Análise dos algoritmos de ordenação:

Entre os diversos algoritmos de ordenação existente, foram selecionados 6 para o experimento: BubbleSort, SelectionSort, InsertionSort, MergeSort, QuickSort e ContingSort.

Os testes foram feitos em seis vetores de inteiros, sendo três de tamanho 20 e três de tamanho 2000. Foram criados então, de cada tamanho, um vetor já ordenado crescentemente, um aletório e um ordenado decrescentemente.

Após 1010 execuções de cada algoritmo em cada vetor, e desprezando-se as primeiras 10 execuções em cada experimento, obteve-se os seguintes tempos (em ns).

TEMPOS DE EXECUÇÃO (*ns*) DOS ALGORITMOS DE ORDENAÇÃO						
ALGORITMO	CRESCENTE		ALEATORIO		DECRESCENTE	
	N = 20	*N = 2000*	*N = 20*	*N = 2000*	*N = 20*	*N = 2000*
BUBBLE SORT	134	751136	209	3326419	265	1439066
SELECTION SORT	129	400956	201	1206705	218	1601399
INSERTION SORT	31	5493	159	495260	166	937735
MERGE SORT	863	114743	975	221289	971	113694
QUICK SORT	264	1451614	189	108989	397	3021708
COUNTING SORT	107	20597	109	7718	107	7522

Pela tabela acima, é fácil ver que o algoritmo de pior performance é o BubbleSort, e o de melhor performance é o CountingSort.

Para uma melhor visualização dos resultados, vamos analisar os dados através dos gráficos abaixo:

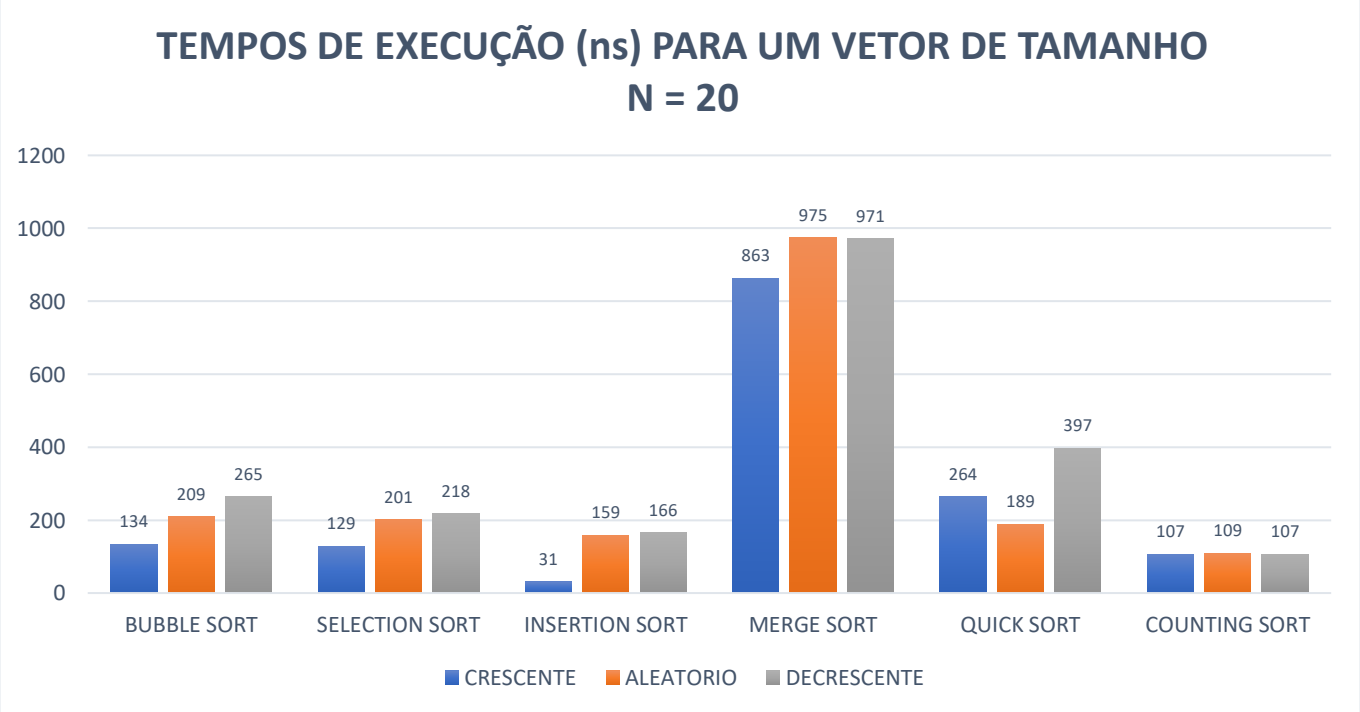

Gráfico (1)

Pelo *Gráfico 1* podemos notar que os algoritmos Bubble, Selection e Insertion têm desempenhos bem parecidos. Já o QuickSort se diferencia dos demais pelo fato de ter um melhor desempenho no vetor aleatório. Quanto ao CountingSort vemos que seu tempo de execução praticamente não varia em relação à ordenação inicial do vetor. O MergeSort por sua vez é o que tem o pior desempenho de todos com um vetor de tamanho 20.

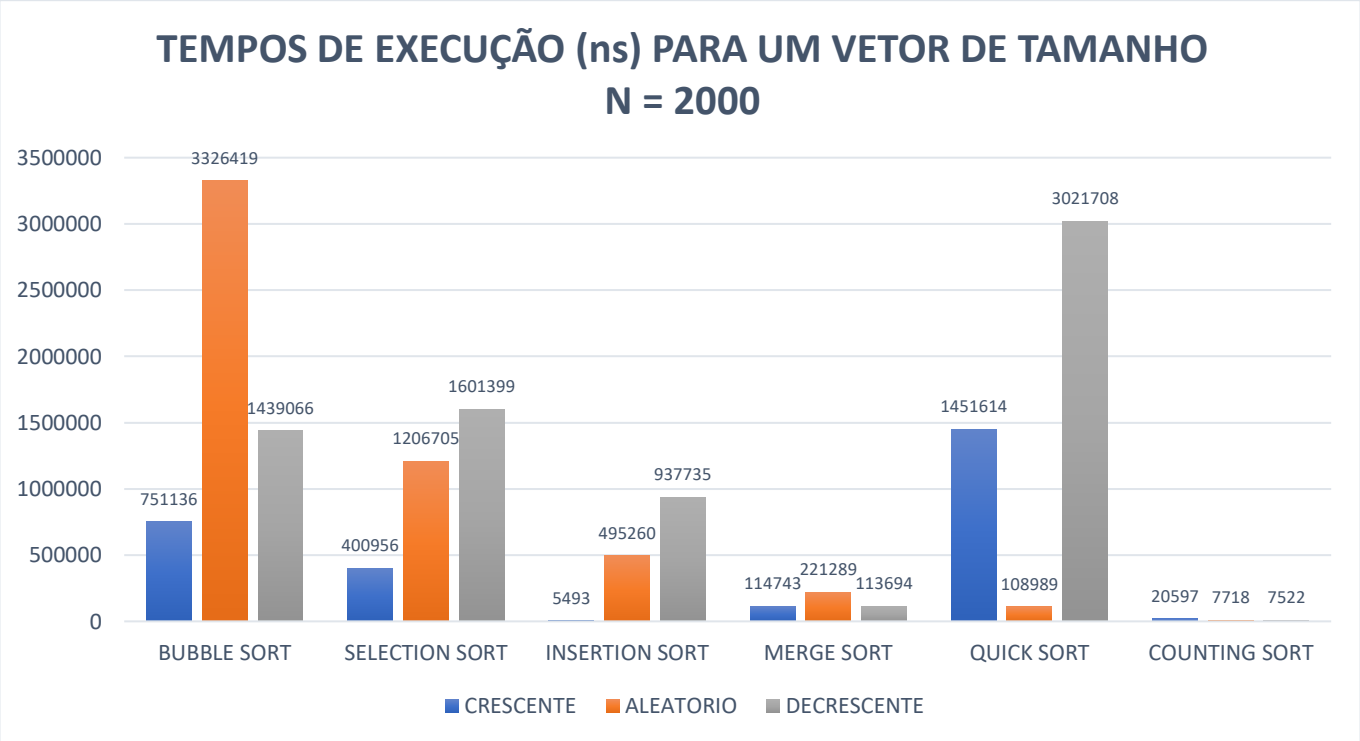

Gráfico (2)

Podemos ver agora pelo *Gráfico (2)* que a situação se inverte no que diz respeito ao MergeSort, que agora passa a ter melhores desempenhos para todas as condições iniciais, perdendo apenas para o CountigSort. O QuickSort manteve sua característica de ter um ótimo desempenho para um vetor aleatório, mas para os vetores já ordenados seu desempenho piorou muito, chegando a ser tão ruim quanto o BubbleSort, que dessa vez é quem apresenta os piores resultados.

Essa tendência se mantém para vetores maiores, o que é bem razoável já que o Bubble, Selection e Insertion têm, assintoticamente, tempos de $O(n^2)$. O MergeSort é da ordem de $O(n \log n)$. O QuickSort, quando o vetor está ordenado, é do tipo $O(n^2)$, porém tem um desempenho de $O(n \log n)$ quando o vetor é aleatório. Já o CountigSort é linear, o que o faz ter um desempenho melhor q todos, mas é menos usado devido a algumas restrições que possui.