



Pesquisa e Ordenação de Dados

(Ordenação de Dados - Radix Sort)

Maurício Witter

euiciowr@mail.ufsm.br



UFSM

Universidade Federal de Santa Maria

Departamento de Tecnologia da Informação

RADIX SORT

- **Definição:**
- **Ordenação estável:**
 - O algoritmo de ordenação Radix Sort preserva a ordem relativa dos elementos que têm o mesmo valor de chave, ou seja, não troca valores duplicados.
- **Big-O**
 - $O(nk)$: onde n é o número de elementos e k é o número de valores que cada dígito pode ter. E a complexidade de espaço é $O(n+k)$.
- **Princípio:**
 - Utiliza o algoritmo Counting Sort como sub-rotina, o counting sort utiliza um array auxiliar de contagem para classificar o array pela contagem e não por comparação. O Radix Sort classifica os valores pegando o número menos significativo de todos os elementos e vai fazendo a contagem pelo Counting Sort para trocar os elementos de posição de acordo com a contagem e itera sobre o array de valores até chegar no valor mais significativo e ordenar sem comparações.

RADIX SORT

- **Aplicação:**
 - Onde é/foi aplicado? É/Foi utilizado em alguma linguagem de programação, SGBD ou em outra tecnologia/aplicação/situação real do mundo real?
- IBM card sorter
 - Ordenar cartões que tinham de 1 a 9 colunas e podiam ser ordenados pelo radix partindo dos dígitos menos significativos para os mais significativos até ordenar por completo.
- Algoritmo DC3 (Kärkkäinen-Sanders-Burkhardt)
 - Manipular grandes quantidades de dados como arrays de sufixos, é usado para Bibliometria e pesquisa de sequências de DNA, por exemplo.
- Não é aplicado em linguagens pois consome bastante memória dependendo do caso e não é tão eficiente dependendo da base.

RADIX SORT

- **Funcionamento:**
- **Complexidade:**
 - $O(d*(n+k))$: n é o número de itens para ordenar, d é o número de dígitos que cada item tem e k é o número de valores que cada dígito tem.
- O Radix Sort ordena os elementos por meio de um único dígito, começando pelo final, pelos dígitos menos significativos, ele ordena de forma crescente ou decrescente e repete o processo até chegar no primeiro dígito mais significativo.
- **EXEMPLO: $A = [44, 65, 4, 676]$**
 - $i = 0 ::$ **044, 065, 004, 676**
 - $i = 1 ::$ **044, 004, 065, 676**
 - $i = 2 ::$ **004, 044, 065, 676**
 - $i = 3 ::$ **4, 44, 65, 676**

RADIX SORT

- **Implementação:**
 - [Código completo encontra-se no Github. Link aqui.](#)
 - <https://github.com/rwietter/radixsort>

RADIX SORT

■ Teste de Mesa:

```
public class Radixsort {  
    CountingSort cs = new CountingSort();  
    int getMax(int array[], int n) {  
        int max = array[0];  
        for (int i = 1; i < n; i++)  
            if (array[i] > max)  
                max = array[i];  
        return max;  
    }  
  
    public void radixSort(int array[], int len) {  
        int max = getMax(array, len);  
        for (int place = 1; max / place > 0; place *= 10)  
            this.cs.countingSort(array, len, place);  
    }  
}
```

Teste de mesa está em imagem externa pois não cabe aqui, foi feito de forma mais direta para não dar muito grande.

Segue o link:

[teste de mesa Radix Sort](https://raw.githubusercontent.com/rwietter/radixsort/master/assets/table_test_radix_sort.png?token=ALFPCQ636BMOIULYPBJF5P27AZ73Q)

https://raw.githubusercontent.com/rwietter/radixsort/master/assets/table_test_radix_sort.png?token=ALFPCQ636BMOIULYPBJF5P27AZ73Q

RADIX SORT

- **Referências:**
 - [Radix Sort Algorithm](#)
 - [Radix Sort](#)
 - [Radix Sort Algorithm](#)
 - [Radix Sort Algorithm](#)