

Сортировка Шелла

Материал из Википедии — свободной энциклопедии

Сортировка Шелла (англ. *Shell sort*) — алгоритм сортировки, являющийся усовершенствованным вариантом сортировки вставками. Идея метода Шелла состоит в сравнении элементов, стоящих не только рядом, но и на определённом расстоянии друг от друга. Иными словами — это сортировка вставками с предварительными «грубыми» проходами. Аналогичный метод усовершенствования пузырьковой сортировки называется сортировка расчёской.

Содержание

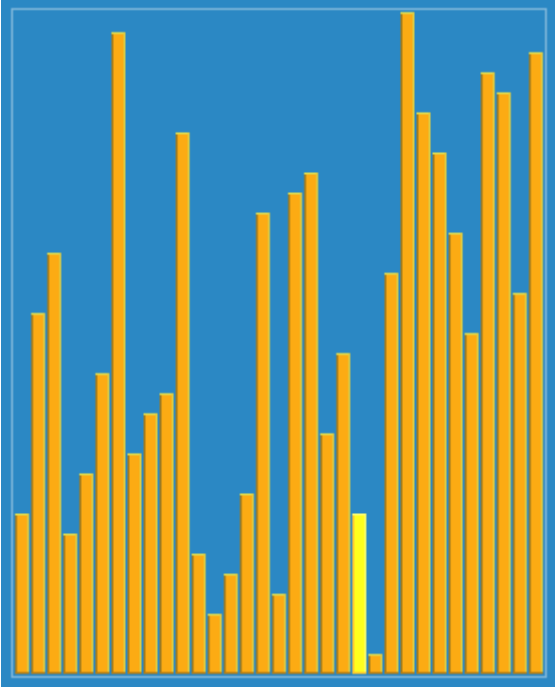
- Описание
- История
- Пример
- Выбор длины промежутков
- Реализация на C++
- Реализация на C
- Реализация на Java
- Реализация на Python
- Примечания
- Ссылки

Описание

При сортировке Шелла сначала сравниваются и сортируются между собой значения, стоящие один от другого на некотором расстоянии ***d*** (о выборе значения ***d*** см. ниже). После этого процедура повторяется для некоторых меньших значений ***d***, а завершается сортировка Шелла упорядочиванием элементов при ***d*** = 1 (то есть обычной сортировкой вставками). Эффективность сортировки Шелла в определённых случаях обеспечивается тем, что элементы «быстрее» встают на свои места (в простых методах сортировки, например, пузырьковой, каждая перестановка двух элементов уменьшает количество инверсий в списке максимум на 1, а при сортировке Шелла это число может быть больше).

Невзирая на то, что сортировка Шелла во многих случаях медленнее, чем быстрая сортировка, она имеет ряд преимуществ:

Сортировка Шелла



Сортировка с шагами 23, 10, 4, 1.

Предназначение	Алгоритм сортировки
Структура данных	Массив
Худшее время	$O(n^2)$
Лучшее время	$O(n \log_2 n)$
Среднее время	зависит от выбранных шагов
Затраты памяти	$O(n)$ всего, $O(1)$ дополнительно



Сортировка Шелла на примере

- отсутствие потребности в памяти под стек;
- отсутствие деградации при неудачных наборах данных — быстрая сортировка легко деградирует до $O(n^2)$, что хуже, чем худшее гарантированное время для сортировки Шелла.

История

Сортировка Шелла была названа в честь её изобретателя — [Дональда Шелла](#), который опубликовал этот алгоритм в [1959](#) году.

Пример

Пусть	дан	список	Исходный массив	32 95 16 82 24 66 35 19 75 54 40 43 93 68	
			После сортировки с шагом 5	32 35 16 68 24 40 43 19 75 54 66 95 93 82	6 обменов
			После сортировки с шагом 3	32 19 16 43 24 40 54 35 75 68 66 95 93 82	5 обменов
			После сортировки с шагом 1	16 19 24 32 35 40 43 54 66 68 75 82 93 95	15 обменов

$A = (32, 95, 16, 82, 24, 66, 35, 19, 75, 54, 40, 43, 93, 68)$ и выполняется его сортировка методом Шелла, а в качестве значений d выбраны **5, 3, 1**.

На первом шаге сортируются подписки A , составленные из всех элементов A , различающихся на 5 позиций, то есть подписки $A_{5,1} = (32, 66, 40)$, $A_{5,2} = (95, 35, 43)$, $A_{5,3} = (16, 19, 93)$, $A_{5,4} = (82, 75, 68)$, $A_{5,5} = (24, 54)$.

В полученном списке на втором шаге вновь сортируются подписки из отстоящих на 3 позиции элементов.

Процесс завершается обычной сортировкой вставками получившегося списка.

Выбор длины промежутков

Среднее время работы алгоритма зависит от длин промежутков — d , на которых будут находиться сортируемые элементы исходного массива ёмкостью N на каждом шаге алгоритма. Существует несколько подходов к выбору этих значений:

- первоначально используемая Шеллом последовательность длин промежутков: $d_1 = N/2, d_i = d_{i-1}/2, d_k = 1$ в худшем случае, сложность алгоритма составит $O(N^2)$;
- предложенная Хиббардом последовательность: все значения $2^i - 1 \leq N, i \in \mathbb{N}$; такая последовательность шагов приводит к алгоритму сложностью $O(N^{3/2})$;
- предложенная Седжвиком последовательность: $d_i = 9 \cdot 2^i - 9 \cdot 2^{i/2} + 1$, если i четное и $d_i = 8 \cdot 2^i - 6 \cdot 2^{(i+1)/2} + 1$, если i нечетное. При использовании таких приращений средняя сложность алгоритма составляет: $O(n^{7/6})$, а в худшем случае порядка $O(n^{4/3})$. При использовании формулы Седжвика следует остановиться на значении $\text{inc}[s-1]$, если $3 \cdot \text{inc}[s] > \text{size}$.^[1];
- предложенная Праттом последовательность: все значения $2^i \cdot 3^j \leq N/2, i, j \in \mathbb{N}$; в таком случае сложность алгоритма составляет $O(N(\log N)^2)$;
- эмпирическая последовательность Марцина Циура (последовательность [A102549](#) в [OEIS](#)): $d \in \{1, 4, 10, 23, 57, 132, 301, 701, 1750\}$; является одной из лучших для сортировки массива ёмкостью приблизительно до 4000 элементов.^[2];
- эмпирическая последовательность, основанная на [числах Фибоначчи](#): $d \in \{F_n\}$;

- все значения $(3^j - 1) \leq N$, $j \in \mathbb{N}$; такая последовательность шагов приводит к алгоритму сложностью $O(N^{3/2})$.

Реализация на C++

```
template< typename RandomAccessIterator, typename Compare >
void shell_sort( RandomAccessIterator first, RandomAccessIterator last, Compare comp )
{
    for( typename std::iterator_traits< RandomAccessIterator >::difference_type d = ( last - first ) / 2; d != 0; d /= 2 )
        for( RandomAccessIterator i = first + d; i != last; ++i )
            for( RandomAccessIterator j = i; j - first >= d && comp( *j, *( j - d ) ); j -= d )
                std::swap( *j, *( j - d ) );
}
```

Реализация на C

```
// BaseType - любой перечисляемый тип
// typedef int BaseType - пример
void ShellsSort(BaseType *A, unsigned N)
{
    unsigned i, j, k;
    BaseType t;
    for(k = N / 2; k > 0; k /= 2)
        for(i = k; i < N; i++)
        {
            t = A[i];
            for(j = i; j >= k; j -= k)
            {
                if(t < A[j - k])
                    A[j] = A[j - k];
                else
                    break;
            }
            A[j] = t;
        }
}
```

Реализация на Java

```
public class ShellSort {
    public void sort (int[] arr) {
        int increment = arr.length / 2;
        while (increment >= 1) {
            for (int startIndex = 0; startIndex < increment; startIndex++) {
                insertionSort(arr, startIndex, increment);
            }
            increment = increment / 2;
        }
    }

    private void insertionSort (int[] arr, int startIndex, int increment) {
        for (int i = startIndex; i < arr.length - 1; i = i + increment) {
            for (int j = Math.min(i + increment, arr.length - 1); j - increment >= 0; j = j - increment) {
                if (arr[j - increment] > arr[j]) {
                    int tmp = arr[j];
                    arr[j] = arr[j - increment];
                    arr[j - increment] = tmp;
                } else {
                    break;
                }
            }
        }
    }
}
```

Реализация на Python

```
def shellSort(array):
    increment = len(array) // 2
    while increment > 0:

        for startPosition in range(increment):
            gapInsertionSort(array, startPosition, increment)

        print("После инкрементации размера на", increment, "массив:", array)

        increment //= 2

def gapInsertionSort(array, low, gap):
    for i in range(low + gap, len(array), gap):
        currentvalue = array[i]
        position = i

        while position >= gap and array[position - gap] > currentvalue:
            array[position] = array[position - gap]
            position = position - gap

        array[position] = currentvalue
```

Примечания

1. *J. Incerpi, R. Sedgewick*, «Improved Upper Bounds for Shellsort», J. Computer and System Sciences 31, 2, 1985.
2. Marcin Ciura *Best Increments for the Average Case of Shellsort* (<http://sun.aei.polsl.pl/~mciura/publikacje/shell-sort.pdf>)

Ссылки

- *Д. Кнут*. Искусство программирования. Том 3. Сортировка и поиск, 2-е изд. Гл. 5.2.1. ISBN 5-8459-0082-4
- Анимированное представление алгоритма сортировки Шелла (<http://www.sorting-algorithms.com/shell-sort>)
- Представление алгоритма сортировки Шелла в виде танца (видео) (<https://www.youtube.com/watch?v=CmPA7zE8mx0>)

Источник — https://ru.wikipedia.org/w/index.php?title=Сортировка_Шелла&oldid=98539970

Эта страница в последний раз была отредактирована 9 марта 2019 в 13:31.

Текст доступен по лицензии [Creative Commons Attribution-ShareAlike](#); в отдельных случаях могут действовать дополнительные условия.

Wikipedia® — зарегистрированный товарный знак некоммерческой организации [Wikimedia Foundation, Inc.](#)