2025 华南理工大学程序设计竞赛题解

2025 华南理工大学程序设计竞赛出题组

South China University of Technology

2025 年 3 月 29 日

赛题分析

预期难度:

Easy:AMLE

Easy-Medium:GBD

Medium:HKF

Medium-Hard:CIJ

实际过题人数情况:

A	B	C	D	E	F	G	H	I	J	K	L	M
72	37	0	7	48	8	46	2	2	0	0	44	23

黑白三消 1

此题方法很多，这里只介绍其中一种。
考虑贪心，从左到右放零件：
如果这个位置的左边已经放了两个同色零件，那当前位置只能放异色零件，否则就放剩余数量更多的那个零件。
中间如果存在需要放的零件没有了，便说明此时无解。
时间复杂度：$O(n)$。

黑白三消 2

此题方法很多，这里只介绍其中一种。
先构造一个 n 个点的尽量大的凸包（没有三点共线）:
第一步放置三个白传感器 $C_1(0,0), C_2(10^9,0), C_3(0,10^9)$，然后做一个从 C_3 到 C_2 的弧线:
只需要使得对于每个 $i \in (3,n]$ 都有:

- $x(C_i) - x(C_{i-1})$ 递减
- $y(C_i) - y(C_{i-1})$ 递减

最后在三角形 $C_1C_2C_3$ 内部随机 m 个点作为黑传感器即可，可以验证这样的方式获取到的 $n+m$ 个点存在三点共线的概率极低，可以通过本题。
时间复杂度: $O(n+m)$。

要 Go 了吗!!!

首先，当游戏某方获胜时，必然是当前字符串某个后缀与胜者的一个字符串匹配，审视有关的数据结构，AC 自动机可以很好完成这项任务，先把双方的字符集一起建立出一个 AC 自动机，构建出 Trie 图，再思考后面怎么做。

接着，一步步思考，Trie 图上部分节点的状态是确定的，到达这些节点意味着当前字符串后缀与某个模式串匹配，无论是先手到达这个节点，还是后手到达这个节点，胜负情况都已唯一确定。我们先把这些已知胜负的节点标记出来：一共有两种。第一种是模式串的结尾节点，第二种是该节点代表的字符串的后缀是模式串的节点。

换句话说，当一个节点在 Fail 树或 Trie 树上存在祖先是模式串的结尾节点时，该节点已知胜负。这里要注意根据题目描述，当一个节点既匹配先手模式串，又匹配后手模式串时，视为先手获胜。

要 Go 了吗!!!

然后，继续想办法处理其它点的状态，若我们能把根节点的状态求出，即求出本题答案。双方都以最优策略情况下，轮到先手操作时必然想转移到先手必胜节点，后手亦然。设状态 $f_{i,0}, f_{i,1}$，第二维取 0 表示先手到达 i 节点，取 1 表示后手，$f = 1$ 表示先手必胜，$f = 0$ 表示后手必胜，$f = -1$ 表示未知。f 取值只有这三种可能，初始 f 均视为 -1。我们可以先赋值之前已讨论过胜负状态的节点。

若我们想把 $f_{i,j}$ 的值从 -1 改为 1 或 0，这里以 $j = 0$ 为例:

- 要么 i 在 Trie 图上所有转移均为后手必胜节点（$\forall\ next \mid f_{next,1} = 0$），此时 $f_{i,0}$ 应改为 0。
- 要么 i 在 Trie 图上存在着转移为先手必胜节点（$\exists\ next \mid f_{next,1} = 1$），此时 $f_{i,0}$ 应改为 1。

这时我们发现，若某个 f 值需要从 -1 改为 1 或 0，意味着它的转移状态存在变化，那么我们可以每修改一个 f 值时，便检查所有可转移到该节点的点的状态，即建出 Trie 图的反边，便可判断是否需要更新新的 f 值。

要 Go 了吗!!!

若所有被修改过的 f 值的反向转移都被检查过后，剩余的 f 仍为 -1 的状态便表示平局状态，因为在双方都明智的操作下该状态无法转移到某人胜利的节点。

最后我们检查根节点的 f 值即可知道最终游戏的结果，时间复杂度为 $O(\sum|S| \times |\sum|)$。

值得一提的是，虽然 Trie 图中存在环，但是在遍历修改的过程中不会陷入无限循环，因为当你访问到一个 $f \neq -1$ 的状态时，你不会再次将该状态放到你存放准备检查反向转移的状态的容器中。

罗马斗兽场

- 首先，我们发现当 $i = 1, 2, n$ 时，不可能阻止从 1 走向 i，方案数为 0，这些 i 值若放到下述做法中需要特殊讨论，为了方便直接赋值，不另行讨论。
- 求出阻止 $1 \to i$ 的方案数不大好做，先把问题转化成用总方案数减去允许 $1 \to i$ 的方案数。
- 若 1 与 i 用绳子连接，在合法方案中只要挨着绳子走必能从 1 走到 i。更进一步，若 1 与 i 所连线段被任意其它绳子穿过，则必不能走到 i。因此，允许 $1 \to i$ 的方案中，不能出现绳索分别连接直线 $Line_{1,i}$ 两侧的点，斗兽场被分为一个 i 个顶点的凸包和一个 $n - i + 2$ 个顶点的凸包，这两个凸包间不能连绳索。
- S 无论怎么分割，凸包的顶点都是原来圆上 n 个点的子集，不可能出现三点共线。无论形状，m 个顶点的凸包的合法方案数只与 m 有关。
- 显然的，凸包的边上是否连接绳索对于其它绳索能否连接无影响，对是否允许 $1 \to i$ 亦无影响，那么我们可以随意决定连接与否，这部分的方案数是 2^m。

罗马斗兽场

- 将除去边界的绳索方案数设为 f_m。若 f 已知，则允许 $1 \to i$ 的方案数为 $2^n \times f_i \times f_{n-i+2} \times 2($ 代表 $1, i$ 绳索的连接与否 $)$，总方案数为 $2^n \times f_n$，即可 $O(1)$ 求出阻止 $1 \to i$ 的方案数。
- 接下来我们设法求出 f，考虑 **dp**。
- 考虑枚举 1 顶点连接的所有顶点中编号最小的为 j，由于不涉及边界上的连接情况，j 的取值范围为 $[3, i-1]$，当然，我们别忘记考虑 1 顶点不与其它顶点连接的情况。以 $i = 5$ 为例，如下图：

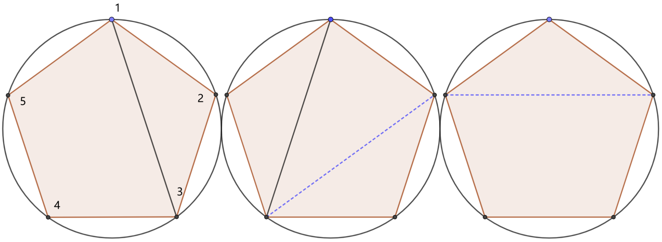
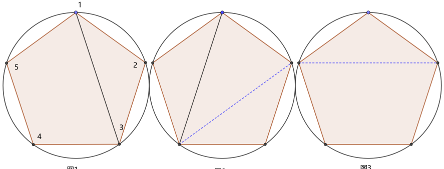


图1 图2 图3

罗马斗兽场



- 情况 1（图 1）：$j=3$ ，凸包被划分成右侧三角形与左侧 $i-1$ 顶点凸包，方案数为 f_{i-1}。
- 情况 2（图 2）：$j\in[4,i-1]$，凸包被划分为右侧 j 顶点凸包与左侧 $i-j+2$ 顶点凸包，右侧凸包中 1 不能再与其他顶点相连，否则违背 j 为编号最小的前提。为便于计算右侧方案数，我们可以将 1 顶点视作不存在，方案数为 f_{j-1}，而此时 $Line_{2,j}$（图中虚线）作为新凸包的边界，既非原边界，亦未被新凸包的方案考虑，我们要补上，其情况只有连接与否，无论其情况均不会对其它连接产生阻碍。最终方案为 $2\times f_{j-1}\times f_{i-j+2}$。

罗马斗兽场

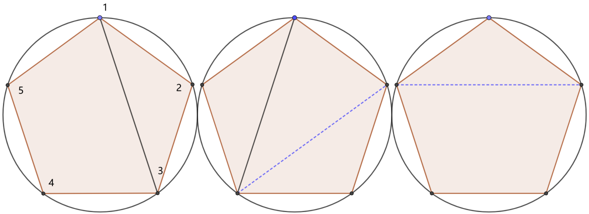


图1 图2 图3

- 情况 3（图 3）：1 未与其它结点连线，类似于情况 2 的，我们忽略 1 顶点，补回新边界 $Line_{2,n}$，方案数为 $2 \times f_{i-1}$。
- 初始情况为 $f_1 = f_2 = 1$。
- 综上，我们可以 $O(n^2)$ 预处理出 $f_1, f_2, f_3, \cdots, f_n$，再 n 次 $O(1)$ 求出 n 个答案。

让火焰净化一切!

简略题意：给你两个数 x 和 y，在每回合中，等概率使其中一个正数减 8，y 不为正数后则每回合只会使 x 减 8，问使 x 减到 0 及以下的回合数的期望，误差不超过 6 位小数。

提示：这里询问的是概率期望，而非排列的频率。每种攻击排列的出现概率并非等可能的，两者易发生混淆。在错误的理解方式下，样例 1 的答案不变，样例 2 的答案变为 $\frac{10}{3}$。还请仔细斟酌。

让火焰净化一切！

题解：首先，对 x 和 y 进行预处理，分别转化为两者可以承受的攻击次数上限 $\lceil \frac{x}{8} \rceil$ 和 $\lceil \frac{y}{8} \rceil$ 。而后，构建状态矩阵 $dp_{x,y}$，其中 $dp_{x,y}$ 代表 Arthas 承受 x 次攻击且其随从承受 y 次攻击这一事件所发生的概率。设 $dp_{0,0} = 1$，由此，易得到下面的递推关系：

$$dp_{i,j} = \begin{cases} \frac{dp_{i-1,j}}{2}, & j = 0 \vee i = x \\ \frac{dp_{i,j-1}}{2}, & j = y \\ \frac{dp_{i-1,j}}{2} + \frac{dp_{i-1,j}}{2}, & else \end{cases}$$

$j = 0$ 和其他情况都是好理解的，而为什么 $i = x$ 和 $j = y$ 的情况要特殊处理呢？因为此后发生的情况已经唯一确定了，也为了方便计算回合数。最后的答案为：

$$ans = \sum_{j=0}^{y-1} dp_{x,j} \times (x + j) + \sum_{i=0}^{x-1} dp_{i,y} \times (x + y)$$

让火焰净化一切!

如果出题人调整 x 和 y 的数据范围，使得时空间进一步受限呢？

方案 1

采用滚动数组，每轮只保留存储上一个状态 $y-1$ 和当前状态 y 的数组，可以把空间降到 $O(x)$，但是并没有改变时间复杂度。

方案 2

采用数学方法，空间复杂度降到 $O(1)$，时间复杂度降到 $O(x+y)$。

$$ans = \sum_{i=x}^{x+y-1} \frac{C_{i-1}^{x-1} \times i}{2^i} + \sum_{i=y}^{x+y-1} \frac{C_{i-1}^{y-1} \times (x+y)}{2^i}$$

公式如上，有兴趣的同学可以自行探讨（仍要注意边界的特判）。

最大公约数问题

离线处理

首先把所有询问读入，按右端点排序处理，做扫描线。

考虑支配情况

虽然区间总数是 $O(n^2)$ 的，但是有许多区间被别的区间支配。
例如存在一个区间 $[l,r]$，同时存在一个区间 $[l-1,r]$，且它们区间 $\gcd$ 一样且同时被询问区间包含，那么可以认为区间 $[l,r]$ 无用，也即被 $[l-1,r]$ 所支配。

后缀 $\gcd$ 的性质

每次后缀向左扩充一个点时，$\gcd$ 要么不变，要么变为它的因子。即要么不变，要么至少变为原来的 $\frac{1}{2}$。所以后缀 $\gcd$ 最多变化 $O(\log V)$ 次。

最大公约数问题

有用的区间数数量级

由于后缀 gcd 最多变化 $O(\log V)$ 次，所以有用的区间数最多是 $O(n \log V)$ 个，对这些区间利用线段树做扫描线，即可做到复杂度 $O(n \log n \log V)$ 解决问题。

考虑特殊情况

注意到有些完全支配的区间会被询问区间所断开，所以需要特殊计算这些情况。只需要固定左端点，找到每次 gcd 变化前的最大区间即可，每次都会计算 $O(\log V)$ 个区间。

快速找到 gcd 变化的位置可以使用**二分** $+$**st** 表实现，总的时间复杂度为 $O(n \log n \log V)$。

Butterfly Delusion

题意

$n \times n$ 的点阵，初始全为白点，只能在黑点转弯，m 次询问，包括将白点变为黑点和判断两点是否可达。

题解

观察发现，当处于某行且方向为左右时可以到达该行任意点，处于某列时同理，因此当经过黑点 (i, j) 并进行转弯时，可以认为从第 i 行任意点能到达第 j 列任意点。
这启发我们使用并查集：当新增一个黑点时，将其所在行的并查集和所在列的并查集合并；当查询时，注意需要判断四种出发和抵达的情况，分别为起点、终点所在行与行，行与列，列与行，列与列。当至少有一种情况属于同一并查集时，这两点可达，否则不可达。时间复杂度为 $O(m \times \alpha(n))$。

Border Phony

先给出结论:

- 当 $m = 0$ 时，答案为 $abbb \cdots bbb$。
- 当 $m = n - 1$ 时，答案为 $aaa \cdots aaa$。
- 当 $2m < n$ 时，答案为 $aa \cdots aabb \cdots bbaa \cdots aa$（$m$ 个 a,$n - 2m$ 个 b,m 个 a)。
- 其他情况及无解见后页。

Border Phony

其他情况及无解

二分每一次询问的 Border 均会 $\geqslant \frac{n}{2}$。
所以 Border 对应的周期都 $\leqslant \frac{n}{2}$。
由强周期引理可以推得，它们的周期是可以不断做 $\gcd$ 运算的。
首先这个串的周期最长是 $n-m$，这个周期在下面的过程中因为做 $\gcd$ 运算，所以它只会变小，但是又因为 m 是整个串的最长 Border，所以 $n-m$ 是最短的一个周期，即后面使得周期变小的操作都是不合法的。
二分的过程中确定了部分长度是 Border，部分长度不是 Border。
对于是 Border 的长度，判断这个 Border 带来的周期和 $n-m$ 做 $\gcd$ 运算会不会使其变小，会则无解。
对于不是 Border 的长度，判断这个 Border 的周期是否是 $n-m$ 的倍数，如果是也是无解的。
那么只需要构造一个周期是 $n-m$ 的串即可。
可以通过随机等方法进行构造，构造结束后再跑一次题目的二分判断是否符合要求。

开盒达人

根据几个步骤的定义，分别构建逆操作 InvSubBytes, InvShiftRow, InvMixColumn, 并对输出进行逆计算得到 [input XOR RoundKey], 再异或上输入即可还原 RoundKey, 其中 MixColumn 矩阵的逆矩阵如下所示，可以用 $GF(2^8)$ 有限域乘法的定义推得。

14	11	13	9
9	14	11	13
13	9	14	11
11	13	9	14

城市供电网络

首先对这个树进行重链剖分。

邻域分类

首先这是一个树的邻域查询问题，并不好做，所以重链剖分后，把操作点的邻点分类为三种点：重儿子，轻儿子，父亲。
重儿子和父亲分别只有一个，所以可以单独特殊处理。
轻儿子的数量可能有 $O(n)$ 个，考虑对于每一个点的轻儿子利用数据结构维护。

重链剖分性质

只有被修改点到根路径上的点的轻儿子会发生变化。而这条路径上最多只有 $O(\log n)$ 个轻儿子。从修改点开始跳重链的顶端，就能找到所有轻儿子进行修改。

城市供电网络

选择合适的数据结构

对每个节点使用动态开点线段树/树状数组来维护轻儿子的信息。查询的时候在线段树/树状数组上二分即可找到 mex。
在修改时，在动态开点线段树/树状数组中减去节点权值变更前的子树最小值，再加上节点权值变更后的子树最小值。查询子树最小值可以用线段树，是一个简单的单点修改，区间查询。
总共修改 $O(\log n)$ 次轻儿子，每次修改的代价是 $O(\log n)$，所以一次修改权值的代价为 $O(\log^2 n)$。
查询的时候查询重儿子的最小值，子树外的最小值，插入线段树中，在线段树上二分查询 mex，然后把值删去，时间复杂度为 $O(\log n)$。

注意动态开点线段树/树状数组值域范围

注意到查询点的 mex 不会超过这个点的度数，所以每个点的动态开点范围为 $0 - d_i$ 即可，否则空间复杂度有可能达到 $O(n \log^2 n)$，甚至 $O(n \log n \log V)$。

总的时间复杂度为 $O(q \log^2 n)$，空间复杂度为 $O(n)$。

又一个计数问题

显然这是一个数位 dp+ 容斥的计数题目。

数位 dp

总共有 $0, 1, 2..., 9$ 十个数字，那么对于一个数字 x 的数位出现这些数字的情况有 1023 种，
利用差分和数位 dp 求出 $[1, x]$ 区间里，这 1023 种分别有多少个元素。

容斥计数

枚举超集，累加每种情况的超集总共会有多少个元素出现，即对于一个二进制码 $mask$，求出所有数位集合中至少包含 $mask$ 的数字个数，记作 $S(mask)$。
若固定候选集合 $mask$，保证选出的每个数字都包含 $mask$，那么数组的公共出现数字至少包含 $mask$，但同时也有可能有 $mask$ 以外的数字，所以考虑容斥。

$$ans = \sum_{|mask| \geqslant d} (-1)^{|mask|-d} \binom{|mask|}{d} \binom{S(mask)}{n}$$

单组测试数据时间复杂度 $O(length(num) \times 2^2 \times 2^{10} \times 10 + 3^{10})$。常数很小，轻松通过。

演绎法

题意整理

需要找出两个点，使得给定的错误点均可以到达这两个点。
且未给定的非错误点不能到达这两个点。
“可以到达”即可以通过有向边直接或间接移动到目标点。

题解

考虑 $n \leqslant 200$，可以使用不太超过 $O(n^3)$ 的算法。
可以建立反向的图，利用搜索先预处理出对于每个点，能到达它的点有哪些，每个点都需要 $O(n+m)$，共计 $O(n^2+nm)$。
然后两两枚举点，判断可以到达他们的点的交集是否完全等同于给定的错误点，如果是就可以输出答案了。

12:9 再现

处理特殊情况

不可能有任意一方的得分大于 K。
m 必须要小于 K。
当 $n = m = 0$ 时，答案为 1。
不可能发生 $n = 0, m > 0$ 的情况。

分类讨论当前轮胜者

记两者胜率分别为 $P_s = \frac{p}{p+q}, P_b = \frac{q}{p+q}$。
前者为当前轮胜者的概率是 $P_1 = \binom{n+m-1}{n-1} P_s^n P_b^m$。
后者为当前轮胜者的概率是 $P_2 = \binom{n+m-1}{n-1} P_b^n P_s^m$。
所以 $ans = P_1 + P_2$。
预处理组合数和快速幂计算结果即可。