“商汤杯”不咕不孤双人挑战赛决赛题解

A.超级素数

这题是次签到题。

超级素数的限制其实比较强。直接暴搜所有的素数即可，可以发现10^{14}以内只有将近3000个超级素数。

如果不打表的话，也可以先筛出10^7以内的所有素数，在搜索的时候用判断素数。

B.树论

首先根据 SG 函数的理论，我们可以将每个棋子单独拿出来看做一个游戏，整棵树的 SG 值就是所有棋子的 SG 值的异或。那么每个节点上一旦每有两颗棋子，我们都可以删去两颗棋子，因为其 SG 值一定一样，对最后的答案的影响为 0。这时候不难发现，我们就不需要关心每个节点上的棋子的具体数量，而只需要关心这个数量的**奇偶性**就行了。

那么**对棋子个数的操作**就变成了链翻转和子树翻转（这里的翻转指棋子数量奇偶性之间的翻转）。

接下来我们先来研究一下，这个 SG 函数到底是啥。

首先，叶子节点的 SG 值显然是 0。注意到在这个游戏中一个点的 SG 值是其子树内所有点的 SG 值的 mex，那么稍加思索就能发现，其实这个 SG 函数就是其**子树内最深的叶子节点到子树根所经过的边数**。这个用数学归纳法可以轻易证明，所以这里就不赘述了。

然后我们可以分析一下，这个 SG 函数在树上有什么性质。

> 引理 1：对于所有 n 种根的状态，对于任意一个点 x，其 SG 值仅有不超过两种取值。

先考虑定义 "x 在 y 的方向"为，y 到 x 最短路径上第一个点的编号。则考虑求出每个点 x 距其最远的一个叶子节点 u，以及另一个叶子节点 v，满足 x 到 v 的距离最大且 v 在 x 的方向与 u 在 x 的方向不同。则 x 的 SG 值仅可能是 u 或者 v 到其的距离。

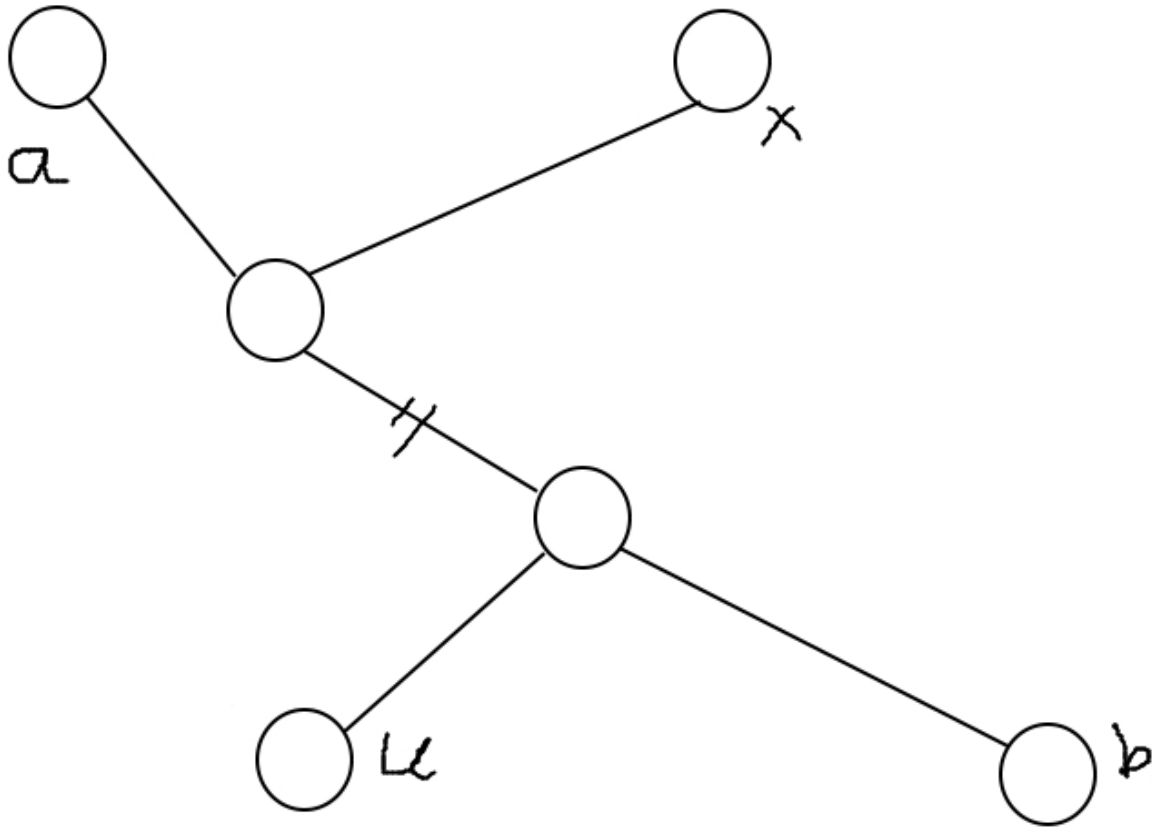
因为考虑根节点 rt，若 rt 在 x 的方向与 u 在 x 的方向不同，则 SG 值就是 u 到 x 的距离。否则，rt 为根时，u 就不在 x 的子树里头，而此时 v 就是 x 子树内距离 x 最远的叶子节点。则 x 到 v 的距离就成了 x 的 SG 值。证毕。

考虑对于一个点 x，这两种取值就是以 x 为根时，其所有儿子的子树中深度最大值中的最大值和次大值。不难想到这个问题可以使用一个简单的换根 dp 预先计算出来，故不赘述。

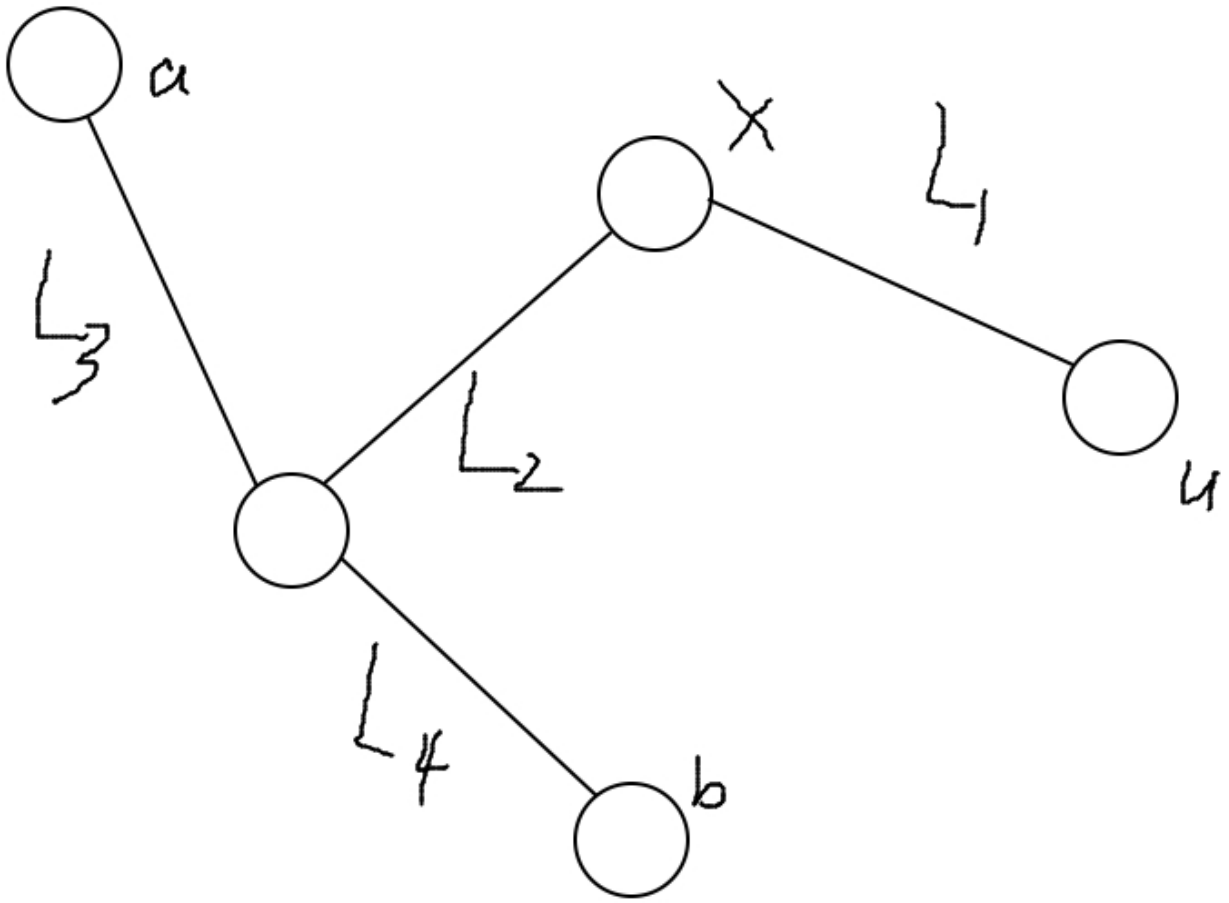
考虑每个点的 SG 值只可能取到两种取值之后，我们再深入思考这个取值的分布有没有什么规律。

> 引理 2：u 号点一定是树的某条直径的一个端点。

考虑反证法。若有一条直径是 $a \to b$，且 $x \to u$ 的长度比 $x \to a$ 和 $x \to b$ 的长度都要长。考虑 $x \to u$ 与 $a \to b$ 交与不交两种情况：



考虑交的时候，由上图可以看出，若 $x \to b$ 的长度没有 $x \to u$ 长，则 $a \to u$ 一定比 $a \to b$ 长。则不满足直径的条件。



考虑不交的情况：则有 $L_1 > \max(L_3, L_4) + L_2 > \max(L_3, L_4)$。则 $L1 + L2 + \max(L3, L4) > L3 + L4$ 一定成立，则 $a \to b$ 一定不能是直径，与假设也矛盾。

所以 u 一定是直径的一个端点。

> 引理 3：一棵树的直径中点唯一。

这个证明起来非常容易，自证不难。

发现上面几条引理之后，我们就可以想到：除了长度为奇数时的中点，树上每个点 x 的 u 号点都是往直径中点方向的！

那么当我们以直径中点为树根时，所有点（除了直径中点）的 `SG` 值都恰好取到 g_x！

那么这个时候，我们从直径中点换根至 x 时，路径上的点的 `SG` 值都恰好会变成 f_x！

那么对于换根操作，我们也可以通过默认根节点为直径中点的方式，让这个操作也变成翻转操作（这里的翻转指 `SG` 值的取值在 f 和 g 之间翻转）！

这样，我们就可以利用树剖加上线段树维护两种翻转标记，快速维护出任意点为根时，每个点的 SG 值了。

但是，我们注意到，如果直径长度是偶数的时候，直径中点在一条边上。这可不太妙。

一种可行的解决方案是建立一个虚点，中点连向两边的点，然后钦定虚点的 f 值与 g 值为 0。

当然，随便钦定两边的一个作为根，特殊考虑它的情况也是可行的。

考虑到两种操作是两个不同维度上的翻转。那么维护一个 2×2 的矩阵，维护对应情况的 SG 值的异或和即可。

C.树上游戏

如果固定一个根，那么就可以轻松使用树形DP解决了。

用 f_i 表示先手在该点子树内最优解， g_i 代表后手在该点子树内的最优解。

转移方程：$f_i = \max\{v_i - w_{i,j} + g_j\}$，后手的同理。

叶子节点 i 初值为 $f_i = g_i = v_i$ 。

然后我们考虑如何换根。

首先由于DP方程是取$\max$型的，那么我们需要在一个点同时记录最大值和次大值（不严格）。

然后在换根的时候通过比较当前dp值和最大值的关系，来考虑是否需要替换为次大值。

时间复杂度 $O(n)$。

D.子集

考虑如何求答案。

令 $l=\lfloor\frac{L-1}{x}\rfloor,r=\lfloor\frac{R}{x}\rfloor$，则 $S_{[L,R]}(x)=2^{r-l}\sum\limits_{i=l+1}^{r}ix=2^{r-l+1}\cdot x\cdot(r+l+1)(r-l)=K.$

考虑直接枚举 K 的因子进行验证，暴力枚举复杂度为 $O(\sqrt{K}\cdot T)$，无法通过

注意到一个数的因子个数不会太多，因此先对 K 进行质因数分解然后搜出所有因子。

实际上……本题出题人一开始的做法是对 l,r 进行数论分块，然后就被踩爆了……应该没人写这个做法吧……

~~（出题人已经送去ICU了）~~

E.计算几何再入门

首先对点进行极角排序，不妨令第一个凸多边形的起点是1，枚举第二个凸多边形的起点k，做dp，$dp[i][j]$表示第一个凸多边形最后一个点是i，第二个凸多边形最后一个点是j，两个凸多边形面积和的最大值，注意到如果考虑了前m个点的dp状态，那么$m-1$必然也在dp数组中的某一维中，故转移为

$$dp[i][j]+w(1,i,\max(i,j)+1)\to dp[\max(i,j)+1][j]$$
$$dp[i][j]+w(k,j,\max(i,j)+1)\to dp[i][\max(i,j)+1]$$

其中$w(a,b,c)$表示以a,b,c为三个顶点的三角形的面积，可以用叉积求出。答案就是$\max(dp[i][n],dp[n][i])$

时间复杂度$O(n^3)$，可以通过本题

F.长度

这是本场比赛的签到题。

对于每一个填写操作，将区间转化为点用单调栈存储，显然对于一个时间更晚的操作，会覆盖掉所有的操作点在本次操作右边的操作。那么插入时用单调栈维护区间起始位置随时间递增的单调性即可。

势能分析发现维护单调栈时总删除元素的次数严格小于插入元素次数，故总时间复杂度为$O(n)$，可以通过本题。

G.等比数列求和

有一个这样的结论：我们可以把 $1,2,3,\cdots,P-1$，用原根的幂来唯一的表示，即 $x^{y_1},x^{y_2},x^{y_3},\cdots,x^{y_{P-1}}$，这些幂可以被视作一个随机序列。

我们考虑分块来做。那么需要一个基本操作，假设我们已经求出一个序列分别对 P 取模后的值 $a_1,a_2,\cdots,a_s$，现在要让每一个元素都乘以 T 之后分别对 P 进行取模并求和。我们可以考虑 $a_1T \bmod P=a_1T-\lfloor\frac{a_1T}{P}\rfloor*P$，注意到 $a_i<P$，即 $\lfloor\frac{a_1T}{P}\rfloor<T$，那么我们可以枚举该值，拆出 $O(T)$ 个询问，每次询问形如一个长为 s 子区间，值在某个区间范围内的数有多少，以及他们的和是多少。在 $O((s+T)\log s)$ 的时间内解决。

那么我们考虑找到一个块 $[is,(i+1)s)$ 内最小数 $T_i=x^{k_i}$，则该块内的数可以被表示为 $(x^{is-k_i})\cdot x^{k_i},\cdots,(x^{is+s-1-k_i})\cdot x^{k_i}$，这样就可以使用上述基本操作来解决，注意到处理后的指标范围在 $[-s,s]$ 内，那么我们只需要预处理出一个长度为 $2s$ 的区间即可。

把所有询问离线下来，即可做到复杂度 $O(s\log s+\sum k_i\log s)$。我们需要估计 $\sum k_i$ 的级别。随机取 s 个数，最小数的期望大小为 $O(\frac{P}{s})$，故 $\sum k_i=O(\frac{P^2}{s^2})$。取 $s=P^{\frac{2}{3}}$，得到最优复杂度 $O(P^{\frac{2}{3}}\log P)$。

接着要想办法求出所有的 p_i，我们可以从 2 开始枚举，对这些数字求解离散对数，直到所有的 p_j 都被找到位置。求离散对数可以使用大步小步。直接进行求解会很慢，注意到我们其实只需要对所有的质数求解离散对数就可以了，合数的离散对数可以直接从质数的离散对数得到。

H.循环

通过观察或者打表可以看出

$\frac{1}{7} = 0.\dot{1}42857\dot{}$

$\frac{2}{7} = 0.\dot{2}85714\dot{}$

$\frac{3}{7} = 0.\dot{4}28571\dot{}$

$\frac{4}{7} = 0.\dot{5}71428\dot{}$

$\frac{5}{7} = 0.\dot{7}14285\dot{}$

$\frac{6}{7} = 0.\dot{8}57142\dot{}$

可以发现p等于7的时候$1, 4, 2, 8, 5, 7$分别各出现了1次

继续研究可以发现不是$p = 7$有这个性质17,19等也有这个性质，而13的循环节是6位,分子是
$1, 3, 4, 9, 10, 12$和$2, 5, 6, 7, 8, 11$分别具有这个性质

而根据价值的定义，这两个小数的价值相等。

为什么会出现循环？考虑我们做除法的过程计算$\frac{x}{p}$，先令$x' = 10x$，再把x'除以p的商当做下一位，余数当成下一次的x''，在进行下一次除法。我们可以发现，当当前x与初始x相同的时候，我们就找到了一个循环节，而在这个过程中所有出现过得x，价值都和初始的x相等！

于是我们可以把$[1, p - 1]$所有的整数划分成若干个独立的环。就相当于我们用每个环的长度算出了这个环所有x的答案均摊每算一个的复杂度是$O(1)$!

所以说我们每次只需要找一个没有算过的数，暴力算出整个环的答案然后把换上的数标记为算过了即可

时间复杂度$O(n)$,可以通过本题。

I.随机数组

单点随机加 v 的操作，可以转化为区间加上 v 除以区间长。

我们考虑同时对区间序列和数列分块，设块大小为 k ，则会分 $\frac{n}{k}$ 个块。

为了降低加操作在整块的复杂度，我们发现对数列产生的影响只和 v 相关，那么我们就用 sum_i 来记录第 i 块所加过的 v 。

并且预处理 $f_{i,j}$ 代表第 i 块内每加 1 对数列位置 j 的贡献，使用差分就可以在 $O(\frac{n^2}{k})$ 的时间内处理出来。

然后考虑块外的修改和询问，区间加操作时有 mk 次，但是查询时只有 m，差分后使用分块即可做到单次修改 $O(1)$ ，单次询问 $O(\sqrt{n})$

总复杂度为 $O(\frac{n^2}{k}+mk+m\sqrt{n})$，注意到 n,m 同阶，取 $k=\sqrt{n}$ ，总复杂度为 $O(n\sqrt{n})$

J.抱团取暖

这个问题分为两个部分，第一部分在环上，但在有熊抱在一起后，就会由环变成链，注意到链两端的熊的决策是一定的，故第二部分要解决链上的问题。

先考虑链上的问题，记长度为 n 的链期望孤独值为 $f(n)$ 。考虑每个子区间对它的贡献：

不与左右边界相邻的每一个长度为 i 的子区间是等价的，所以我们可以只枚举长度进行转移，**需要注意的是**，但如果当左右抱在一起的熊包含端点处的熊时需要除掉它决策带来的概率，这样的区间有 $n-i-5$ 和 2 个；

形似下方

O O .. | | .. O O

O O | | .. O O

O O .. | | O O

O O | | O O

所有长度小于等于 $n-2$ 的子区间与边界相邻时，只需要另一边的熊抱在一起，且由于端点处的熊决策一定，故必定是子区间内所有熊朝另一端点的方向抱，**需要注意的是**，当区间长度达到 $n-2$ 时，另一个端点熊也会被包含进来，同样需要除掉它的贡献。

形似下方

| | .. O O

O O .. | |

| | O O

这样一共会有 7 种转移，导致表达式过于冗余，故略去，细节可以参照代码。表达式可以用分治FFT进行优化至 $O(n\log^2 n)$。

代码中的 $p(i)$ 表示产生一个长为 i 的子区间的概率，此时区间左右两对熊抱在一起，而中间的 i 只熊不能抱在一起，故必定将分为左右两段，左段的熊全部向左边抱，右边的熊全部向右边抱，共有 $i+1$ 中决策，故

$$p(i)=p^2(1-p)^2\sum_{j=0}^{i}p^j(1-p)^{i-j}$$

不难看出这是一个卷积的形式，用NTT就可以在 $O(n\log n)$ 的复杂度内解决。

对于拆环成链，我们可以利用同样的思想，只需要注意熊全部往顺时针或逆时针抱的情况即可

$$ans=\left(p^n+(1-p)^n\right)ans+n\cdot q(n-2)\cdot f(n-2)+n\sum_{i=1}^{n-4}p(i)f(i)$$

其中 $q(n-2)$ 表示长度为 n 的环上仅有一对熊抱团的概率，故

$$q(n-2)=p(1-p)\sum_{i=0}^{n-2}p^i(1-p)^{n-2-i}$$

综上，我们可以在 $O(n\log^2 n)$ 的复杂度内解决第二部分问题。事实上，作为瓶颈的第二部分问题，可以利用一些trick，如 $\sum p^jq^{i-j}=q^i\sum(\frac{p}{q})^j$，将卷积优化成前缀和的形式，就可以在线性时间内解决本题。