Нужно больше энергии

Авторы задачи и разработчики: Дмитрий Грунтов и Александр Яцук

Базовая идея решения — использовать динамическое программирование. Первая подзадача при этом решается аккуратным перебором (например, рекурсивным перебором) всех вариантов расположения криокамер. Тогда как во второй подзадаче можно использовать динамику наподобие $\texttt{dp}[i][j][lst][up]$, где i — количество уже расставленных криокамер, j — количество энергетически выгодных криокамер среди первых i, lst — расстояние последней криокамеры от стены, и up — флаг (либо 0, либо 1), означающий, имело ли место возрастание расстояний от стены между последними двумя камерами (то есть, может ли последняя криокамера стать энергетически выгодно расположенной, если следующую расположить ближе).

Пересчитывать такую динамику можно по следующим формулам:

$$\texttt{dp}[i][j][lst][0] = \texttt{dp}[i-1][j][lst][*] + \sum_{prv=lst+1}^{x} (\texttt{dp}[i-1][j][prv][0] + \texttt{dp}[i-1][j-1][prv][1])$$

$$\texttt{dp}[i][j][lst][1] = \sum_{prv=1}^{lst-1} \texttt{dp}[i-1][j][prv][*]$$

Здесь под $[up = *]$ имеется в виду сумма значений динамики в состояниях с $[up = 0]$ и $[up = 1]$. Идея следующая — если последняя криокамера не дальше предыдущей, тогда предыдущая либо на том же расстоянии и не расположена энергетически выгодно, либо на большем расстоянии и расположена энергетически выгодно (j без последней меньше на 1) тогда и только тогда, когда камера перед ней была ближе к стене ($[up = 1]$). Во втором же случае достаточно просто просуммировать состояния, в которых предыдущая камера расположена строго ближе, и тогда она в любом случае не расположена энергетически выгодно, поэтому j не уменьшается. Такое решение за $\mathcal{O}(nx^2k)$ решало вторую подзадачу, только если не пытаться завести статический массив размера $500 \times 500 \times 500$.

Для решения третьей подзадачи можно заметить, что используемые в формулах пересчета динамики суммы можно поддерживать и считать за $\mathcal{O}(1)$, используя префиксные суммы $\texttt{ps}[i][j][lst][up] = \sum_{t=1}^{lst} \texttt{dp}[i][j][t][up]$. При пересчете динамики внутри одного слоя (i, j) по возрастанию lst можно было считать такие префиксные суммы параллельно с основной динамикой, что позволяет получить решение за $\mathcal{O}(nxk)$.

Чтобы не получить вердикт `ML` в четвертой подзадаче, достаточно избавиться от первого измерения массива `dp`. Действительно, формула пересчета для $\texttt{dp}[i]$ всегда обращается к $\texttt{dp}[i-1]$, поэтому достаточно хранить только два последних слоя, тем самым сокращая используемую память с $\mathcal{O}(nxk)$ до $\texttt{O}(xk)$. Время решения при этом не меняется.

Достаточно аккуратное решение четвертой подзадачи проходило и пятую, но если нет, то можно было добавить различные оптимизации, например, считать реже выполнять операцию взятия по модулю $10^9 + 7$, либо хранить только один слой, и пересчитывать `dp` только через `ps`, таким образом следуя логике «(посчитать `dp` слоя i) $\to$ (посчитать `ps` слоя i) $\to$ (посчитать `dp` слоя $i + 1$) $\to \ldots$». Асимптотика времени решения все та же, $\mathcal{O}(nxk)$.