Анализ быстрой сортировки (сортировки Хоара)

Проведем анализ временной сложности быстрой сортировки (сортировки Хоара).

Основная процедура сортировки имеет вид:

```
Procedure QuickSort(l, r : Integer);
Var x, i, j : Integer;
Begin
  If l<r Then Begin
    x:=A[(l+r) Div 2]; {выбор значения барьерного элемента}
    i:=l; r:=j;
    Repeat
     {перестановка элементов: слева от барьерного –
       элементы, меньшие его, а справа – большие}
      While (A[i]<x) Do Inc(i);
      While (A[j]>x) Do Dec(j);
      If i<=j Do Begin
        Swap(A[i], A[j]);
        Inc(i); Dec(j)
      End
    Until i>j;
```

Действия на входе в рекурсию

```
    QuickSort(l, j); {рекурсивный вызов для левой части}
    QuickSort(i, r); {рекурсивный вызов для правой части}
```

Два рекурсивных вызова

```
  End
End;
```

Время перестановки элементов в рассматриваемой части пропорционально длине этой части, т. е. в общем виде для части из n элементов $D(n)=\Theta(n)$.

Действий на выходе из рекурсии алгоритмом не предусмотрено, т. е. время соединения полученных решений $S(n)= \Theta(1)$.

В процедуре присутствует два рекурсивных вызова, для выполнения аналогичных действий для двух полученных в результате перестановки элементов частей массива (не обязательно равной длины), т. е. $T(n_1)$ и $T(n_2)$. Заметим, что $n_2 \approx n-n_1$.

Если в массиве один элемент, то происходит выход из процедуры сразу после проверки условия ($l<r$), т. е. $T(1)= \Theta(1)$.

Таким образом, получили рекуррентное соотношение вида:

$$T(n)=\begin{cases} T(n_1)+T(n_2)+\Theta(n)+\Theta(1), & n>1 \\ \Theta(1), & n=1 \end{cases}.$$

В общем случае $n_2=n-n_1$.

Упростим данное рекуррентное соотношение с учетом свойств асимптотических обозначений:

$$T(n) = \begin{cases} T(n_1) + T(n - n_1) + \Theta(n), & n > 1 \\ \Theta(1), & n = 1 \end{cases}.$$

Решить данное рекуррентное соотношение рассмотренными выше методами не удается, так как в соотношении присутствует формально введенный параметр n_1. Для того чтобы «избавиться» от него, рассмотрим работу алгоритма в лучшем и в худшем случаях.

Лучший случай

Характеризуется делением рассматриваемой части массива при каждом рекурсивном вызове пополам. Таким образом, $n_1=n/2$, $n\text{-}n_1=n/2$, и рекуррентное соотношение с учетом свойств асимптотических обозначений принимает вид:

$$T_{лучший\ случай}(n) = \begin{cases} 2 \cdot T\left(\dfrac{n}{2}\right) + \Theta(n), & n > 1 \\ \Theta(1), & n = 1 \end{cases}.$$

Таким образом, получено рекуррентное соотношение, аналогичное рекуррентному соотношению сортировки слиянием, т. е. $T_{лучший\ случай}(n)=O(n\cdot\log_2 n)$.

Худший случай

Худший случай для времени работы алгоритма характеризуется выбором в качестве барьерного элемента при каждом рекурсивном вызове максимального (или минимального) элемента рассматриваемой части. Тогда сортируемое множество элементов каждый раз разбивается на две части таким образом, что в одной части оказывается один элемент, а в другой – все остальные. Таким образом, $n_1=1$, $n\text{-}n_1=n\text{-}1$, и рекуррентное соотношение с учетом свойств асимптотических обозначений принимает вид:

$$T_{худший\ случай}(n) = \begin{cases} T(n-1) + \Theta(n), & n > 1 \\ \Theta(1), & n = 1 \end{cases}.$$

Найдем методом итераций значение верхней оценки, предварительно раскрыв асимптотические обозначения и взяв лишь их правые части.

$$T_{худший\ случай}(n) \le T(n-1) + c_1 \cdot n \le T(n-2) + c_1 \cdot (n-1) + c_1 \cdot n \le$$

$$\le T(n-3) + c_1 \cdot (n-2) + c_1 \cdot (n-1) + c_1 \cdot n \le \ldots$$

$$\ldots \le T(1) + c_1 \cdot 2 + \ldots + c_1 \cdot (n-2) + c_1 \cdot (n-1) + c_1 \cdot n \le$$

$$\le c_2 + c_1 \cdot 2 + \ldots + c_1 \cdot (n-2) + c_1 \cdot (n-1) + c_1 \cdot n =$$

$$= c_2 + c_1 \cdot (2 + \ldots + (n-2) + (n-1) + n) = c_2 + c_1 \cdot \frac{(2+n)\cdot(n-1)}{2} =$$

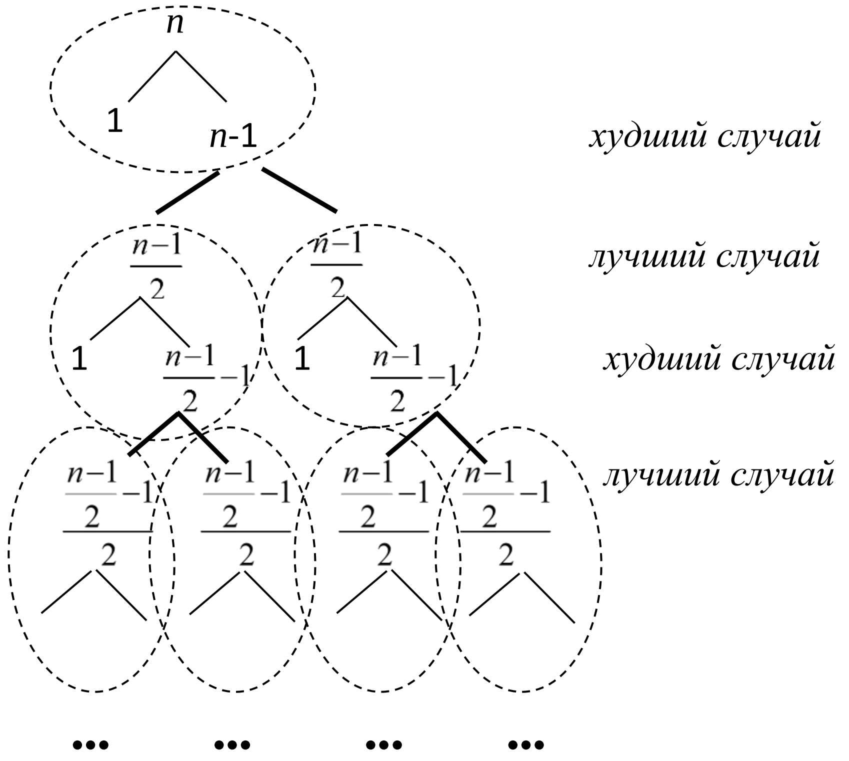
$$= c_2 + c_1 \frac{n^2 + n - 2}{2} \quad - \quad \textit{квадратичная сложность}$$

Таким образом, $T_{худший\ случай}(n)=O(n^2)$.

Возникает вопрос: правомерно ли называть сортировку Хоара быстрой сортировкой? Для ответа на этот вопрос рассмотрим работу алгоритма в среднем случае. Если оценка среднего случая окажется ближе к оценке лучшего случая, то сортировку можно будет считать быстрой.

Средний случай

Рассмотрим гипотетический средний случай, когда происходит чередование худшего и лучшего случаев. Дерево делений массива будет иметь следующий вид:



Данное дерево является аналогом двоичного дерева с составными вершинами, каждая из которых имеет сложность $\Theta(n)$, где n – длина рассматриваемой части массива. Как видно из рисунка, каждый раз массив делится на две равные части, т. е. пополам, таким образом, рекуррентное соотношение времени работы алгоритма в среднем случае имеет вид:

$$T_{средний\ случай}(n) = \begin{cases} 2 \cdot T\left(\dfrac{n}{2}\right) + \Theta(n) + \Theta(n), & n > 1 \\ \Theta(1), & n = 1 \end{cases}$$

Найдем методом итераций значение верхней оценки, предварительно раскрыв асимптотические обозначения и взяв лишь их правые части.

$$T_{средний\ случай}(n) \le 2 \cdot T\left(\frac{n}{2}\right) + c_1 \cdot n + c_3 \cdot n = 2 \cdot T\left(\frac{n}{2}\right) + (c_1 + c_3) \cdot n \le$$

$$\le 2 \cdot \left(2 \cdot T\left(\frac{n}{4}\right) + c_1 \cdot \frac{n}{2} + c_3 \cdot \frac{n}{2}\right) + (c_1 + c_3) \cdot n = 4 \cdot T\left(\frac{n}{4}\right) + 2 \cdot (c_1 + c_3) \cdot n \le \ldots$$

$$\ldots \le 2^i \cdot T\left(\frac{n}{2^i}\right) + i \cdot (c_1 + c_3) \cdot n \le$$

$$\left\langle \text{предположим, что } \frac{n}{2^i} = 1, \text{ т.е. } i = \log_2 n \right\rangle$$

$$\leq 2^{\log_2 n} \cdot c_2 + (c_1 + c_3) \cdot n \cdot \log_2 n =$$

$$= (c_1 + c_3) \cdot n \cdot \log_2 n + c_2 \cdot n$$

Таким образом, $T_{средний\ случай}(n)$=O($n \cdot \log_2 n$), т. е. средний и лучший случаи асимптотически эквивалентны, и сортировку Хоара можно считать быстрой.