“统信杯”第十七届黑龙江省大学生程序设计竞赛

题解

吉林大学命题组

May 15, 2022

Overview

A. Bookshelf Filling

首先题面里要求的最多是从右边取 $m-1$ 本书放到顶上，可能有一些队伍没有看到导致 wa 了几发。

我们可以发现如果顶上横着摆放的书的宽度大于底下剩余书的宽度，那么肯定是不优的。因为取走的书本数小于 m，所以底下至少有一本高度为 b 的书，所以当顶上的高度大于底部时，我们可以从顶上取回高度为 b 的书放回到下面，直到底下的宽度大于顶部为止。

那么我们就可以认为顶部书本的宽度总是小于底部书本的宽度。于是我们可以二分底部的 B 类书本剩余数量 x，这样就知道了底部的宽度为 $n+x$，那么可以通过简单的计算算出顶部空白部分最多 B 类书本为 $(b-a)\times\lfloor\frac{n}{b}\rfloor+(h-b)\times\lfloor\frac{n+x}{b}\rfloor$，判断 B 类书本总和是否大于等于 m 即可。

A. Bookshelf Filling

此题也存在 $O(1)$ 做法。但实现有一些细节。

B. Lovely Fish

把'1' 看成 1，把'0' 看成-1，原题等价于，任意前缀和和后缀和都应该非负的。
先考虑暴力的写法，
1. 先保证所有前缀和为非负数，根据贪心的思路，为了让把后缀和变为非负数时操作次数尽可能少，可以确定加'1' 的位置要尽可能后面，即，只有当 $sum_i = 0$ 且 $S_{i+1} =' 0'$ 时，我们需要在 S_i 和 S_{i+1} 之间添加一个 $'1'$
2. 让后缀和都变成非负数，只需要看步骤 1 后，所有后缀和中最小值就行了。

B. Lovely Fish

考虑正解，
设 1 到 i 的和为 $sum[i]$（即前缀和）
一段合法的串满足：

1. 所有的 $sum[i] \geq 0 (i \in [1, n])$。（每个前缀和和 $>= 0$）
2. 所有的 $sum[n] - sum[i] \geq 0$ $(i \in [1, n])$。（每个后缀和 ≥ 0）

因为对于每一个加'1' 可以转换成把 $sum[i]$ 到 $sum[n]$ 都加了 1，（因为插入的那些位置 p 肯定都是 $sum[p] = 1$ 的情况，可以不用考虑这些点，直接特殊考虑全 0 的情况就好了）

1. 使所有的 $sum[i] \geq 0$ 的贡献显然是 $max\{-sum[i], (i \in [0, n])\}$
2. 这时候肯定会有一个最大的 $sum'[i]$(操作后)，把 $sum'[n]$ 加到 $sum'[i]$ 的贡献就是 $sum'[i] - sum'[n]$

B. Lovely Fish

对于第二步中，这个最大的 $sum'[i]$ 就是：$max\{sum[j]-sum[i],(i,j\in[0,n],i<j)\}$
因为对于每一个 $sum'[i]=sum[i]-min\{sum[j],j\in[0,i)\}$
（注意：一定是右边的减左边的）
而 $sum'[n]$ 就是 $sum[n]+max\{-sum[i],(i\in[0,n])\}$
答案就是 2 个操作加起来就行了，也就是说，
$ans=max\{sum[j]-sum[i],(i,j\in[0,n],i<j)\}-sum[n]$
特殊的，如果全是 0，那么在第二步找的 $sum'[i]$ 不存在，而插入的 1 会产生影响，
因此答案会变成：$-sum[n]+1$
综合俩个情况，最后可以得到
$ans=max(1,max\{sum[j]-sum[i],(i,j\in[0,n],i<j)\})-sum[n]$
而对于计算 $sum[j]-sum[i]$ 这部分可以离线 + 并查集实现接近 $O(n)$ 的复杂度

C. Tree Division

Set node 1 as the root, then:

- $\forall u, v \in A (u \neq v)$, if v is in the subtree of u, then $a_u < a_v$
- $\forall u, v \in B (u \neq v)$, if v is in the subtree of u, then $a_u > a_v$

For any node t, define S_t as the set of nodes in the subtree of node t. Based on the greedy strategy:

- If $t \in A$, then a_t is the minimum value of nodes in $A \cap S_t$. Find a possible division to minimize the maximum value of nodes in $B \cap S_t$.
- If $t \in B$, then a_t is the maximum value of nodes in $B \cap S_t$. Find a possible division to maximize the minimum value of nodes in $A \cap S_t$.

C. Tree Division

$dp[n][2]$, set the first dimension as the node t and the second dimension as the $t \in A$ or $t \in B$ state. We can maintain the minimized/maximized value in the process of DP. Then the DP transfer equation is:

- $dp[u][0] = max_{fa_v=u}\{min(a_u < a_v ? dp[v][0] : Inf, a_u < dp[v][1] ? a_v : Inf)\}$
- $dp[u][1] = min_{fa_v=u}\{max(a_u > a_v ? dp[v][1] : -Inf, a_u > dp[v][0] ? a_v : -Inf)\}$

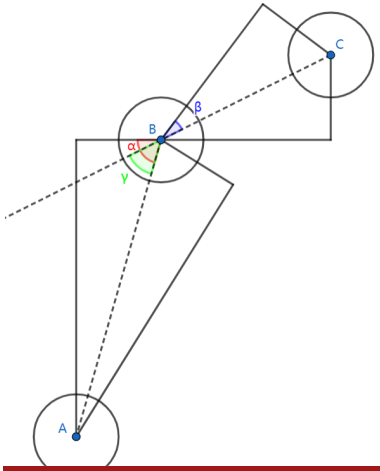
if $dp[1][0] \neq Inf$ or $dp[1][1] \neq -Inf$, then node 1 is valid.

D. Collision Detector

我们可以先计算 O_1 碰撞 O_2 的在 O_2 上的碰撞点范围，也就是计算两个圆的交叉公切线的切点。
然后计算 O_2 能够撞到 O_3 的角度区间，同样可以通过求它们的公切线得到。
然后计算碰撞点反方向角度区间和 O_2 碰撞 O_3 的角度区间在 $[-\pi, \pi]$ 上是否有交就行了。

D. Collision Detector

还有一个精度较高且更加简洁做法，验题人 @Magi_karp 在验题时使用了这个方法，比赛是也应该有队伍用类似方法过了。



D. Collision Detector

图片见上个 ppt。我们只需要判断 $\alpha+\beta$ 是否大于 γ 即可。其中

$$\cos\beta=\frac{\sqrt{BC^2-4}}{BC}\quad \sin\beta=\frac{2}{BC} \tag{1}$$

$$\cos\alpha=\frac{2}{AB}\quad \sin\alpha=\frac{\sqrt{AB^2-4}}{AB} \tag{2}$$

$$\cos\gamma=\frac{\vec{AB}\cdot\vec{BC}}{|AB||BC|} \tag{3}$$

$$\cos(\alpha+\beta)=\frac{2\sqrt{BC^2-4}-2\sqrt{AB^2-4}}{|AB||BC|} \tag{4}$$

D. Collision Detector

这里要注意 $\alpha+\beta$ 范围在 $(0,\pi]$
则如果 $2\sqrt{BC^2-4}-2\sqrt{AB^2-4}<\vec{AB}\cdot\vec{BC}$，输出 `yes`，否则输出 `no`.

E. Exclusive Multiplication

首先需要推出

$$f(a \times b) = \frac{f(a) \times f(b)}{\gcd(f(a), f(b))^2} \tag{5}$$

然后我们令

$$g(n) = \sum_{n=\gcd(b_i, b_j)} f(b_i) \times f(b_j) \tag{6}$$

那么最后的答案就是

$$\frac{1}{2} \times (\sum_{d} \frac{g(d)}{d^2} - n) \tag{7}$$

E. Exclusive Multiplication

为了计算 $g(n)$，我们令

$$G(n)=\sum_{n|d} g(d)=\sum_{n|\gcd(b_i,b_j)} f(b_i)f(b_j)=\left(\sum_{n|b_i} f(b_i)\right)^2 \tag{8}$$

$G(n)$ 可以快速算出，然后通过莫比乌斯反演计算得到 $g(n)$。

$$g(n)=\sum_{n|d} \mu(\frac{d}{n})G(d) \tag{9}$$

F. 342 and Xiangqi

方法一：

在 7 个象的可到达点位上任选 2 个的方案数为 $\binom{7}{2} = 21$ 种，我们可以把两只象的每种位置状态设为点，相互可一步到达的状态之间连一条长度为 1 的边，用 Floyd 算法预处理两两点之间的最短路即可。

方法二：

设两象的初位置分别为 a_1, b_1，末位置分别为 a_2, b_2，位置 x, y 之间的最短路为 $f(x, y)$，则最终答案一定为 $\min\{f(a_1, b_1) + f(a_2, b_2), f(a_1, b_2) + f(a_2, b_1)\}$，即单独考虑每只象去到每个位置的情况再求和。不难想象，由于两象被视作相同且移动顺序任意，我们总能通过两种不同的位置对应方法和移动顺序的选取使得路径不出现交叉，从而满足要求。

G. Chevonne's Necklace

题意：一串珠子，每个珠子上有一个数字 c_i，每次可以选中一个珠子，然后删去从这个珠子开始顺时针相邻的 c_i 个珠子。问最多能删去的珠子数量，以及这样的方案数。（此处方案的定义为选中珠子的集合）

先说一下做法，直接 c_i 为重量，n 为容量，0 为价值做背包，然后方案数就是背包的方案数。（其他本质相同的做法也 ok）

G. Chevonne's Necklace

这样子可能会让大家产生疑惑，因为明显在珍珠项链上的要求更紧，可能你删去一个选中的珠子不小心顺带把另外一个需要选中的珠子给提前删掉了，导致无法按照背包意义下的方案构造题意中的方案。

注意到此题在描述方案的时候用的是“集合”的概念，在这里便暗示了取珍珠的顺序并不是很重要的一件事情，进而可能去想到正确做法。

G. Chevonne's Necklace

下证：背包选出来的任意方案都为符合题意的方案；即：我们来对背包选出来的某个珍珠集合构造题意中的删珍珠方法。

我们将选中的珍珠用下标表示为 $p_1 p_2 \ldots\ldots p_k$，我们有 $\sum c_{p_i} \leq n$，我们再用 d_{p_i} 表示 p_i 与项链上顺时针方向的下一个选中的珍珠、即 $p_{i \bmod n+1}$ 间的距离，那么我们可以发现，只有当前 $c_{p_i} \leq d_{p_i}$ 才能删去 p_i 这个珍珠。

事实上，每次都至少有一个珍珠满足 $c_{p_i} \leq d_{p_i}$（因为 $\sum c_{p_i} \leq n$，根据鸽巢原理$\cdots\cdots$），所以每次总能选中一个待删除珍珠删去，直到删完为止。故构造完毕，解法的正确性得证。

H. Kanbun

方法一：按要求模拟即可。设 R_i 表示位置为 i 的左括号所对应的右扩号的位置。对于一个 $[L, R]$ 区间，若 $L > R$ 则过程结束，否则根据第 i 个位置上的字符决定输出及接下来递归的区间。整个过程从 $[1, n]$ 区间开始。

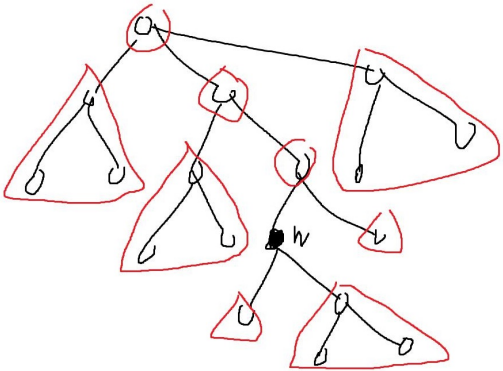
方法二：注意到原题等价于将左括号的下标在其对应的右括号之后输出，显然不同的括号间不会产生影响，故我们在括号匹配的过程中用栈记录未被匹配的左括号位置，在其对应的右括号输出后，输出该左括号的位置。

I. Equal Sum Arrays

考虑 $f(n)$ 的通项，我们可以把构造一个和为 n 的正项数组转化为在一个长度为 n 的序列形成的 $n-1$ 个空隙中任意插板，由此得到 $f(n)=2^{n-1}$。

J. JOJO's Happy Tree Friends

首先需要发现这样的一个性质：在同一个区域内的期望是一样的。



然后我们可以将同一个区域的结点合并，获得一个简单的树。

J. JOJO's Happy Tree Friends

然后我们将结点分成两类，一类是在根结点 1 到目标结点 w 路径上的点，记为 A 类点。除此以外的点记为 B 类点。
对于 B 类点，其合并后期望可以通过下面的式子计算

$$E(u) = \frac{1}{n}\left(sz(u)E(u) + \sum_{(v,w)\in path(1,u)} (sz(v) - sz(w))E(v)\right) + 1 \tag{10}$$

J. JOJO's Happy Tree Friends

接下对于 A 类点，我们来尝试去推导父亲和儿子结点之间的数值关系。我们先令 $S(u) = \sum_{v \in subtree(u)} E(v)$。
那么对于一个 A 类点 u，其期望可以写成

$$E(u) = \frac{1}{n}\left(S(u) + \sum_{(v,w)\in path(1,u)} (sz(v) - sz(w))E(v)\right) + 1 \tag{11}$$

经过变换得到

$$S(u) = nE(u) - \left(\sum_{(v,w)\in path(1,u)} (sz(v) - sz(w))E(v)\right) - n \tag{12}$$

J. JOJO's Happy Tree Friends

根据 $S(u)$ 的定义，我们知道

$$S(u) = E(u) + \sum_{v \in ch(u)} S(v) \tag{13}$$

对于 B 类儿子，我们可以直接计算期望，对于 A 类儿子（只有一个，记为 nxt），$S(nxt)$ 用 nxt 代入 (12) 式得到。然后将所有东西代入 (13) 式，我们获得了

$$S(u) = E(u) + \left(\sum_{v \in ch(u), v \in B} sz(v) E(v) \right) + nE(nxt) - \left(\sum_{(v,w) \in path(1,nxt)} (sz(v) - sz(w)) E(v) \right) - n \tag{14}$$

J. JOJO's Happy Tree Friends

联立 (12) 和 (14)，解出 $E(nxt)$，于是我们获得了 $E(nxt)$ 关于同层 B 类结点和父亲 u 的表达式

$$E(nxt) = \frac{(sz(u) - sz(nxt) + n - 1)E(u) - \sum_{v \in ch(u), v \in B} sz(v)E(v)}{n} \tag{15}$$

J. JOJO's Happy Tree Friends

接下来我们设根结点的期望为 x，即 $E(1) = x$。然后以如下的顺序计算其他结点的 E 关于 x 的表达式。

从根节点出发，深度从浅到深，每层先计算 B 类结点的期望，然后计算 A 类结点的期望。因为 B 类结点期望表达式只和到根的路径上的结点的期望有关，而 A 类结点除了需要到根的路径上的结点的期望，还需要同层 B 类点的期望。

然后我们将所有点的期望，代入 $path(1, w)$ 路径上除 w 外任意一点的期望表达式，即可解出 x，最后将 x 代回所有点得到每个点的期望 $E(u)$。

总时间复杂度 $O(n)$。

K. Monkey Joe

不妨考虑树上每个点对于答案的贡献。
考虑点对 (x, y)，$val_y \leqslant val_x$，贡献就是 $val_x \times$ 包含 (x, y) 的路径数量。
那么对于每一个 x，只需要求出每一个 y 对其贡献系数即可。以 1 为根建树，可以将 y 分为 3 类:

1. 位于 x 的子树中
2. y 是 x 的祖先
3. y 并不在 x 的子树中且 y 不是 x 的祖先

对于第一种情况，查询 x 的每一个儿子的子树中有多少权值小于 x 的点 y 并计算贡献 (y 的贡献为 $size[y]$，即以 y 为根的子树大小)，这是一个二维数点问题。对于第二种情况，可以用类似树状数组的数据结构维护祖先到 x 的这条链。对于第三种情况，也用二维数点来进行维护并减掉 1 中的信息即可。标程在 *DFS* 序上构建了主席树来进行二维数点的维护和查询，时间复杂度为 $O(n \log n)$

L. Let's Swap

我们设取前缀 l_1 的操作为 A，取 l_2 的操作为 B。首先可以发现连续相同的两个操作可以被抵销，那么操作的方式只能为 $ABABAB\cdots$ 或者 $BABABA\cdots$。考虑到 $(AB)^{-}1 = BA$，于是最后可以归纳成两种情况：$(AB)^n$ 或者 $A(AB)^n$，其中 n 可以为负数。

然后我们可以发现，一个 AB 操作或者 BA 操作，在原串上等价于进行循环左移 $|l_1 - l_2|$ 个单位或循环右移 $|l_1 - l_2|$ 个单位。那么合法的循环位移长度为 $k|l_1 - l_2| \bmod n$，于是我们取 $d = gcd(|l_1 - l_2|, n)$，对于两种情况分别以 d 的单位循环位移 $\frac{n}{d}$ 次，用 kmp 或 hash 判断即可。