2021 年度训练联盟热身训练赛第一场题解

兰州大学

2021 年 3 月 7 日

A. Weird Flecks, But OK

应用最小圆覆盖算法三次（每个平面一次）。`Welzl` 算法的运行时间为 $\mathcal{O}(N)$（预期）。其他方法也同样有效。

可以发现，一个圆可以由两个对映点或三个非共线点唯一地确定。

朴素的方法：对于每个平面，尝试由 $\mathcal{O}(N^2)$ 和 $\mathcal{O}(N^3)$ 枚举确定所有圆。对于每个圆，检查是否所有 N 个点都在里面。

这个方法是 $\mathcal{O}(N^4)$，因为 $N \leq 5000$ ，所以太慢了。

一旦我们将问题简化为二维，我们就可以通过二分搜索最小封闭圆的半径。

对于任何覆盖所有点的有效圆，我们可以平移它，使至少两个读入的点位于圆的边界上。然后，对于每个点，考虑该点在其边界上的所有圆。每个点都包含在一定的角度范围内。如果存在一个包含所有点的角，那么这个半径是可以达到的。

时间复杂度：$\mathcal{O}(N^2 \log(1000))$。

考虑以下函数：$f(x,y)$ 是圆心为 (x,y) 并覆盖输入中的所有点的圆的最小半径。对 $f(x,y)$ 求一次值只需 $O(N)$ 时间。

现在我们把问题简化为求 $f(x,y)$ 的最小值。

我们可以通过做两个嵌套的三元搜索来做到这一点：对于一个固定的 x，$f(x,y)$ 是凸函数。

加速这个解决方案的一个启发式方法是在计算 $f(x,y)$ 的值之前先取点的凸包。

时间复杂度：$\mathcal{O}(N\log^2(1000P)$。

从集合 P 和空集合 R 中的所有点开始，这里 R 定义为一组边界定义点（最多 3 个）。

$Welzl(P,R)$：如果 P 为空或 $|R|=3$，返回由 R 定义的最小圆。

随机选择点 $p\in P$。

用 $P\backslash\{p\}$ 和 R 递归。

如果生成的圆 C 包含 p，则返回 C。

否则，用 $P\backslash\{p\}$ 和 $R\cup\{p\}$ 再次尝试（递归），并返回那个圆。

时间复杂度：预期时间为 $\mathcal{O}(n)$。

B. Code Names

这似乎是一个 NP-hard 问题——寻找最大独立集。

我们可以把这个问题简化为二部图最大匹配（最大流）问题。

识别出二部结构，构造边，利用网络流找到最大匹配。

定义：如果两个词不是 `swap-free` 的，它们就是邻居。

识别出二部图结构。

- 构造一个邻居之间有边的图。
- 每一对单词都有着相同的字母但是字母顺序不同，类似与排列。
- 排列具有奇偶性——等于逆序数的奇偶性。
- 交换（任何）两个字母都会改变奇偶性。因此，如果两个词具有相同的奇偶性，它们不能相邻；如果两个词具有不同的奇偶性，它们可能相邻。

在 $\mathcal{O}(n^2)$ 的时间内构造出这个图。

一旦我们定义了 n 个单词上的二部图结构，最大独立集数就是 $n-m$（其中 m 是二部图中的最大匹配数）。

使用任何算法进行最大二部匹配，可以在时间复杂度最差为 $\mathcal{O}(n^3)$ 中找到 m。

C. New Maths

递归地从最高有效位回溯到最低有效位，反之亦然。

可知，N 的最高有效位由 a 的最高有效位决定；同样，N 的第二有效位由 a 的第两有效位决定。

设 a_k 为 a 的最高有效位。

那么由于 $a_k^2 \equiv N_{2k} \pmod{10}$，这限制了 a_k 的选择最多只有两种。

类似的性质也适用于 a_i：a_i 的选择受到所有 $j > i$ 和取模同余的 a_j 的选择的限制。

测试可以确认进行通过递归回溯找到 a 的可行数字的速度足够快。

D. Some Sum

考虑连续四个数，由于一定只包含 2 或 2 的倍数个奇数和 2 或 2 的倍数个偶数，那么他们的和一定是偶数。

而一个奇数和一个偶数的和一定是奇数，那么数字个数模 4 为 2 时和一定为奇数。

其余情况可以通过移位改变奇偶性，因此为 Either。

E. Early Orders

我们按顺序从前往后在这 n 个数之中选 k 个数出来，一个自然的想法就是每次尽量选取字典序最小的数，让较大的数在答案序列中尽量靠后。

我们可以用类似单调队列的思想去实现:

用 `ans[]` 存储答案，当枚举到第 i 个数 a_i 的时候，当 `ans[]` 之中没有 a_i 时，将其与 `ans[]` 的最后一位 b 做比较，如果 $a_i < b$ 且 a_i 之后还有 b，那么删去 b，接着比较，直到在 `ans[]` 中找到不满足条件的数为止，此时插入 a_i，枚举下一个数。

F. Pulling Their Weight

考虑应用前缀和。

设 $sum(x) = \Sigma_{a_i \le x} a_i$, $A = \max_{1\le i\le n}\{a_i\}$, 则所求即为满足 $sum(x-1) = sum(A) - sum(x)$ 的 x 的最小值。预处理出 $sum(1)...sum(A)$，之后由低到高枚举 x 验证即可。

G. Birthday Paradox

令 $m = \sum_{i=1}^{n} c_i$，即总人数。易得总情况数为 365^m。

对于符合题意的情况数，

- 我们可以首先考虑出现的日期的情况数，为 C_{365}^{n}。
- 然后考虑将所有人分为符合题意的 n 组的情况数（排除组内重复），为 $\frac{m!}{\prod_{i=1}^{n}(c_i!)}$。
- 最后考虑将确定的 n 个日期与 n 个组一一对应的情况数，考虑到某些 c_i 可能相同，具有相同 c_i 的组之间相互交换对应日期和不同的分组方式之间存在重复，在此引入数组 d，表示数组 c 中所有不同的 c_i 出现的次数（例：若 $c = (1, 1, 1, 5, 5)$，则 $d = (3, 2)$），则计算可得此时情况数为 $\frac{n!}{\prod(d_j!)}$。

综上可得，答案为

$$\log_{10}\left(\frac{1}{365^m} \cdot C_{365}^{n} \cdot \frac{m!}{\prod_{i=1}^{n}(c_i!)} \cdot \frac{n!}{\prod(d_j!)}\right)$$

利用对数的运算性质，可以预处理出 1 到 36500 的阶乘以 10 为底的对数。时间复杂度为 $\mathcal{O}(n)$。

H. On Average They’re Purple

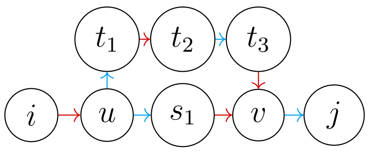
本题看似复杂，实则不难。

由题意得，Alice 希望 Bob 所走路径上颜色改变的最小次数最大，我们不妨假设所走路径相邻两条边颜色总是不同的。此时不难发现，走从 1 到 n 的最短路最优。当最短路径长度为 k 时，最小次数为 $k-1$，且 Bob 在这种情况下答案不可能比 $k-1$ 更大。那么对于 Alice 而言，只要保证每条路径上颜色改变的次数都不小于 $k-1$ 即可取得最小次数上限 $k-1$。不难发现，符合要求的构造方案总是存在的。

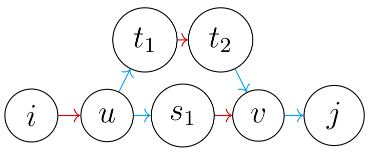
本题可以 BFS 求得最短路长度。时间复杂度 $\mathcal{O}(n+m)$。

构造参考思路：

- 对于最短路 s 的一条扩展路径 t（从 u 到 v），若扩展路径与最短路在 u,v 之间的距离相差偶数：



- 对于最短路 s 的一条扩展路径 t（从 u 到 v），若扩展路径与最短路在 u,v 之间的距离相差奇数：



不难发现，途径非最短路颜色改变次数总不小于途径最短路颜色改变次数。

综上，最小次数上限 $k-1$ 总能取得，故答案为 $k-1$。

I. Full Depth Morning Show

设该树的根节点为 u，

$$
\begin{aligned}
a_u &= \Sigma_v d_{u,v} \\
b_u &= \Sigma_v t_v d_{u,v}
\end{aligned}
$$

则对于节点 u，答案为 $t_u a_u + b_u$。

考虑与 u 相邻的节点为 u'，设以 u' 为根节点的子树节点集（包括 u'）为 $S(u')$，则有

$$\begin{aligned} a_{u'} &= a_u + (n - 2|(S(u'))|)d_{u,u'} \\ b_{u'} &= b_u + (\Sigma_v t_v - 2\Sigma_{v\in S(u')} t_v)d_{u,u'} \end{aligned}$$

J. This Ain't Your Grandpa's Checkerboard

签到题

直接模拟即可

K. Solar Energy

注意到 $f_i(a)$ 是由一系列一次函数和常函数构成的分段函数，$f(a)$ 自然也是，而且 $f(a)$ 的分界点集 D 即为所有 $f_i(a)$ 分界点集 D_i 的并集。

显然答案为 $\max_{x\in D} f(x)$。

接下来我们只需考虑如何快速求 $f(a)$ 分界点的函数值了。

令 $D = \{d_1, d_2, \ldots, d_m\}$，$k_i$ 为 $f|_{x\in[d_i, d_{i+1}]}$ 的斜率。

则 $f(d_{i+1}) = f(d_i) + k_i(d_{i+1} - d_i)$。

而 k_i 实际上是很容易求的，这样我们便可以 $\mathcal{O}(m) = \mathcal{O}(n)$ 递推了。

下面举一个例子：

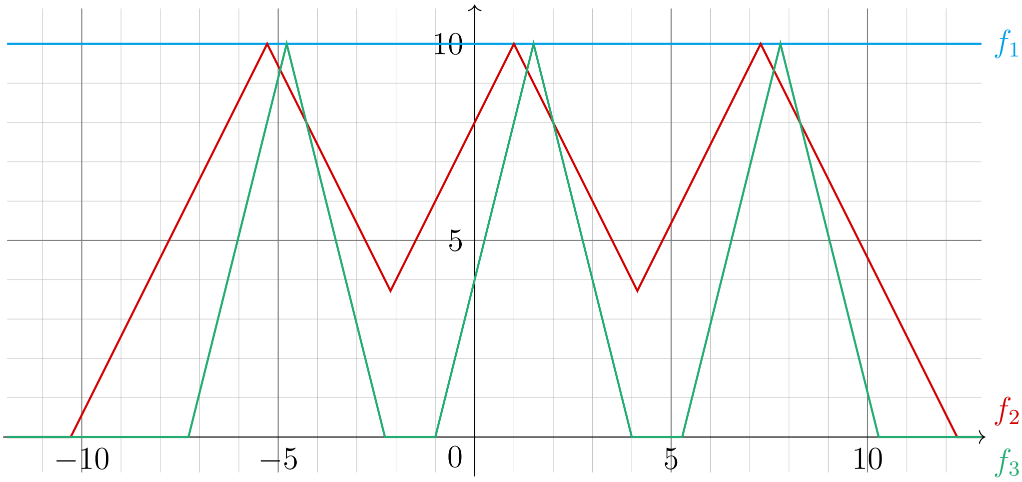
输入：

```
3
10 0 0.5
10 2 1
10 4 1.5
```

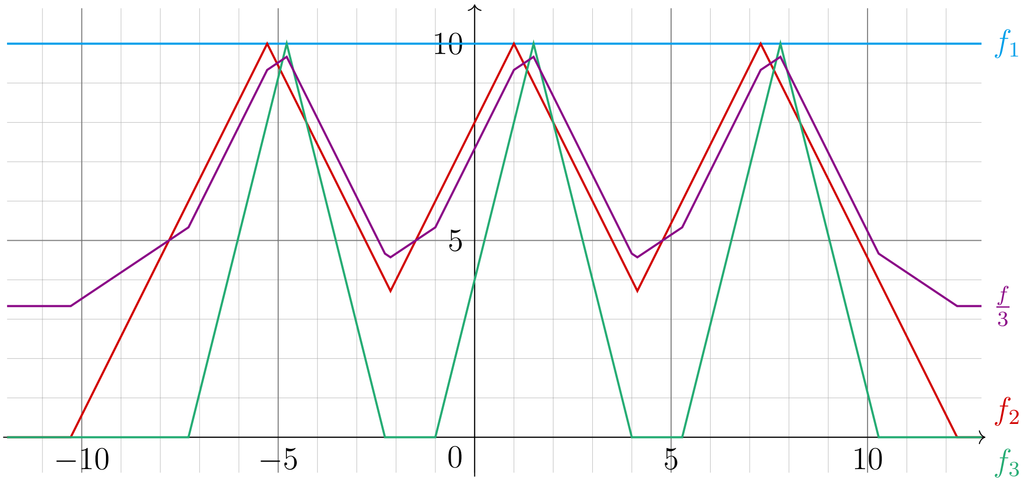
我们列出对应函数：

$$\begin{aligned} f_1(a) &= \max\{0, 10\} \\ f_2(a) &= \max\{0, 10 - 2|a - 1|, 10 - 2|2\pi - |a - 1||\} \\ f_3(a) &= \max\{0, 10 - 4|a - 1.5|, 10 - 4|2\pi - |a - 1.5||\} \end{aligned}$$

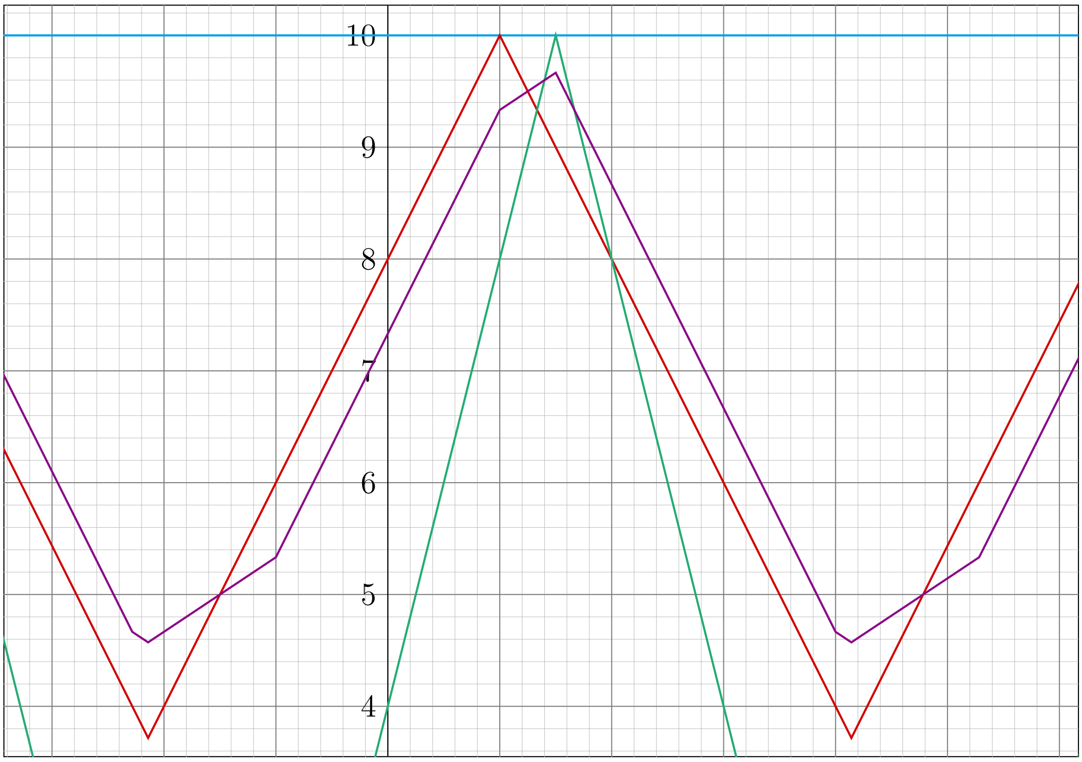
画个图：



我们把 f 也加进去：



放大一些：



可以证明 f 的图像存在“局部周期”，即

$$(\exists a, t \in \mathbb{R}, \forall x \in [a, a+t)), f(x) = f(x+t)$$

而且在不考虑 $f \equiv 0$ 的那些情况下，只有有限组 a, t。因此我们没有必要枚举所有的 d_i，只需要取遍其所有函数值不恒为 0 的“局部周期”即可，我们可以通过选择 f_i 的一部分进行加和来完成该操作。

一种可行的操作是只选 f_i 沿 a 正方向的后两个“峰”，即

- 对 `_/\_/\_/\_` 型图像，只需取 `/\_/\_` 部分即可。
- 对 `_/W\_` 型图像，只需取 $\wedge$`/\_` 部分即可。

记录对应间断点和斜率变化，即可 $\mathcal{O}(n)$ 递推