The 2nd Yuzusoft Cup Stage 1: Shantou Tutorial

绫地宁宁、椎叶紬、相马七绪

Mar 28th, 2024

1 禁断之门

实际上 $n+1$ 是诈骗。

我们不难发现，假设 $p_{1\sim t}$，$q_{1\sim t}$ 内部元素两两不同，假设 $f(p,q)=\sum_{i=1}^{t}\sum_{j=\min(p_i,q_i)}^{\max(p_i,q_i)}b_j$。

那么假设使得 $f(p,q)$ 最大值为 $f_0=f(p_0,q_0)$。

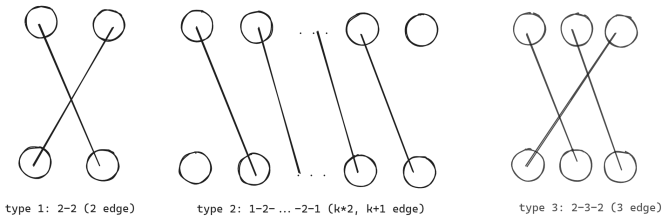
那么 $f(A)\le(\sum_{i=1}^{n}a_i)f_0$，这个在 A 每一行依次为 $p_0,q_0,p_0,q_0,\dots$ 的时候可以取到等。

也就是我们只需要求 $f(p,q)$ 的最大值 f_0 即可。

我们视作为一种类似于“二分图最小权匹配的东西”，连边 (x,y) 需要的代价是 $\sum_{j=\min(x,y)}^{\max(x,y)}b_j$。

首先我们都知道这玩意是是二分图匹配那一类的东西，这个问题容易通过费用流解决。那么也就是这个问题的答案有凸性。我们可以 wqs 二分，转成每次多选一条边会多一个收益 M，最后需要最大化收益。

观察我们会怎么连边，通过手枚一下跑一下暴力，发现可能的情况无非就几种：



我们考虑 dp，f_i 表示只考虑前 i 个左/右节点的连边，最大收益。

首先 $1,3$ 方案的转移是简单的。2 方案只需要维护个前缀最小即可。

复杂度：$\mathcal{O}(m\log V)$。

2 黑暗

熟知二项式前缀和可以 $O(1)$ 转移。

3 情侣给我烧了

根据简单的推式子，我们大概是想求：

$$\sum_{i=0}^{n}(-2)^i \times \binom{2(n-i)}{n-i} \times \frac{1}{i!}$$

假设 $f(x)=\sum_{i=0}\frac{(-2x)^i}{i!}=e^{-2x}$，$g=\sum_{i=0}\binom{2i}{i}x^i=\frac{1}{\sqrt{1-4x}}$。

那么我们实际上就是想要求：$H(x)=\frac{e^{-2x}}{\sqrt{1-4x}}$ 的第 n 项系数。通常对带有 e^x 的生成函数求导有奇效，我们尝试一下：

$$H'(x)=\frac{-2e^{-2x}\sqrt{1-4x}+2xe^{-2x}(1-4x)^{-\frac{1}{2}}}{1-4x}=-2H(x)+\frac{2xH(x)}{1-4x}$$

也就是：

$$(1-4x)H'(x)=8xH(x)$$

到了这里就可以拆每一项系数得到递推式，就可以 $\mathcal{O}(n)$ 递推了。

4 青春有悔

我们需要求 $f(x)$ 的前 n 项系数和，实际上就是求 $[x^n]\dfrac{f(x)}{1-x}$。

考虑单个询问怎么做，每个考试的生成函数相当于 $\dfrac{1-x^{a+1}}{1-x}$，我们把这些东西全部乘起来，就是 $\dfrac{1}{(1-x)^{n+1}}\times\prod(1-x^{a_i+1})$。

前面的东西展开就是 $\sum_{i=0}\binom{i+n}{i}x^i$，后面那一坨类似侯公主背包，直接取 ln，熟知 $\ln(1+x)=\sum_{i=1}\dfrac{(-1)^{i-1}}{i}x^i$，对于所有 $\ln(1-x^{a_i+1})$ 求和可以先用个桶存起来在跑，复杂度是 $\mathcal{O}(n\ln n)$。

考虑修改，假设原来的生成函数是 F，我们现在就是要求 $[x^r]\dfrac{1-x^p}{1-x^q}F(x)$。我们把分子拆开，我们只需要求 $[x^r]\dfrac{F(x)}{1-x^q}=F[x^r]+F[x^{r-q}]+\ldots$，这里直接根号分治即可。

5 请神粉兔

对这种区间查询的问题可以考虑使用扫描线。从小到大枚举 r，当 r 从 $r'-1$ 转移到 r' 时，维护每个 l 的答案。更新所有可能的 $x \le r$，假设与 $s[x,r]$ 完全相等且 L 最小的字串 $s[L,R]$。我们可以用线段树维护这个新的串所带来的答案的改变。

然后我们怎么找所有有用的 x 呢？我们尝试取枚举 R，那么上面维护 $s[x,r], s[L,R]$ 产生贡献的部分就可以变成 $s[1,r], s[1,R]$ 的 lcs（最长公共后缀）。

我们对建一个 SAM，假设 $s[1,i]$ 在 SAM 对应点为 u_i。那么上面 lcs 的长度相当于 $len(\text{lca}(u_r, u_R))$。如果两个 $R_1 < R_2$ 对应的 lca 相同必然会选择 R_2 对应的节点（前者 lcs 被后者包含）。

也就是，大概的算法就是：

- SAM 每个节点维护一个变量 p 表示上个在其子树内被访问的 u_i。- 对 u_r，把 1 到 u_r 路径上所有 p 与 r 计算贡献。- 把 1 到 u_r 路径上所有 p 设置为 u_r。

这玩意类似于 LCT 的 assert，我们也可以用树链剖分简单维护。

6 美好的每一天 不~ 连续的存在

首先合并两个连通块并且维护连通块颜色出现次数为奇数的个数的东西，有两个能够实现的算法，一个是线段树合并，一个是 ‘bitset‘。

下面假设 n, q 同阶。

我们考虑先搞出来一个低于 $\mathcal{O}(n^2)$ 的算法。

考虑莫队，由于删除一个点难以实现，所以我们得改成撤销。一个小问题就是回滚莫队的时候，假设所有询问的 l 都在 $[p, q]$（一个块当中），那么 $[p, q]$ 当中这个块当中的点就可能会被加入/撤销 $\mathcal{O}(q)$ 次，如果这个块当中刚好有个菊花中心就炸了。

注意到给 $[l+1, r]$ 加入 l 的复杂度不弱于 $[l+1, n]$ 加入 l。而且依次加入 $n, n-1, \ldots, 1$ 的复杂度是 $\mathcal{O}(n \log n)$ 的。于是我们可以先模拟一下倒着加入的复杂度，按照这个复杂度作为权重分块。

有个小问题，复杂度很大的点所处的块权值很高。但是实际上这个块必然只包含其一个点。而且所有询问 l 必须经过这个点，所以有个比较简单的结局方案就是把 r 端点纳入扩展的点列当中。这样子复杂度就是 $\mathcal{O}(n\sqrt{n} \log x)$ 的。

线段树合并常数很大，必然过不去。我们想使用 ‘bitset‘ 思想的算法引入减少常数，我们下面提供一种类似 ‘bitset‘ 启发式合并的东西。

考虑一种融合了线段树合并，‘bitset‘ 的算法，32 叉树！

具体是这样子的，我们建一个 4 层的树，每层除了叶子都都有 32 叉，如果这个节点的子树内没有任何关键点我们就不去建。我们把叶子缩到第 3 层的节点上去。每层节点维护一个 son 表示有元素的子树下标集合，维护 $a_{*,32}$ 表示儿子下标。对于合并 x, y 节点：

- 如果 x, y 位于第 3 层，直接让 $son_x \leftarrow son_x$ XOR son_y；

- 否则：

- 对每个 i，$i \in son_x, i \in son_y$，合并 $a_{x,i}, a_{y,i}$，并且更换 $a_{x,i}$ 标号。

- 对于每个 i，$i \notin son_x, i \in son_y$，直接让 $a_{x,i} \leftarrow a_{y,i}$ 即可。

这样子我们就能做到 $\mathcal{O}(n\sqrt{n} \log_w x)$ 的东西。随便卡卡常就过去了。

这边推荐几种本质优化的方式：

- 每个原树节点维护 siz_x，表示其连通块出现奇数次颜色数量，合并 x, y 时，$siz_x \leftarrow siz_x + siz_y$，对于重复的部分，在上面第一点直接减掉 $2 \times \text{popcount}(son_x \text{ AND } son_y)$ 即可。- 撤销部分，原来的并查集和 siz 数组开个备份就行了。- 其他实现方面的问题可以见代码。

由于是 Ynoi 就不放代码了，我的代码比较丑陋（写了 6k），需要的私聊我吧。