# NOI 2024 联合省选模拟赛　题解

Wu_Ren

2024 年 2 月 22 日

## 目录

**1 chef** **1**
1.1 subtask 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1
1.2 subtask 2,3 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1
1.3 subtask 4 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1
1.4 subtask 5 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1

**2 draw** **2**
2.1 subtask 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2
2.2 subtask 2 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2
2.3 subtask 3 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

**3 message** **3**
3.1 subtask 1 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
3.2 subtask 2,3 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
3.3 subtask 4 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

# A chef（chef）

记 $A_i, B_i$ 前缀和为 $SA_i, SB_i$。

## 1. subtask 1

枚举一道菜的步数,二分求得另一道菜的最大步数,取前缀 max 即可,复杂度 $O(M\log N)$。

## 2. subtask 2,3

此时 $P_i, Q_i \geq 1$。

记已做完第一道菜前 $i$ 步，第二道菜前 $j$ 步的最大收益为 $f_{i,j}$。

考虑从 $f_i$ 变到 $f_{i+1}$ 的过程，首先考虑 $f_{i,j} \to f_{i+1,j}$ 的转移，相当于给一段前缀加上 $P_{i+1}$。

然后考虑 $f_{i+1,j} \to f_{i+1,j+1}$ 的转移，注意到满足 $SA_{i+1}+SB_{j+1} \leq T_{j+1} \wedge f_{i,j+1}-f_{i,j} \neq Q_{j+1}$ 的 $j$ 只有 $O(1)$ 个，所以考虑维护 $f_i$ 的差分即可。

复杂度 $O((N+M)\log(N+M))$。

subtask2 的情况维护起来更为简单。

## 3. subtask 4

这档分是给一些 $O((N+M)\log^2(N+M))$ 的解法或者常数过大的正解的。

## 4. subtask 5

记考虑我们从 $(0,0)$ 出发，每次进行第一道菜的制作步骤相当于往左走一步，进行第二道菜的制作步骤相当于往上走一步，最后走到 $(N,M)$。

对于第一道菜，如果想要拿到 $P_i$ 的价值，记 $x=\max\{x \mid SB_x+SA_i \leq S_i\}$，则我们的路径必须严格在 $(i-1,x+1)$ 这个点下方，那么我们预先给答案加上 $P_i$，然后加入一个价值为 $-P_i$ 的位于 $(i-1,x+1)$ 的点。

对于第二道菜，如果想要拿到 $Q_i$ 的价值，记 $x=\max\{x \mid SB_i+SA_x \leq T_i\}$，则我们的路径必须在 $(x,i)$ 上方（可以恰好经过），则加入一个价值为 $Q_i$ 的位于 $(x,i)$ 的点。

现在我们问题变成了，求一条 $(0,0)$ 到 $(N,M)$ 的折线，使得这条折线下方的点的价值和最大，传统的树状数组维护差分即可，复杂度 $O((N+M)\log(N+M))$。

# B draw（draw）

## 1. subtask 1

当 $a=b$,可以发现存在一个最优解使得任意一个点只会被染一次。那么给每个格子开两个点 $H_{i,j},V_{i,j}$,表示这个点是横着染了还是竖着染了,连边 $\forall 1 \le i \le n, (S, H_{i,j}, a), (H_{i+1,j}, H_{i,j}, b), (S, H_n$ 分别表示如果 $(i,j)$ 竖着染了,则需要支付 $a$,如果 $(i-1,j)$ 竖着染了且 $(i+1,j)$ 没有,则支付 $b$,以及 $(n+1,j)$ 不能染。类似地有连边 $\forall 1 \le j \le m, (V_{i,j}, T, a), (V_{i,j}, V_{i,j+1}, b), (V_{i,m+1}, T, +\infty)$。同时连边 $(H_{i,j}, V_{i,j}, c)$ 表示如果这个点既没有横着染也没有竖着染则支付 $c$ 来单点染，复杂度 $O((nm)^3)$。

## 2. subtask 2

考虑轮廓线 dp，对于轮廓线上的点维护 $0/1, 0/1$ 表示是否有未结束的涂黑/涂白操作，复杂度 $O(nm4^m)$，常数优秀的代码都足以通过。

## 3. subtask 3

考虑从 subtask1 继续：

我们可以设 $Hb_{i,j}, Vb_{i,j}, Hw_{i,j}, Vw_{i,j}$，分别为是否被黑/白的竖着/横着的线段覆盖了。

以黑色竖线为例，花费是 $a\sum Hb_{i,j} + b\sum Hb_{i,j}\overline{Hb_{i+1,j}}$。

然后考虑图形的限制，对于一个要是黑色的点，花费就是 $c\overline{Hb_{i,j}Vb_{i,j}} + \infty(Hw_{i,j} + Vw_{i,j})$，后面是因为如果染了白色就染不上黑色了。

对于白色的点，花费为 $c(Hb_{i,j}\overline{Vw_{i,j}} + Vb_{i,j}\overline{Hw_{i,j}}) + \infty Hb_{i,j}Vb_{i,j}$，后面是因为一个格子只能染两次。

这些可以建成最小割模型求解，具体的，假设 $S$ 是源点，$T$ 是汇点，对每个变量建一个点。

把 $Hw_{i,j}$ 和 $Vb_{i,j}$ 的意义取反，所有式子都变成 $cX\overline{Y}$ 的形式，那么可以看作 $Y$ 向 $X$ 连了一条边权为 $C$ 的边。

对于 $cX$ 的权值，看作 $S$ 向 $X$ 连了一条边权为 $c$ 的边，对于 $c\overline{X}$，看作 $x$ 向 $T$ 连了一条边权为 $c$ 的边。

复杂度 $O((nm)^3)$，网络流常数优异。

# C message（message）

## 1. subtask 1

给各种简单 dp 的。

## 2. subtask 2,3

考虑可以归纳证明，我们一定可以把所有密文分成两个部分，使得每个部分中，对于任意长度为 $x$ 的段，都不会包含超过一个完整的密文。

那么我们现在只需要考虑有 $n$ 个密文，要求任意长度为 $x$ 的段不包含超过一个完整的密文，然后需要的时间最小值。

考虑我们确定了所有密文的相对位置时，假如第 $i$ 个密文从 $s$ 时间开始，那么第 $i+1$ 个密文最早从 $s+x-a_{i+1}+1$ 开始，发现第 $i+1$ 个线段越早开始越好，那么直接取下界即可，那么最后答案就是 $\sum_{i=2}^{n}(x-a_i+1)+a_n=\sum_{i=2}^{n-1}(x+1-a_i)+x+1$。

先特判某部分 $n\le 1$ 的情况，现在就变成了，删去最短的四条线段，把其他线段分成两部分，使得两部分分别的和的最大值最小。

那么设 $C=\frac{\sum w_i}{2}$，那么就是有 $n$ 个大小为 $w_i$ 的物品，现在取若干个物品，求在总大小不超过 $C$ 的情况下，总大小最大能取到多少。

简单背包就是 $O(nx\min(n,x))$。

## 3. subtask 4

记 $W=\max\{w_i\}, b=\max\{b\mid\sum_{i=1}^{b}w_i\le C\}$，则称 $\begin{cases}x_i=1 & i\le b\\ x_i=0 & i>b\end{cases}$ 为截断解。

一组解为平衡解由如下递归定义：

- 截断解是平衡解。
- 如果 $x$ 是平衡解且 $\sum x_iw_i\le C$，则任取 $x_i=0(i>b)$，令 $x_i:=1$，该解也是平衡解。该操作称为平衡插入。
- 如果 $x$ 是平衡解且 $\sum x_iw_i>C$，则任取 $x_i=1(i\le b)$，令 $x_i:=0$，该解也是平衡解。该操作称为平衡删除。

显然最优解一定是平衡解，记平衡解集合为 balance。

记 $f_{s,t}(w)$ 为 $\max\{\sum x_iw_i\mid x\in\text{balance}\wedge\forall i<s,x_i=1\wedge\forall i>t,x_i=0\wedge\sum x_iw_i\le w\}$。若 $f_{s,t}(w)=w$，则称 $(s,t,w)$ 是可达的。

如果存在 $s\ge s',t\le t'$，且 $(s,t,w),(s',t',w)$ 都是可达的，那么后者显然不如前者，记 $s_t(w)=\max\{s\mid f_{s,t}(w)=w\}$，如果这样的 $s$ 不存在，记 $s_t(w)=0$。假如我们能求出

$s_t(w)$，那么 $\max\{w \mid s_n(w) \neq 0, w \leq c\}$ 就是答案。

假如 $t = b$，此时唯一平衡解为截断解，记 $\overline{w} = \sum_{i=1}^{b} w_i$，故有 $\forall i \in [C - W + 1, C + W]\backslash\{\overline{w}\}, s_b(i) = 0, s_b(\overline{w}) = b + 1$。

考虑我们枚举 $t = b + 1, b + 2, \ldots, n$，那么有如下转移：

- $\forall i \in [C - W + 1, C + W], s_t(i) \leftarrow s_{t-1}(i)$。
- $\forall i \in [C - W + 1, C], s_t(i + w_t) \leftarrow s_{t-1}(i)$，即进行一次平衡插入。
- $\forall i \in [C + W, C + 1], j < s_t(i), s_t(i - w_j) \leftarrow j$，即进行一次平衡删除。

注意到第三个转移，如果 $j < s_{t-1}(i)$，那么一定有 $s_{t-1}(i - w_j) \geq j$，故我们可以把 $j$ 只枚举到 $\max(s_{t-1}(i), 1)$。

此时第一个和第二个转移复杂度都是 $O(nW)$,第三个转移复杂度为 $O(\sum_{i=c+1}^{c+W} \sum_{t=b+1}^{n} s_t(i) - s_{t-1}(i)) = O(nW)$，所以总复杂度就是 $O(nW) = O(nx)$。