Сортировка камней

Kailiak Eugene

10/09/2016

1. Опишем оптимальный алгоритм
   У нас есть массив из n камней. Тогда разобьём его на части, в которых по k камней (в последнем куске будет немного меньше камней, но это не влияет на решение задачи). В каждой из этих частей камни могут находиться на любом месте, потому что находятся друг от друга не дальше k. Тогда отсортируем каждую такую часть обычной сортировкой слиянием за $O(k \log k)$. Так как всего таких отрезков $\frac{n}{k}$, то мы сделаем всё это за $O(n \log k)$
   Дальше будем делать такие операции:
   Сначала сольём две первых части так же, как мы это делали в сортировке слиянием, потому что они отсортированы. А далее каждую итерацию:

   (a) Возьмём первую часть (первые $k * m$ камней, где $m \in \mathbb{N}$, которые представляют из себя отсортированную последовательность камней), получившуюся после слияния.

   (b) Выделим из неё последние k камней

   (c) Сольём эти k камней со следующей частью: отсортированной хпоследовательностью k камней, идущей после первой части.

   Докажем, почему это работает. Допустим, что из первой части надо было взять не k камней, а больше. Пусть m-ый камень - тот, который находится в первой части под номером m, если считать справа налево. Тогда если камень с номером $m > k$ попал не на своё место после слияния, то это значит, что до этого хотя бы один камень из второй части перебросили в первую часть. Значит, его место в отсортированном массиве находится дальше, чем на k от изначального, потому что он должен был переместиться как минимум через $m - 1$ камень, а $m > k$. Значит, сортировка выполнится верно.

2. Докажем ассимптотику алгоритма
   Как мы уже выяснили, ассимпотика первой части алгоритма происходит за $O(n \log k)$ Далее произошло ещё $\frac{n}{k} - 1$ слияние, в каждом из которых сливались два массива из k элементов (или меньше, если это была последняя часть массива). Тогда получившаяся ассимптотика алгоритма:
   $$O(n \log k + \frac{n}{k} \cdot 2k) = O(n \log k + 2n) = O(n \log k)$$

3. Докажем, что это самый быстрый алгоритм
   Разобьём массив на m частей, идущих подряд друг за другом и состоящих из k камней, где $m = n/k$, а деление целочисленное. Пусть $l = n\%k$. Тогда в конце останется l камней, не попавших ни в одну часть. В каждой части камень может находиться на любом месте этого отрезка по условию. Поэтому если мы рассмотрим перестановки камней в этих частях, то все эти варианты будут листьями искомого дерева решений. Какие-то листья из дерева решений

мы не взяли, но мы делаем "оценку снизу". Мы взяли $g = k!^m$ листьев, каждый из которых является листом в реальном дереве решений. Тогда

$$h_{opt} \leq \log_2 g = \log_2 k!^m = m \log_2 k! = \Omega(m \cdot k \log k)$$

Заметим так же, что $m \cdot k = n - l$, а раз $k \leq n$, то $l \leq \frac{n}{2}$ Тогда

$$m \cdot k \log_2 k = (n - l) \log_2 k \geq \frac{n}{2} \log_2 k \Rightarrow \Omega(m \cdot k \log k) = \Omega(n \log k)$$

Значит, лучше, чем $n \log k$ сделать нельзя. Что и требовалось доказать.