Алгоритмы 11

Максим Пасько

29 апреля 2018 г.

Задание 1

Решим динамикой: будем на каждом i-ом шагу смотреть на последовательность, оканчивающуюся на a_i, а также хранить максимальную последовательность на нашем шаге. Для $k = 1$ $S_{end} = a_1 = S_{all}$.

1) $i_{all} = i_{end} = 1, j_{all} = j_{end} = 1$
2) for$(k = 2, k \leq n; ++k)$
3) if$(S_{end} + a_k > a_k)$ then $j_{end} = k$
4) else $i_{end} = j_{end} = k$
5) $S_{end} = max(S_{end} + a_k, a_k);$
6) $if(S_{all} > S_{end})$ $theni = i_{all}, j = j_{all}$
7) else $i = i_{end}, j = j_{end}$
8) $S_{all} = max(S_{all}, S_{end})$
9) $print(i, j);$

Задание 2

Решим динамикой: пусть $f(p, q)$ - максимальная длина общей подпоследовательности последовательностей $x[1]...x[p]$, $y[1]...y[q]$. Тогда очевидно, что:

1) $x[p] = y[q] \to f(p, q) = f(p-1, q-1) + 1;$
2) $x[p] \neq y[q] \to f(p, q) = max(f(p-1, q), f(p-1, q-1));$

Он корректен, т.к. рассматривает все варианты развития событий, и работает динамикой.
Асимптотика, понятно, складывается из произведения размера массивов, т.к. максимум сравнений мы проведём как раз nm.

Задание 3

Очевидно, что если мы можем набрать сумму $S - v_i$, тогда можем набрать и S, а значит если мы будем идти по массиву из S элементов, значение каждой ячейки которого равно 0, то будет два варианта:

1) $i = v[j] \Rightarrow a[i] = 1$
2) $i > v[j] \Rightarrow a[i] = a[i - v[j]]$

Тогда, для каждого $i \leqslant s$ мы будем знать, можно ли набрать такую сумму, если в итоге в ячейке $a[s]$ будет стоять значение 1.(1 - да, 0 - нет)

Очевидно из допущения, что если $s - v_i = \sum_{j=1}^{n} \alpha_j v_j$, то $s = v_i + \sum_{j=1}^{n} \alpha_j v_j$

Имеем два цикла, в каждом шаге которых делаем сравнения за постоянное время, а значит данный алгоритм займет время $O(ns)$.