УДК 004.021:811.111

Belyi E., Lapko O.

Popular Sorting Algorithms

Belarusian National Technical University
Minsk, Belarus

What is a sorting algorithm and why do we need it? In computer science, it is very common to sort the input data. Therefore, it is very important to find the most optimal way of sorting, which is the main concern. There are a lot of sorting algorithms known to men now, but the article deals with the most famous and popular ones.

Quick sort. Qsort was invented in 1960 by Tony Hoare. The algorithm was developed according to the Divide and Conquer method according to the principle of direct movement. It works thanks to recursion. The function receives an array. Then the function selects some pivot element, and then splits the array into two arrays. The first array contains elements which are less than the pivot one, and the right one contains larger elements. Then the function calls itself and gives these two arrays as input. Best case is O (n), average O (n logn) worst O (n ^ 2). The disadvantage is that the algorithm does not have stable variants, an outdated method has been used, to which a better alternative has long been found. There advantages of this algorithm worth mentioning. First of all it's easy to write, it uses a small amount of memory. Moreover, it works faster than its counterparts.

Tree Sort. This sort of sort uses the principles of a binary tree. A binary tree is a kind of dynamic structure consisting of roots, nodes and leaves. The topmost element is the middle element in such a structure. Two nodes extend from it, and more nodes extend from the nodes, and so on to the extreme

lower elements - leaves. Typically, the left branch is smaller than the root, and the right branch is larger. The principle of this sorting is that we are trying to build a binary tree from the array given to us. And then we perform a depth-first traversal with the formation of a new, already sorted array. Best case is O (n logn), average O (n logn) worst O (n ^ 2). The biggest disadvantage of this sort is the tree's imbalance. When the root element is incorrectly defined, some of the branches of the tree grows longer, which leads to degradation of the sorting and reduces the time to O (n ^ 2) time complexity. Therefore, numerous modifications have been written for this sort to allow rebalancing of the tree. Thanks to this modification, the algorithm becomes one of the few stable sorts, and the time complexity is reduced to: the best case is O (n logn), the average is O (n logn), the worst is O (n ^ 2) Its disadvantage is its complicated implementation. The advantage of this algorithm is its stability. It also should be mentioned that it is one of the fastest algorithms.

Radix sort. Bitwise sorting is applied in as many steps as there are bits in the largest element. There are many types of this algorithm, but the most functional of them is LSD (least significant digits). It works exactly as its name suggests. First, it sorts the numbers by units, then sorts by tens, keeping the results of the previous sorting and so on until the last digit. There is also a type MSD (most significant digits), working towards the lower digits. A stable sort that has a time complexity of O (w * n), where w is the number of bits that need to be spent to store the maximum number. Its disadvantage is high time complexity. The advantages are the ease of writing and stability.

Bucket Sort. Pocket (Block) sort uses the same method of operation as qsort. For a successful sort, we must make sure that the array data obeys a uniform distribution law. If so, then the algorithm has one of the fastest time complexity. Like

many algorithms, this sorting works in two stages. The first stage - the algorithm creates an array of arrays into which it writes ranges of numbers (baskets). It then walks through the array and writes the values to the bucket data. Since in the preliminary stage we confirmed that the array obeys uniform distribution law, then we expect that there will be few values in each basket. After that the second and final stage comes - sorting takes place inside all the baskets and the generation of the final array. The disadvantages are the complexity of writing this algorithm and the instability of its work. The advantage is the linear time complexity of the O (n) algorithm.

And finally, let's move on to the most advanced sorting at the time of this article - TimSort. It was invented in 2002 by Tim Peterson. It is a hybrid algorithm consisting of insertion sort cooperation and modernized merge sort. This sort is one of the few quicksort algorithms with stable results. One of these sorts is the binary tree sort, which has already been described above. The main idea of this algorithm is that the array given to us may already have some sorted parts. TimSort finds or recreates these parts at the stage of reading the array. The algorithm itself works in three stages. The first stage - after entering the data, we perform a partial sorting and split the array into several practically sorted arrays. The second step is to perform a secondary insertion sort, which works fine for this data type. And the final step is to generate a new array of their sorted arrays using merge sort. This is how the currently best sorting algorithm works, which has already been recognized and used in the standard Java, Python and Kotlin libraries. Since this is a fairly new algorithm, serious flaws have not been identified yet. The advantages of this algorithm include relative ease of writing, stable operation of the algorithm with time complexity O (n logn) [1].

Count sort. Count sort is a linear sort that uses memory. It works only with whole data sets of small diversity. The

principle of the algorithm is that it creates an array from the minimum element to the maximum. If the index of an array element coincides with the value of the array, then the value of the array element with this index is increased by one. Therefore, this algorithm has a linear tricky execution. In the worst cases, the program just doesn't work.

Shell Sort. It was invented in 1959 by Donald Schell. The algorithm works on the principle of enumeration by the direct displacement method. It works cyclically. In each cycle, a certain number is selected, in what follows we will call it a step. Next, we go through the array and change the elements to the left and right of the pointer on the left and right sides. Thus, the algorithm can swap several elements. Its disadvantages are old method of work and low productivity. But at the same time it is more stable than qsort and easier to write.

Bubble sort. Probably all of you have heard of bubble sort, the most primitive sorting of the existing ones. We go through the array as many times as there are elements in it, decreasing it each time. Thus, its time complexity is O (n^2). Its disadvantages are low performance, outdated method, primitiveness. Its advantage is that it is easy to write.

Thus, in this article, the key sorting algorithms, their advantages and disadvantages, the principle of their operation, the methods and ideas laid down by the developers, the time complexity were analyzed.

References:

1. Timsort [Electronic resource]. – Mode of access: https://bugs.python.org/file4451/timsort.txt. – Date of access: 16.04.2021.