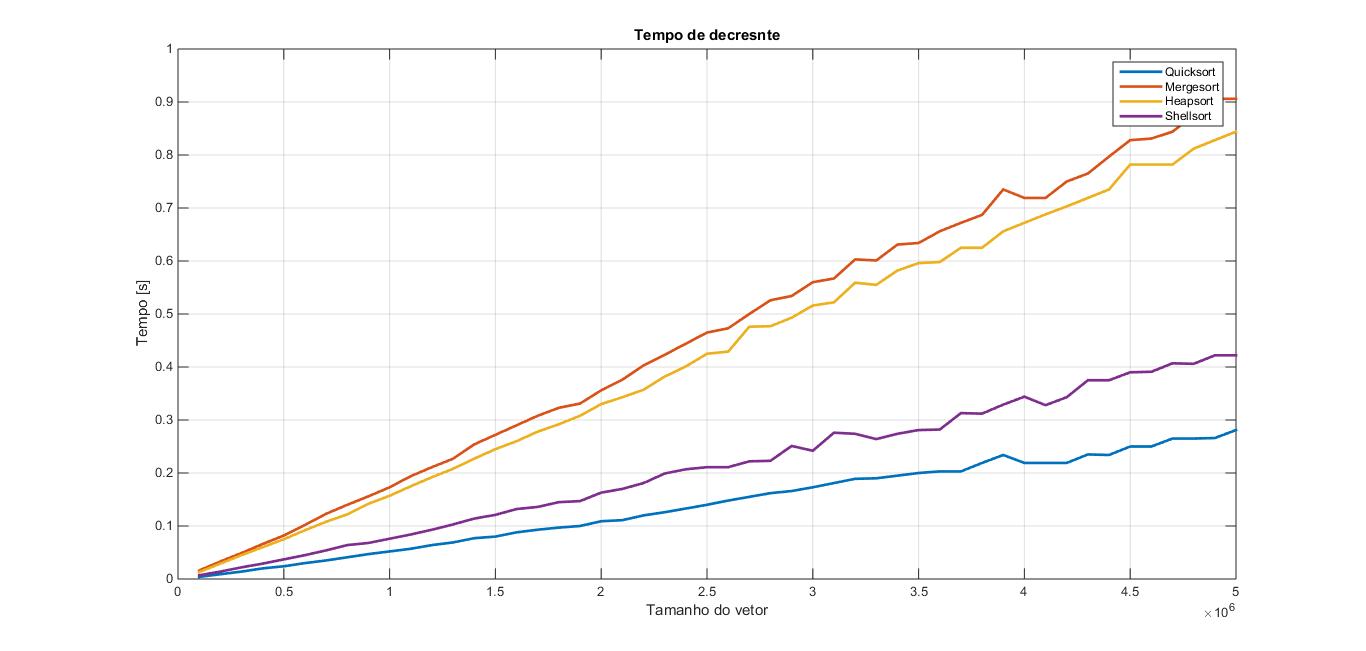
I7\_860 overclock 3.8ghz



Intel dualcore 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 tematica 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 algoritimo entende melhor como o processador trabalha e soube utilizar isso para ordenar mais rapidamente um vetor, pagando o preco de nao ser estavel.