ICPC Jinan 2022题解

A. Tower

首先注意到先加减再除二一定可以调整成先除二再加减并且步数不会变多。

其次注意到我们选择的数除二除完之后，一定会通过加减变成其中的中位数。通过这个可以发现，最后的数一定是某个数除二得到，一共有 $O(n \log W)$ 种取值。

所以枚举最后的取值，然后计算每个数变成这个取值的最小步数，然后去最小的几个即可。

直接做的时间复杂度是$O(n^2 \log W(\log W + \log n))$，可以通过本题。也可以加一些简单的优化做到 $O(n^2 \log W)$。

这个题似乎可以做到 $O(n \log^3 W)$ 左右，但看起来不一定跑得过 $O(n^2 \log W)$。

B. Torch

记前面的人是 Alice，后面的人是 Bob。定义序列 t_i，如果Alice走Bob没走，那么 $t_i = 1$，如果Alice没走Bob走了，那么 $t_i = -1$，否则 $t_i = 0$。t_i 可以看成每秒 Bob 会比 Alice 多走多少步。记两个人的距离是 $1 + s_i$，那么每次都有 $s_i = \max(s_{i-1} + t_i, 0)$。

原问题相当于多组询问 s_i 的值，因为可以根据 Alice 的位置和 s_i 算出 Bob的位置。s_i 的值实际上等于 t_i 的最大后缀和。

最大后缀和是一个可以合并的结构，只需要离散化一下，然后求出每个前缀的信息和整段的信息。回答的时候算一下多个循环节加上最后一部分的信息即可。

时间复杂度 $O(n \log n)$。

C. DFS Order 2

一个点在DFS序的什么位置只和它到根的路径上的节点的访问顺序有关。

对于一个节点 u ，假设有 m 个儿子，其中一个儿子是 v，我们需要求出它的儿子任意排列，其中 v 在第 $k + 1$ 位的方案数。可以先背包，求出在除了 v 这个儿子里面选了 i 个点，size 的总和为 k 的方案数，那么将方案数乘上 $i!(m - 1 - i)!$ 相乘就是答案。如果对于每个儿子，直接求背包，时间复杂度是 $O(n^4)$ 的。正确的做法是先求出所有儿子的背包，然后对于每个儿子，除掉这个物品即可。

求完每个点每个儿子的背包之后，对于一个点的答案，将它到父亲的路径上所有点背包并起来即可。

时间复杂度 $O(n^3)$。

Bonus：这个题可以做到 $O(n^{2.5})$。

D. Frozen Scoreboard

可以发现，每个 `?` 提交可以贡献的罚时一定是一段区间（240分钟第一次提交通过到299分钟最后一次提交才通过）。所以只需要枚举哪些提交通过了，然后看总罚时在不在对应的区间的和里即可，输出方案比较简单。

时间复杂度 $O(nm2^m)$，可以通过。也可以优化到 $O(n2^m)$ 或者更低的时间复杂度，因为本质是个子集和问题，

E. Identical Parity

注意到隔 k 个的数奇偶性一定要相同，可以把数字每隔 k 个分成一组。问题变成了能不能把 $1 \sim n$ 这些数字分到这些组里，使得每组数字奇偶性都相同。

那么一共有 $n \bmod k$ 个大小为 $\lfloor n/k \rfloor + 1$ 的组和 $k - (n \bmod k)$ 个 大小为 $\lfloor n/k \rfloor$ 的组。相当于要判断 $x \cdot (\lfloor n/k \rfloor + 1) + y \cdot (\lfloor n/k \rfloor) = \lfloor n/2 \rfloor$ 是否存在 $0 \le x \le n \bmod k, 0 \le y \le k - (n \bmod k)$ 的整数解即可。可以通过设出这个方程的通解，然后用不等式计算一下范围即可。

时间复杂度 $O(1)$。

F. Grid Points

这个题的算法比较简单，首先二分斜率，然后二分横坐标，统计对应的区域里面有多少个整点即可。

统计的时候，简单多边形可以拆成 $O(n)$ 个梯形的加减。对于每个梯形，斜率在某个范围内的点也可以拆成不超过两个梯形。对于每个梯形，用类欧几里得统计某段直线下面的整点即可。

时间复杂度 $O(n \log^2 W)$。但是实现起来比较麻烦，特别是边界和顶点的处理。

G. Quick Sort

首先可以发现，交换次数是 $O(n \log n)$的。令 $T(n)$ 表示交换次数，那么有 $T(n) = T(x) + T(n - x) + \min(x, n - x)$，其中 x 表示两段的大小。但是如果直接模拟，那么快排就会有退化的问题，所以是不行的。

那么接下来只要快速模拟这个交换的过程，或者说保证分治的时候只和短的那边复杂度相关即可。做法比较多：

- 用线段树维护区间最大/最小值，从而快速找到下一个需要换的元素。
- 维护一下每个数所在的位置和每个位置的数字。然后枚举短的那边的数字和值域在短的那边的数字所在的位置就可以确定哪些位置需要交换。

第一种方法比较好实现，但是常数较大。第二种方法不需要数据结构，只需要sort，但是写起来有点细节，因为 pivot 可能被划分到左边或者右边，但是比第一种方法快 10 倍左右。

时间复杂度 $O(n \log^2 n)$。使用第二种方法可以简单得优化到 $O(n \log^2 n / \log \log n)$ 或者更低的时间复杂度，因为比较均匀的时候可以直接模拟，不需要排序。

H. Set of Intervals

一个关键的观察是只有至多不超过 4 个区间有用，我们只关心左端点的最小值 l_1，次小值 l_2，右端点最大值 r_1，次大值 r_2 所在的那些区间。特别的，当 $n \ge 4$ 的时候，如果 l_1, r_1 不在同一个区间，那么只有 $[l_1, l_2), (r_2, r_1]$ 内的区间之间无法形成。否则左端点在 $[l_1, l_2)$，右端点在 $(r_2, r_1]$ 的区间也无法形成。

有了这个结果之后，我们只需要解决 $n \le 4$ 的问题，做法比较多，比如搜索，状压或者直接分类讨论之类的。

I. Shortest Path

主要，如果路径如果大于等于 $4n + 1$，那么我们可以找到最短的边 (u, v)，然后将起点到 u，v 到终点的路径替换成边数同奇偶的最短路，多出来的就反复走这条边即可。

所以如果边数不超过 $4n$，直接暴力做。对于大于等于 $4n + 1$ 的情况，考虑枚举最短边，根据奇偶性，能得到类似 $kx + b$ 的东西，跑个半平面交/凸包/李超线段树即可。

时间复杂度 $O(nm)$。

J. Skills

首先注意到在最优解里面，一个技能如果开始学习了，那么中间过程一定不会减到 0 以下，否则不如学习别的技能。所以一个技能开始学习了就可以忽略不能减到 0 以下的限制，不学习就扣出没学习的天数，否则不算分数。

然后注意到一个技能开始学习了，不会停止学习超过 $2\sqrt{W}+O(1)$ 天。比如一个技能 200 天没有学习，不如 100 天的时候学一下，那么会损失掉当天的收益，但是后面 100 天每天会少损失 100 点，所以不亏。

然后可以用动态规划 dp_{i,k_1,k_2,k_3} 表示第 i 天，三个技能分别有 k_1,k_2,k_3 天没有学习，其中 $k_i=0$ 表示从未开始学习过，不需要算分数。k_1,k_2,k_3 里面至少有一个是 1，所以状态数是 $O(nW)$的。

总时间复杂度 $O(nW)$。注意实现得太差（比如多乘了很多个3，或者额外一维状压每个技能有没有开始学习）会有一些常数问题。

Bonus：这个题可以做到 $O(n^2)$。

K. Stack Sort

通过分析可以发现，我们要把连续一段弹到同一个栈里。考虑两个连续的数 $x,x+1$，如果在一开始的序列里面 x 在前面，那么不能在同一个栈里，否则可以。所以答案就是统计多少个 x 在 $x+1$ 前面。

时间复杂度 $O(n)$。

L. Tree Distance

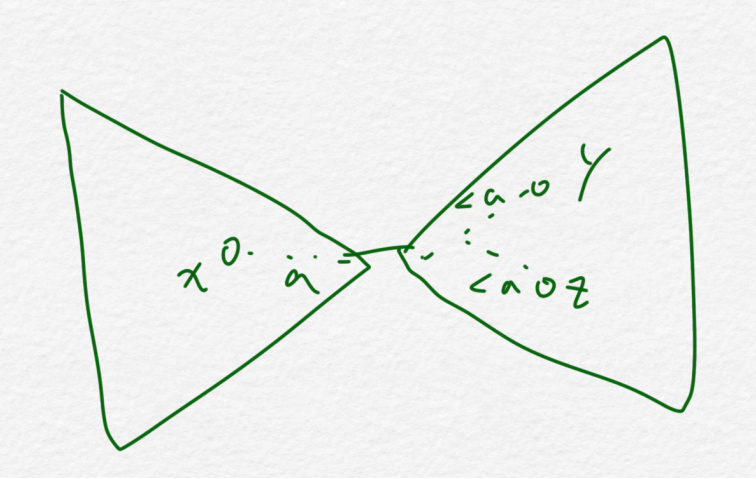
对树进行树分治，尝试找出少数支配对，使得只需要计算这些支配对对答案的贡献即可。

为了方便起见，讨论时默认采用边分治，实际上可以用其他树分治实现同样的功能。

分治时拆出左右两边两个子问题，对左边的每个 x，假设 x 距离分治中线的距离为 a，我们计算 x 于右边的每个距离分治中线的距离不超过 a 的 y 的贡献。

这里引入的条件距离分治中线的距离不超过 a 是不会让我们少统计信息的，因为假设右边的点 y' 距离分治贡献超过 a，则枚举 x 时我们不会计算 x 和 y' 的贡献，而当我们枚举到右边的点 y'，计算左边的点与 y' 的贡献时，一定会计算到这里我们没计算到的 x 和 y' 的贡献。

这里我们的结论是对每个 x，只需要计算 x 与距离分治中线的距离不超过 a 的所有点的编号的前驱后继的贡献即可支配所有点对之间的贡献。



考虑两个点 y,z，假设 $x<y<z$，则可以发现 x 和 z 在树上的距离比 y 和 z 在树上的距离大，同时 x 和 y 在编号上的距离比 y 和 z 在编号上的距离大，这里构成了支配关系，所以不需要考虑 x 和 z 的贡献，对 $z<y<x$ 同理不需要考虑 z 和 x 的贡献。

三元组 (x,y,z) 表示 x 和 y 两个点在树上距离为 z。这样树分治时，对当前树分治需要 DFS 到的每个点 x，只会有 $O(1)$ 个三元组 (x,y,z) 不被支配，总共 $O(n\log n)$ 个三元组。每次询问即给出 l,r，求被 $l\le x\le y\le r$ 中的 (x,y,z) 的 z 的 $\max$，直接扫描线+线段树维护区间 $\max$ 就可以了。

总时间复杂度 $O(n\log^2 n+m\log n)$。

M. Best Carry Player

容易发现按什么顺序都不会改变进位的次数，直接模拟即可。

时间复杂度 $O(n\log W)$。