Atcoder Grand Contest 018 解説

writer : maroonrk

平成 29 年 7 月 23 日

For International Readers: English editorial starts from page 6.

A : Getting Difference

A_1,A_2,...,A_N 全ての最大公約数を G とします。また、A_1,A_2,...,A_N の最大値を M とします。このとき、K の書かれたボールを箱に入れることができるための必要十分条件は、K が G で割り切れる M 以下の整数であることです。後はこれを証明します。どんな操作をしても、箱の中のボールに書かれた数の最大公約数が G のままであり、また箱の中のボールに書かれた数の最大値が M のままであることは簡単にわかります。よって、必要性は示されました。次に、十分性を説明します。二つのボールに書かれた差を取るという操作を繰り返すと、ユークリッドの互除法のような操作が行えます。そのため、ある二つのボールの最大公約数のボールを箱に入れることが可能になり、特に、すべての数の最大公約数である G の書かれたボールを箱に入れることができます。そのあとは、M と G の差を取り、それと G の差を取り、…と繰り返すことで、M,$M-G$,$M-2G$,...$2G$,G の書かれたボールはすべて箱に入ります。よって、K の書かれたボールを箱に入れることができました。

B : Sports Festival

まず、すべてのスポーツを実施すると仮定して、どのスポーツに何人の人が参加するか調べます。ここで、最も参加人数の多いスポーツがスポーツ P で、その参加人数が Q 人だとします。当り前ですが、この段階で、参加人数の最大値が Q の解が得られています。よって、あと考えるべきは、参加人数の最大値が Q 未満の解です。そして、そのような解において、スポーツ P は必ず実施されません。なので、Q 未満の解を求めるには、スポーツ P を候補から外した上で、もう一度同じ問題を解きなおせばよいです。これを再帰的に行っていけば、最終的な解を求めることができます。再帰するごとに候補の数が減るので、再帰は合計 M 回呼ばれます。誰がどのスポーツを選ぶのか愚直にシミュレーションすると $O(NM)$ かかるので、全体で $O(NM^2)$ でこの問題は解けました。なお、シミュレーションを工夫すると、全体で $O(NM)$ でこの問題を解くこともできます。

C : Coins

まず初めに、コインを持っている人を全員、持っている金のコインの枚数 – 持っている銀のコインの枚数 で昇順にソートし、左から右へ並ばせておきます。すると、最適解の一つにおいて、金

のコインをくれる全ての人は、銀のコインをくれる全ての人よりも右に立っていることがわかります。これは、そうでない場合、つまり、ある二人の人がいて、左に立っている方が金のコインを、右に立っている方が銀のコインをくれる場合、左から銀のコイン、右から金のコインをもらうように変えても損しないことからわかります。上の主張を言い換えると、ある整数 K が存在して、列の中で左の K 人のうち、Y 人から銀のコイン、$K-Y$ 人から銅のコインをもらい、列の中で右の $X+Y+Z-K$ 人のうち、X 人から金のコイン、$Y+Z-K$ 人から銅のコインをもらう、という形の最適解が存在することを言っています。よって、すべての K について、左側からもらうコインの最大枚数、右側からもらうコインの最大枚数を求めればよいです。ある K について、左側からもらうコインの枚数を最大化することを考えます。このとき、左側の人の中で、持っている銀のコインの枚数 – 持っている銅のコインの枚数 の大きい方から Y 人選び、彼らから銀のコインを、残りから銅のコインをもらうのが最適な戦略になります。K を小さい方から大きい方へと動かしてゆき、持っている銀のコインの枚数 – 持っている銅のコインの枚数 を優先度付きキューなどで管理すれば、K を一つ大きくする際に $O(log(X+Y+Z))$ で、左からもらうコインの最大枚数を更新できるので、全ての K について、合計で $O((X+Y+Z)log(X+Y+Z))$ で求めることができます。右側からもらうコインの最大枚数についても、同様にして求めることができます。よってこの問題は、$O((X+Y+Z)log(X+Y+Z))$ で解けました。

D : Tree and Hamilton Path

完全グラフ上での移動ではなく、木上で移動するものと考えます。まず、各辺について、その辺を通る回数の上限を考えます。辺 i で木を切った時に、残る二つの木のうち小さいものの頂点数を S_i とすると、辺 i を通る回数は、小さい方の木に入って出るという操作が高々 S_i 回しか行えないため、高々 $2\times S_i$ 回です。以下では、ここで求めた、$2\times S_i$ を、辺 i の上限と呼ぶことにします。ここで、2 通りの場合分けを行います。

$S_i = N/2$ となる辺がある場合。$S_i = N/2$ なる辺を辺 e と呼ぶことにします。辺 e を通る回数は、先ほど求めた上限では N 回ですが、そもそも、N 頂点のパスは当然 $N-1$ 回しか移動しないので、辺 e を通る回数は高々 $N-1$ 回です。ここで、辺 e で木を切ったあとに残る二つの木の頂点を、それぞれ、X_1,X_2,...,$X_{N/2}$、Y_1,Y_2,...,$Y_{N/2}$ と置きます。特に、辺 e が結ぶ二つの頂点が X_1 と $Y_{N/2}$ であるようにします。すると、$X_1 \to Y_1 \to X_2 \to Y_2 \to ...X_{N/2} \to Y_{N/2}$ というハミルトンパスを考えると、このパスに沿って移動した場合、辺 e 以外のすべての辺について、その辺を通る回数が上限に達しており、また辺 e を $N-1$ 回通っています。よってこれが最適な動き方だと分かり、最適解を求めることができます。

$S_i = N/2$ となる辺がない場合。木の重心（この場合、必ず一意に決定する）を G とおきます。G に接続する辺について、先ほど求めた通る回数の上限を足し合わせると $2(N-1)$ となりますが、これらの辺を通る回数の合計は、$2(N-1)$ 回にはなりません。なぜなら、G に接続する辺すべてを上限いっぱいの回数通ることにすると、G 以外の頂点からパスが始まることも終わることもできなくなり、明らかにありえないからです。よって、G に接続する辺のうち、少なくとも 1 本は、上限より少ない回数しか通りません。ここで、G に接続する辺のうち、最も短い辺を e とします。辺 e 以外の辺はすべて上限いっぱいの回数通り、辺 e だけは上限 -1 回通るような動き方があれば、これは明らかに最適な動きです。そして、このような動きは必ず可能です。木から G を取り除いた後に残る木の頂点数を、T_1,T_2,...,T_K(ここで、$T_1 \geq T_2 \geq ...T_K$) とおきます。木の重心の性質と、$S_i = N/2$ となる辺がない、という条件から、$T_1 \leq T_2 + T_3 + ... + T_K$ が成り立ちます。すると、

頂点の列であって、列の中で隣接する頂点間のパスに必ず G が含まれているようなものであって、G で始まり、好きな頂点で終わるものを作ることができます。特に、辺 e が結ぶ頂点のうち G でない方をパスの最後におくことができ、このようなパスは、辺 e 以外の辺を上限回、辺 e を上限 -1 回通っているので、最適な動きになります。

上で上げたどちらの場合も、木の重心を求めたあと、簡単な処理を施すだけで求めることができるので、この問題は $O(N)$ で解けました。

E : Sightseeing Plan

問題を要約すれば、次のようになります。

> 3 つの長方形領域が北西から南東に向かって並んでいる。それぞれの長方形領域から 1 マスずつ選び、選んだ 3 つのマスを通る最短経路の個数の総和を求めよ

上の問題にとりかかる前に、まず、次の問題を考えてみましょう。

> 整数 W, K が与えられる。マス $(1, 1)$ から出発し、マス (x, K) で終わる最短経路の数を、$1 \leq x \leq W$ を満たす全ての x について足し合わせた値を求めよ。

この問題の答えは、マス $(1, 1)$ からマス $(W, K + 1)$ へ向かう最短経路の個数になります。これは、二項係数をこねるか、グリッドをグッと睨むとわかります。

続いて、次の問題を考えてみましょう。

> 整数 W, H が与えられる。マス $(1, 1)$ から出発し、マス (x, y) で終わる最短経路の数を、$1 \leq x \leq W$,$1 \leq y \leq H$ を満たす全ての x, y について足し合わせた値を求めよ。

この問題の答えは、マス $(1, 1)$ からマス $(W + 1, H + 1)$ へ向かう最短経路の個数 -1 になります。これは、先ほどの問題の答えを使って、再びグリッドをグッと睨むとわかります。

最後に、次の問題を考えてみましょう。

> 整数 P, Q, R, S が与えられる。マス $(1, 1)$ から出発し、マス (x, y) で終わる最短経路の数を、$P \leq x \leq Q$,$R \leq y \leq S$ を満たす全ての x, y について足し合わせた値を求めよ。

この問題の答えは、

$$
\begin{aligned}
&+ \text{マス } (1, 1) \text{ からマス } (P, R) \text{ へ向かう最短経路の個数} \\
&- \text{マス } (1, 1) \text{ からマス } (P, S + 1) \text{ へ向かう最短経路の個数} \\
&- \text{マス } (1, 1) \text{ からマス } (Q + 1, R) \text{ へ向かう最短経路の個数} \\
&+ \text{マス } (1, 1) \text{ からマス } (Q + 1, S + 1) \text{ へ向かう最短経路の個数}
\end{aligned}
$$

になります。これは、二次元累積和と同じ考え方を用いると、先ほどの問題の答えから導けます。

以上の結果を用いると、元の問題において、3 つある長方形領域のうち真ん中以外の 2 つは、長方形ではなくて正負の重みのついた 4 つのマスと考えてもよいとわかります。よって、次のような問題が解ければよいです。

マス A、長方形領域、マス B が北西から南東に向かって並んでいる。マスからマスへの最短経路全てについて、通る長方形領域内のマスの数を重みとして足し合わせた値を求めよ。

ここで、次の問題を考えます。

長方形領域、マス A が北西から南東に向かって並んでいる。長方形領域の最も北西のマスからマス A への最短経路すべてについて、通る長方形領域内のマスの数を重みとして足し合わせた値を求めよ。

この問題は、長方形領域から出る位置を決め打ちすると、通る長方形領域内のマスの数が一定になっているので、長方形領域から出る位置を全通り試せば答えを求めることができます。適切な前処理を行っておけば、長方形領域から出る位置 1 つにつき $O(1)$ で答えが求まるので、全体で $O($座標の値$)$ で求まります。

この問題が解ければ、二次元累積和と同じ考え方を用いて、元の問題も同様にして解くことができます。これで、全体で $O($座標の値$)$ で答えを求めることができ、この問題は解けました。

F : Two Trees

ある整数列 X_1,X_2,...,X_N が条件を満たしていたとします。そして、頂点 v が木 1 では C_v 個の子を、木 2 では D_v 個の子を持っていたとします。条件が満たされている場合、$1 \equiv -1 \pmod 2$ なので、$X_v + C_v \equiv X_v + D_v \equiv 1 \pmod 2$ となります。ここで、明らかに、$C_v \equiv D_v \pmod 2$ がわかります。つまり、$C_v \equiv D_v \pmod 2$ が全ての頂点について成り立っていることが、条件を満たす整数列が存在することの必要条件になります。

次に、すべての頂点について $C_v \equiv D_v \pmod 2$ が満たされている場合、実際に条件を満たす整数列を作れることを示します。まず、X_v の偶奇を調べます。そして、$X_v \equiv 0 \pmod 2$ となる X_v については、$X_v = 0$ と決定してしまいます。次に、$X_v \equiv 0 \pmod 2$ となる X_v については、$X_v = 1, -1$ とすることにします。ここで、木 2 のことはいったん忘れて、木 1 において条件を満たすようにすることを考えます。N 頂点からなるグラフを考え、以下の条件を満たすようなマッチングを作ることを考えます。

- 木 1 の任意の部分木について、その部分木内に、$X_v = 1, -1$ となるような頂点が $2K + 1$ 個あった時に、そのうち $2K$ 個がマッチングを作る。

このようなマッチングは、DFS で求めることができます。こうしてできたマッチングには、次のような性質があります。

- 各ペア (u, v) について、$X_u = -X_v$ となっていれば、整数列 X_1,X_2,...,X_N は木 1 に関して条件を満たす。

これは、マッチングの性質から、どの部分木についてもペア同士で値を打ち消しあうことからわかります。今、木 1 に関して条件が満たされるようにしましたが、同じようなマッチングを木 2 についても求めてみます。そして、木 1 で求めたマッチングと木 2 で求めたマッチングを、グラフの辺だと思って一つのグラフにまとめてみます。こうして作られたグラフは、次の性質を持っています。

- 各辺 $e = (u, v)$ について、$X_u = -X_v$ となっていれば、整数列 X_1,X_2,...,X_N は木 1、木 2 両者に関して条件を満たす。

ここで、二つのマッチングを重ね合わせてできたグラフは、奇数長閉路を持つことができないことから、二部グラフです。よってこのグラフにおいて、各辺 $e = (u, v)$ について、$X_u = -X_v$ となる値の割り振りが可能です。辺を張る DFS や二部グラフに値を割り振る DFS は $O(N)$ で行えるので、この問題は $O(N)$ で解けました。

Atcoder Grand Contest 018 Editorial

writer : maroonrk

平成 29 年 7 月 23 日

A : Getting Difference

Let G be the greatest common divisor of A_1,A_2,...,A_N. Also, let M be the maximum of A_1,A_2,...,A_N.

When two integers x, y are both multiples of G, their absolute difference $|x - y|$ is also a multiple of G. Also, when $x, y \leq M$, $|x - y| \leq M$. Thus, all integers in the box is always a multiple of G, and at most M.

Next, let's prove that we can construct all such integers. Let X be the smallest integer in the box. If all integers in the box are multiples of X, it means that $X = G$. Otherwise, we can find an integer $Y = kX + Z$ that is not a multiple of X. Using this, we can construct $(k-1)X+Z, (k-2)X+Z, \ldots, Z$, so we got a new integer Z that is smaller than X. By repeating this process, we will eventually get G. Once we get G, we can construct $M, M-G, M-2G, \ldots, G$, and these are everything we want to get.

In summary, we shuold output "POSSIBLE" if and only if K is a multiple of G and $K \leq M$.

B : Sports Festival

First, suppose that we select all sports. In this case, let P be the sport that is chosen by the largest number of participants, and let this number Q. Obviously, the answer of the original problem is at most Q.

Can the answer be smaller than Q? In case we choose the sport P, at least Q people will choose this sport, no matter what subset of the remaining $M-1$ sports we choose. Thus, from now on, we can assume that we never choose the sport P. Recursively, we can solve a problem with N participants and $M-1$ sports.

Since each step of the recursion reduces the number of sports by one, the recursion terminates after M steps. In total, this solution works in $O(NM^2)$. If we implement this solution efficiently, it's also possible to solve this problem in $O(NM)$.

C : Coins

First, we sort the people in the increasing order of (the number of gold coins) minus (the number of silver coins), from left to right. That is, we assume that $A_1 - B_1 \leq A_2 - B_2 \leq \ldots$. In one optimal solution, all gold people (people who give Snuke gold coins) are to the right of all silver people. (Otherwise, if a gold person g is to the left of a silver person s, even if Snuke receives silver coins from g and gold coins from s, the total number of coins will not decrease.)

Therefore, there exists an integer K, such that

- Among the leftmost K people, there are Y silver people and $K - Y$ bronze people.
- Among the remaining people, there are X gold people and $Y + Z - K$ brone people.

For simplicity, let's assume that each person has exactly zero bronze coins. Otherwise, we can replace a person (A_i, B_i, C_i) with $(A_i - C_i, B_i - C_i, 0)$, and add C_i to the final answer. In this case, for a fixed K, we want to compute the sum of following values:

- The sum of Y largest integers among $B_1, \ldots, B_k$.
- The sum of X largest integers among $A_{K+1}, \ldots, A_{X+Y+Z}$.

It's not hard to compute these values for all K in $O((X + Y + Z)log(X + Y + Z))$ time with priority queues.

D : Tree and Hamilton Path

Instead of Hamiltonian Paths on the complete graph, imagine that we visit all vertices of the tree in some order. How many times do we pass through an edge i? If we cut the tree by the edge i, we get two connected components. Let S_i be the size of the smaller of those two components. It's easy to see that we can pass through this edge at most $2S_i$ times. Thus, the answer is at most $\sum 2S_iC_i$. Actually we can prove that the answer is always $\sum 2S_iC_i$ if we consider Hamiltonian Cycles, but in case of Haniltonian Paths, the answer is slightly smaller.

In this problem, centroids play an important role. It is well-known that there are two types of centroids:

Case I. There is a unique edge e such that $S_e = N/2$

In this case, $2S_e = N$, but obviously, we can pass through the edge e at most $N-1$ times. Thus, the answer is at most $\sum 2S_iC_i - C_v$.

On the other hand, we can achieve this value in the following way. Let X_1,X_2,...,$X_{N/2}$ be vertices in one component, and Y_1,Y_2,...,$Y_{N/2}$ be vertices in the other component. In particular, X_1 and $Y_{N/2}$ are the two endpoints of e. Then, the path $X_1 \to Y_1 \to X_2 \to Y_2 \to \cdots \to X_{N/2} \to Y_{N/2}$ achieves the bound.

Therefore, in this case, the answer is $\sum 2S_iC_i - C_v$.

Case II. There is a unique vertex G such that $S_G < N/2$

Suppose that G is incident to k edges $e_1, \ldots, e_k$. If we remove G from the tree, k connected components will appear: call them $T_1, \ldots, T_k$. Here T_i corresponds to e_i.

We know that $S_i = |T_i|$. If we want to pass through the edge e_i $2S_i$ times, we must start the path outside T_i and end the path outside T_i. It is impossible to satisfy this for all i: then, we must start/end the pass outside $T_1, \ldots, T_k$, but there's only one such vertex (G). Thus, the answer is at most $\sum 2S_iC_i - \min\{C_{e_1}, \ldots, C_{e_k}\}$.

Without loss of generality, we can assume that $\min\{C_{e_1}, \ldots, C_{e_k}\} = C_{e_1}$. We construct a path of length $\sum 2S_iC_i - C_{e_1}$. Let G, v be the endpoints of e_1. Here, we use the following fact (the proof is not very hard, an exercise for readers):

> There are $M \geq 2$ balls of various colors. In each color, there are at most $M/2$ balls. Then, we can always arrange balls in a circle such that no two balls of the same color are adjacent.

Because of this, we can arrange all $N-1$ vertices of the tree except for G in a circle, such that no two adjacent vertices are from the same T_i. We split this cycle next to v, and construct a sequence that ends with v, and append G to the beginning of the sequence. This sequence achieves the cost $\sum 2S_iC_i - C_{e_1}$.

Therefore, in this case, the answer is $\sum 2S_iC_i - \min\{C_{e_1}, \ldots, C_{e_k}\}$.

In both cases, this solution works in $O(N)$.

E : Sightseeing Plan

The problem asks the following:

> You are given three rectangular regions A, B, C. Choose a point from each region, and count the number of paths that passes through all chosen points. Compute the sum of these values.

First, we state some easy lemmas. Let $C(x, y) = \frac{(x+y)!}{x!y!}$ be the number of paths from $(0, 0)$ to (x, y).

> $\sum_{y=0}^{Y} C(X, y) = C(X + 1, Y)$

Consider a path from $(0, 0)$ to $(X+1, Y)$. It passes through an edge between (X, k) and $(X+1, k)$ for some k, and the number of such paths is $C(X, k)$.

> $\sum_{x=0}^{X} \sum_{y=0}^{Y} C(x, y) = C(X + 1, Y + 1) - 1$

Apply the previous lemma twice.

> $\sum_{x=X_1}^{X_2} \sum_{y=Y_1}^{Y_2} C(x, y) = C(X_1, Y_1) - C(X_1, Y_2 + 1) - C(X_2 + 1, Y_1) + C(X_2 + 1, Y_2 + 1)$

This is similar to a well-known trick where you compute the sum in a subrectangle by adding/subtracting rectangles four times.

In the original problem, we choose the start from a rectangle region. However, because of this lemma, we can assume that there are only four (weighted) possible positions for the start. In particular,

- $(X_1 - 1, Y_1 - 1)$ with weight $+1$.
- $(X_1 - 1, Y_2)$ with weight -1.
- $(X_2, Y_1 - 1)$ with weight -1.
- (X_2, Y_2) with weight $+1$.

Similarly, the goal is also considered as four points, so we can solve the original problem by solving the following simpler problem $4 \times 4 = 16$ times:

> You are given a point A, a rectangular region B, and a point C. Choose a point from B, and count the number of paths that passes through all chosen points. Compute the sum of these values.

We should count the number of paths from A to C that passes through B at least once. However, if the size of the intersection of a path P and the rectangle B is k, this path should be counted with multiplicity k.

The following works:

- For each $X_3 \leq x \leq X_4$, count the number of paths that enters B by $(x, Y_3 - 1) \to (x, Y_3)$ with multiplicity $-(x + Y_3)$.
- For each $Y_3 \leq y \leq Y_4$, count the number of paths that enters B by $(X_3 - 1, y) \to (X_3, y)$ with multiplicity $-(X_3 + y)$.
- For each $X_3 \leq x \leq X_4$, count the number of paths that leaves B by $(x, Y_4) \to (x, Y_4 + 1)$ with multiplicity $x + Y_4 + 1$.
- For each $Y_3 \leq y \leq Y_4$, count the number of paths that leaves B by $(X_4, y) \to (X_4 + 1, y)$ with multiplicity $X_4 + y + 1$.

In total, this solution works in $O(MAXCOORDINATE)$, with pre-computation of factorials and inverses of factorials.

F : Two Trees

Since $1 \equiv -1 \pmod 2$, we know the parity of the sum of each subtree. Suppose that a vertex v has C_v children in the first tree. Then, X_v must be even if C_v is odd, and X_v must be odd if C_v is even. Similarly, if a vertex v has D_v children in the first tree, X_v must be even if D_v is odd, and X_v must be odd if D_v is even.

If $C_v \equiv D_v \pmod 2$ for all v, the answer is always "POSSIBLE", as we describe later. Otherwise, the answer is "IMPOSSIBLE".

Author's solution

Construct a new graph with $2N + 1$ vertices in the following way:

- The vertices are named $P_1, P_2, \ldots, P_N, Q_1, Q_2, \ldots, Q_N, R$.
- If there is an edge between i and j in the first tree, add an edge between P_i and P_j.
- If there is an edge between i and j in the second tree, add an edge between Q_i and Q_j.
- If p is the root of the first tree, add an edge between P_p and R.
- If q is the root of the first tree, add an edge between Q_q and R.
- If X_v is odd, add an edge between P_v and Q_v.

This graph is connected and the degree of each vertex is even, so it has a Eulerian Cycle. Find one Eulerian Cycle, and direct it in an arbitrary way. Define $X_v = 0$ if X_v is even, $X_v = 1$ if the Eulerian Cycle is directed from P_v to Q_v, and $X_v = -1$ if the cycle is directed in the opposite way. It's easy to see that this satisfies the constraints.

Admin's solution

As we see, we know the parity of X_v. If $X_v \equiv 0 \pmod 2$, we decide $X_v = 0$, and call it "even variable". Otherwise, we decide that X_v is either 1 or -1, and call it "odd variable".

Now, let's forget about the second tree for a while, and try to satisfy the condition in the first tree. We want to find some disjoint pairs of odd variables with the following property:

- For each subtree T of the first tree, if T contains $2K + 1$ odd variables, there are K pairs among those odd variables.

Such pairs can be easily found by a DFS.

Then, for each pair (u, v), let's force a constraint that $X_u = -X_v$. If all pairs satisfy this condition, obviously, the sum in each subtree will be either 1 or -1.

Now, let's construct a graph with N vertices $1, 2, \ldots, N$. If u and v are paired in the first tree, add a red edge between vertices u and v. Since the pairs are disjoint, the red edges form a matching of the graph. Similarly, do the same for the second tree, and add a blue edge between two vertices paired in the second tree. Now this graph has two matchings: red and blue.

Consider a cycle in this graph. Since a red edge and a blue edge must appear alternately in this cycle, this graph doesn't contain odd cycles, so it is bipartite. This means that we can satisfy constraints $X_u = -X_v$ for both red and blue edges at the same time. This is what we want to get.