MAC121 - Algoritmos e Estruturas de Dados I

Universidade de São Paulo

Segundo Semestre de 2020

Ordenação em tempo linear - radixsort

Ordenação indireta

Radixsort

Problema: Considere que queremos ordenar n números inteiros, cada um com no máximo d dígitos.

1245	42**3**0	9**9**00	2**12**1	**9**19
4230	13**9**0	9**1**9	8**12**2	**11**99
919	99**0**0	2**1**21	7**12**2	**12**45
1390	12**7**0	1**32**2	4**13**3	**12**70
3175	21**2**1	8**12**2	7**17**1	**13**22
2121	71**7**1	7**12**2	3**17**5	**13**90
1322	13**2**2	4**23**0	1**19**9	**21**21
4133	81**2**2	4**13**3	4**23**0	**31**75
9900	71**2**2	1**24**5	1**24**5	**41**33
1199	41**3**3	1**27**0	1**27**0	**42**30
7171	12**4**5	7**17**1	1**32**2	**71**22
8122	31**7**5	3**17**5	1**39**0	**71**71
7122	9**1**9	1**39**0	9**90**0	**81**22
1270	11**9**9	1**19**9	0**91**9	**99**00

Radixsort

```
para cada um dos dígitos, começando no menos
significativo
  ordene os números por este dígito
```

Animação do algoritmo

Ordenação indireta

Muitas vezes os elementos que estão ordenando são muito grandes (structs, ou vetores grandes) e não desejamos movimentá-los no algoritmo de ordenação. Neste caso se usa a ideia de ordenação indireta.

Mantemos um vetor `ord` que vai armazenar na posição i a posição que $v[i]$ deverá ocupar na ordenação final.

para $i = 0, \ldots, n-1$

```
  ord[i] = i;
```

As comparações entre os elementos do vetor são feitas, então, indiretamente:

```
if (v[ord[i]] > v[ord[j]])
```

E as trocas são, então, feitas no vetor `ord`.

Exemplo de ordenação indireta

```
void bubblesort (int *v, int *ord, int n) {
  int i, j, aux;
  for (i = 0; i < n; i++) ord[i] = i;
  for (i = 0; i < n; i++)
    for (j = 0; j < n - i - 1; j++)
      if (v[ord[j]] > v[ord[j + 1]]){
        aux = ord[j+1];
        ord[j+1] = ord[j];
        ord[j] = aux;
      }
}
```

Os elementos de `v` permanecem nas mesmas posições. O vetor `ord` guarda a ordem dos elementos na ordenação.