2024 省选模拟赛

目录

A 红岸基地（base）

1. 算法 1

枚举答案, 复杂度 $O(2^n)$, 期望得分 20 。

2. 算法 2

输出无解, 期望得分 5。

3. 算法 3

考虑 Subtask 4, 即设 $n = 2^a \times x, k = 2^b \times y$, 其中 x, y 是奇数, a, b 是整数, 有 $a > b$ 。

1. 若 $b = 0$, 答案显然可以是 0 和 1 交替的序列。
2. 若 $b > 0$, 可以发现 2^b 个 0 和 2^b 个 1 交替的序列是合法的。

期望得分 20。

4. 算法 4

在 n 是奇数或 $n = k$ 时, 显然无解。结合算法 3 , 现在可以假设 $a, b > 0$, 即 n, k 为偶数。

1. 若 $a = 1$, 考虑修改一个发射台的状态相当于将修改包含它的 k 个子段的奇偶性。为了得到目标状态, 需要有 $\frac{n}{2}$ 个子段为奇, $\frac{n}{2}$ 是奇数, 而 k 是偶数, 因此无法做到。
2. 若 $a > 1$,
3. 若 $k \leq \frac{n}{2}$, 相距 k 将一个发射台置为高频率, 每个发射台会贡献 k 个奇子段, 当总数将到达或超过 $\frac{n}{2}$ 时, 可以合理选择最后一个高频率的位置使得总数恰好为 $\frac{n}{2}$ 。
4. 若 $k > \frac{n}{2}$, 同 $k_1 = n - k$ 的情况。

期望得分 100。

5. 算法 Ω

从全 0 出发, 每次随机修改一个位置, 用线段树维护, 当遇到合法状态则输出, 一定次数 (10^5) 仍无解则认为无解。不知道为什么是对的。这个做法可以通过所有测试数据, 原本会被 $12MB$ 卡空间, 现在放过去了。

B 曲率飞船（ship）

首先这个代价的计算方式可以很自然地想到一笔画。

假设我们让第 i 条边经过 t_i 次, 那么我们的限制是 $t_i \equiv -w_i \bmod K$, 由一笔画的经典结论, 代价是“每个点的出边权和与入边权和之差的绝对值”之和的一半, 即 $\frac{1}{2}\sum_u |\sum_{(u,v)\in E} t_{u,v} - \sum_{(v,u)\in E} t_{v,u}|$。这样这个“至少覆盖几次”的问题就变成了“给每条边赋 t_i”的问题了。

但是现在的限制在边上, 我们却需要最小化点的出度入度差, 这显得很奇怪。所以我们把“随便赋 t_i”转变成每条边有初始边权 $-w_i \bmod K$, 我们现在每次可以选一条边 $+K$, 再转化成每次选两个点 x, y 满足 x 到 y 有边, 然后令 $a_x \leftarrow a_x + K, a_y \leftarrow a_y + K$, 其中 a_x 为 a 的出边权和减去入边权和。

更进一步, 如果 x 到 y 有路径我们就可以执行上述操作。这引导我们思考什么样的 x, y 是满足条件的。

仔细观察题目中的条件, 缩一下强连通分量, 那么拓扑序小的强连通分量一定能到达拓扑序大的强连通分量！

于是整道题可以转化为：给定一个序列 a, a 被划分成了若干段, 现在每次可以选择两个 i, j 满足 i 所在的段不大于 j, 然后令 $a_i \leftarrow a_j - K, a_i \leftarrow a_j - K$, 最终要最小化 $\sum |a_i|$。

那么这个从左到右扫一遍反悔贪心即可。维护一个优先队列表示当前段以及前面段剩下的负数, 来了一个新的段的时候可以把前面的负数翻成正的并把新段的正数翻成负的, 处理完这一段之后就把正数加进答案里扔掉。

时间复杂度 $O(n \log n)$。

C 智子纠缠（proton）

1. 算法一

对于 $N, M \leq 2 \times 10^5$ 的数据, 直接模拟过程并用并查集维护连通性。容易发现时刻 $N+M$ 之后图的连通性不会改变 (此时所有编号差整除 $\gcd(N, M)$ 的点对都是连通的)。

时间复杂度 $\mathrm{O}((N+M)\log(N+M))$ 或 $\mathrm{O}((N+M)\alpha(N+M))$, 可通过子任务一。

2. 算法二

考虑 $K = 0$ 。观察发现当 N, M 互质时, 前 $N+M-1$ 条边构成了一棵树。进一步地, 加入的前 $N+M-\gcd(N, M)$ 条边总是构成一座森林, 且之后的边都不影响连通性。（该结论可以直接用下文的类欧算法归纳证明）故可以直接使用等差数列求和公式计算答案。

时间复杂度 $\mathrm{O}(\log(N+M))$, 可通过子任务二。

3. 算法三

考虑 $L = R$ 的情形。记 V 为输入的 K 条边的端点集合, G 为时刻 L 时去掉这 K 条边后的二分图。则显然 G 的连通块数是已知的, 故只需求出 V 中每个点在 G 中所在连通块的编号再对连通块合并即可。不妨令一个连通块的编号为它当中最小的点编号 ($L < N$ 的情况是平凡的, 而 $L \geq N$ 时编号相同的点显然在同一个连通块内)。

将边看作点, 这等价于 L 个点中差为 N 和 M 的连边, 求一个点所在连通块中的最小点编号。

不妨设 $N < \min(M, L)$ 。则最后的 N 个点对前 $L-N$ 个点的连通性的贡献相当于把所有差为 $M-N$ 的点对之间连边（同时还需将最后 N 个点中的 x 分别连向 $x-N$)。进而我们可以将 N, M, L 的问题转化为 $N' = N, M' = M-N, L' = L-N$ 的新问题。使用类欧几里得算法即可解决。

时间复杂度 $\mathrm{O}(K\log(N+M+K))$, 可通过子任务三。

4. 算法四

显然我们只需求出原图关于插入时间的的最小生成树（或 Kruskal 重构树), 再对每条边计算对答案的贡献。

回顾算法三的过程：它其实模拟了一个按边权从大到小插入并建立最大生成树的过程, 最终每次 x 向 $x-N$ 连边所构成的边集即等价于 Kruskal 重构树。

将点的标号反向再进行算法三, 即每次将 $x \in [B, B+N-1]$ 分别向 $x+N$ 连边 (初始时 $B=0$), 再令 $(B, N, M, L) \leftarrow (B+N, N, M-N, L-N)$ 。这样既可得到 Kruskal 重构树。

只需维护这个过程中关键点集合 V 所在连通块的代表元的变化并记录这些连通块的每次合并 (即求出 V 中节点在 Kruskal 重构树上的虚树), 再对 K 条输入边和所有虚树上分叉点对应的边跑最小生成树即可。

直接维护是 $\mathrm{O}(K \log K \log(N+M+K))$ 的, 其中 $\log K$ 是因为排序和去重。类欧过程中将排序替换为 $\log\left(\frac{M}{N}\right)$ 次归并排序即可去掉 $\log K$ 。

时间复杂度 $\mathrm{O}(K \log K \log(N+M+K))$ 或 $\mathrm{O}(K \log(N+M+K))$ 。