新试题 解题报告

雅礼中学 刘剑成

1 试题大意

有一场比赛，如果你能在这场比赛中获得第i名，那么你可以得到p_i的奖金。

现在你已经提前了解了其他$n-1$名选手与你自己的各项能力，并且根据他们的能力分别计算出了他们得分的概率。这场考试的满分为1分，最低为0分，分数可以为任意小数。对于第i个人，他得x分的概率，与t_i次函数$f_i(x)$成正比。

如果一个人的函数为$f(x)=2$，那么他获得任意分数的概率都是相等的；如果一个人的函数为$f(x) = 2-x$，那么他获得越高的分的概率就越低，且他获得0分的概率是获得1分的概率两倍。

现在你需要计算你能在这场比赛中期望能得到多少的奖金。由于分数可以为小数，所以无需考虑排名相等的情况。

答案对998244353（$7\times 17\times 2^{23}+1$，一个质数）取模。

时间限制：清澄 8sec/UOJ 6sec

空间限制：64MB

2 输入格式

输入共$n+2$行。

第一行包含一个整数n。

在第二行有n个整数，第i个数代表p_i。

接下来$n-1$行，每行第一个数为t_i，代表第i个人所对应的函数是一个t_i-1次函数，接下来t_i个实数，第j个数代表该函数中x_j-1项的系数。

最后一行，第一个数为t_n，代表你所对应的函数是一个t_n-1次函数，接下来t_n个实数，第j个数代表该函数中x_j-1项的系数。

3 输出格式

输出1行，包含一个整数，表示你期望能获得的奖金。

4 数据范围

<table>
<tr><th>编号</th><th>n</th><th>$S=\sum t_i$</th><th>特殊限制</th></tr>
<tr><td>1</td><td rowspan="2">2</td><td>5</td><td rowspan="10">无</td></tr>
<tr><td>2</td><td>10</td></tr>
<tr><td>3</td><td rowspan="4">10</td><td>40</td></tr>
<tr><td>4</td><td>60</td></tr>
<tr><td>5</td><td>80</td></tr>
<tr><td>6</td><td>100</td></tr>
<tr><td>7</td><td rowspan="4">100</td><td>400</td></tr>
<tr><td>8</td><td>600</td></tr>
<tr><td>9</td><td>800</td></tr>
<tr><td>10</td><td>1000</td></tr>
<tr><td>11</td><td rowspan="10">500</td><td rowspan="4">4000</td><td>所有名次的奖金相等</td></tr>
<tr><td>12</td><td>所有人的函数相同</td></tr>
<tr><td>13</td><td rowspan="2">除了第一名与最后一名其他人的奖金都相等</td></tr>
<tr><td>14</td></tr>
<tr><td>15</td><td>1500</td><td rowspan="6">无</td></tr>
<tr><td>16</td><td>2000</td></tr>
<tr><td>17</td><td>2500</td></tr>
<tr><td>18</td><td>3000</td></tr>
<tr><td>19</td><td>3500</td></tr>
<tr><td>20</td><td>4000</td></tr>
</table>

保证所有人的函数在$[0,1]$的范围以内大于等于0，输入的所有实数仅有2位小数，且除了常数项以外的系数绝对值均小于5，常数项的绝对值小于30。

所有数据均为随机生成。

p_i的范围在0到10000之间。

5 算法介绍

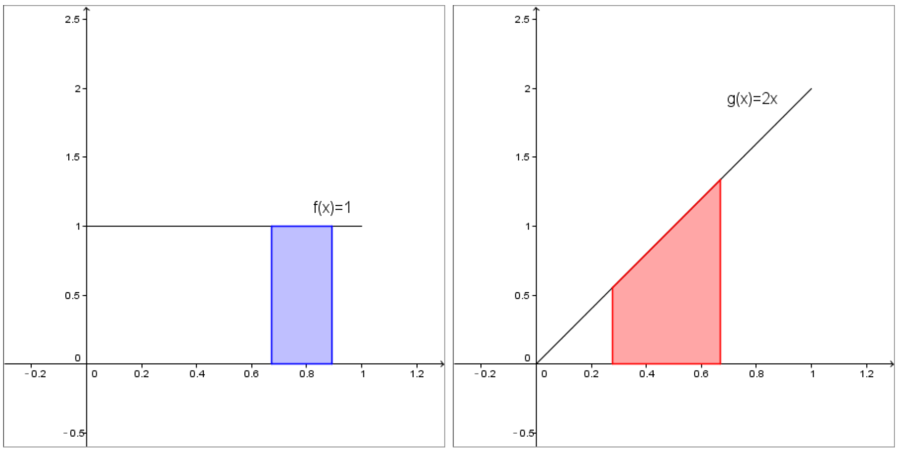
5.1 算法1

在10%的数据中，只有2个人，即我们只需要求出第一个人有多大概率在第二个人以下，再乘上获得的奖金即可。

对于第i个人，他的分数落在$[l,r]$以内的概率$P_i(l,r)$为：

$$P_i(l,r) = \frac{\int_l^r f_i(x)\,dx}{\int_0^1 f_i(x)\,dx} \tag{1}$$

直观上看，该结果就是函数在$[l,r]$这一段中与x轴所构成区域的面积，除以函数在$[0,1]$中与x轴所构成区域的面积。即函数在$[l,r]$这一段的定积分除以在$[0,1]$这一段的定积分，如下图所示：



染色区域的面积除以函数在$[0,1]$的总面积即为分数落在左右边界内部的概率

由于函数乘上一个常数对答案并没有任何影响，则我们可以将所有的函数除以它在$[0,1]$中与x轴所构成区域的面积，则有：

$$P_i(l,r) = \int_l^r f_i(x)\,dx \tag{2}$$

假设答案不需要对998244353取模，考虑一种并不完美的做法：

确定一个很小的距离Δd，将每Δd的距离分为一段，枚举你所获得的分数落在哪一段内，通过计算另一个人的分数在你之下的概率，求和即为你超过另一

个人的概率P：

$$P \approx \sum_{t=\Delta d}^{1} P_1(0, t-\Delta d) P_2(t-\Delta d, t) \tag{3}$$

$$= \sum_{t=\Delta d}^{1} \int_0^{t-\Delta d} f_1(x)\,dx \int_{t-\Delta d}^{t} f_2(x)\,dx \tag{4}$$

可以发现，当Δd取到无限接近于0的时候，$\int_{t-\Delta d}^{t} f_2(x)\,dx$等于$f_2(x)$的不定积分在$t$处的导数，即为$f_2(t)$。

记$f_i(x)$的原函数中，满足$F_i(0) = 0$的原函数为$F_i(x)$，则有：

$$P = \int_0^1 F_1(x) f_2(x) dx \tag{5}$$

由于本题中函数均为多项式函数，则对于函数：

$$f(x) = a_0 + a_1 x^1 + a_2 x^2 + ... + a_N x^N$$

它的原函数为：

$$F(x) = a_0 x + \frac{a_1}{2} x^2 + \frac{a_2}{3} x^3 + ... + \frac{a_N}{N+1} x^{N+1}$$

所以我们只需要求出$F_1(x) f_2(x)$的值，再求一次定积分即可。

时间复杂度：$O(S^2)$
空间复杂度：$O(S)$

5.2 算法2

对于接下来20%的数据，有$n = 10$，那么考虑枚举哪些人分数比你高，哪些人分数比你低。

定义$U = 1, 2, 3, ..., n$，假设所有分数比你高的人构成了集合A，则分数比你低的人构成了集合$\complement_U A$，套用**算法1**的计算方法，则有：

$$P = \int_0^1 \prod_{i}^{i\in A} F_i(x) \prod_{i}^{i\in \complement_U A} [1 - F_i(x)]\, f_n(x)\,dx \tag{6}$$

接下来，只需统计当前超过了多少人，计入答案即可。

时间复杂度：$O(2^n S^2)$
空间复杂度：$O(S)$

5.3 算法3

我们观察获得第i名的总概率：

$$P_i = \sum_{A\subseteq\{1,2,\ldots,n-1\}}^{|A|=i} \int_0^1 \prod_i^{i\in A} F_i(x) \prod_i^{i\in \complement_U A} [1-F_i(x)]\, f_n(x)\, dx \tag{7}$$

$$P_i = \int_0^1 f_n(x) \sum_{A\subseteq\{1,2,\ldots,n-1\}}^{|A|=i} \prod_i^{i\in A} F_i(x) \prod_i^{i\in \complement_U A} [1-F_i(x)]\ dx \tag{8}$$

即我们可以先求出：

$$\sum_{A\subseteq\{1,2,\ldots,n-1\}}^{|A|=i} \prod_i^{i\in A} F_i(x) \prod_i^{i\in \complement_U A} [1-F_i(x)]$$

接下来乘上$f_n(x)$再积分即为获得第i名的概率。

考虑动态规划，我们依次枚举第i个函数，判断它是乘上$F_i(x)$还是$1-F_i(x)$。可以设出状态$dp_{i,j}(x)$代表已经到了第i个人，且有j个人的分数比你低的时候，当前的多项式为$dp_{i,j}(x)$，

则有递推式：

$$dp_{i,j}(x) = F_i(x)dp_{i-1,j-1}(x) + [1-F_i(x)]\,dp_{i-1,j}(x) \tag{9}$$

接下来，我们只需要暴力进行多项式乘法，递推即可。使用滚动数组可以将空间优化至$O(nS)$。

时间复杂度：$O(nS^2)$

空间复杂度：$O(nS)$

5.4 Bonus数据

5.4.1 数据11

当所有名次的奖金相等时，无论获得任意名次都必然获得p的奖金，所以可以直接输出p。

时间复杂度：$O(1)$

空间复杂度：$O(1)$

5.4.2 数据12

当所有人的概率函数相等时，每个人获得第i名的概率是相同的，所以我们只需输出$\sum_{i=1}^{n}\frac{p_i}{n}$即可。

时间复杂度：$O(n)$
空间复杂度：$O(1)$

5.4.3 数据13-14

对于数据点13和14，我们只需要计算出获得第一名与最后一名的概率，再乘上他们奖金与2到$n-1$名奖金的差值即为答案。

获得第一名的概率为：

$$P_1 = \int_0^1 \prod_{i=1}^{n-1} F_i(x)\, f_n(x)\, dx \tag{10}$$

获得最后一名的概率为：

$$P_n = \int_0^1 \prod_{i=1}^{n-1} [1 - F_i(x)]\, f_n(x)\, dx \tag{11}$$

我们只需要直接暴力计算多项式乘法即可。

时间复杂度：$O(S^2)$
空间复杂度：$O(S)$

5.4.4 一个更优的解法

可以发现，在求第一名与最后一名的概率时，所需要做的即为求n个多项式的乘积。由于本题中模数的特殊性，很容易想到使用快速傅里叶变换（FFT）优化乘法。

对于所有的i满足$1 \le i < n$，将$F_i(x)$进行离散傅里叶变换（DFT），即用一个数组记录当x分别为$\omega, \omega^2, \omega^3, ..., \omega^L$[1]时，函数的值分别为多少。

如：新的A_k等于原A函数$A(\omega^k)$的值；新的$F_{i,k}$等于原F_i函数$F_i(\omega^k)$的值。

[1] ω指原根，即满足$\omega^i \equiv 1$，当且仅当i为L的倍数；L为2的若干次幂

对于多项式$C(x) = A(x) + B(x)$，在进行DFT之后有：

$$C_i = A_i + B_i$$

同理，对于$C(x) = A(x) \times B(x)$，在进行DFT之后有：

$$C_i = A_i \times B_i$$

所以在计算多项式的运算时，我们可以将两个多项式进行DFT，在进行若干运算过后通过逆DFT转换回来，即为多个多项式的运算结果。

转换的时间复杂度为$O(L \log L)$，单次计算的时间复杂度为$O(L)$。[2]

如果我们直接使用FFT，需要转换$2n$次，计算n次，是$O(nS \log S + nS)$的，这样的时间复杂度反而更劣了。

注意到如果直接一个一个多项式乘过去，到最后会出现长度为$O(S)$的多项式乘上长度为$O(1)$的多项式，这样并不是最优的方法。所以我们可以每次将n个多项式分为两半，将两边的多项式分别乘起来之后再相乘，这样的复杂度为：

$$T(S) = 2T\left(\frac{S}{2}\right) + O(S \log S) \tag{12}$$

可以算出总复杂度为$O(S \log^2 S)$。

时间复杂度：$O(S \log^2 S)$

空间复杂度：$O(S)$

5.5 算法4

同样的，我们在递推求$dp_{i,j}$的时候，可以考虑用FFT进行优化。

在我们将F_i进行DFT之后，转移方程9变为：

$$dp_{i,j,k} = F_{i,k} dp_{i-1,j-1,k} + (1 - F_{i,k})\, dp_{i-1,j,k} \tag{13}$$

接下来，我们确定i与k，即将整个dp的第二维与第三维交换一下，则转移方程变为：

$$dp_{i,k,j} = F_{i,k} dp_{i-1,k,j-1} + (1 - F_{i,k})\, dp_{i-1,k,j} \tag{14}$$

[2]FFT的具体实现过程可以戳这里http://en.wikipedia.org/wiki/Fast_Fourier_transform

可以发现，数列$dp_{i,k}$的生成函数$G_{i,k}(x)$与数列$dp_{i-1,k}$的生成函数$G_{i-1,k}(x)$的关系为：

$$G_{i,k}(x) = [F_{i,k}x + (1 - F_{i,k})]\, G_{i-1,k}(x) \tag{15}$$

由于$G_{0,k}(x) = 1$，则可推导出：

$$G_{n-1,k}(x) = \prod_{i=1}^{n-1} [F_{i,k}x + (1 - F_{i,k})] \tag{16}$$

对于这个式子，我们可以直接使用如5.4.4所说的分治套FFT，即可在$O(n\log^2 n)$的时间内求出$G_{n,k}(x)$。

接下来我们只需要用$O(nS\log S)$的时间预处理出所有$F_{i,k}$的值，再枚举所有的k，求出所有的$dp_{n-1,k,j}$，最终转换回来即为函数$dp_{n-1,j}(x)$。

时间复杂度：$O(nS\log^2 n)$

空间复杂度：$O(nS)$

5.6 算法5

换个角度，假设对于转移方程9我们直接套用生成函数，则对于$dp_{i,j}(x)$的二维生成函数$G_i(x, y)$，有：

$$G_i(x, y) = [F_i(x)y + [1 - F_i(x)]]\, G_{i-1}(x, y) \tag{17}$$

又由于$G_0 = 1$，则可以推导出：

$$G_{n-1}(x, y) = \prod_{i=1}^{n-1} [F_i(x)y + [1 - F_i(x)]] \tag{18}$$

则最终我们需要求的即为若干个二元多项式乘起来之后所得的多项式。

对于二元多项式的DFT，相当于分别枚举x与y在取不同的值时，函数的值为多少，所以我们只需要先固定y，对于y的若干次幂之前关于x多项式进行FFT，再对于每个x不同的取值，对关于y的多项式进行FFT即可，对于一个$N \times M$的二元多项式进行FFT的时间复杂度为$O(NM(\log N + \log M))$。

有了二元多项式的DFT，接下来我们可以沿用之前的思路——分治套FFT来解决当前问题。

需要注意的是，假设我们的问题规模是$n \times S$的，由于数据是随机的，当我们将n个不等式等量的分成两份时，它们乘起来的项数也大约是差不多的，则此时的时间复杂度为：

$$T(n,S) \quad = \quad 2T\left(\frac{n}{2},\frac{S}{2}\right)+O(nS\log S) \tag{19}$$

可以发现，此时分治套FFT的时间复杂度并没有$\log^2$项了，时间复杂度仅为$O(nS\log S)$。

在求出G_{n-1}之后，我们只需要用$dp_{n-1,j}(x)$乘上$f_n(x)$再积分即为答案。

至此，问题完美解决。

时间复杂度：$O(nS\log S)$

空间复杂度：$O(nS)$

6 试题总结

本题的考察点在于对积分以及FFT的运用，由于在模数上对于题目的解法有一定的提示，大大降低了本题的思维难度，总体来说属于NOI中等难度。

就本场互测而言，saffah的题目是非传统题，lyy的题目在代码难度以及思维难度上均较难，所以从某种意义上来说，花部分时间来写此题的正解是一种比较稳妥的做法。当然，对于我来说一种较好的考试策略，并不一定适合于其他人，比如你在非传统题上有自己的见解，或者在数据结构上有独特的造诣，那么你也可以选择花费更多的时间在其他两题上。

对于集训队的其余12人来说大部分人都有能力拿到满分，基本上所有人都可以得到70分——前提是在大家在考试中给本题分配了合理的时间。然而由于互测的计分改为参加测试即可获得1分，导致了部分选手在一定程度上对考试的轻视，并没有认真参考，所以在此不对具体人数进行预测。