ARC100 / ABC102 解説

writer : maroonrk

平成 30 年 7 月 1 日

For International Readers: English editorial starts from page 8.

A : Multiple of 2 and N

N が偶数のとき、答えは N です。また、N が奇数のとき、答えは $2N$ です。
C++による解答コード

B： Maximum Difference

数列の最大値を最小値を求め、その差を答えればよいです。最大値と最小値が一致する場合、添字が異なる、という条件を無視してしまうように思えますが、最大値と最小値が一致するということは、すべての要素が等しいので、問題ありません。

C++による解答コード

C : Linear Approximation

$B_i = A_i - i$ と定義すれば、問題は、自由に整数 b を選び、$abs(B_i - b)$ の総和を最小化する問題です。ここで、b を B_i の中央値にするのが最適であることがわかります。中央値でない場合は、中央値に近づけることで損をしないことからこれはわかります。よって、B_i をソートして中央値を求め、実際に答えを計算すればよいです。ソートがボトルネックとなり、この問題は $O(NlogN)$ で解けます。

D : Equal Cut

予め累積和を求めておき、ある区間の和を $O(1)$ で求められるようにしておきます。

切り込みを入れる 3 箇所のうち、真ん中の切れ込みを 1 つ固定したとします。真ん中の切れ込みで切ったあとの 2 つの数列を L,R とします。あとは、L と R をそれぞれ 1 回ずつ切ることを考えます。

L の切り方を考えると、切ったあとの 2 つの数列の要素の総和ができるだけ近いほうが良いことがわかります。よって、真ん中の切れ込みを固定すると、左の切れ込みの位置を決めることが出来ます。真ん中の切れ込みの位置が、i 番目の要素と $i+1$ 番目の要素の間の時、最適な左の切れ込みの位置を $F(i)$ とおきます。このとき、$F(i)$ は i に関して単調増加であり、尺取り方を用ることで $F(i)$ をすべての i に対して $O(N)$ で求めることが出来ます。

右の切れ込みについても、同様にして求めることが出来ます。これで、すべての真ん中の切れ込みの位置に対して、残りの 2 つの切れ込みの位置を求めることが出来ました。あとは、実際にそれぞれの場合に対して答えを計算すれば、答えが求まります。

よって、$O(N)$ でこの問題は解けました。

E : Or Plus Max

各 x について、$max\ A_i + A_j\ s.t.\ (i\ or\ j) = x$ が求められれば、あとは単に累積 max を取れば良いので、この問題は簡単です。しかし、これを求めるのは難しいです。そこで代わりに、各 x について、$max\ A_i + A_j\ s.t.\ (i\ or\ j) \subseteq x$ を求めます。ただしここで便宜的に、$a \subseteq b$ を、a を bit で表したときの 1 の位置が、b を bit で表したときの 1 の位置の部分集合となっている、の意味で使います。これを求めたあと累積 max を求めても、正しい答えは得られます。($(i\ or\ j) \subseteq x \rightarrow (i\ or\ j) \leq x$ なので)

これを求めるためには、各 x について、$i \subseteq x$ なる i について A_i を列挙したときの上位 2 つが求まれば良いです。逆から見ると、各 i について、$i \subseteq x$ なるすべての x について、A_i で上位 2 つを更新する、という操作ができれば良いです。これは、高速ゼータ変換の要領で、高速に行うことができます。よってこの問題は、$O(N2^N)$ で解くことが出来ます。

F : Colorful Sequences

カラフルとは限らない、長さ N で、各要素が 1 以上 K 以下の整数列すべてについて、A を含む個数の総和を求めるのは簡単です。A が登場する位置が $N-M+1$ 通り考えられ、それを一つ固定すると、残りの $N-M$ 要素を自由に決定できるので、その位置に A を含む整数列の総数は K^{N-M} 個になります。

あとは、カラフルでない長さ N の整数列すべてについて A を含む個数を数える問題を解きます。これが解ければ、引き算をして答えが求まります。

まず、そもそも A がカラフルであった場合、答えは 0 です。

残りのパターンを考える前に、次の問題を解いておきます。

> 長さ N のカラフルでない整数列の個数を求めよ。

DP をします。$dp[i][j]$ = 長さ i で、最後の j 個の要素はすべて異なるが、最後の $(j+1)$ 個の要素の中には重複があるような整数列の個数 と定めます。この DP の遷移を考えます。$dp[i][j]$ から $dp[i+1][j']$ への遷移にかかる係数は、以下のようになります。

- $j+1 < j'$ のとき: 0
- $j+1 = j'$ のとき: $K-j$
- $j+1 > j'$ のとき: 1

最初のパターンは簡単です。どんな要素を追加しても、最後の $(j+2)$ 個の中に必ず重複は存在するので、このような遷移はありません。次のパターンは、元の数列の最後 j 個以外の要素を追加することで起こる遷移です。追加する要素の候補は $K-j$ 個なので、この遷移になります。最後のパターンは、j' の値を決めると、追加する要素が一意に定まるので、この遷移になります。

この DP を愚直に書くと、状態数 $O(NK)$、遷移 (K) で合計 $O(NK^2)$ となり間に合いませんが、累積和を用いると遷移を高速化でき、$O(NK)$ で解くことが出来ます。

この DP の結果を用いて、A がカラフルでない場合を解きます。

まず、A 内に重複する要素がない場合を考えます。ここで、次の問題を考えてみます。

> 長さ N のカラフルでない整数列すべてについて、長さ M 個の連続する部分列であって、重複する値を含まないものの個数を数える。

この問題では、A の要素には一切関知せず、M だけを考えている点に注意してください。この問題は、カラフルでない数列を数える上記の DP を変形して（長さ M の重複を含まない連続部分列がすでに登場したか、という 0,1 の次元を足す）答えを求めることが出来ます。長さ M で、各要素が 1 以上 K 以下の、重複する要素を含まない整数列は、$N!/(N-M)!$ 通りありますが、この問題において、これら全ては同じ回数だけ数え上げられます。よって、この問題の答えを $N!/(N-M)!$ で割れば、A を含む個数の総和が求まることになります。

最後に、A 内に重複する要素がある場合を考えます。A 内に重複する要素があるので、次の条件を満たす整数 F,B を定義することが出来ます。

- A の先頭 F 個の要素はすべて異なるが、A の先頭 $F+1$ 個の要素の中には重複がある
- A の末尾 B 個の要素はすべて異なるが、A の末尾 $B+1$ 個の要素の中には重複がある

すべての $i(0 \leq i \leq N-M)$ について、A の先頭に i 個の要素を足し、A の末尾に $N-M-i$ 個の要素を足して、カラフルでない整数列を作る方法が何通りあるかを求めます。ここで、長さ $i+F$ で、最後の F 要素がすべて異なるカラフルでない数列の個数、を求めることが出来ます。これは単に、上記の DP の結果を利用すれば良いです。長さ F で、各要素が 1 以上 K 以下の、重複する要素を含まない整数列は、$N!/(N-F)!$ 通りありますが、これらはすべて、先程数えた数列の末尾 F 個として均等に登場します。よって、割り算をすることで、長さが $i+F$ で、最後の F 要素が A の先頭 F 個と一致するカラフルでない数列の個数が求まります。同様にして、長さが $B+N-M-i$ で、先頭の B 要素が A の末尾 B 個と一致するカラフルでない数列の個数が求まります。この 2 つの通り数を掛け合わせれば、ある i に対する答えを求められます。ここで数えた先頭、末尾に足される数列の組合せが必要十分であることは、簡単に確かめられます。

これで、すべてのパターンを数え上げることが出来ました。最初の DP がボトルネックとなり、$O(NK)$ でこの問題は解くことが出来ます。

Atcoder Regular Contest Editorial

writer : maroonrk

July 1, 2018

A : Multiple of 2 and N

If N is even, the answer is N; otherwise the answer is $2N$.

C++ solution

B : Maximum Difference

Print max minus min.

C++ solution

C : Linear Approximation

Define $B_i = A_i - i$. Then, we want to minimize the value of the sum of $abs(B_i - b)$ by choosing a variable b.

It turns out that the optimal value of b is the median of B_i. This is because, if b is not the median, if we continuously change b to the direction of the median, the sum of $abs(B_i - b)$ never increases.

We can get the median by sorting the array B. This solution works in $O(NlogN)$ time.

D : Equal Cut

Let's pre-compute prefix sums (and now we can get interval sums in $O(1)$.

We need to make three cuts. Let's fix the position of the second (the middle) cut. Now we have two parts, and we need to make one more cut for each of the two parts.

Let L, R be the sum of all elements in left and right parts, respectively. Let L_1, L_2 be the sum of all elements in the two smaller parts we get from the left part (define R_1, R_2 similarly). How should we choose L_1, L_2?

Notice that we should always make a cut that minimizes the value of $|L_1 - L_2|$, regardless of the value of R_1 and R_2. No matter what the values of R_1 and R_2, this choice always minimizes $max\{L_1, L_2, R_1, R_2\}$ and maximizes $mub\{L_1, L_2, R_1, R_2\}$. (This is because the average of L_1 and L_2 is always $L/2$. The value of $max\{L_1, L_2\}$ becomes the closest to $L/2$ (and the smallest) when $|L_1 - L_2|$ is minimized. The same observation holds for the min.)

Thus, once we decide the position of the second cut, we can uniquely determine the positions of the first cut and the third cut. Let $F(i)$ be the optimal position of the first cut when the second cut is made at position i. Since $F(i)$ is monotonously increasing, we can compute this function by using two-pointers. Similarly, we can compute the position of the third cut.

This solution works in $O(N)$ time.

E : Or Plus Max

In this editorial, for two integers a and b, "$a \subseteq b$" means that the set of positions of ones in the binary representation of a is a subset of that of b.

Ideally, for each x, we want to compute $max\ A_i + A_j\ s.t.\ (i\ or\ j) = x$. Then, we can compute the solutions as prefix maximums of this array. However, this is difficult.

Instead, for each x, compute $max\ A_i + A_j\ s.t.\ (i\ or\ j) \subseteq x$. Notice that the prefix maximums of this array give the same results because $(i\ or\ j) \subseteq x \rightarrow (i\ or\ j) \leq x$.

To compute this, for each x, we want to get the two largest values of A_i such that $i \subseteq x$.

Let's use an algorithm similar to Fast Zeta Transform. Imagine a set S_x, the (multi)set of all A_i such that $i \subseteq x$. We are only interested in the largest two elements of S_x.

- For each k from 0 to $N-1$,
  - For each x from 0 to 2^{N-1} such that the k-th bit of x is zero,
    - $S_{x|(1<<k)} := S_{x|(1<<k)} \cup S_x$
    - If $S_{x|(1<<k)}$ contains more than two elements, discard small elements while it has more than two elements.

This solution works in $O(N2^N)$ time.

F : Colorful Sequences

It's easy to compute the total number of occurrences of A in (not necessarily colorful) all sequences of length N: since there are $N - M + 1$ possibilities for the position of A and you can freely choose the remaining $N - M$ elements, this is $(N - M + 1)K^{N-M}$.

From now on, let's compute the total number of occurrences of A in non-colorful sequences of length N. The answer to the original problem is $(N - M + 1)K^{N-M}$ minus this number.

As a warm-up, consider the following problem:

> Count the number of non-coloful sequences of length N.

Define $dp[i][j]$ as the number of non-colorful sequences of length i, such that the last j elements are pairwise distinct but the last $j+1$ elements are not. The transition coefficients from $dp[i][j]$ to $dp[i+1][j']$ is as follows:

- In case $j + 1 < j'$, 0.
- In case $j + 1 = j'$, $K - j$.
- In case $j + 1 > j'$, 1.

This is $O(NK^2)$ with straightforward implementation, but with prefix sums we can do it in $O(NK)$.

Let's return to the problem. Consider three cases:

Case 1. A is colorful.

Obviously, the answer is 0.

Case 2. All elements in A are pairwise distinct. A is not colorful.

Consider the following problem:

> For each non-colorful sequence of length N, count the number of contiguous subsequences of length M whose elements are pairwise distinct. Find the sum of these counts.

Note that the elements in A are irrelevant to this problem: we only care the value of M. This problem can be solved in the almost same way as the DP solution in warm-up problem. Notice that each of $K!/(K - M)!$ sequences of length M occur in all non-colorful sequences the same number of times. Thus, the solution to the original problem is the solution to this problem, divided by $K!/(K - M)!$.

Case 3. There are duplicated elements in A. A is not colorful.

Define F, B as follows:

- The first F elements of A are pairwise distinct, but the first $F+1$ elements are not.
- The last B elements of A are pairwise distinct, but the last $B+1$ elements are not.

For each $i (0 \leq i \leq N-M)$, let's compute the number of ways to make a non-colorful sequence by attaching i elements to the beginning of A and $N-M-i$ to the end of A (the answer is the sum of these values). Notice that we can handle two directions independently. That is, this is the product of (the number of ways to make a non-colorful sequence (of length $M+i$) by attaching i elements to the beginning of A) and (the number of ways to make a non-colorful sequence by attaching $N-M-i$ elements to the end of A). These values can be computed in the same way as the warm-up problem.

For all three cases, this solution works in $O(NK)$ time.