Разбор задачи «D. Сообщающиеся сосуды»

Автор задачи Антонов Вадим
Авторы разбора Антонов Вадим, Батузов Кирилл

Перед тем, как рассматривать решения, докажем 2 леммы.

Лемма 1. *Пусть в правильном решении задачи уровни жидокости в отсеках получились равными p_i. Тогда выполнены следующие утверждения:*

1. $\sum_i p_i = C$
2. $\forall i\ p_i \geq p_{i+1}$
3. *если* $p_i > p_{i+1}$ *и* $p_{i+1} > 0$, *то* $H_i = p_i$
4. *если* $p_i < H_i$, *то* $\forall j : j > i$ *будет выполнено* $p_j = 0$
5. $p_i \geq 0$

Доказательство. Данные утверждения являются формализацией условия задачи. □

Лемма 2. *Существует единственное распределение жидкости, удовлетворяющее условиям леммы 1.*

Доказательство. Пусть существуют два различных распределения жидкости p_i и q_i, удовлетворяющие условиям леммы 1.

Тогда, из 1 следует, что существуют такие индексы x и , что, без ограничения общности $p_x > q_x$, $p_y < q_y$. Тогда найдется такой индекс i, что $x \leq i < y$ и $p_i > q_i$, $p_{i+1} \leq q_{i+1}$.

Из $p_i > q_i \geq q_{i+1} \geq p_{i+1}$ по условию 2 следует, что $p_i > p_{i+1}$ и по условию 3 получаем, что $H_i = p_i$.

Из $H_i = p_i$ и $p_i > q_i$ следует, что $q_y = 0$ (по условию 4). Но $q_y > p_y \geq 0$ (из условия 5). Противоречие. □

Решение на 30 баллов

Для решения задачи на 30 баллов применим метод динамического программирования.

Предположим, что $H_0 = \infty$ и $H_N = \infty$, а в C_i будет храниться ответ для i-го столбца.

Введём два дополнительных массива: $Sum_i = \sum_{j=1}^{i} C_i$ и A_i^j - высота воды в столбцах между перегородками i и j после установления стационарного состояния, если в столбцах между перегородками с i по j был установлен одинаковый уровень. Из определения следует, что Sum_i не может привышать C.

Пусть посчитаны A_i^j для всех значений, меньших j. Тогда для того, чтобы посчитать A_i^j требуется вычислить $a = min(H_i, H_j)$, $b = \max_{k \in (i,j)} H_k$ и $c = \frac{C - Sum_{i-1}}{j-i}$. Понятно, что если $b \leq a$ и $b \leq c$, то $A_i^j \in [b, a]$, а если $b > a$ или $b > c$, то такого A_i^j не существует. В случае, если A_i^j существует, то оно будет равно c, если $c \leq a$ и a — иначе (т.е. установившийся уровень воды не может быть больше, чем отверстия в перегородках i и j, и не может быть меньше, чем максимальная высота отверстия в перегородках между i и j).

Как же среди всех существующих A_i^j выбрать то, которое подходит? Исходя из лемм, нужно выбрать то A_i^j, для которого $A_i^j * (j-i) + Sum_{i-1}$ будет максимально. После определения нужного A_i^j можно посчитать Sum_j.

Исходя из этого алгоритма, понятно, что для его работы требуется проинициализировать Sum_0 нулём.

Давайте посчитаем количество операций, требуемых для этого решения. В таблице A_i^j $O(N^2)$ ячеек. Для заполнения одной ячейки требуется $O(N)$ операций. Получается $O(N^3)$.

Решение на 60 баллов

Предположим, что Петя поставил немного другой эксперимент. Пусть он поставил на каждое отверстие кран и открывал их по одному. Сначала он открыл кран в перегородке между первым и вторым отсеком, дождался пока вода достигнет состояния равновесия, затем открывал кран в перегородке между вторым и третьем отсеками, опять ждал и так далее. Легко видеть, что распределние воды, полученное в таком эксперименте удовлетворяет условиям леммы 1, а следовательно, по лемме 2, оно совпадает с распределением, которое нас просят узнать в задаче. Покажем, как его найти.

Пусть Петя уже открыл первые $L-1$ кранов и дал воде прийти в состояние равновесия. Если изобразить уровень воды, то он будет напоминать лестницу:

$$p_1 = p_2 = \cdots = p_{i_1} > p_{i_1+1} = p_{i_1+2} = \cdots = p_{i_2} > \cdots > p_{i_f+1} = p_{i_f+2} = \cdots = p_L,$$

причем $p_{i_j} = H_{i_j}\ \forall j = 1..f$.

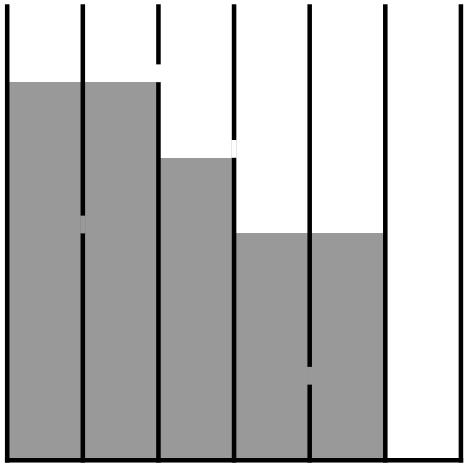

Рис. 1: Уровень воды в первых L секциях

Пердположим Петя открыл кран, между отсеками L и $L+1$. При этом вода начнет перетекать из последней «ступеньки» в секцию $L+1$, а уровень воды в других «ступеньках» (то есть в секциях с первой по i_f) меняться не будет. При перетекании из последней «ступеньки» в новый отсек уровень воды секциях последней «ступеньки» будет равномерно уменьшаться до тех пор, пока не произойдет одно из следующих событий:

1. уровень воды в секии $L+1$ и в секциях последней «ступеньки» выровняется
2. уровень воды в «ступеньке» достигнет одного из кранов, расположенных внутри нее (мы будем считать, что кран между последней «ступенькой» и следующей за ней секцией также расположен внутри последней «ступеньки»)

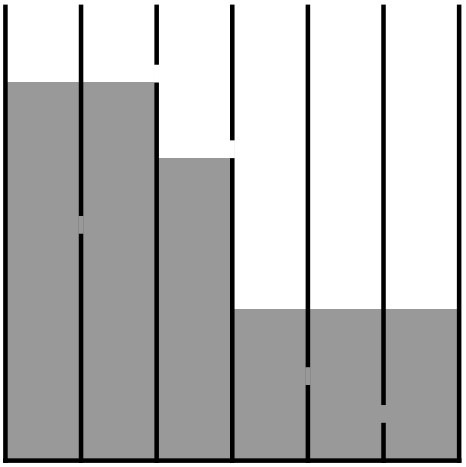

Рис. 2: Случай 1

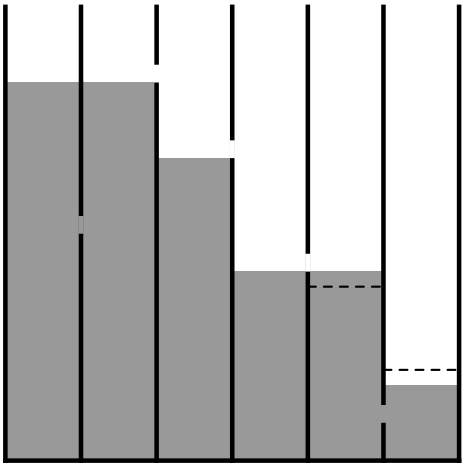

Рис. 3: Случай 2

Рис. 4: Уровень воды в случае 1 (рис. 2) и 2 (рис. 3). Во втором случае пунктиром показано изменение уровня воды в дальнейшем.

В первом случае вода достигла равновесия и Петя может окрывать следующий кран. Во втором случае из последней «ступеньки» образуются две новых. Из последней из них вода продолжет перетекать в секцию $L + 1$ по тому же закону.

Сформулируем алгоритм. Начальное состояние — одна «ступенька» из первой секции с уровнем воды C, откыт один кран.

Шаг алгоритма:

1. Вычислить, сколько воды должно перетечь из «ступеньки» в следующую секцию до наступления события 1.

2. Вычислить, сколько воды должно перетечь из «ступеньки» в следующую секцию до наступления события 2.

3. Если событие 1 наступит раньше, чем событие 2, то расширить «ступеньку» на одну секцию, пересчитать уровень воды, открыть следующий кран.

4. Если событие 2 наступит раньше, чем событие 1, то разбить «ступеньку» на две, пересчитать уровень воды в этих ступеньках и в следующей секции.

Шаг необходимо повторять до тех пор, пока не будет получен окончательный ответ.

Очевидно, что в ходе алгоритма границы последней ступеньки движутся только вправо, причем на каждом шаге одна из них изменяется. Значит алгоритм выполнит $O(N)$ шагов.

Пункт 1 шага алгоритма легко реализовать за $O(1)$ — необходимо решить линейной уравнение с одной неизвестной. В пункте 2 необходимо найти кран, расположенный наиболее высоко внутри последней «ступеньки» (именно по нему она будет разделена). Простая реализация требует $O(N)$ операций. Пункты 3 и 4 также легко реализуются за $O(N)$ операций, поскольку необходимо записать новый уровень воды во все секции. Таким образом, при реализация данного алгоритма будет работать за $O(N^2)$ операций.

Правильное решение

Покажем, как реализовать предыдущее решение за $O(N)$.

Начнем с пунктов 3–4. Заметим, что в ходе шага алгоритма уровень воды изменяется только в секциях, входящих в состав последней «ступеньки», и следующей за ней секции. Причем, уровень воды во всех секциях, входящих в состав «ступеньки», одинаков. Значит, нам достаточно хранить этот уровень, а в ответ мы его запишем в пункте 4 в те секции, которые перейдут из последней «стпеньки» во вновьобразованную предпоследнюю. Уровень воды во всех «ступеньках», кроме последней в ходе алгоритма изменяться не может, поэтому на запись ответа суммарно будет затрачено $O(N)$ времени. Все остальные действия в пунктах 3–4 выполняются за $O(1)$.

Остался пункт 2. Предположим, что у нас есть массив h:

- h_1 — кран, расположенный вутри последней «ступеньки» и имеющий наибольшую высоту. Среди всех таких — самый правый.
- h_i, где $i > 1$ — кран, расположенный внутри последней «ступеньки» правее крана h_{i-1} и имеющий наибольшую высоту. Среди всех таких — самый правый.

Первый элемент массива h позволит реализовать пункт 2 за $O(1)$. Однако возникает другая проблема — в пунктах 3 и 4 необходимо пересчитывать этот массив. Покажем, как это сделать эффективно.

С пунктом 4 все просто — необходимо отбросить первыйэлемент массива. Мы, конечно, не будем двигать содержимое массива, а просто сдвинем его начало. Это требует $O(1)$ времени.

В пункте 3 воспользуемся идеей, аналогичной идее, используемой в алгоритме Грэхема построения выпуклой оболочки. Пусть при добавлении новой секции к «ступеньке» добавился кран W. Пока последний кран в массиве h расположен ниже крана W будем удалять последний элемент массива h. Затем добавим W в конец массива h. Удаление одного элемента из конца массива требует $O(1)$ времени. Поскольку за каждый шаг алгоритма в массив h добавляется не более одного элемента, а всего шагов $O(N)$, то суммарно на все удаления будет затрачено $O(N)$ вермени.

Таким образом весь алгоритм будет работать за $O(N)$.