Сортировка выбором

Слёлин А.В.

17 мая, 2024 г.

Описание алгоритма

Пусть дан массив $A[0..n-1]$, такой что $n = \text{length}[A]$, причём индексация начинается с нуля для облегчения понимания кода.

Рассмотрим случай когда нам нужно отсортировать массив A в неубывающую последовательность; случай, когда нужно отсортировать в невозрастающую последовательность аналогичен.

Алгоритм сортировки выбором заключается в том, чтобы из массива $A[0..n-1]$ выбрать наименьший элемент с индексом i и поменять его с нулевым элементом. Затем найти наименьший элемент в массиве $A[1..n-1]$ и поменять его с первым элементом. Таким образом мы уменьшаем массив из которого выбираем наименьший элемент и в итоге получаем отсортированную последовательность.

Пример

Пусть массив $A = [8, 3, 12, 16, 10, 4, 1, 9, 7, 14]$. i - означает начальный индекс массива A, с которого начинается поиск наименьшего элемента (подсвечено красным цветом). Элементы с индексами меньшими i уже отсортированы (подсвечиваются зелёным цветом); наименьший найденный элемент подсвечивается жёлтым цветом.

i = 0:

$$[8, 3, 12, 16, 10, 4, 1, 9, 7, 14];$$

$$[1, 3, 12, 16, 10, 4, 8, 9, 7, 14];$$

i = 1:

$$[1, 3, 12, 16, 10, 4, 8, 9, 7, 14];$$

$$[1, 3, 12, 16, 10, 4, 8, 9, 7, 14];$$

i = 2:

$$[1, 3, 12, 16, 10, 4, 8, 9, 7, 14];$$

$$[1, 3, 4, 16, 10, 12, 8, 9, 7, 14];$$

i = 3:

$$[1, 3, 4, 16, 10, 12, 8, 9, 7, 14];$$

$$[1, 3, 4, 7, 10, 12, 8, 9, 16, 14];$$

i = 4:

$$[1, 3, 4, 7, 10, 12, 8, 9, 16, 14];$$

$$[1, 3, 4, 7, 8, 12, 10, 9, 16, 14];$$

i = 5:

$$[1, 3, 4, 7, 8, 12, 10, 9, 16, 14];$$

$$[1, 3, 4, 7, 8, 9, 10, 12, 16, 14];$$

i = 6:

$$[1, 3, 4, 7, 8, 9, 10, 12, 16, 14];$$

$$[1, 3, 4, 7, 8, 9, 10, 12, 16, 14];$$

i = 7:

$$[1, 3, 4, 7, 8, 9, 10, 12, 16, 14];$$

$$[1, 3, 4, 7, 8, 9, 10, 12, 16, 14];$$

i = 8:

$$[1, 3, 4, 7, 8, 9, 10, 12, 16, 14];$$

$$[1, 3, 4, 7, 8, 9, 10, 12, 14, 16].$$

Теперь очевидно, что последний элемент является наибольшим в массиве A, так что массив A отсортирован. Для получения невозрастающей последовательности следует изменить знак сравнения.

Код сортировки

Листинг 1: selectionSort

```
1      public static void selectionSort(int[] A, boolean reverse) {
2          for (int i = 0; i < A.length - 1; ++i) {
3              int index = i;
4              for (int j = i; j < A.length; ++j)
5                  if (reverse ? A[index] < A[j] : A[index] > A[j])
                       index = j;
6
7              int tmp = A[i];
8              A[i] = A[index];
9              A[index] = tmp;
10         }
11     }
```

В переменной index хранится индекс наименьшего элемента в подмассиве $A'[i..n-1]$.

Исходный код

Представим весь код Main.java для тестирования алгоритма.

Листинг 2: Main

```
1  public class Main {
2      public static void selectionSort(int[] A, boolean reverse) {
3          for (int i = 0; i < A.length - 1; ++i) {
```

```
4               int index = i;
5               for (int j = i; j < A.length; ++j)
6                   if (reverse ? A[index] < A[j] : A[index] > A[j])
                        index = j;
7
8               int tmp = A[i];
9               A[i] = A[index];
10              A[index] = tmp;
11          }
12      }
13
14      public static void print(int[] A) {
15          for (int i = 0; i < A.length; ++i)
16              System.out.print(A[i] + " ");
17
18          System.out.println();
19      }
20
21      public static void main(String[] args) {
22          int[] A = {8, 3, 12, 16, 10, 4, 1, 9, 7, 14};
23
24          print(A);
25          selectionSort(A, false);
26          print(A);
27      }
28  }
```

Сложность

Во-первых, заметим, что мы не используем дополнительную память, не считая некоторых переменных, поэтому сразу можно сказать, что пространственная сложность алгоритма сортировки выбором состовляет $O(1)$.

Рассмотрим сразу все случаи - лучший (на вход подаётся отсортированный массив), средний (на вход подаётся неотсортированный массив) и худший (на вход подаётся отсортированный в обратную сторону массив). При всех случаях мы линейно проходимся в цикле for по всем элементам массива A (кроме последнего), а в нём проходимся в цикле for по подмассиву $A' = [i..n-1]$. Поэтому временная сложность алгоритма сортировки выбором равна $\Omega(n^2)$, $\Theta(n^2)$, $O(n^2)$ в лучшем, среднем и худшем случаях соответственно.

Временная сложность			Пространственная сложность
Худший	Средний	Лучший	Худший
$O(n^2)$	$\Theta(n^2)$	$\Omega(n^2)$	$O(1)$

Таблица 1: **Резюме**

Комментарии

Алгоритм сортировки выбором проще даже, чем сортировка пузырьком, но он остаётся очень медленным даже в лучшем случае, хоть и не использует память. Использовать алгоритм стоит только в учебных целях.