Shell Sort in data structure and algorithm

Shell Sort is a highly efficient sorting algorithm based on insertion sorting algorithm (Insertion Sort). This algorithm avoids the case of swapping positions of two distant elements in the selection algorithm (if the smaller element is in the right position quite far from the larger element on the left).

What is Shell Sort?

Shell Sort is a highly efficient sorting algorithm based on **insertion** sorting algorithm **(Insertion Sort)** . This algorithm avoids the case of swapping positions of two distant elements in the selection algorithm (if the smaller element is in the right position quite far from the larger element on the left).

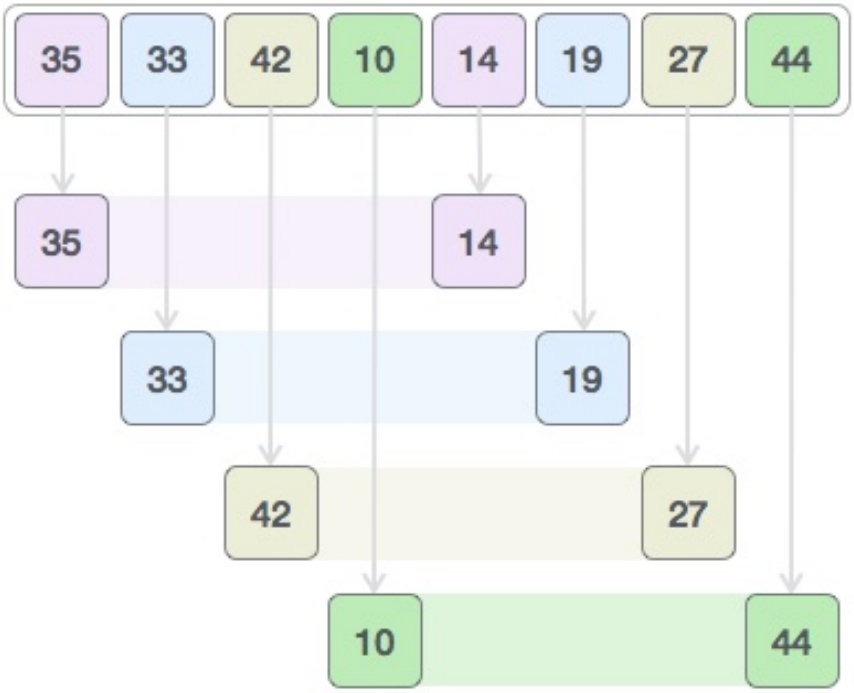
First, this algorithm uses a sorting algorithm on elements that are far apart, then arranging elements with a narrower distance. This distance is also known as **interval** - the number of positions from one element to another. This range is calculated based on the following Knuth formula:

h = h * 3 + 1 *trong ?ó:* h l à Kho ? ng (interval) v ? i gi á tr ? ban ?â u l à 1

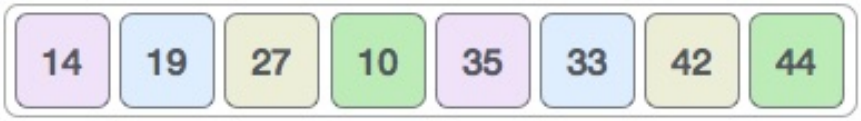
This algorithm is quite effective for medium sized data sets when the worst case complexity and the average case are O (n), where n is the number of elements.

How Shell Sort works

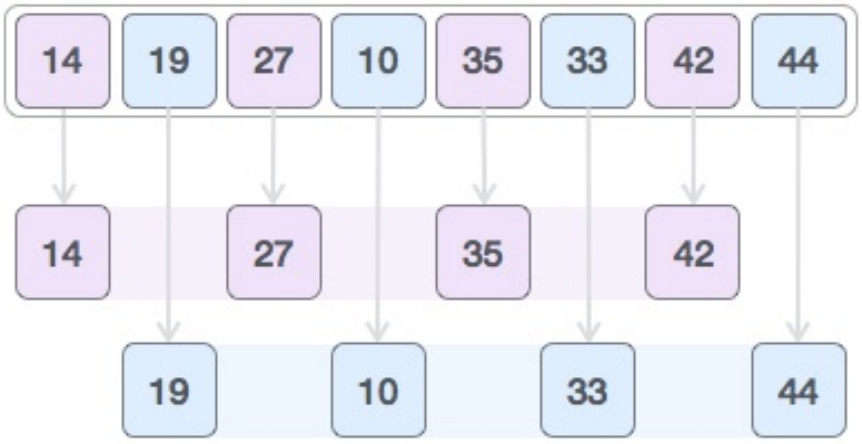
To make it easier to learn, below I provide illustrations for how Shell Sort works. We use an array of values ??as shown below. Suppose the initial value of interval is 4. For example, for element 35, with a range of 4, the remaining element will be 14. Therefore we will have pairs of values ??{35, 14}, { 33, 19}, {42, 27}, and {10, 14}.



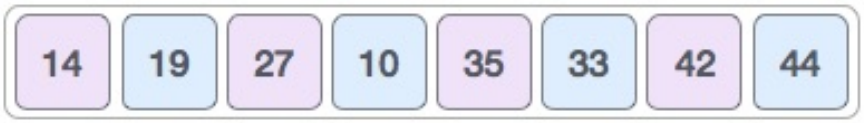
Compare these values ??together in sub-lists and swap them (if needed) in the original array. After this step, the new array will be empty as follows:



Then, take the value of 2 interval and with this distance will give two sub-lists: {14, 27, 35, 42}, {19, 10, 33, 44}.



Continue to compare and swap values ??(if needed) in the original array. After this step, the array will look like this:



Finally, we arrange this remaining array with interval equal to 1. Shell Sort uses an insertion sort algorithm to sort the array. Below is an illustration of each step.

As shown above, you see that we only need 4 swaps to sort out this remaining array.

Algorithm for Shell Sort

Now we will monitor the algorithm for Shell Sort:

```
 B??c 1 : Kh?i t?o giá tr? h B??c 2 : Chia list thành các sublist nh? h?n t
??ng ?ng v?i h B??c 3 : S?p x?p các sublist này b?i s? d?ng s?p x?
p chèn (Insertion Sort) B??c 4 : L?p l?i cho t?i khi list ?ã ???c s?p x?p
```

Sample algorithm for Shell Sort

From the above steps we can design a sample algorithm for Shell Sort as follows:

```
 B ? t  ?? u h à m shellSort ()  A  :  m ? ng c á c ph ? n t ?
 /* Tính toán giá tr?  Kho?ng (interval)
 */ while  interval    A . length  / 3  th ? c hi ?
 n :  interval  =  interval  * 3 + 1  k?
 t th ú c  while while  interval  > 0  th ? c hi ?
 n : for  outer  =  interval ;  outer    A . length ;  outer  ++  th ?
 c hi ? n : /* ch?n giá tr? ??
 chèn */  valueToInsert  =  A [ outer ]  inner  =  outer ; /*d?ch chuy?n ph
?n t? sang ph?
i*/ while  inner  >  interval  - 1 &&  A [ inner  -  interval ] >=  valueToInser
? t th ú c  while /* chèn giá tr? vào v?
 trí trên */  A [ inner ] =  valueToInsert k ?
 t th ú c  for /* Tính toán giá tr?  Kho?ng (interval)
 */  interval  = ( interval  - 1 ) / 3 ;  k? t th ú c  while  K ?
 t th ú c h à m
```

According to Tutorialspoint

Previous lesson: Mixing algorithm (Merge Sort)

Next lesson: Quick Sort (Quick Sort)