《硬币游戏》解题报告

成都市第七中学 杨宁远

2022.10.07

1 题目大意

小A现在有n个硬币排成一行，最开始都是正面朝下，并且从左到右依次编号为$1, 2, 3, \ldots, n-1, n$。

小A决定玩一个游戏：小A会先进行k次操作，每次选定两个正整数s, t满足$1 \leq s < t \leq n$然后翻转s号和t号硬币。

在k次操作完成后，小A会选定一个值$p(0 \leq p \leq n)$。我们假设左边p个硬币(编号为$1 \sim p$)中朝上的硬币有a个，右边$n-p$个(编号为$p+1 \sim n$)中朝上的有b个。

除此之外，小A还有一个美观度函数H，以及两个常数x, y。小A一局游戏的分数被定义为$val = x^p y^{n-p} H(a)$。现在小A想求出所有游戏的分数和。

当然如果仅仅是求出所有游戏的分数和有点太无趣了，因此小A还规定了a, b的具体值。我们定义$F(x, y)$为所有满足$a = x, b = y$的不同游戏的分数之和。两个游戏不同当且仅当某次操作的(s, t)不同或者最后选择的p不同。

小A并不仅仅满足于求出$F(x, y)$。他又突发奇想，给了两个数ra, rb，想让你求出$ans_k = \sum_{i=0}^{ra} F(i, rb)$，其中$k$表示进行了$k$次操作。

小A除了是一个游戏爱好者之外，还是一个精益求精，奋发向上的人。因此他会要求你求出$ans_0, ans_1, ans_2, \ldots, ans_{m-1}, ans_m$。

答案对998244353取模。

2 数据范围

对于所有数据，保证$1 \leq n, x, y, H(i) < 998244353, 0 \leq ra, rb, m \leq 10^5, ra + rb \leq n$。

子任务编号	特殊性质	分值
1	$n, m \leq 5000$	10
2	$ra, rb, m \leq 5000$	20
3	$ra, rb \leq 5000$,性质A	20
4	性质A	10
5	$ra, rb, m \leq 5 \times 10^4$	15
6	无	25

性质A：$H(x)$只有一个位置非零。

每个子任务可能会依赖所有数据范围为其严格子集的子任务。

时空限制：1.5s, 512MB

2.1 解题思路

2.1.1 算法一

我们首先发现如果我们知道游戏的最终结果，那么过程的方案数只与有多少个硬币朝上有关。

我们先考虑如何求出方案数。定义$dp_{i,j}$表示操作了i步，当前有j个正面。那么我们枚举每次翻转时翻转了多少个朝下的，可以得到一个显然的转移：

$$dp_{i,j} = \binom{n-j+2}{2} dp_{i-1,j-2} + (n-j)j dp_{i-1,j} + \binom{j+2}{2} dp_{i-1,j+2}$$

接下来我们计算出所有不同的满足$a = i, b = rb$的最终结果的分数和

$$V(i, rb) = \sum_{p=0}^{n} \binom{p}{i} \binom{n-p}{rb} x^p y^{n-p}$$

那么最终答案就是 $\sum_{i=0}^{ra} \frac{1}{\binom{n}{i+rb}} dp_{k,i} V(i, rb) H(i)$。

时间复杂度：$O(n \times ra + nm + m^2)$。期望得分：10。

2.1.2 算法二

我们还是沿用算法一的思路，即利用方案数只与朝上硬币数有关的性质，将解题步骤拆为计算出所有$V(i, rb)$以及计算$dp_{i,j}$。

我们现在考虑找到一个复杂度与n无关的算法。由于第二步复杂度为m^2，因此我们只需优化第一步的做法。

我们发现V函数有一个奇妙的性质：

$$(n-i-j)V(i,j) = (i+1)V(i+1,j) + (j+1)V(i,j+1)$$

我们可以通过组合意义得到这个式子。$V(i,j)$表示对于所有分界点情况，在左边有i个朝上，右边j个的结果分数之和。现在我们考虑增加一个朝上的硬币，并标记为红色。那么我们有$(n-i-j)$种选择这个硬币的方式，也就是说现在总共有$(n-i-j)V(i,j)$种情形。

而显然，这枚硬币一定在左边或右边，因此我们也可以通过$(i+1)V(i+1,j)$表示红色硬币在左边的情形，$(j+1)V(i,j+1)$表示在右边的情形，加起来就可以得到所有方案。因此等式得证。

通过这个等式，我们可以得出一个递推式：$V(i,j+1) = \frac{(n-i-j)V(i,j) - (i+1)V(i+1,j)}{j+1}$。

接下来我们考虑如何计算$V(i,0)$。注意到对于$i>0$，有：

$$\begin{aligned}V(i,0)&=\sum_{p=0}^{n}\binom{p}{i}x^p y^{n-p}\\&=\sum_{p=1}^{n}(\binom{p-1}{i}+\binom{p-1}{i-1})x^p y^{n-p}\\&=\frac{x}{y}(V(i,0)-\binom{n}{i}\cdot x^n+V(i-1,0)-\binom{n}{i-1}\cdot x^n)\end{aligned}$$

有了这两个结论之后，我们就可以得出一个$O((ra+rb)^2+m^2)$的做法。即我们先利用上式计算出$\forall 0\le i\le ra+rb$的$V(i,0)$，然后通过递推算出所有剩下的$V(i,j)$。

注意当 $x=y$ 时需要特判，此时 $V(i,0)=\sum_{p=0}^{n}\binom{p}{i}x^n=\binom{n+1}{i+1}x^n$。

时间复杂度：$O((ra+rb)^2+m^2)$。期望得分：30。

2.1.3 算法三

我们考虑如何优化第二步，即计算$dp_{i,j}$。

我们先去掉s,t的大小限制，于是现在问题等价于$2k$次操作，每次选一个s，翻转s号硬币。记新问题中有i个朝上的方案数为$S_{k,i}$。我们可以利用生成函数来表示这个问题，写出方案数关于k的EGF：

$$pre_k=[\frac{x^{2k}}{2k!}](\frac{e^x+e^{-x}}{2})^{n-i}(\frac{e^x-e^{-x}}{2})^i$$

那么利用多项式快速幂，我们就可以对于一个固定的j得到所有$S_{i,j}$。我们也可以通过多项式乘法，在$O(m(ra+rb)\log)$的复杂度下求出所有$S_{i,j}$。

考虑利用容斥来得到原问题的答案。我们首先需要减去所有$s=t$，也就是新问题中第$2i-1$步与$2i$步选中的硬币相同的方案数。我们可以枚举有多少对这种情形，得到递推式：

$$dp_{i,j}=S_{i,j}-\sum_{k=0}^{i-1}dp_{k,j}\binom{i}{k}n^{i-k}$$

注意，容斥后由于我们并没有要求$s<t$，而只是要求$s\neq t$，因此最终$dp_{i,j}$需要除掉2^i。

当然，我们也可以求出乘上H后的式子后再进行容斥。

直接容斥是$O(m^2)$的，但是我们可以通过分治FFT在$O(m\log^2)$的复杂度内解决或多项式求逆在$O(m\log)$的复杂度内解决，具体见算法五的后半部分。

时间复杂度：$O((ra+rb)^2+m\log)$。可以通过子任务3。

结合算法二可以通过子任务1,2,3，有50分。

2.1.4 算法四

现在的瓶颈变成了第一步。我们考虑如何快速求出所有$V(i,rb)$。

我们观察一下我们上面的那个递推式中所有$V(i,0)$对于$V(x,y)$的贡献($i \le x+y$且$x \le i$)。发现系数为：

$$\frac{(-1)^{i-x}(n-i)^{\underline{x+y-i}}i^{\underline{i-x}}\binom{y}{i-x}}{y!}$$

于是，在知道$V(i,0)$的情况下，我们可以利用多项式乘法对于固定的y得到所有$V(x,y)$，此处不再赘述。

于是第一步的时间复杂度降为$O((ra+rb)\log)$。

结合前面的算法可以拿到60分。

2.1.5 算法五

我们现在的瓶颈在于第二步。即，我们需要快速求出如下式子：

$$\sum_{i=0}^{ra}(\frac{e^x+e^{-x}}{2})^{n-i-rb}(\frac{e^x-e^{-x}}{2})^{i+rb}H(i)V(i,rb)$$

我们将后面的$H(i)V(i,rb)$记为$C(i)$。也就是求出以下的式子：

$$(\frac{e^x+e^{-x}}{2})^{n-ra-rb}(\frac{e^x-e^{-x}}{2})^{rb}(\sum_{i=0}^{ra}(\frac{e^x+e^{-x}}{2})^{ra-i}(\frac{e^x-e^{-x}}{2})^{i}C(i))$$

我们利用多项式快速幂可以轻松得到前面的$(\frac{e^x+e^{-x}}{2})^{n-ra-rb}(\frac{e^x-e^{-x}}{2})^{rb}$。问题在于后面$\sum_{i=0}^{ra}(\frac{e^x+e^{-x}}{2})^{ra-i}(\frac{e^x-e^{-x}}{2})^{i}C(i)$的求法。

我们令$y = e^x$，现在问题变成了求出$\frac{1}{y^{ra}}\sum_{i=0}^{ra}(\frac{y^2+1}{2})^{ra-i}(\frac{y^2-1}{2})^{i}C(i)$。我们可以利用分治FFT来解决，此处不再赘述。

我们得到了一个关于y的多项式$G(y)$，我们需要还原得到x的多项式$F(x)$。

假设$G(y)=\sum_i a_i y^{p_i}$，同时定义新的多项式$H(x)$满足$[x^i]H(x)=i![x^i]F(x)$。那么有：

$$i![x^i]F(x)=[x^i]H(x)=\sum_j a_j p_j^i$$

也就是说：

$$H(x)=\sum_i \frac{a_i}{1-p_i x}$$

同样地，我们可以利用分治FFT同时维护分子与分母来解决。此处同样不再赘述。

我们称容斥之前的答案为pre_i。最后容斥时会有

$$ans_i = pre_i - \sum_{k=0}^{i-1} ans_k \binom{i}{k} n^{i-k}$$

$$\sum_{k=0}^{i} \frac{ans_k}{k!(i-k)!n^k} = \frac{pre_i}{i!n^i}$$

那么我们定义$RF(x) = \sum_i \frac{ans_i}{i!}x^i, RG(x) = \sum_i \frac{1}{i!}x^i, RH(x) = \sum_i \frac{pre_i}{i!}x^i$。于是有

$$RF(x)RG(x) = RH(x)$$

$$RF(x) = RH(x)RG^{-1}(x)$$

利用多项式求逆即可解决。时间复杂度：$O((ra+rb)\log^2 + m\log)$。

注意直接实现常数可能会略大，但是我们发现很多地方多项式都只有偶数处有值，因此可以将常数减半。

期望得分：100分。

3 参考文献

[1]https://qoj.ac/contest/845/problem/2834