Условието беше вдъхновено по истински случай

Анализ на задача Близначки

Увод. Преди да затворите анализа, като видите броя на страниците, ще ви успокоя, като кажа, че това е просто, защото са описани различните подходи за частични точки и защото има примерчета ☺. Реалното обяснение на авторовото решение е 2 скромни странички. За удобство ще наричаме отделните въпроси към журито просто въпроси, а групите на които ги задаваме – рундове.

Първа подзадача. Първото и най-просто решение, което ще разгледаме, е да използваме **N** въпроса, като в **i**-тия питаме само за **i**-тата снимка. То ще хване 5% за **P=1** и **L=N**.

Можем да намалим броя въпроси за сметка на броя рундове, като използваме двоично търсене. Нека проследим отговорите на въпросите **{1}, {1,2}, {1,2,3}, …, {1,2…,N}**. Очевидно, ако **S** е номерът на търсената снимка, то първите **S-1** резултата ще са отрицателни, а останалите положителни. Това решение ще хване 15% за **P=log(N)=16** и **L=log(N)**.

(*) Тук с **log(N)** означаваме най-голямото К, за което $2^{K-1} \leq N$ или по друг начин казано, броят битове на **N** в двоична бройна система.

(*1) Преди да продължим със следващата идея ще отбележим едно важно наблюдение за **T=1**, спрямо което след това ще ни е по-лесно да намираме отговора, а именно – едно решение, използващо 1 рунд може еднозначно да определи дали дадена снимка е отговор. Следователно всяка снимка, която не е, участва в поне един въпрос, който не съдържа специалната снимка или не участва в поне един въпрос, в който участва специалната снимка. Иначе няма как да сме сигурни дали точно тя не е отговора.

Дойде време и за най-доброто решение. В интерактивни задачи нерядко се случва да трябва да разгледаме нещата в двоична бройна система. Нека търсената снимка се представя като $S = \overline{b_1 b_2 b_3 \dots b_{\log(N)}}$ в двоична бройна система, а произволна друга снимка като $W = \overline{c_1 c_2 c_3 \dots c_{\log(N)}}$. Очевидно все за някоя позиция i $(1 \leq i \leq \log(N))$ двете числа имат различни битове. Това дава предпоставка да направим следните **log(N)** въпроса: в **i**-тия от тях участват всички числа със сетнат бит на **i**-та позиция. Така, ако $\overline{b_1 b_2 b_3 \dots b_{\log(N)}}$ и $\overline{c_1 c_2 c_3 \dots c_{\log(N)}}$ се различават на позиция **pos** имаме два случая:

- $b_{pos} = 1$ и $c_{pos} = 0$. Положителен въпрос, в който снимката W не участва => тя не е отговор.
- $b_{pos} = 0$ и $c_{pos} = 1$. Отрицателен въпрос, в който снимката W участва => тя не е отговор.

Пример: N=10
{0001,0011,0101,0111,1001} , {0010,0011,0110,0111,1010} , {0100,0101,0110,0111} , {1000,1001,1010}
В червено са оцветени въпросите с положителен отговор, а в синьо са оцветени общите сетнати битове. Всяко число без $0101_{(2)}=5_{(10)}$ изгаря на принципа от (*1).

Крайният брой използвани въпроси за **N=50 000** е **log(N)=16**. Решение за 96%, използващо до 32 въпроса би могло да бъде да имаме двойно повече групи (освен за сетнати битове да имаме и за несетнати битове, но в случая едните са достатъчни).

Втора подзадача. Всички решения от първа подзадача могат лесно да се модифицират, за да работят и при **T=2**, когато двете снимки имат поредни номера. Да помислим как би изглеждало оптималното. Разглеждаме по-малкото число отдясно-наляво. Докато текущият бит е единица по-голямото число ще има нула на тази позиция, защото иначе разликата между двете числа ще стане повече от 1. В момента, в който стигнем до първата нула по-голямото число ще е със сетнат бит. Оттам нататък битовете на двете числа съвпадат.

Пример: 110011001000
110011000111

За да разберем до коя позиция се различават двете числа ще направим същите **log(N)** въпроса за всяка позиция, но освен за сетнати битове ще ни трябват и още **log(N)** за несетнатите. Ако и двата въпроса за дадена позиция са положителни, числата се различават в битовете. Предвид че в общият случай броят на въпросите за **T=2** е доста по-голям, това решение спокойно ще си вземе 100%.

Трета подзадача. Да започнем отново с най-простото решение, използващо **L=N** въпроса. Както в първа подзадача отново хваща 5%.

Следващо подред е двоичното търсене. Нека търсените две снимки са с номера **S** и **W** (**S < W**). Да разгледаме префиксите **{1}, {1,2}, {1,2,3}, …, {1,2…,N}** и суфиксите **{N}, {N,N-1}, …, {N,N-1,…,1}**. Първите **S-1** префикса ще са отрицателни както и първите **N-W** суфикса. Тук са възможни две решения. Едното използва **P=2×log(N)** рунда и получава 10%, докато второто пуска двете двоични едновременно и подава по два въпроса наведнъж, като така използва само **P=log(N)** рунда и хваща 15%.

(*2) Тук наблюдението е по-различно. За да сме сигурни, че една снимка не е търсената, тя трябва да участва във въпрос с отрицателен резултат. Ако не участва във въпрос с положителен резултат, не можем да кажем нищо, тъй като тук имаме две снимки.

Остава да модифицираме и решението, разглеждащо битове. Знаем, че **S** и **W** все се различават в някой бит. Нека разделим числата на две групи спрямо този бит. Сведохме задачата до две задачи с **T=1**. Отново ще имаме **2×log(N)** въпроса по два за всеки бит. Така ще намерим позиция, на която **S** и **W** се различават. Оттам нататък подхода е същият както при **T=1**, като трябва да имаме **2×log(N)** въпроса, решаващи задачата за **T=1**, за всяка позиция, тъй като не знаем предварително къде точно се различават.

Оптимизация 1: Може да съобразим, че веднъж щом намерим **S,** лесно можем да изчислим и **W**, използвайки резултатите от първите **2×log(N)** въпроса, и вместо по **2×log(N)** въпроса на позиция (за две задачи с **T=1**) ползваме двойно по-малко. Освен това при получените подгрупи за **T=1** за дадена позиция можем да си спестим една, тъй като всички числа от групата имат един общ бит.

Оптимизация 2: Последната оптимизация, която можем да направим е като в задачите с **T=1** питаме само за битовете, които още не знаем. В действителност, ако от първите **2×log(N)** въпроса сме стигнали до i-тата двойка, значи търсените числа съвпадат в първите i-1 бита, а в i-тия се различават. Интересуват ни само битовете от i+1-вия нататък.

Крайната сметка за общия брой използвани въпроси възлиза на **L = 2×log(N) + log(N)-1 + log(N)-2 + … 1 = 2×log(N) + log(N) × (log(N) - 1) / 2 = log(N) × (log(N) + 3) / 2.** За **N=50 000** това са 152 въпроса. С това задачата е решена.

Пример: N=9 S=0010 W=0110
Спрямо бит 4:
{0010,0100,0110,1000} : не е нужно да решаваме T=1 и тук
{0001,0011,0101,0111,1001} : {0011,0111} {0101,0111} {1001}

Спрямо бит 3:
{0001,0100,0101,1000,1001} : не е нужно да решаваме T=1 и тук
{0010,0011,0110,0111} : {0110,0111} {}

Спрямо бит 2:
{0001,0010,0011,1000,1001} : не е нужно да решаваме T=1 и тук
{0100,0101,0110,0111} : {}

Спрямо бит 1:
{0001,0010,0011,0100,0101,0110,0111} : не е нужно да решаваме T=1
{1000,1001,} : не е нужно да решаваме T=1

{група с несетнат бит на дадена позиция} : не са нужни подгрупи
{група със сетнат бит на дадена позиция} : подгрупи за T=1 след дадената позиция

Да погледнем резултатите за втората позиция. Там и двете групи са положителни, следователно във всяка от тях има по една специална снимка. Намираме, че $0110_{(2)}=6$ е едната снимка, като решим задачата за T=1 с подгрупите на втората позиция. След това лесно откриваме и $0010_{(2)}=2$, знаейки на кои позиции двете числа се различават в битовете и на кои не.

За любознателните 1. Съществуват и други интересни решения с така наречената коренова декомпозиция:

(T=1) Разделяме снимките на групи от по $\sqrt{N}$ и съответно за всяка от тях използваме по един въпрос. На втория рунд взимаме всички снимки от групата, с положителен резултат и използваме още $\sqrt{N}$ въпроса, за да открием и отговора.

Пример: N=9.
Рунд 1: {1,2,3} , {4,5,6} , {7,8,9}
Рунд 2: {7} , {8} , {9} => Отг. 8

Можем да подобрим предното решение, използвайки само един рунд, същия брой въпроси $(2 \times \sqrt{N})$ и (*1). Нека отново имаме $\sqrt{N}$ групи. Така $(\sqrt{N} - 1) \times \sqrt{N}$ от нормалните снимки вече участват във въпрос, който е с отрицателен отговор. Остават тези, които са били в една група с търсената снимка. Тъй като тя може да е всяко число при второто разпределение на групи не бива две снимки, които са били в една група, отново да са заедно, защото, ако техния въпрос се окаже пак положителен, няма как да сме сигурни коя снимка е търсената. Затова ще разпределим снимките по следния начин: Ако снимка **P** е била на позиция **p** $(1 \le p \le \sqrt{N})$ в групата **G** $(1 \le G \le \sqrt{N})$ при първия рунд, то при втория такъв ще е на позиция **G** в групата **p**.

Пример: N=9.
Рунд 1: {1,2,3} , {4,5,6} , {7,8,9} , {1,4,7} , {2,5,8} , {3,6,9}
Всички снимки участват във въпрос с отрицателен резултат, което остава 6-тата като единствен възможен отговор. Това ще хване около 62% за големите тестове.

(T=2) Първият рунд ще е за стандартните $\sqrt{N}$ групи. Възможно е обаче две от тях да се окажат положителни и на втория рунд да трябва да ги проверим с още $2 \times \sqrt{N}$ въпроса. Така имаме **P=2** рунда и **L** $\approx 3 \times \sqrt{N}$ въпроса. Решението за T=2, използващо само един рунд, е значително по-забавно. Използваме същата стратегия от **T=1**. Нека направим някои означения:
g(i)=(i-1)/sqrt(N)+1 , p(i)=(i-1)%sqrt(N)+1.

(*) С sqrt(N) отбелязваме $\lfloor\sqrt{N}\rfloor$ ($\lfloor x\rfloor$ е най-голямото цяло число по-малко или равно на x).

Пример: N=9

i	1	2	3	4	5	6	7	8	9
g(i)	1	1	1	2	2	2	3	3	3
p(i)	1	2	3	1	2	3	1	2	3

Отделно нека с G(i,p) отбележим групата, в която участва числото **i** при p-тото разбиване (имаме 3 разбивания на по $\sqrt{N}$ групи). Казахме, че първото и второто разбиване ще са същите от **T=1**, т.е. G(i,1)=g(i) , G(i,2)=p(i). Да отбележим двете търсени снимки с **S** и **W**. Ако имаме, че g(S)=g(W) то всички нормални снимки, които са били заедно с **S** и **W** при първото разбиване ще се разпръснат в различни групи при второто разбиване и еднозначно ще можем да намерим **S** и **W** без трето разбиване. Аналогично, същото се отнася и за случая p(S)=p(W).

Пример: N=9 S=1 W=3
Рунд 1: {1,2,3} , {4,5,6} , {7,8,9} , {1,4,7} , {2,5,8} , {3,6,9}
При второто разбиване 1 2 и 3 са в различни групи и съответно снимка 2 няма как да попадне пак в група със специална снимка.

Остава случаят, в който g(S)≠g(W) и p(S)≠p(W). Тук вече имаме проблем. Да разгледаме снимките **A** и **B** такива, че g(A)=g(S) p(A)=p(W) g(B)=p(S) и p(B)=p(S) (ако има такива, защото когато **N** не е точен квадрат не винаги има). При първото разбиване **A** е в групата на **S**, а **B** в групата на **W**. При второто разбиване **A** е в групата на **W** и **B** е в групата на **S**. Това е проблем, защото не установяваме, че **A** и **B** са нормални снимки. Хубавото е, че само **A** и **B** са проблемни (доказателство оставяме на читателя).

Пример: N=9 S=1 W=8 A=2 B=7
Рунд 1: {1,2,3} , {4,5,6} , {7,8,9} , {1,4,7} , {2,5,8} , {3,6,9}

Трябва третото разбиване да е такова, че 2 и 7 да попаднат в групи без специални снимки. Тук идва момент на експериментиране. Нека G(i,3)=(g(i)+p(i))%sqrt(N)+1. Да видим какво се случва с **A** и **B**. Няма как **A** да е в група с **S**, защото (g(A)+p(A))%sqrt(N)+1=(g(S)+p(W))%sqrt(N)+1 ≠ (g(S)+p(S))%sqrt(N)+1 (уговорихме се, че p(S)≠p(W)). Аналогично **A** няма как да е в група с **W** или **B** да е в група с някоя от специалните снимки, с което **A** и **B** изгарят и сме готови. Това ще хване около 65% за големите тестове.

i	1	2	3	4	5	6	7	8	9
G(i,3)	3	1	2	1	2	3	2	3	1

Пример: N=9 S=1 W=8
Рунд 1: {1,2,3} , {4,5,6} , {7,8,9} ,
{1,4,7} , {2,5,8} , {3,6,9} ,
{2,4,9} , {3,5,7} , {1,6,8}.

Единствената досадна подробност за решения с коренова декомпоцизия е, че при **N**=30**,** правят много въпроси. Тук можем да използваме 3 рунда, като с подобна идея разбием числата на 4 групи, после положителната група на 3 други и накрая положителната на още 3 (3*3*4>30). С това максимално оптимизираните решения с коренова ще хващат към 60%.

За любознателните 2. Случаят с **T=1** няма по-добро решение, но при **T=2** нещата стоят по съвсем различен начин. Оказа се, че има решение с китайската теорема за остатъците, което използва към 130 въпроса и е сравнително лесно за разбиране и писане (можете да погледнете **pesho.cpp**), но неговата коректност се доказва само опитно, вместо теоретично, пък и задачата беше достатъчно трудна и с 152 въпроса.

За любознателните 3. Може би се чудите защо беше цялата история с малките тестове. Причината е, че има рандомизирани решения, които се държат доста добре на големите и могат да бъдат спрени само на малките с адаптивен грейдър, който на първия рунд нагласяв възможно най-гадно снимките, в зависимост от въпросите, които е получил, и чак тогава отговаря. Забележете, че така решения с P=2 трудно могат да бъдат спрени, дори и рандомизирани. Ако някой е успял да чийтне задачата в крайна сметка, се надявам поне да не е било тривиално :) .

Автор: Александър Гатев
Решение с теория на числата: Петър Петров