17. Анализ быстрой сортировки

Описание алгоритма

Быстрая сортировка работает следующим образом:

1. Выбирается опорный элемент (pivot) из массива.
2. Массив делится на две части:
   - В первой части находятся элементы, которые меньше или равны опорному элементу.
   - Во второй части — элементы, которые больше опорного элемента.
3. После деления каждая из частей сортируется рекурсивно.

Анализ сложности

1. Время работы в лучшем случае

В лучшем случае опорный элемент всегда делит массив на две равные части. Тогда на каждом уровне рекурсии количество элементов, подлежащих сортировке, будет уменьшаться вдвое. Время работы будет оцениваться как:

$$T(n) = 2T(n/2) + O(n)$$

где $T(n/2)$ — это время сортировки двух половин массива, а $O(n)$ — время на деление массива и размещение элементов относительно опорного. Решение этого рекуррентного соотношения даёт:

$$T(n) = O(n \log n)$$

Таким образом, в лучшем случае алгоритм работает за время $O(n \log n)$.

2. Время работы в худшем случае

В худшем случае опорный элемент оказывается либо на минимальной, либо на максимальной позиции, деля массив на одну большую часть и одну маленькую. Это приводит к тому, что на каждом уровне рекурсии остаётся лишь один элемент, а сортировка происходит за $O(n)$ времени на каждом уровне. Таким образом, время работы алгоритма в худшем случае оценивается как:

$$T(n) = T(n-1) + O(n)$$

Решение этого рекуррентного соотношения даёт:

$$T(n) = O(n^2)$$

Таким образом, в худшем случае алгоритм работает за время $O(n^2)$.

3. Время работы в среднем случае

Средний случай предполагает, что опорный элемент делит массив на примерно равные части. Время работы алгоритма в среднем случае оценивается как:

$$T(n) = 2T(n/2) + O(n)$$

Поскольку в среднем случае алгоритм работает так же, как в лучшем, его время работы также будет $O(n \log n)$.