2024“钉耙编程”中国大学生算法设计超级联赛（4）题解

By Claris

2024 年 7 月 29 日

1 超维攻坚

Shortest judge solution: 2707 Bytes.

如果一个点集最终存活，那么它的凸包边界上及内部的所有点一定都存活。$O(2^n)$ 枚举一个点集 S，计算它的凸包，将边界上和内部的点都找到，标记 S' 为可行解即可。需要特判凸包退化为一个平面或者一条直线的情况。

求凸包可以通过枚举三个点 A, B, C 来得到所有面，如果所有点都在它同侧那么它是凸包上的一个面。时间复杂度 $O(2^n n^4)$，常数为 $\frac{1}{96}$。

2 黑白边游戏

Shortest judge solution: 3055 Bytes.

博弈 DP，设 $f[i][S]$ 表示当前处于第 i 轮，在该轮操作之前图中每条边的颜色情况是 S 时，经过后续操作先手与对手分差的最大值，则 $f[i][S] = \max\{w(a_i, b_i, T) - f[i+1][T]\}$，其中 T 与 S 仅有一条边颜色不同，$w(a, b, S)$ 表示黑边边权为 a、白边边权为 b 时 S 图中所有点对的最短路长度之和。

共有 $\frac{n(n-1)}{2}$ 条边，S 共有 $2^{\frac{n(n-1)}{2}}$ 个，时间复杂度为 $O(2^{\frac{n(n-1)}{2}} n^2 m)$，不能接受。

注意到任意调换 S 中点的编号不会影响计算，不妨把黑边看成不存在的边、白边看成存在的边，则只需要考虑所有不同构的图作为状态，当 $n = 8$ 时只有 12346 个状态。

接下来考虑如何预处理出所有不同构的图以及它们之间的所有转移。对于一个图 G，计算每个点的度数 d_i 以及每个点所有邻居的度数可重集 d'_i，首先将所有点按照二元组 (d_i, d'_i) 排序，如果一段二元组相等，则对这一段点的顺序进行全排列搜索，并在所有搜索出的情况中取邻接矩阵字典序最小的图作为 G 的最小表示。显然两个图同构当且仅当它们的最小表示相等，只需将所有图都转化成最小表示。于是可以从 0 条边开始，按照 $1, 2, \ldots, \frac{n(n-1)}{2}$ 的顺序依次扩展出 $k = 1, 2, \ldots, \frac{n(n-1)}{2}$ 条边的所有不同构的图的集合，并记录对应转移。

3 最优 K 子段

Shortest judge solution: 1264 Bytes.

二分答案，判断能否划分出 k 个不相交子段使得每段长度都是质数且权值和至少为 mid。

设 $s_i = a_1 + a_2 + \cdots + a_i$，从左往右贪心进行划分：维护一个集合 T，从左往右依次往 T 中加入每个下标，假设当前考虑到了下标 i 但是还没往 T 中加入 i，如果此时发现 T 中存在一个 j 满足 $s_i - s_j \geq mid$ 且 $i - j$ 是质数，说明当前已经可以得到一段满足条件的子段，将 T 清空。重复上述过程直至找到 k 个满足条件的子段。在此过程中，容易发现只需考虑 $i - j$ 是质数且 s_j 最小的 j。如果用 `std::set` 按 s 维护 T，那么只需从小到大暴力枚举前若干小的作为 j，直至 $i - j$ 为质数，这是因为数据随机分布，结合质数密度可得期望需要遍历 $O(\log n)$ 个 j。

时间复杂度 $O(n \log n \log ans)$。

4 分组

Shortest judge solution: 1360 Bytes.

设 $L = A \cup B, R = C \cup D$。规定 a_1 一定属于 L，$O(2^{n-1})$ 枚举剩下 $n-1$ 个数每个数分别属于 L 还是 R。对于 L 以及 R 内部的划分成两个集合的方案，可以使用 01 背包。

设 L, R 的元素异或和分别为 sum_L, sum_R，枚举 L 的一个可能的子集异或和 A 以及 R 的一个可能的子集异或和 C，则 $B = A \oplus sum_L, D = C \oplus sum_R$。不妨强制规定 $w_A \geq w_B, w_C \geq w_D$，有两种可能的情况：

- $w_A \geq w_C$，则极差为 $w_A - \min(w_B, w_D)$，需要维护 w_D 的最大值。
- $w_A \leq w_C$，则极差为 $w_C - \min(w_B, w_D)$，需要维护 w_B 的最大值。

按 w 排序后记录维护 w_B 以及 w_D 的前缀最大值即可。

时间复杂度 $O(2^{n+m-1})$。

5 多层血条

Shortest judge solution: 708 Bytes.

签到模拟题，只需暴力处理第 $hp - m + 1$ 至第 hp 点生命值。

6 延时操控

Shortest judge solution: 2218 Bytes.

由于己方和敌方之间不会互相成为障碍，可以将 m 减去 k，把问题转化为 $k = 0$ 即无延迟的情况，最后再让己方游走 k 步。设 $f[i][px][py][dmg][ex][ey]$ 表示目前进行了 i 个回合，己方位于 (px, py)，敌方扣除了 dmg 点生命值，且敌方位于 (ex, ey) 的方案数，枚举下一轮的指令进行转移，时间复杂度 $O(mn^4hp)$，不能接受。

令 (dx, dy) 表示初始情况下己方与敌方的横纵坐标差，注意到 (ex, ey) 与 $(px + dx, py + dy)$ 的曼哈顿距离不超过 dmg，因此只需记录 (ex, ey) 相对 $(px + dx, py + dy)$ 的坐标差而不用记录 (ex, ey)，大幅降低状态数。

时间复杂度 $O(mn^2hp^3)$。

7 序列更新

Shortest judge solution: 1065 Bytes.

取 lim 为 b 中第 x 大元素，对于每次操作，考虑以下两种维护手法：

- 枚举 a 中所有值不超过 lim 的下标 i，用 $b_{(i+k) \bmod n}$ 更新它。
- 枚举 b 中所有值大于 lim 的下标 i，用它更新 $a_{(i-k) \bmod n}$。时间复杂度 $O(x)$。

由于数据随机，考虑计算 a 中一个位置期望需要经历多少次第一类操作，它的值才会大于 lim。假设经历了 i 次操作仍然不超过 lim，等价于 $i+1$ 个随机数均不超过 lim，概率为 $(1-\frac{x}{n})^{i+1}$，故期望操作次数为：

$$\sum_{i\geq 0}(1-\frac{x}{n})^{i+1} \approx \frac{1}{\frac{x}{n}} = \frac{n}{x}$$

综上可得期望时间复杂度为 $O(\frac{n^2}{x}+nx)$，当 $x=\sqrt{n}$ 时为最优复杂度 $O(n\sqrt{n})$。

8 魔法卡牌

Shortest judge solution: 4976 Bytes.

建一张 n 个点的有向图，分别表示每张卡片，连边方式如下：

- 若 i 选择正面可以确定 j 的朝向，连黑边 $i \to j$。
- 若 i 选择背面可以确定 j 的朝向，连白边 $i \to j$。

根据初始连边信息，可能存在一些点只有一种可行的朝向，将这些点优先处理掉。接着忽略所有重边后，称一个点的出度为它连出去的黑白边数量之和。如果所有点的出度都不超过 2，那么把图视作无向图后，每个连通块要么是环，要么是链，可以分别使用环形 DP 或者线性 DP 在 $O(n)$ 时间内求出答案。否则，令 $T(n)$ 表示 n 个点的图的求解时间复杂度：

- 如果存在一个点的出度至少为 4，即 k 条黑边，$4-k$ 条白边，那么它选择正面可以确定 $k+1$ 个点，选择背面可以确定 $4-k+1$ 个点，有

$$\begin{aligned} T(n) &= \max_{k=0}^{4}\{T(n-1-k)+T(n-1-(4-k))\} \\ &= \max\{T(n-1)+T(n-5), T(n-2)+T(n-4), T(n-3)+T(n-3)\} \end{aligned}$$

- 否则，所有 n 个点的出度都不超过 3。观察连边方式，对于一个点的两种朝向，它的邻居要么直接确定，要么出度减少 1。设 $F(i,j,k)$ 表示一度点、二度点、三度点的个数分别为 i,j,k 时的最坏时间复杂度，枚举所有可能进行转移，可以发现 $F(i,j,k) \leq T(i+j+k)$。

当图中存在出度至少为 3 的点时，选择两种朝向对应的剩下局面的 T 值之和最小的点递归求解子问题即可。需要注意的是确定一个点的朝向后，可能会带来一系列连锁反应确定更多点的朝向，可以通过位运算模拟 BFS 来实现。

时间复杂度 $O(T(n))$，$T(60) \approx 1.4 \times 10^7$，而且只是个上界，通过对图结构更深入的分析可以得到更好的结果。

9 昵称检索

Shortest judge solution: 1300 Bytes.

对于一个“昵称”来说，它由名字和生日两个部分拼接而成，如果它是 S 的子序列，那么名字匹配的位置应该尽可能靠左，生日匹配的位置应该尽可能靠右。

预处理出 $nxt[i][j]$ 表示 $S[i,n]$ 中字符 j 最靠左的出现位置的下标。利用 nxt 数组，对于每个名字可以求出从左往右贪心匹配时，它的最后一个字符在哪个下标最小的位置匹配上，记为 a_i；同理可以倒着贪心求出每个生日的第一个字符在哪个下标最大的位置可以匹配上，记为 b_i。

问题现在转化为统计有多少对 (i,j) 满足 $a_i < b_j$。枚举 i，利用前缀和统计 j 的个数即可。

10 矩阵的周期

Shortest judge solution: 2373 Bytes.

将 $\mathbf{M}$ 看作一张 n 个点的有向图的邻接矩阵，则 $f(i,j,k)$ 表示是否存在一条从点 i 出发到点 j 的经过恰好 k 条边的路径。枚举起点 i 终点 j，分类讨论：

- 如果 i 无法到达 j，那么显然答案 $=1$。
- 如果 i 和 j 在同一个强连通分量内，那么该强连通分量内存在若干个长度不超过 n 的简单环，周期为所有环长的 gcd，说明答案不超过 n。
- 否则，考虑 i 到 j 的某条路径经过的那些强连通分量，这条路径的周期为途径的所有强连通分量周期的 gcd。此时继续讨论两种情况：
  - i 所在强连通分量或 j 所在强连通分量至少有一个的大小不为 1：所有路径的周期都是起点或终点所在强连通分量周期的约数，因此答案不超过 n。
  - 否则，考虑所有 i 到 j 除去起点终点外不相交的路径，答案不超过这些路径周期的 lcm，可以粗略用所有可能途径的强连通分量大小的 lcm 作为上界。

现在来考虑一个点对的答案上界是多少，假设除去起点和终点还剩下 k 个点，那么问题等价于找出一些正整数使得它们总和不超过 k，且 lcm 最大，记为 $F(k)$，可以 DP 求出 $F(60-2)=510510$。

现在再来考虑所有点对的答案上界之和是多少，在此忽略上界显然是 n 的点对，那么最坏情况下有 k 个起点、k 个终点，剩下 $n-2k$ 个点构成途径点，可估计出所有点对的答案上界之和 $G(n)=\max_{k=1}^{\lfloor \frac{n}{2} \rfloor}\left\{k^2F(n-2k)\right\}$，可得 $G(60)=4504500$。

首先求出 $\mathbf{M}^t$，使步数充分大以保证进入周期。可以通过同余最短路证明 $t \le O(n^2)$；方便起见，也可以直接通过快速幂求出 $\mathbf{M}^{2^{64}}$。

枚举起点 i，估计出它到每个终点的答案上界 $bound_j$，找出其中的最大值 k，暴力从 $\mathbf{M}^t$ 开始模拟 $2k$ 步，然后对于每个 j 通过 KMP 算法求出前 $2\cdot bound_j$ 位的字符串的最小周期即可。每一步模拟中，如果进行朴素的矩阵乘向量，是 $O(n^2)$ 的。一方面可以通过位运算除掉一个 w；另一方面可以将点的编号按 $\log n$ 进行分块，对于每块 $O(2^{\log n})=O(n)$ 预处理出每个子集的边表的并集，在矩阵乘向量时除掉一个 $\log n$。

时间复杂度 $O(G(n)+\frac{n^3F(n-2)}{w\log n})$。

11 找环

Shortest judge solution: 3201 Bytes.

Karp 算法，建立超级源点 0，向 1 到 n 连边权为 0 的单向边，设 $f[i][j]$ 表示从源点出发经过 i 条边到达 j 的最短路，则：

$$ans = \min_{i=1}^{n}\{\max_{j=0}^{n}\frac{f[n+1][i]-f[j][i]}{n+1-j}\}$$

其中 $f[n+1][i] \neq +\infty$。

由于边权是 998244353 的 0 到 $n-1$ 次幂，在 998244353 进制下考虑 $f[][]$ 的计算过程，只会涉及单点加 1、字典序比大小以及拷贝，可以使用可持久线段树维护区间 Hash 值来实现。

在最后计算答案时，可以通分后比较分子来实现，此时只涉及两棵线段树的减法以及乘以一个正整数，都可以很方便地计算 Hash 值。需要注意的是，计算过程中可能出现某些位的差值变为了负数，此时不需要考虑退位，因为这种情况一定不优。

时间复杂度 $O((nm+n^2)\log n)$。

12 寻找宝藏

Shortest judge solution: 1750 Bytes.

设 f_i 表示 $(0,0)$ 到 (i,p_i) 的最大收益，则 $f_i = v_i + \max_{j<i, p_j<p_i}\{f_j\}$，可以通过扫描线 + 树状数组在 $O(n\log n)$ 时间内求出。同理可以求出 g_i 表示 (i,p_i) 到 $(n+1,n+1)$ 的最大收益，再令 h_i 表示经过 (i,p_i) 的最大收益，有 $h_i = f_i + g_i - v_i$。

对于一个询问中的禁区矩形 $[x_1,x_2]\times[y_1,y_2]$，答案来自以下几种情况：

- 经过了某个坐落在 $[x_2+1,n]\times[1,y_1-1]$ 的点 x，答案为 h_x。
- 经过了某个坐落在 $[1,x_1-1]\times[y_2+1,n]$ 的点 x，答案为 h_x。
- 先经过 $[1,x_2]\times[1,y_1-1]$，再经过 $[x_2+1,n]\times[y_1,n]$。令前者最后一个经过的点为 a，后者第一个经过的点为 b，则答案为 f_a+g_b。
- 先经过 $[1,x_1-1]\times[1,y_2]$，再经过 $[x_1,n]\times[y_2+1,n]$。令前者最后一个经过的点为 a，后者第一个经过的点为 b，则答案为 f_a+g_b。

以上四种情况均能通过扫描线 + 树状数组在 $O((m+n)\log n)$ 时间内求出。