Algoritmo Proposto: Double Sort

Explicação da Lógica:

O algoritmo proposto tem como ideia, separar o Array em duas partes, e ordenar cada uma das partes com algoritmos de ordenação diferentes já existentes, e então, organiza-los a partir de uma espécie de “IntercalaComSentinela”.

Ilustração do Algoritmo Proposto:

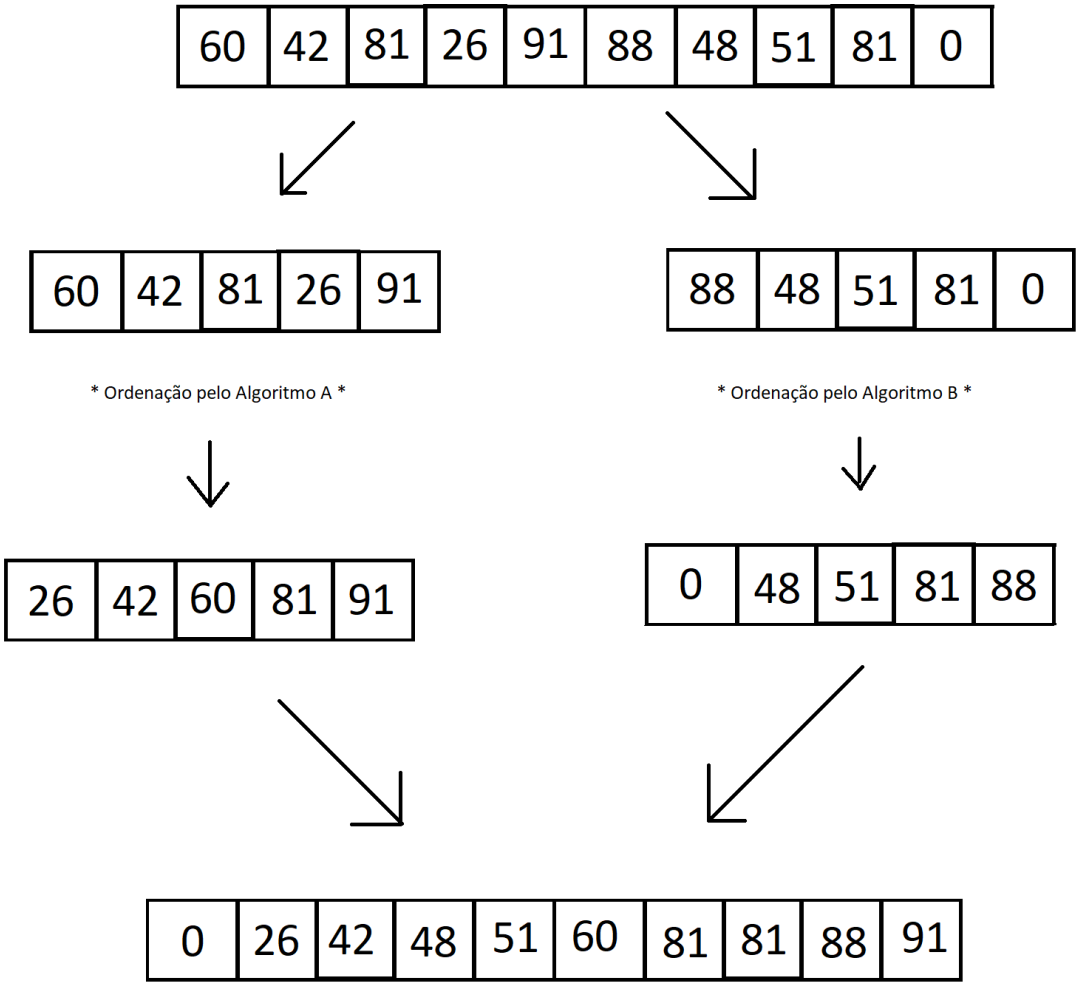

Imagem 1 – Ilustração Visual do Algoritmo Proposto

Estudo da Complexidade

Algoritmo Utilizado:

```
371  void OrdenaDoubleSort(int arrayA[], int n){
372
373      int i, j, k;
374      int arrayESQ[n/2];
375      int arrayDIR[n/2];
376
377      for(i=0; i<n/2; i++){
378          numComparacoesDS++;
379          arrayESQ[i] = arrayA[i];
380      }
381
382      for(i=0; i<n/2; i++){
383          numComparacoesDS++;
384          arrayDIR[i] = arrayA[i+(n/2)];
385      }
386
387      insertionSortDouble(arrayDIR, n/2);
388      OrdenaBubbleSortDouble(arrayESQ, n/2);
389
390      i=0;
391      j=0;
392      arrayESQ[n/2] = 100000;
393      arrayDIR[n/2] = 100000;
394
395      for(int k=0; k<n; k++){
396          numComparacoesDS++;
397          numTrocasDS++;
398          if(arrayESQ[i] < arrayDIR[j]){
399              arrayA[k] = arrayESQ[i];
400              i++;
401          } else {
402              arrayA[k] = arrayDIR[j];
403              j++;
404          }
405      }
406  }
```

Imagem 2 – Código em ‘C’ do Algoritmo Proposto

Como a proposta do algoritmo é que ele possa utilizar qualquer um dos algoritmos de ordenação já existentes. O estudo da complexidade

pode mudar a partir da escolha do programador. De forma geral, o algoritmo tem como complexidade o Θ do algoritmo mais pesado entre os dois utilizados. Como podemos ver no estudo abaixo:

LINHA	COMPLEXIDADE
373	3
374	1
375	1
377	1 + (n/2+1) + n/2
378	n/2
379	n/2
382	1 + (n/2+1) + n/2
383	n/2
384	n/2
387	Θ((Algoritmo A) / 2)
388	Θ((Algoritmo B) / 2)
390	1
391	1
392	1
393	1
395	1 + (n+1) + n
396	n
397	n
398	n
399	n/2
400	n/2
402	n/2
403	n/2
TOTAL =	Θ((Algoritmo A) / 2) + Θ((Algoritmo B) / 2) + 11n + 15
Θ =	Θ(Algoritmo mais "Lento")

Imagem 3 – Estudo da Complexidade para qualquer caso de Algoritmos utilizados

Na imagem Ilustrativa do algoritmo proposto, foram utilizados os algoritmos “Insertion Sort” e o “Bubble Sort”. Este é o estudo para o algoritmo utilizando estes dois métodos de ordenação:

LINHA	COMPLEXIDADE
373	3
374	1
375	1
377	$1 + (n/2+1) + n/2$
378	$n/2$
379	$n/2$
382	$1 + (n/2+1) + n/2$
383	$n/2$
384	$n/2$
387	$\Theta(n^2/2)$
388	$\Theta(n^2/2)$
390	1
391	1
392	1
393	1
395	$1 + (n+1) + n$
396	n
397	n
398	n
399	$n/2$
400	$n/2$
402	$n/2$
403	$n/2$
TOTAL =	$n^2 + 11n + 15$
Θ =	$\Theta(n^2)$

Imagem 4 – Estudo da Complexidade para o Algoritmo Proposto utilizando o “Insertion Sort” e “Bubble Sort”