SUCLA -

Sorting Using C Lovely Algorithms.

BRUNO, FILIPE, MATHEUS, VICTOR HUGO

Bubble Sort

A classificação por bolha, ou Bubble Sort, é um algoritmo básico para organizar uma sequência de números ou outros elementos na ordem correta. O método funciona examinando cada conjunto de elementos adjacentes na string, da esquerda para a direita, trocando suas posições se estiverem fora de ordem. O algoritmo então repete esse processo até que possa percorrer toda a string e não encontrar dois elementos que precisem ser trocados.

Tamanho médio da Lista	Número médio de comparações	Número médio de trocas	Tempo médio gasto
1000~10k			
10K~100K			
100k~1M			
1M~10M			

Comb Sort

O algoritmo Comb sort (ou CombSort ou ainda algoritmo do pente) é um algoritmo de ordenação relativamente simples, e faz parte da família de algoritmos de ordenação por troca. O Comb Sort melhora o Bubble Sort, e rivaliza com algoritmos como o Quick Sort. A ideia básica é eliminar as tartarugas ou pequenos valores próximos do final da lista, já que em um bubble sort estes retardam a classificação tremendamente.

Tamanho médio da Lista	Número médio de comparações	Número médio de trocas	Tempo médio gasto
1000~10k			
10K~100K			
100k~1M			
1M~10M			

Insertion Sort

A classificação por bolha, ou Bubble Sort, é um algoritmo básico para organizar uma sequência de números ou outros elementos na ordem correta. O método funciona examinando cada conjunto de elementos adjacentes na string, da esquerda para a direita, trocando suas posições se estiverem fora de ordem. O algoritmo então repete esse processo até que possa percorrer toda a string e não encontrar dois elementos que precisem ser trocados.

Tamanho médio da Lista	Número médio de comparações	Número médio de trocas	Tempo médio gasto
1000~10k			
10K~100K			
100k~1M			
1M~10M			

Merge Sort

Sua ideia básica consiste em Dividir (o problema em vários subproblemas e resolver esses subproblemas através da recursividade) e Conquistar (após todos os subproblemas terem sido resolvidos ocorre a conquista que é a união das resoluções dos subproblemas). Como o algoritmo *Merge Sort* usa a recursividade, há um alto consumo de memória e tempo de execução, tornando esta técnica não muito eficiente em alguns problemas.

Tamanho médio da Lista	Número médio de comparações	Número médio de trocas	Tempo médio gasto
1000~10k			
10K~100K			
100k~1M			
1M~10M			

Shell Sort

O shell sort, às vezes chamado de “ordenação por incrementos diminutos”, melhora a ordenação por inserção ao quebrar a lista original em um número menor de sub listas, as quais são ordenadas usando a ordenação por inserção. A forma única como essas sub listas são escolhidas é a chave para o shell sort. Em vez de quebrar a lista em sub listas de itens contíguos, o shell sort usa um incremento i, às vezes chamado de gap, para criar uma sub lista escolhendo todos os itens que estão afastados i itens uns dos outros

Tamanho médio da Lista	Número médio de comparações	Número médio de trocas	Tempo médio gasto
1000~10k			
10K~100K			
100k~1M			
1M~10M			

Quick Sort

O quick sort adota a estratégia de divisão e conquista. A estratégia consiste em rearranjar as chaves de modo que as chaves "menores" precedem as chaves "maiores". Em seguida o quick sort ordena as duas sub listas de chaves menores e maiores recursivamente até que a lista completa se encontre ordenada

Tamanho médio da Lista	Número médio de comparações	Número médio de trocas	Tempo médio gasto
1000~10k			
10K~100K			
100k~1M			
1M~10M