Теоретическое решение задачи В

Алгоритм решения и доказательство его правильности

Докажем, что для того чтобы ролик был максимально эффективным и правдоподобным, необходимо и достаточно того, чтобы в ролике принимали участие работники, чьи коэффициенты востребованности складывают не строгую наибольшую убывающую подпоследовательность (далее НУП) ($K = \{k_1, k_2, \ldots, k_n\}$), где главная последовательность это множество ($Q=\{q_1, q_2, \ldots, q_m\}$) коэффициентов востребованности, отсортированных по убыванию времени работы в компании своих обладателей.

Докажем условие достаточности. Так как в НУП порядок не меняется, элементы входят в множество K в той же последовательности, что и в множество Q. Из условия задачи следует, что $t(q_1) \geq t(q_2) \geq \ldots \geq t(q_m)$, где t – время работы в компании, следовательно, $t(k_1) \geq t(k_2) \geq \ldots \geq t(k_n)$. Исходя из наших утверждений, K – убывающая последовательность, то есть $k_1 \geq k_2 \geq \ldots \geq k_n$. Так как для решения задачи должно выполняться условие для всех пар работников, снимающихся в ролике: $t(qi) \geq t(qj)$ *AND* $qi \geq qj$, а в нашем множестве K выполняется условие $t(k_1) \geq t(k_2) \geq \ldots \geq t(k_n)$ *AND* $k_1 \geq k_2 \geq \ldots \geq k_n$. Условие задачи выполняется при любых двух k_i и k_j. Исходя из этого мы можем говорить, что для решения задачи достаточно нахождение НУП, ч. т. д..

Докажем условие необходимости. Предположим, что в ролик могут входить не только работники множества K, но и работники множества Q не входящие в K. Возьмем такого работника с коэффициентом q_i и рассмотрим для него 3 случая:

1. $k_j = q_{i-s}$, $k_{j+1} = q_{i+h}$, $s > 0$, $h > 0$, $s \leq i$, $h \leq m-i$. То есть в общем множестве Q, отсортированном по времени работы в компании, q_i будет находиться левее k_{j+1} и правее k_j. Из сортировки множества Q выплывает то, что $t(k_j) \geq t(q_i) \geq t(k_{j+1})$, где t – время работы в компании. Из условия задачи мы знаем что если $t(q_i) \geq t(q_j)$, то и $q_i \geq q_j$, следовательно мы можем утверждать, что $k_j \geq q_i \geq k_{j+1}$, а то есть $\{k_1, k_2, \ldots, k_j, q_i, k_{j+1}, \ldots, k_n\}$ является не строгой убывающей подпоследовательностью, и множество K не являлось НУП, что нарушает логику наших суждений, и значит этот случай является тождественно неверным.
2. $k_j = q_{i+h}$, $h > 0$, $h \leq m-i$. То есть в множестве Q q_i будет находиться левее k_j. Из доказанного выше следует, что $t(q_i) \geq t(k_j)$ и $q_i \geq k_j$, а следовательно, $\{q_i, k_1, k_2, \ldots, k_n\}$ является не строгой убывающей подпоследовательностью и множество K не являлось НУП, тождество неверно
3. $k_j = q_{i-s}$, $s > 0$, $s \leq i$. Аналогично второму, тождество тоже неверно.

Исходя из этого мы можем говорить, что нельзя найти такого работника q_i, который должен сниматься в ролике, но не входит в множество K, то есть можно утверждать что нахождение НУП необходимо для решения данной задачи, ч. т. д..

Теперь опишем непосредственно алгоритм решения задачи. Перебираем всех работников в порядке возрастания времени работы в компании, для каждого работника в цикле с n итераций применим вставку его коэффициента востребованности k в заранее созданный массив *mas* длиной $n+1$, с изначальными значениям его элементов $-\infty$ и нулевым элементом $+\infty$ и тогда первый вставленный коэффициент будет на позиции 1 (отсчет с нуля) . Вставляем число k в массив *mas* как можно правее, заменяя значение первого числа, меньшего k, методом двоичного поиска. Так же записываем в массив pos на каждой итерации цикла позицию работника в исходном массиве на то же место, куда была произведена вставка в массив *mas*. В массив *prev* на каждой итерации записываем на место индекса работника индекс работника за которым он следует в нашей НУП (у первого работника остается индекс-ссылка -1).

Восстановление ответа. Восстанавливаем ответ, проходясь по массиву *prev,* начиная с элемента, указанного в массиве *pos,* под индексом, соответствующим длине конечной НУП (*len*) и заканчивая элементом со значением -1. Все значения записываем в массив *answer*. В конечном итоге переворачиваем его и получаем нужный нам массив из порядковых номеров работников, которые должны сняться в ролике.

Временная сложность

Наш алгоритм состоит из двух шагов:

1. Основа алгоритма. Перебор всех работников, отсортированных по времени работы в компании, временная сложность – O(n). И для каждого сотрудника выполняется поиск места вставки в упорядоченный массив длиной n+1, который выполняется двоичным поиском, временная сложность которого как известно – O(log (n+1)). Итоговая временная сложность – O(n log (n+1)) = O(n log n).
2. Восстановление ответа. Восстановление ответа работает с помощью цикла *while*. Количество выполнений данного цикла равняется длине НУП, то есть оценка худшего случая равняется O(n), когда весь исходный массив и является наибольшей убывающей подпоследовательностью.

Итоговая временная сложность — O(n log n + n) = O(n log n).

Затраты памяти

Для реализации описанного выше алгоритма требуется только исходный массив эффективности сотрудников размера n, вспомогательные массивы *mas, pos* размером n+1 и массив *prev* размером n, а также константное число вспомогательных переменных (таких как *len, first, last, mid*, . . .).

Таким образом, итоговые затраты памяти — O(n).