题意

给定 k 以及系数序列 $w_0 \sim w_{2^{k-1}-1}$。定义一个 $n \geq k$ 阶排列 p 的权值 $val(p) = \prod_{i=1}^{n-k+1} w_{f(p_i,p_{i+1}\dots p_{i+k-1})}$，其中 $f(a_1,a_2\dots a_k) = \sum_{i=1}^{k-1} 2^{i-1}[a_i < a_{i+1}]$。

给定 n，计算所有 n 阶排列的权值和 $\bmod 998244353$。

$2 \leq k \leq 4$，$k \leq n \leq 10^5$，$0 \leq w_i < 998244353$。

输入格式：

第一行两个整数 n, k。

第二行 2^{k-1} 个整数，第 i 个整数表示 w_{i-1}。

输出格式：

一行一个整数，表示答案 $\bmod 998244353$。

数据范围

Subtask	$n \leq$	$k =$	分值
1	10	4	5
2	20	4	10
3	10^5	2	5
4	100	3	10
5	4000	3	10
6	4×10^4	3	15
7	10^5	3	5
8	2000	4	10
9	4×10^4	4	10
10	10^5	4	20

对于所有数据：$2 \leq k \leq 4$，$k \leq n \leq 10^5$，$0 \leq w_i < 998244353$。

说明

本题应该原本是一位不愿透露姓名的群友准备投给集训队互测的题目，遗憾的是他在今年 NOI 中发挥失常没有进入集训队。但他仍然希望将这道题目分享给大家，于是他将此题交给我代投。在此感谢他提供本题的 idea/solution/std。

下面有两份题解，一份是我写的，一份是他给我的题解。

两份题解大体上的方向是相似的，不过我的题解可能更详细一些。

solution 1

用 $b_1 \sim b_{n-1}$ 描述一个排列相邻对的上升/下降关系，即 $b_i = [p_i < p_{i+1}]$，那么 $val(p) = \prod_{i=1}^{n-k+1} w_{\overline{b_i, b_{i+1} \ldots b_{i+k-2}}}$。枚举序列 b，这样对应序列 b 的所有 p 的 $val(p)$ 都一致且已经被确定，乘上这样 p 的数量加和即可得到答案。

要计算对应 b 的 p 的数量显然是 LOJ 上的不等关系：考虑对 `>` 限制容斥，拆成无限制减去 `<`，记 $ways(b)$ 为 $n!$ 除以 b 中所有极长 1 连续段的长度加 1 的阶乘之积。那么对应 b 的 p 的数量可以写成 $\sum_{c_1 \sim c_{n-1}, \forall 1 \le i < n, c_i \ge b_i} ways(c) \prod_{i=1}^{n-1} (-1)^{b_i + c_i}$。

现在可以认为是要枚举所有长度为 $n-1$ 的 01 序列对 (b, c)，计算对应贡献和。我们将 c 拆分为一个 `0` 接上一些 `1` 这样结构的拼接（`1` 数量可以为 0），每个部分称为一个段，最开始可以额外再接一些 `1`（也可以不接），这样的划分是唯一的。考虑这样由若干个 `1` 不断拼上新的一段的过程，每次拼上新的一段的时候同时确定这一段的 b 是什么，因为 b 有跨段的贡献但其实只需要知道每一段最后 $k-2$ 个 b 就足够了，所以做一个 2^{k-2} 个状态的状压。

现在考虑往已有的 01 序列对 (b, c) 之后再给 c 拼上一个段的过程，可以发现只要知道已有的 b 的后 $k-2$ 个字符（即枚举这 $k-2$ 个字符），可以 DP 求出所有 $val_{i,T}$ 表示所有新的 c 段长度为 i，选择拼上某种 b 段且拼完后整体 b 的后 $k-2$ 个字符是串 T 的所有方案的权值和（权值是新的段的 $(-1)^{b_i+c_i}$ 之积除以新的段的长度的阶乘）。这里我们假设已有的部分至少有 $k-2$ 个字符，所以每拼上一个新的字符都需要计入以这个位置为结尾的长度为 $k-1$ 的 b 子段的贡献。假设加入段前 b 的后 $k-2$ 个字符是 S，加入后是 T，把所有这样的方案对应的权值和写成 OGF（x 元记录段长），把 $(2^{k-2})^2$ 个 OGF 写成一个 $2^{k-2} \times 2^{k-2}$ 的多项式矩阵 A，可以发现这样非常有利于描述段的拼接。

接下来考虑如何计算答案，直接枚举 c 的前 2^{k-2} 个字符，那么再往后面拼段就满足我们 “每次拼段时至少已有 $k-2$ 个字符” 的假设了。但注意到拼段要求必须以 `0` 开头，所以如果有 c 的方案第 $k-1$ 个字符是 `1` 这样会被忽略，所以还要进一步枚举从 $k-1$ 开始的极长 1 连续段的长度（也可以为 0），同样 DP 算出所有可能的长度为 i 的 b 根据后 $k-2$ 个字符划分后每一种的权值和。那么我们只需要知道 $\forall 0 \le i < n, 0 \le S < 2^{k-2}$，已知已有部分的 b 后 $k-2$ 个字符是 S，再拼一些段满足长度和为 i 的所有方案的权值和，即可解决原问题。

那么这其实是要计算 $(\sum_{k=0}^{\infty} A^k) \bmod x^n$，我们知道这就等于 $\frac{1}{I - A} \bmod x^n$。使用类似多项式求逆的牛顿迭代，可以变为要进行模 $x^1, x^2, x^4 \ldots x^n$ 意义下各 $O(1)$ 次的 $2^{k-2} \times 2^{k-2}$ 的系数为多项式的矩阵的矩阵乘法，如果先对每个位置的多项式做 DFT，然后点乘加和后一起 IDFT 回来，就可以做到总长 $O(4^{k-2}n)$ 级别的 DFT 以及额外的 $O(8^{k-2}n)$ 的点乘代价。

所以矩阵求逆的部分可以做到 $O(4^{k-2}n \log n + 8^{k-2}n)$，而前面预处理多项式矩阵 A 以及最终计算答案时的 DP 是 $O(4^{k-2}n)$ 的。所以该问题在 $O(4^{k-2}n \log n + 8^{k-2}n)$ 的时间复杂度内得到解决。

如果你觉得你被卡常了，请依次检查以下几件事：

1. 确保你的矩阵大小是 $2^{k-2} \times 2^{k-2}$ 的。
2. 确保你的矩阵乘法的 DFT/IDFT 总长是 $O(4^{k-2}n)$ 的。
3. 如果还是不行，换一个快一点的 FFT 板子。

solution 2

因为题面略有修改，下面认为 $k = 4$ 且权值数组的名字被换成了 a。

对于一个排列 $\{\pi_i\}$ 我们有权值

$$\prod_{i=1}^{n-3}(a_{000}[\pi_i<\pi_{i+1}<\pi_{i+2}<\pi_{i+3}]\\+a_{001}[\pi_i<\pi_{i+1}<\pi_{i+2}>\pi_{i+3}]\\+a_{010}[\pi_i<\pi_{i+1}>\pi_{i+2}<\pi_{i+3}]\\+a_{011}[\pi_i<\pi_{i+1}>\pi_{i+2}>\pi_{i+3}]\\+a_{100}[\pi_i>\pi_{i+1}<\pi_{i+2}<\pi_{i+3}]\\+a_{101}[\pi_i>\pi_{i+1}<\pi_{i+2}>\pi_{i+3}]\\+a_{110}[\pi_i>\pi_{i+1}>\pi_{i+2}<\pi_{i+3}]\\+a_{111}[\pi_i>\pi_{i+1}>\pi_{i+2}>\pi_{i+3}])$$

考虑「容斥」，也就是考虑转化为钦定一些连续的上升段附带系数。

我们可以这样来做这件事：将上式中所有 $[\pi_{i+j}>\pi_{i+j+1}]$ 换成 $(1-[\pi_{i+j}<\pi_{i+j+1}])$。

然后我们展开成

$$\prod_{i=1}^{n-3}(c_{111}[\pi_i<\pi_{i+1}<\pi_{i+2}<\pi_{i+3}]\\+c_{110}[\pi_i<\pi_{i+1}<\pi_{i+2}]\\+c_{101}[\pi_i<\pi_{i+1}][\pi_{i+2}<\pi_{i+3}]\\+c_{100}[\pi_i<\pi_{i+1}]\\+c_{011}[\pi_{i+1}<\pi_{i+2}<\pi_{i+3}]\\+c_{010}[\pi_{i+1}<\pi_{i+2}]\\+c_{001}[\pi_{i+2}<\pi_{i+3}]\\+c_{000})$$

的形式。

那么我们考虑去展开这个乘积式，其中并在一起的连续上升段就是我们要拿来容斥的东西。具体地，如果连续段的长度为 $l_1+l_2+\cdots+l_m=n$，则方案数就是 $\frac{n!}{l_1!l_2!\ldots l_m!}$。

我们考虑对每个 i 选择哪一项，并将其后三个间隔是否有符号记入状态。那么显然不连续当且仅当某个 i 后面第一个间隔没有符号，或者最后三个间隔没有符号。

我们需要将连续的部分乘上长度的阶乘的倒数作为贡献，因此不妨将连续的上升段并在一起作为一个单位考虑。

对于一个连续段 $[l,r]$，我们需要计算：$(l-1,l),(r,r+1)$ 没有符号，$(l,l+1),(l+1,l+2),(r+1,r+2),(r+2,r+3)$ 是否有符号的条件下的方案数。这显然只跟 $r-l+1$ 相关，可以直接递推，只需同时将 $(r,r+1)$ 是否有符号记入状态即可。

然后连接连续段的过程只需要考虑填满了 $[1,r]$，$(r,r+1)$ 没有符号，$(r+1,r+2),(r+2,r+3)$ 是否有符号。枚举连续段的长度转移，并特殊处理最后一段，立刻有 $O(n^2)$ 解法。

进一步，注意到这是半在线卷积，所以使用 CDQ 分治 NTT 即可 $O(n\log^2 n)$ 或 $O\left(\frac{n\log^2 n}{\log\log n}\right)$ 等。也可以写作 GF 通过求逆或者牛顿迭代求解，时间复杂度 $O(n\log n)$。

部分分设计

Sub1 可以全排列枚举 $O(n!poly(n))$。

Sub2 则是可以枚举 b 序列 $O(2^n n)$。

Sub3 $k=2$ 的情况等价于计算一行欧拉数，这非常经典，可以做到 $O(n\log n)$。

Sub4 注意到只要记录一段前缀最后 $k-1$ 个元素在这段前缀中的排名，再加入一个数的贡献是很好计算的，可以得到一个 $O(n^k poly(n))$ 的 DP。

Sub5,Sub8 只要得到了 $O(poly(2^k)poly(n))$ 的做法即可通过（这里的 $poly(n)$ 应该都是 n^2，所以 $n=4000$ 应该是合理的）。

Sub6,Sub9 只要得到了 $O(poly(2^k)npoly(\log n))$ 的做法应该就容易通过。

Sub7,Sub10 则是在 Sub6,Sub9 的基础上进一步区分 $poly(2^k)$ 和 $poly(\log n)$ 的次数大小。

考虑到如果直接进行分类讨论有可能可以通过 $k=3$，为了照顾这样的选手将 $k=3/k=4$ 分开了。

同时常数较劣的做法应该也更容易通过 $k=3$ 的对应范围的 n，减少因卡常损失的分数。

杂谈

赛时发现大家纷纷使用了半在线卷积以及高超的卡常技巧通过了本题，我只能说牛顿迭代就是不行，不如半在线卷积。

参考资料

无。