简单 NOIP 模拟赛 题解

ZZH的游戏 (game)

首先考虑 $O(n\log n)$ 的做法。

先二分答案，接下来只需要判断是否存在一种方案，使得任意时刻两人棋子所在位置编号的和不超过 x 。

操作可以看成每次 ZZH 连续移动任意次，再由 GVZ 移动任意次。如果存在一个人连续移动结束后，棋子所在位置编号大于移动开始时的编号。考虑最后一次这样的操作，因为之后每个人连续移动结束时，棋子编号都会变小，因此每个人可以移动的区域只会扩大不会缩小。假设这次操作是 GVZ 做的，考虑删去 GVZ 这次以及之后若干次的操作，使得 GVZ 下一次操作的结束位置小于这次操作的起始位置，之后将下一次操作改为从现在的位置移到结束位置。这样之后因为 ZZH 每次操作之后位置编号都会变小，GVZ 可以移动的区域只会扩大不会缩小。因此 GVZ 的操作一定合法。而 GVZ 每次操作后的棋子位置会比以前小，因此这样一定更优。

因此一定存在一种最优操作方式，每个人连续移动结束后，棋子所在位置编号都小于移动开始时的编号。

因此每个人可以移动的区域只会扩大不会缩小。此时可以发现，每次移动到当前能到的点中编号最小的一定最优。

考虑在两棵树上分别记录当前可以移动到的点，以及与可以移动到的点相邻但因为权值限制不能到达的点。记录当前两个人分别可以移动到的点编号的上限。轮流操作两棵树，每次处理这棵树之前的上限到现在的上限区间中，所有与可以移动到的点相邻但因为权值限制不能到达的点，从这些点开始dfs，同时更新上面的东西以及这棵树能到的点中的最小编号，再更新另外一棵树能移动到的点的编号上限。

如果两棵树都能到1则存在合法方案。如果在连续的两次操作中，两棵树的最小编号都没有变化，则不存在合法方案。

这样就得到了 $O(n\log n)$ 的算法。

考虑初始设答案 $x=s+t$ ，开始进行上面的操作。如果出现了不能继续移动的情况，则令 x 加一，并更新两棵树可以移动到的点编号的上限，之后继续操作。当两棵树都到达1时， x 就是答案。

复杂度 $O(n)$

ZZH与背包 (knapsack)

考虑折半搜索。预处理后 $n-S$ 个物品是否选得到的所有情况的体积排序后的结果，对于每一个询问，枚举前 S 个物品是否选，然后在前面排序的结果上二分。

这样的复杂度为 $O(\sqrt{q2^n}*(n+\log q))$

考虑第一部分，实际上需要做的相当于给出 n 个物品，将 2^n 种所有物品选或不选的情况得到的体积和排序。

假设当前已经对于前 i 个物品，将 2^i 种情况排好了序，考虑加入第 $i+1$ 个物品。

此时的 2^{i+1} 种情况可以分为两种：

1. 不拿第 $i+1$ 个物品，这部分已经排好序了。
2. 拿第 $i+1$ 个物品，注意到这部分的每个体积和减去 v_{i+1} 后与上一部分的体积和 一一对应，因此这部分排序之后的结果等于上部分排序后将每个数加上 v_{i+1} 的结果。

接下来再将这两部分归并即可，这一步的复杂度为 $O(2^{i+1})$

因此将后 $n-S$ 个数的 2^{n-S} 种情况排序的复杂度为 $2^1+2^2+\ldots+2^{n-S}=O(2^{n-S})$

然后考虑将第一部分排序，此时拿一个物品相当于将 l,r 减去 v_i 。

先将所有询问拆开，变为询问总体积不超过 $l-1,r$ 的方案数。然后将这 $2q$ 个询问排序。之后依次考虑每个物品选不选，可以使用上面的方式将每一步变为两个数组归并。

因为权值总和只有 4×10^{10} ，询问编号不超过 1000 ，可以将这两个放进一个 `long long` 以减小常数/空间。

复杂度 $O(\sqrt{q2^n})$

ZZH与计数 (counting)

首先考虑只有一个 v_i 非0的情况，可以发现有如下结论

对于一个数 i ，在若干次操作后，对于每一对 x,y ，所有满足在 i 为 1 的位上 1 的个数为 x ，在 i 为 0 的位上 1 的个数为 y 的数，i 变为它们中的任意一个的概率是相同的。

这个结论在 $m=0$ 时显然成立，且如果结论对于 $m=i$ 成立，容易推出结论对于 $m=i+1$ 成立，因此结论对于所有 m 都成立。

考虑dp，设 $dp_{t,j,k}$ 表示操作 t 次后，变为在 i 为 1 的位上 1 的个数为 j ，在 i 为 0 的位上 1 的个数为 k 的数的概率，这样的状态数是 $O(k^2)$ 的，可以 $O(k^4)$ 枚举所有情况。得到转移方程，然后矩阵快速幂得到 m 次之后的情况。

对于一般的情况，可以发现 $bitcount$ 相同的数的dp是一样的。枚举 $bitcount(i)$ ，对于每一种做一遍矩阵快速幂，就可以得到对于每一个数 i ，以及每一对 x,y ，i 变为一个满足在 i 为 1 的位上 1 的个数为 x ，在 i 为 0 的位上 1 的个数为 y 的数的概率。这部分的复杂度是常数极其小的 $O(n^7\log m)$

接下来考虑算出期望，对于一对 i,x,y ，设 $bitcount(i)=ct$ ，它会贡献到的数是所有将 i 二进制表示中 y 个位从 0 变成 1 ，将 $ct-x$ 个位从 1 变成 0 （一位只能操作一次），可以得到的所有数，称i 变为一个满足在 i 为 1 的位上 1 的个数为 x ，在 i 为 0 的位上 1 的个数为 y 的数的概率为 $(i,y,ct-x)$ 的贡献。

考虑一位一位的dp，设 $f_{i,s,j,k}$ 表示当前考虑了前 $i-1$ 位的变化，当前的数是 s ，还需要将 j 个位从 0 变成 1 ，将 k 个位从 1 变成 0 ，所有能转移到这个状态的 (i,x,y) 的贡献和。可以发现答案就是 $f_{n+1,0,0,0},\ldots,f_{j+1,2^n-1,0,0}$

初值 $f_{0,s,j,k}$ 即为 $(s,k,bitcount(s)-j)$ 的贡献。考虑第 i 位的转移

如果 s 的二进制第 i 位为1，则有

$$f_{i+1,s,j,k}=f_{i,s,j,k}+f_{i,s-2^i,j+1,k}$$

否则，有

$$f_{i+1,s,j,k}=f_{i,s,j,k}+f_{i,s+2^i,j,k+1}$$

可以发现，对每一个 i dp时，只有 $s,s\ xor\ 2^i$ 两个位置会互相影响，因此数组可以只开后三维，每次对于互相影响的两个位置直接计算新的 f 。

复杂度 $O(n^3 2^n)$ ，如果只计算有用的状态 $(j+k\le n-i)$ ，有一个较小的常数。

总复杂度 $O(n^7\log m+n^3 2^n)$

ZZH的旅行 (journey)

设 dp_i 表示从 i 出发并游览 i ，得到的最大有趣度。

那么可以得到暴力的转移：

$dp_u = max(0, max_{v \in subtree\ of\ u}(dp_v + (a_u - dis(u,v))b_v))$

记 $d_i = dis(1,i)$ ，则有 $dp_u = max(0, max_{v \in subtree\ of\ u}(dp_v - b_v d_v + (a_u + d_u)b_v))$

因此可以将点 u 看成一条 $y = dp_u - b_u d_u + b_u x$ 的直线，转移可以看成在子树内所有直线在 $x = a_u + d_u$ 处 y 的最大值。

从下往上做这个过程，相当于在每个子树内维护一个凸壳，支持合并两个凸壳以及在凸壳中加入一条直线。

这部分可能有多种做法，这里只描述出题人的做法。

考虑维护一棵动态开点的李超树，因为询问只有 n 次，可以支持 $O(\log n)$ 的插入和询问。

考虑合并两棵李超树，类似线段树合并的方式，先合并两侧子树，对于这个点上的两条直线，保留这个点对应区间值点上值较大的一条，将另外一条插入到这个点的子树中。

考虑这样的复杂度，合并时每经过一个两棵树上都有线段的点，一定会有一条线段插入到子树中。

插入的时候每向下移动一次，插入的线段最后在树上的深度就会增加 1。

因为深度最多只有 $O(n\log n)$ ，因此最多插入 $O(n\log n)$ 次，且深度增加的和不超过$O(n\log n)$

因此这样的复杂度是 $O(n\log n)$