Estrutura de Dados II

Estudo Complementar I

Nome: Hygor Xavier Araújo

Matrícula: 200965363

TimSort – Algoritmo de Ordenação:

O TimSort é um algoritmo de ordenação hibrido, que utiliza que utiliza características de outros dois algoritmos de ordenação, o insertion sort e o merge sort. Ele possui uma boa performance com muitos tipos de dados do mundo real.

Ele foi projetado para tirar proveito de ordenações parciais já existentes na maior parte dos dados, é considerado estável, e possui complexidade O(n) no melhor caso, O(nlog n) no caso médio e O(nlog n) no pior caso.

Como ele opera:

O TimSort opera dividindo um vetor de entrada em sub-vetores chamados runs, que são sub-vetores ordenados (de tamanho mínimo >=2) em forma não-descendente ou estritamente descendente. Ele então ordena cada sub-vetor utilizando o insertio sort e depois uni todos os sub-vetores em um único vetor ordenado com o merge sort.

Passos realizados pelo algoritmo:

1. Se o tamanho do vetor for menor que 64 ele usa o insertion sort para ordenar o vetor, caso não seja segue para o passo 2
2. Calcula o valor de minrun (Para um vetor de tamanho N, N/minrun igual a uma potência de 2 ou ligeiramente menor é uma boa escolha)
3. Acha as runs
   1. Se a run for estritamente descendente faz uma ordenação in-place com ela

Se a run tem tamanho menor que minrun o insertion sort é usado para aumentar o tamanho da run até o tamanho de minrun

1. Empilha as runs em uma pilha
2. Faz o merge das runs empilhadas seguindo 2 regras (sejam A, B e C runs na pilha, com A no topo, seguido por B e C)
   1. A > B + C
   2. B > C
3. Se A < B + C, A é intercalado(merge) com B, C e (A+B) são empilhados novamente e ele checa as 2 regras novamente
4. Repete o merge até que o vetor esteja ordenado

Timsort(inteiro vetor[], inteiro tamanho)

Inicio

int minrun;

pilha\_de\_runs pilha[n];

se tamanho < 64 então

Insertionsort(vetor, tamanho);

fim-se;

minrun <- calcula\_minrun(tamanho); //calcula o valor de minrun

empilha\_runs(pilha);

enquanto Verdadeiro faça

A <- desempilha(pilha);

B <- desempilha(pilha);

C <- desempilha(pilha);

se A < B + C então

empilha(pilha, C);

empilha(pilha, merge(A, B));

senão se B < C então

empilha(pilha, merge(B, C));

empilha(pilha, A);

senão

fim;

fim-se;

fim;