ВикипедиЯ

Сортировка перемешиванием

Материал из Википедии — свободной энциклопедии

Сортировка перемешиванием, или Шейкерная сортировка, или двунаправленная (англ. *Cocktail sort*) — разновидность пузырьковой сортировки. Анализируя метод пузырьковой сортировки, можно отметить два обстоятельства.

Во-первых, если при движении по части массива перестановки не происходят, то эта часть массива уже отсортирована и, следовательно, её можно исключить из рассмотрения.

Во-вторых, при движении от конца массива к началу минимальный элемент «всплывает» на первую позицию, а максимальный элемент сдвигается только на одну позицию вправо.

Эти две идеи приводят к следующим модификациям в методе пузырьковой сортировки. Границы рабочей части массива (то есть части массива, где происходит движение) **устанавливаются в месте последнего обмена** на каждой итерации. Массив просматривается поочередно справа налево и слева направо.

Лучший случай для этой сортировки — отсортированный массив ($O(n)$), худший — отсортированный в обратном порядке ($O(n^2)$).

Наименьшее число сравнений в алгоритме Шейкер-сортировки $C = N - 1$. Это соответствует единственному проходу по упорядоченному массиву (лучший случай)

Содержание

Описание алгоритма

Образно алгоритм можно описать так: на каждом шаге основного цикла рассматривается массив a[Left]÷a[Right], после выполнения двух внутренних циклов минимальный и максимальный элемент в исходном массиве перетекают к краям, минимальный в — a[Left], максимальный — в a[Right]. Пусть максимальный элемент имеет индекс k, тогда массив можно изобразить так: a[Left], a[1],..,a[k-1],A[k], a[k+1],..,a[Right];После сравнения A[k] с a[k+1] значение A[k] перейдет в k+1-ую ячейку, после сравнения k+1-й с k+2-й — в k+2-ею, и так далее, пока он не сместится в крайне правое положение с индексом Right. Аналогично для минимального. После выполнения цикла по всем подмассивам он отсортируется. Трассировка программы:

3 1 5 8 1 0 4 6 6 7
3 1 5 8 0 1 4 6 6 7
3 1 5 0 8 1 4 6 6 7
3 1 0 5 8 1 4 6 6 7
3 0 1 5 8 1 4 6 6 7
0 3 1 5 8 1 4 6 6 7 Left=1
0 1 3 5 8 1 4 6 6 7
0 1 3 5 1 8 4 6 6 7
0 1 3 5 1 4 8 6 6 7
0 1 3 5 1 4 6 8 6 7
0 1 3 5 1 4 6 6 8 7
0 1 3 5 1 4 6 6 7 8 Right=10
0 1 3 1 5 4 6 6 7 8
0 1 1 3 5 4 6 6 7 8 Left=3
0 1 1 3 4 5 6 6 7 8

Примеры реализации

C++

```
#include <cstddef>
#include <utility>

template<typename T>
void shaker_sort(T array[], std::size_t size)
{
    for (std::size_t left_idx = 0, right_idx = size - 1;
         left_idx < right_idx;)
    {
        for (std::size_t idx = left_idx; idx < right_idx; idx++)
        {
            if (array[idx + 1] < array[idx])
            {
                std::swap(array[idx], array[idx + 1]);
            }
        }
        right_idx--;

        for (std::size_t idx = right_idx; idx > left_idx; idx--)
        {
            if (array[idx - 1] >  array[idx])
            {
                std::swap(array[idx - 1], array[idx]);
            }
        }
        left_idx++;
    }
}
```

D

```
import std.stdio;
import std.algorithm.mutation: swapAt;

void shaker_sort(T)(ref T[] arr)
{
```

```
    uint le = 0,
         ri = arr.length - 1;

    while (le <= ri)
    {
        foreach(k; le..ri)
            if (arr[k] > arr[k+1])
                arr.swapAt(k, k+1);
        ri--;

        foreach_reverse(k; le+1..ri+1)
            if (arr[k-1] > arr[k])
                arr.swapAt(k, k-1);
        le++;
    }
}
```

C#

```
using System;

namespace SortLab
{
    class Program
    {
        static void Main()
        {
            Sort();
        }

        /*Основная программа*/
        static void Sort()
        {
            int[] myint = { 99, 88, 77, 66, 55, 44, 33, 22, 11, 8, 5, 3, 1 };

            WriteArray(myint);
            ShakerSort(myint);
            WriteArray(myint);

            Console.ReadLine();
        }

        /* Шейкер-сортировка */
        static void ShakerSort(int[] myint)
        {
            int left = 0,
                right = myint.Length - 1,
                count = 0;

            while (left < right)
            {
                for (int i = left; i < right; i++)
                {
                    count++;
                    if (myint[i] > myint[i + 1])
                        Swap(myint, i, i + 1);
                }
                right--;

                for (int i = right; i > left; i--)
                {
                    count++;
                    if (myint[i - 1] > myint[i])
                        Swap(myint, i - 1, i);
                }
                left++;
            }
            Console.WriteLine("\nКоличество сравнений = {0}", count.ToString());
        }

        /* Поменять элементы местами */
        static void Swap(int[] myint, int i, int j)
        {
            int glass = myint[i];
            myint[i] = myint[j];
            myint[j] = glass;
```

```
        }

        /*Вывести массив*/
        static void WriteArray(int[] a)
        {
            foreach (int i in a)
                Console.Write("{0}|", i.ToString());
            Console.WriteLine("\n\n\n");
        }
    }
}
```

JavaScript

```
const swap = (arr, i, j) => {
    const akum = arr[i]
    arr[i] = arr[j]
    arr[j] = akum
}

function shakerSort(array) {
    let leftIndex = 0
    let rightIndex = array.length - 1

    while (leftIndex < rightIndex) {
        for (let idx = leftIndex; idx < rightIndex; idx++) {
            if ( array[idx] > array[idx + 1]) {
                swap(array, idx, idx + 1)
            }
        }
        rightIndex--;

        for (let idx = rightIndex; idx > leftIndex; idx--) {
            if ( array[idx] < array[idx - 1]) {
                swap(array, idx, idx - 1)
            }
        }
        leftIndex++;
    }

    return array
}
```

PHP

```
function cocktailSorting(&$a) {
    $n = count($a);
    $left = 0;
    $right = $n - 1;
    do {
        for ($i = $left; $i < $right; $i++) {
            if ($a[$i] > $a[$i + 1]) {
                list($a[$i], $a[$i + 1]) = array($a[$i + 1], $a[$i]);
            }
        }
        $right -= 1;
        for ($i = $right; $i > $left; $i--) {
            if ($a[$i] < $a[$i - 1]) {
                list($a[$i], $a[$i - 1]) = array($a[$i - 1], $a[$i]);
            }
        }
        $left += 1;
    } while ($left <= $right);
}
```

Java

```java
    public static void shakerSort(int array[]) {
        int buff;
        int left=0;
        int right=array.length-1;
        do {
            for (int i=left; i<right;i++) {
                if (array[i]>array[i+1]) {
                    buff = array[i];
                    array[i] = array[i + 1];
                    array[i + 1] = buff;
                }
            }
            right--;
            for (int i=right; i>left; i--) {
                if (array[i]<array[i-1]) {
                    buff = array[i];
                    array[i] = array[i - 1];
                    array[i - 1] = buff;
                }
            }
            left++;
        } while (left <right);
    }
```

Python

```python
sample = [0, -1, 5, -2, 3]

left = 0
right = len(sample) - 1

while left <= right:
    for i in range(left, right, +1):
        if sample[i] > sample[i + 1]:
            sample[i], sample[i + 1] = sample[i + 1], sample[i]
    right -= 1

    for i in range(right, left, -1):
        if sample[i - 1] > sample[i]:
            sample[i], sample[i - 1] = sample[i - 1], sample[i]
    left += 1

print(sample)
```

Fortran

```fortran
subroutine sort_cocktail(array_size,array)
    integer i,j
    integer last_unsorted, firs_unsorted, exchange
    logical way
    integer,intent(in)      :: array_size
    integer,intent(inout)   :: array(array_size)
    last_unsorted = array_size
    firs_unsorted = 1
    way = .true.
    do j=1,array_size
        if (way) then
            do i=firs_unsorted,last_unsorted-1
                if (array(i) .gt. array(i+1)) then
                    exchange   = array(i)
                    array(i)   = array(i+1)
                    array(i+1) = exchange
                end if
            end do
            last_unsorted = last_unsorted -1
        else
            do i=last_unsorted-1,firs_unsorted,-1
                if (array(i) .gt. array(i+1)) then
                    exchange   = array(i)
                    array(i)   = array(i+1)
                    array(i+1) = exchange
```

```
26                end if
27             end do
28             firs_unsorted = firs_unsorted +1
29         end if
30         way = .not. way
31         if(firs_unsorted .ge. last_unsorted) exit
32     end do
33 end subroutine
```

Ссылки

- Динамическая визуализация 7 алгоритмов сортировки с открытым исходным кодом (https://airtucha.github.io/SortVis/)

Источник — https://ru.wikipedia.org/w/index.php?title=Сортировка_перемешиванием&oldid=96822279