Анализ на задача partition

Тагове: ad-hoc задача, алчни алгоритми, плъзгащ прозорец

На пръв поглед тази задача изглежда трудна – тя наподобява по-специфична версия на задачата за братска подялба[1], която има стандартно решение с динамично програмиране, но то е ефективно само за малки стойности[2]. Оказва се обаче че има различни прости конструкции с линейна сложност.

За яснота ще считаме, че разделяме оригиналното множество A (възможно е да е мултимножество) и за дадено разделяне сме фиксирали реда на елементите – $A_0, A_1, \ldots, A_{N-1}$, където префиксът образува първото множество (на Дени), а оставащият суфикс образува второто множество (на Мартина). Размерът на префикса се определя както е описано в условието: спираме когато сумата стане $\geq \frac{S}{2}$. Важно е да отбележим, че ако S е нечетно, трябва да внимаваме и затова е добре да сравняваме текущата сума s така: $2 * s >= S$ вместо $s >= S/2$. Има тестове, които проверяват за този частен случай.

Решение на първа подзадача – 3 точки

Тук N и стойностите са достатъчно малки, така че почти всяко вярно решение би трябвало да мине (да не получи time limit). Най-лесното решение е да сортираме всички числа и след това да разделим всяка група от равни стойности поравно между двете множества. Възможен е и подход с counting-sort. Всяка стойност се среща четен брой пъти и следователно това е оптимално решение – разделянето ще бъде на две множества с по точно $\frac{N}{2}$ елемента и със сума $\frac{S}{2}$ всяко.

Сложност: $O(TN \log N)$.

Решение на втора подзадача – 3 точки (=0+3)

Можем да използваме факта, че сумите не е необходимо да са особено близки до $\frac{S}{2}$. Така можем лесно да конструираме оптимално решение, защото $2^{N-1} > 2^0 + 2^1 + \cdots + 2^{N-2} \implies 2^{N-1} > \frac{S}{2}$ и така можем да използваме това число като спиращ елемент на която и да е позиция. Това означава, че при четно N можем да поставим 2^{N-1} в разделянето като елемент $A_{N/2-1}$ и тогава префиксът ще съдържа точно $\frac{N}{2}$ числа независимо какви степени на 2 поставим преди него. Аналогично, при нечетно N можем да поставим 2^{N-1} като елемент $A_{[N/2]-1}$ и тогава префиксът ще съдържа $[N/2]$ елемента. Абсолютната разлика между размерите на префикса и суфикса ще бъде 1, което е оптимално при нечетно N.

Сложност: $O(TN)$.

Решение на трета подзадача – 6 точки (=0+3+3)

Тази подзадача комбинира идеите от предишните две. Можем да обобщим аргумента за стойности, които се срещат четен брой пъти, и първо да разделим всички двойки от равни стойности между двете множества (така и сумите, и размерите им са равни засега). Тъй като тук N е по-голямо отколкото в първата подзадача, трябва да използваме подход с counting-sort за откриване на двойките равни стойности.

---

[1] Задачата за братската подялба е за намиране на разбиване на дадено множество на две подмножества със суми възможно най-близки една до друга.

[2] Всъщност тази сложност се характеризира като псевдополиномиално време, т.е. тя е полиномиална, ако не разглеждаме двоичното представяне на стойностите. Ако го разглеждаме, тогава сложността очевидно е експоненциална.

Сега ни остават само уникални степени на 2, които първоначално са се срещали нечетен брой пъти. Можем да направим същото като в предишната подзадача – да използваме най-голямата степен на 2 като спиращ елемент, защото тя е по-голяма от сумата на всички по-малки степени (дори и ако се срещат всички). При четно N можем да я поставим, така че първото подмножество да има точно половината елементи, а при нечетно N – възможно най-близо до половината елементи.

Сложност: $O(TN)$.

Решение на четвърта подзадача – 5 точки (=0+0+0+5)

Можем да използваме стандартната техника на групиране по двойки за намиране на сумата на последователни числа – сумата на 1 и N е равна на сумата на 2 и $N-1$, равна е на сумата на 3 и $N-2$ и т.н. Ако N се дели на 4, разделяме поравно $\frac{N}{2}$-те групи между двете множества. Ако N е четно, но не се дели на 4, просто разделяме една от групите $(x, N+1-x)$ и добавяме $N+1-x$ (по-голямото число) към префикса, а другото към суфикса – това гарантира, че префиксът ще има сума $> \frac{S}{2}$ и това ще стане точно когато включим $N+1-x$ към сумата. Случаят, когато N е нечетно, е подобен – не можем да направим двете множества с равен размер, затова оставяме средната стойност $[N/2]$ по средата, след което правим същото както преди, защото ни остават четен брой стойности $(1, 2, \ldots, [N/2]-1, [N/2]+1, \ldots, N-1)$.

Сложност: $O(TN)$.

Решение на пета подзадача – 4 точки (=0+0+0+0+4)

Тази подзадача е предназначена за решение с пълно изчерпване. Освен това е полезна и за проверка на коректността, тъй като има много сценарии в тестовете. Минаваме през всички пермутации на A и проверяваме дали сме намерили подходящо разделяне. Като разгледаме няколко примера, можем да видим, че минималната абсолютна разлика при четно N винаги е нула, т.е. винаги има начин да се направи разделяне, при което двете множества са с равен размер, а при нечетно N тя винаги е едно. Оказва се, че тази задача е от типа задачи, при които най-оптималното решение винаги е постижимо. Ще обосновем това твърдение в последните две секции. За да минем тази подзадача, трябва да използваме това наблюдение, така че да прекъснем търсенето, когато намерим пермутация, която постига оптималната абсолютна разлика. За повечето A това обикновено се случва много рано в търсенето с пълно изчерпване.

Сложност: $O(TN!N)$.

Решение на шеста подзадача – 13 точки (=0+3+0+0+4+6)

Можем да оптимизираме предишното решение с пълно изчерпване по стандартен начин. Нека си представим, че рекурсивно генерираме пермутациите и едновременно проверяваме разделянето (прекратявайки, когато абсолютната разлика е ≤ 1). Когато сме на i-тото число и текущото частично разделяне все още е валидно, трябва да знаем само кои индекси вече са били използвани, а не реда, в който са били избрани. Следователно можем да използваме мемоизация за маската на индексите. За да хванем тази подзадача, е важно да имплементираме този подход рекурсивно, тъй като има много маски, които няма да разгледаме (например всички маски, които имат поне $\frac{N}{2}+1$ елемента).

Сложност: $O(T2^N N)$.

Решение на седма подзадача – 32 точки (=0+3+0+0+4+6+19)

Тази подзадача е за по-бавни имплементации на по-добрите идеи и за рандомизирания подход. При този подход многократно тестваме случайни пермутации на A, докато намерим подходящо разделяне. Оказва се, че вероятността да намерим такава пермутация е около $\frac{1}{N}$ и следователно очакваният брой опити е $O(N)$. Ще обосновем това твърдение в последната секция.

Сложност: $O(TN^2)$.

Решение на осма подзадача – 63 точки (=0+3+3+0+4+6+19+28)

Можем да подходим алчно към конструирането на разделяне. Нека сортираме числата и първо добавим в префикса най-големите числа в A, докато сумата стане $\geq \frac{S}{2}$ (в намаляващ ред). Ще наричаме тези числа „големи“. Ясно е, че размерът на префикса не може да надвишава $\lceil \frac{N}{2} \rceil$. Ако размерът е точно толкова, значи сме готови. В противен случай можем да започнем да добавяме най-малките числа едно по едно в префикса. След всяка итерация проверяваме дали префиксът трябва да съдържа по-малко елементи. Това се случва, когато сумата на числата в префикса без последното добавено „голямо“ число е $\geq \frac{S}{2}$. Тогава премахваме това число (забележете, че няма да се налага да премахваме друго число, защото само заменяме елементи с по-малки). След това проверяваме дали префиксът има необходимия размер и прекратяваме, ако това е така.

Защо този алгоритъм винаги намира решение? Нека допуснем противното – че накрая нямаме достатъчно числа в префикса и сме свършили с „малките“ числа. Това означава, че сме премахнали всички „големи“ числа, тъй като не премахваме „малки“ числа. Но сумата на всички „малки“ числа е $< \frac{S}{2}$ (не може да е равна, защото тогава техният брой би бил $\lceil \frac{N}{2} \rceil$ и те вече биха образували валидно разделяне), което е невъзможно, тъй като винаги гарантираме, че префиксът има сума $\geq \frac{S}{2}$.

Ще опишем още две идеи. Нека отново сортираме числата в ненамаляващ ред и след това елементите са: $A_0 \leq A_1 \leq \ldots \leq A_{N-1}$. Първо разглеждаме само случая, когато N е четно. Можем да групираме числата в двойки от последователни елементи: $(A_0, A_1), (A_2, A_3), \ldots, (A_{N-2}, A_{N-1})$. Нека добавим в префикса всички по-големи елементи в двойките – $A_1, A_3, \ldots, A_{N-1}$. Твърдим, че това е валидно разделяне. Ясно е, че сумата на тези стойности е $\geq \frac{S}{2}$. Но също така е вярно, че: $A_1 + A_3 + \cdots + A_{N-3} < \frac{S}{2}$. Ако допуснем противното, тогава получаваме противоречие, използвайки факта, че $A_1 \leq A_2, A_3 \leq A_4, \ldots, A_{N-3} \leq A_{N-2}$, което би означавало, че $A_2 + A_4 + \cdots + A_{N-2} \geq A_1 + A_3 + \cdots + A_{N-3} \geq \frac{S}{2}$ и следователно сумата на всички числа без A_0 и A_{N-1} ще бъде $\geq S$. При нечетно N можем да направим същото, като премахнем най-големия елемент и го добавим към префикса след това.

Последната идея е най-стандартната и се използва в подобни случаи. Когато търсим подмножество с някакво свойство, често се оказва, че съществува подмасив с това свойство (което е по-лесно за намиране). Отново сортираме числата в ненамаляващ ред и този път разглеждаме сумата на плъзгащ прозорец с размер $\lceil \frac{N}{2} \rceil$. Ако прозорецът в първата позиция има сума $\geq \frac{S}{2}$, лесно можем да видим, че това ще бъде валидно разделяне, тъй като това са най-малките възможни числа. Следователно интересният случай е, когато тази сума е $< \frac{S}{2}$. Но знаем, че в последната позиция прозорецът ще има сума $\geq \frac{S}{2}$. Следователно съществува позиция, при която сумата на прозореца става $\geq \frac{S}{2}$. Твърдим, че този прозорец образува валидно разделяне. Наистина, премахването на последния му елемент прави сумата $< \frac{S}{2}$, защото предишният прозорец, който е съдържал тези елементи (и още един елемент), е имал сума $< \frac{S}{2}$. Интересно е, че това разделяне по същество е обръщане на двете

множества в разделянето, получено от първата идея.

Също така ще отбележим, че когато се опитваме да намерим решение, винаги мислим за валидно разделяне, което има $\lceil \frac{N}{2} \rceil$ елемента в префикса. Предишните конструкции доказват, че такова решение винаги съществува. В частност, абсолютната разлика между размерите на двете множества е ≤ 1.

Сложност: $O(TN \log N)$.

Пълно решение – 100 точки

Интересно е, че последната идея почти работи директно, когато премахнем сортирането. Достатъчно е да намерим първата позиция, при която сумата на плъзгащия прозорец се променя от $< \frac{S}{2}$ на $\geq \frac{S}{2}$, или обратното. Ако сумата се промени от $<$ към $\geq$, тогава същият аргумент важи за втория прозорец (със сума $\geq \frac{S}{2}$), както и преди. Ако сумата се промени от $\geq$ към $<$, тогава аргументът важи за първия прозорец, но трябва да го разглеждаме в обратен ред, защото е възможно първата стойност в него да е твърде голяма (например, ако имаме $7, 2, 1, 5, 2, 1$, тогава знакът се променя от първия към втория прозорец, но $7, 2, 1$ не е валиден префикс, за разлика от тези числа в обърнат ред).

Но какво се случва, ако знакът не се промени? Нямахме този проблем, когато числата бяха сортирани, защото първият прозорец винаги ни даваше валидно разделяне в този случай. Можем да видим, че всички прозорци ще имат сума $\geq \frac{S}{2}$. Твърдим, че оригиналната пермутация е решение. Ако N е четно, тогава сумата на първия прозорец също е $\leq \frac{S}{2}$, защото последният прозорец (с всички останали числа) има сума $\geq \frac{S}{2}$. Следователно първият прозорец има сума точно $\frac{S}{2}$ и може да бъде префикс за решение. Ако N е нечетно, аналогично получаваме, че сумата на първите $\lfloor \frac{N}{2} \rfloor$ числа е $\leq \frac{S}{2}$, но първият прозорец има сума $\geq \frac{S}{2}$, следователно това е решение.

Можем също така да направим подобно решение, при което прозорецът преминава през всички циклични ротации на пермутацията (като прозореца преминава през суфикса и след това се връща в префикса). Тогава няма да се налага да обръщаме прозореца, защото ако има промяна на знака, винаги ще имаме място, където промяната е от $<$ към $\geq$. Това също обяснява защо рандомизираното решение е толкова добро. За всяка пермутация винаги имаме поне 1 циклична ротация, която ще ни даде оптимално разделяне. Следователно за дадено A има поне $N!/N$ пермутации, които ще работят, тъй като има N циклични ротации на една пермутация (включително оригиналната).

Съществува и обобщаване на първите две идеи без сортиране. Отново първо разглеждаме случая, когато N е четно, и групираме числата в двойки от последователни елементи, както във втората идея. Този път поставяме по-малките стойности от двойките в префикса. Тяхната сума е $\leq \frac{S}{2}$ и ако е $= \frac{S}{2}$, сме готови. В противен случай я коригираме по подобен начин като в първата идея – чрез добавяне на по-големи стойности. Итерираме през двойките и започваме да заменяме по-малките стойности с по-големите. Ако заменим всички стойности, сумата ще стане $\geq \frac{S}{2}$, така че в първия момент, когато това се случи, имаме префикс на валидно решение (важно е да добавим последното число в края на префикса, иначе сумата може да стане голяма твърде рано). Когато N е нечетно, можем просто да добавим последното число към префикса и да видим, че подобно разсъждение остава вярно.

Сложност: $O(TN)$.

Задачата може по подобен начин да бъде решена и когато са позволени нулеви стойности. Това обаче въвежда допълнителни частни случаи и усложнява някои доказателства, което би направило задачата ненужно пипкава.

Също така, подходът в третата подзадача (първо да разделим всички двойки от равни стойности между двете множества) може да се използва и в общия случай. Това работи, защото за всяко мултимножество от числа съществува оптимално решение. Следователно е възможно да разглеждаме само множества с уникални стойности, което понякога може да бъде полезно, но не и за тази конкретна задача.

Реализация: Илиян Йорданов
Идея: Мартин Копчев