COMB SORT

Equipe:

- Jonatas Travessa Souza de Barros
- Lucas Mendes Sonoda
- Natanael da Mota Figueira
- Paulo André Carneiro Fernandes

Roteiro

- Onde usar e como usar
- Tipo de ordenação
- Forma de ordenação
- Complexidade
- Estabilidade
- Vantagens
- Desvantagens
- Código
- Exemplo passo a passo
- Execuções

Onde usar e como usar

- Usado em arrays e listas dinâmicas na ordenação
- necessário definir um fator de encolhimento (valor geralmente 1.3)

Tipo de ordenação

- ordenação completa

Forma de ordenação

- ordenação por troca

Complexidade

- Melhor caso : O(nlogn)
- Caso médio : Ω(N^2/2^p)
- Pior Caso : O(n^2)

Estabilidade

Comb Sort não é um algoritmo de ordenação estável já que não ordena os elementos na mesma ordem em que aparecem inicialmente.

Vantagens

- Não precisa de espaço adicional para ordenar os elementos.
- Funciona melhor que bubble sort, apesar de serem similares.
- A lógica é simples.

Desvantagens

- A complexidade do pior caso é a mesma que a do bubble sort.
- Não funciona bem com grandes quantidades de dados.
- Não é uma ordenação estável.

Código Comentado

```
1   #include <iostream>
2   #include <algorithm> //swap
3
4   using namespace std;
5
6   /* Calcula próximo gap */
7   int next_gap(int gap){
8
9       /* Gap não pode ser menor do que 1 */
10      if (gap <= 1)
11          return 1;
12
13      return gap/1.3;
14  }
```

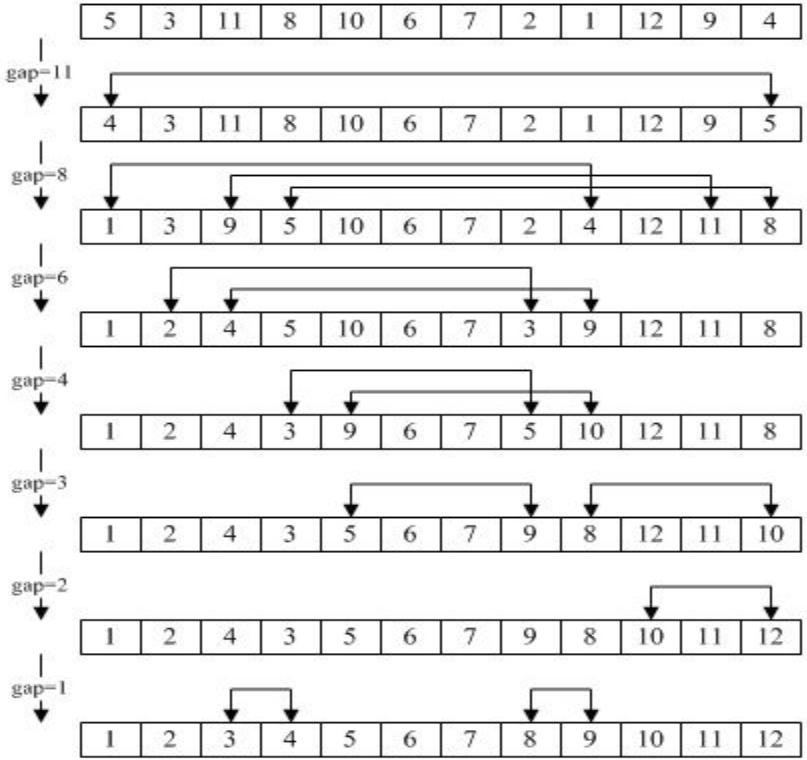
Código Comentado

```c
/* Comb_sort crescente */
void comb_sort_cres(int *v, int size){

    /* tamanho inicial do gap (tamnho  do vetor) */
    int gap = size;

    int swapped = 1;

    /* Ordena até gap == 1 e na última passagem não ter tido troca*/
    while(gap != 1 || swapped == 1){
        gap = next_gap(gap);

        swapped = 0;

        /* Loop de iteração sob o vetor considerando o gap */
        for (int i = 0; i < size - gap; i++){
            if (v[i] > v[i + gap]){
                swap(v[i], v[i + gap]);
                swapped = 1;
            }

        }
    }
}
```

Código Comentado

```
41  /* Comb_sort decrescente */
42  void comb_sort_decres(int *v, int size){
43
44      /* tamanho inicial do gap (tamnho  do vetor) */
45      int gap = size;
46
47      int swapped = 1;
48
49      /* Ordena até gap == 1 e na última passagem não ter tido troca*/
50      while(gap != 1 || swapped == 1){
51          gap = next_gap(gap);
52
53          swapped = 0;
54
55          /* Loop de iteração sob o vetor considerando o gap */
56          for (int i = 0; i < size - gap; i++){
57              if (v[i] < v[i + gap]){
58                  swap(v[i], v[i + gap]);
59                  swapped = 1;
60              }
61
62          }
63      }
64  }
65
```

Exemplo Passo a Passo



Execuções com 100000 números - Configurações

– – –

- Configuração 1: vetor com elementos totalmente desordenados;
- Configuração 2: vetor com elementos ordenados crescentemente;
- Configuração 3: vetor com elementos ordenados decrescentemente;
- Configuração 4: vetor com a primeira metade ordenada crescentemente e a segunda metade ordenada decrescentemente;
- Configuração 5: vetor com a primeira metade ordenada decrescentemente e a segunda metade ordenada crescentemente.

Execuções com 100000 números - Resultados

Referências

- MANISH BHOJASIA. Sanfoundry. Comb Sort Multiple Choice Questions and Answers (MCQs). Disponível em: https://www.sanfoundry.com/comb-sort-multiple-choice-questions-answers-mcqs/. Acesso em: 10 de maio de 2021.

- COMB SORT. **GeeksforGeeks**. Disponível em: https://www.geeksforgeeks.org/comb-sort/. Acesso em: 7 de maio de 2021.

- COMB SORT - ALGORITHMS. **H.urna**. Disponível em: https://hurna.io/academy/algorithms/sort/comb.html. Acesso em: 10 de maio de 2021.

- COMB SORT. **Mycarrerwise**. Disponível em:https://mycareerwise.com/programming/category/sorting/comb-sort. Acesso em: 10 de maio de 2021.

Obrigado