Анализ на задача minmax

Тагове: комбинаторика, сортиране, броячен масив

Формалната постановка на задачата е следната: дадена е редица $a_1, a_2, \ldots, a_n$ и търсим броя валидни избори на четири индекса от редицата k_1, k_2, d_1, d_2, за които са изпълнени следните три условия:

- $k_1 < k_2$
- $d_1 < d_2$
- $\max\{a_{d_1}, a_{d_2}\} = \min\{a_{k_1}, a_{k_2}\}$

Нулева подзадача – 0 точки

Тази задача е оставена за комуникация със системата – ако състезателят забелязва, че решението му връща верен отговор на примерните тестове на машината му, но системата твърди, че решението връща грешен отговор на тези тестове, можем да се уверим, че грешката не е в решението, а нещо механично – примерно грешно въведен вход, работа с невалидни индекси на масиви или подобни.

Решение на първа подзадача - 11 точки

Ограничението над дължината на цялата редица $n \leq 50$ е достатъчно малко, за да позволи директна имплементация на зададената задача да се вмести в тайм лимита – четири вложени цикъла избират четирите индекса k_1, k_2, d_1, d_2 и проверяват нужните условия. Заради четирите цикъла ще оценим времевата сложност на алгоритъма $O(n^4)$.

Решение на втора подзадача - 33 точки

Завишеното ограничение над n ни принуждава да разкараме един от вложените цикла и да свалим сложността на $O(n^3)$. Може да се напише валидно решение чрез премахване на кой да е от циклите, но в авторовия код `minmax_cubic_33pts.cpp` е имплементирано такова, което избира k_1, k_2, d_1 и прави разсъждения за d_2, така че ще се придържаме към тази идея.

След като сме избрали индексите на Калоян, това ни фиксира какво число трябва да седи от двете страни на равенството. Също от циклите вече сме избрали едното число на Дамян. Нека да разгледаме колко е броят на валидните конфигурации $(k_1, k_2, d_1, *)$:

- Ако $\min\{a_{k_1}, a_{k_2}\} > a_{d_1}$, можем да изберем кое да е от числата равни на $\min\{a_{k_1}, a_{k_2}\}$. Техният брой може да вземем директно от броячен масив (имаме допълнителното условие, че $a_i \leq 10^6$), който сме подготвили със самото въвеждане на числата;
- Ако $\min\{a_{k_1}, a_{k_2}\} = a_{d_1}$, можем да се интересуваме само от конфигурациите, в които Дамян избира две равни числа – ако не са равни, то тази конфигурация ще е преброена в предходната точка. Тук авторовото решение преброява всяка двойка от равни елементи два пъти, защото няма как да ги различим – заради това накрая бройката им се дели на 2.
- Ако $\min\{a_{k_1}, a_{k_2}\} < a_{d_1}$, трябва да заключим, че не е възможно да имаме валидна конфигурация с тези индекси, защото най-голямото число на Дамян ще бъде поне a_{d_1}, а това вече е по-голямо от най-малкото число на Калоян.

Решение на третата подзадача - 66 точки

От тук нататък водеща идея ще е да извършим броенето за двете момчета отделно – наистина те са почти независими освен в момента, когато трябва да проверим, че получените им отговори са равни. Това лесно може да се оправи с броячни масиви – авторовото решение ползва такива `kaloyan[]` и `damyan[]`, в които записва колко избора има дадено момче, такива че минимума или максимума му съответно има стойност v. Заради все още ниското ограничение $n \leq 2000$ можем да си позволим да обходим всички двойки избори за индекси. Накрая, за да изброим колко начина има равенството между момчетата да има стойност v, просто ще умножим `kaloyan[v] * damyan[v]`.

Решение на четвърта подзадача - 83 точки

Предишната подзадача ни научи да работим с двете момчета по отделно. Сега ще видим как да не обикаляме абсолютно всички възможни двойки индекси. Работата е почти симетрична за Калоян и Дамян – в анализа ще обясним как да смятаме стойностите за Калоян.

Формално се интересуваме как във времева сложност по-добра от $O(n^2)$ да класифицираме всички двойки спрямо по-малкият им елемент. Искаме отново да подготвим масив `kaloyan[]`, в който v-тата стойност ни показва колко двойки имат минимум точно v. Преформулирано, за всяко v от входната редица, искаме да преброим колко други числа има по-големи или равни на него. Тъй като не се вълнуваме от наредбата на числата можем да си позволим логичния лукс да сортираме редицата – наистина така всички по-големи или равни числа на мен ще са с индекс по-голям или равен от моя! Може би единственото притеснение е като имаме няколко равни елемента как всъщност работим – бихме се притеснявали една и съща двойка да не я преброим два пъти. Тогава обаче даже е удобство, защото можем да преброим двойката само от гледната точка на числото с по-малък индекс. Окончателно, числото $a[i]$ в сортираната редица е минимумът на всички двойки (i, j), за които $i < j$, а те са $n - i - 1$ на брой (индексацията е от нула).

Решение на пета подзадача - 100 точки

Четвъртата подзадача премахва ограничението над стойностите на редицата и ни кара да мислим как да премахнем броячният масив. При внимателен поглед човек вижда, че даже и в четвъртата подзадача той не беше нужен – можем всякакви изчисления за числото v да извършим заедно, обхождайки подинтервала на редицата, съдържащ само стойностите v. Каквито и сметки да правим тук, те няма да влияят на сметките за останалите числа. Поради сортирането, окончателната времева сложност на задачата възлиза на $O(n \log n)$.

Автор: Иван Лупов