Анализ на задача Hedgehog

от Виктор Кожухаров

В задачата е дадена координатна система, както и ябълки и круши поставени на точки в нея. Търси се път, който може да върви само нагоре или надясно, такъв че броят плодове е максимизиран. Ако има няколко такива пътя трябва да се намери такъв, който има най-близък брой ябълки и круши.

Подзадача 1: $N <= 100$, 5 точки

Можем да използваме динамично програмиране, което да помни позицията ни в координатната система, общият брой плодове, които сме събрали, и броят от тях, които са ябълки. За всяка комбинация от тези параметри динамичното програмиране ще помни дали тя е възможна за постигане. Така сложността на това решение става $O(N^4)$.

Подзадача 2: $N <= 400$, 5 точки

Решението от предишната подзадача може да се подобри лесно. Просто премахваме от state-а на динамичното броя плодове и вместо това самата стойност на ДП-то ще е максималния брой плодове, които можем да постигнем. Сложността става $O(N^3)$.

Подзадача 3: $N <= 5000$, 10 точки

Намирането на максималния брой плодове, които могат да се съберат е класическа задача, която може да се реши с Фенуик или сегментно дърво. Минаваме по нарастващо x на плодовете и намираме най-големия брой плодове, които можем да вземем преди сегашния чрез заявка за максимум на префикс.

За да решим третата подзадача можем да модифицираме това решение като пазим във Фенуик-ът освен максималния брой плодове, които можем да получим, също и boolean масив. 1 на дадена позиция в този масив означава, че можем да получим максималния брой плодове с толкова на брой ябълки. Сложността става $O(N^2 logN)$.

Подзадачи 4 и 5: $N <= 5 \times 10^4$, 40 точки

Предишното решение може да се оптимизира като се използва bitset за масива в Фенуик. Тогава сложността става $O(N^2 logN/64)$.

Подзадача 4 може да се реши и с $O(N^2)$ решение, което обаче е по-трудно от решението с Фенуик.

Подзадача 6: $N <= 9 \times 10^4$, 40 точки

За да решим цялата задача, трябва да променим начина, по който я представяме. Можем да забележим, че ако максималният брой плодове, които могат да се съберат за дадена точка е S, то за тази точка има поне един плод, със максимален резултат $S - 1$, от който може да се стигне до нея. Затова можем

да започнем да мислим за плодовете като организирани във слоеве спрямо общия брой плодове, които могат да се съберат свършвайки в тях.

Още повече, не може да се стигне от плод в един слой в плод във същия слой. Това е така, защото тогава единия от тях би имал резултат $S+1$. Следователно, плодовете в един слой могат да се подредят $(x_1, y_1), (x_2, y_2), ..., (x_k, y_k)$, така че $x_1 < x_2 < ... < x_k$ и $y_1 > y_2 > ... > y_k$.

Един плод ще може да използва някакъв интервал $(x_l, y_l), ..., (x_r, y_r)$ от предишния слой. Този интервал може да се намери бързо чрез два показателя. Нека за всеки плод пазим bitset, който пази броя ябълки, които са възможни. Тогава задачата се превръща във бързо пресмятане на битовото ИЛИ върху тези интервали.

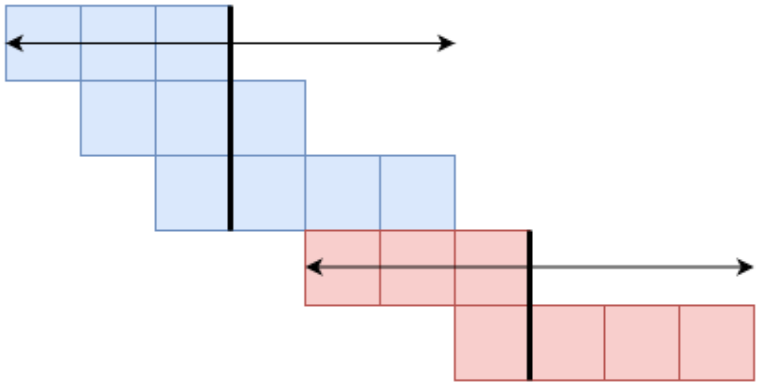
Можем да използваме сегментно дърво за да получим сложност $O(N^2 logN/64)$. Това решение ще получи 60 точки, но не е достатъчно бързо за цялата задача.

Следващото наблюдение, което можем да направим, е че интервалите в един слой са нарастващи, тоест $l_1 <= l_2 <= ... <= l_k$ и $r_1 <= r_2 <= ... <= r_k$. Как бихме могли да решим задачата ако $l_1 <= l_2 <= ... <= l_k <= mid$ и $mid < r_1 <= r_2 <= ... <= r_k$?

Нека пресметнем побитовото ИЛИ за $(i, i+1, ..., mid)$ за всяко $i <= mid$ и за $(mid+1, mid+2, ..., j)$ за всяко $j >= mid+1$. Тогава за да отговорим на заявка за побитовото ИЛИ на интервал от l_j до r_j трябва само да извършим ИЛИ между съответния префикс и съответния суфикс. Повече за тази идея можете да разберете от този линк: `https://usaco.guide/plat/DC-SRQ?lang=cpp`.

Можем да използваме тази техника по следния начин – ще разделим интервалите на групи, така че всяка група да съдържа всички интервали с лява граница по-малка или равна на дясната граница на най-левия интервал в нея. Тоест започваме с най-левия интервал и добавяме в неговата група всички други интервали с $l_i <= r_1$. След това ги премахваме и образуваме групи с останалите интервали. Вижда се, че можем да използваме идеята от предишния параграф за да решим всяка от групите.

Графиката показва разделение на 5 интервала на 2 групи. Сините интервали са в първата група, а червените във втората. С удебелена линия се обозначава разделителната точка mid за всяка от групите.



Каква е сложността на това решение? Всяка клетка се покрива най-много два пъти $\implies$ изпълняваме O(брой плодове в предишния слой) побитови ИЛИ-та. Общата сложност става $O(N^2/64)$.