



省队集训 Day4 solution

——wfj

一些瞎扯

- 出题人是一个 ~~day2~~ 梦游省选爆炸没有资格参加 NOI 的退役狗。
- 出题人是一个没有妹子的单身狗。
- 题面逻辑非常混乱非常优美，可惜主人公没来。
- 出题人很菜，还望大家多多教育。
- 出题人非常毒瘤良心。
- 谢绝一切与题目无关或与题面有关的提问。



y

- 题意： HNOI2018 day2t3 ，把每个城市的两条子边中选一条改成任选 $n-1$ 条边。
- 签到题。不过为什么出这道题呢？
- 因为这是我两个月前在考场上看到的题意。
- ~~当然我就算没看错题也进不了队，因为 T1 也写挂了。~~
- 于是出题人化悲痛为力量，发誓自己一定要想出在考场上看到的题目。
- 出题人水平有限，不保证没有更优秀的做法，欢迎大家踩标算。
- tag：凸优化 dp ，贪心。



0 分做法

- 把 HNOI2018 day2t3 的满分代码蒯过来即可。
- 当然，如果你把本题的标程蒯过去，你也能获得 0 分的好成绩。
- 应该…没有人看错题吧？



subtask1

- 爆搜每条边选不选，然后算贡献，复杂度 $O(C(2n,n)*n)$ 。



subtask2

- 暴力 dp，设 $f[x][y][a][b]$ 表示在 x 子树中选 y 条边，当前根到 x 的链上已经选了 a 条公路， b 条铁路的最小代价。
- 转移非常显然，复杂度 $O(n^3 \cdot d^2)$ 。
- 如果对于 x ， y 枚举到 $\text{size}[x]$ ，复杂度变为 $O(n^2 \cdot d^2)$ 。
- 由于出题人没时间造数据了，所以并没有卡第一种做法。
- 其实这道题的所有数据都是原题数据。



subtask3

- 我们设一个函数 $f(x)$ 表示选了 x 条边的最小代价。
- 打表可以发现， $f(x)$ 单调递减，且减小的速率越来越慢。
- 也就是说，对于 $y \geq x$ ， $f'(y) \leq f'(x)$ 。
- 于是我们可以二分 $f'(x)$ ，即 x 处的斜率。
- 设斜率为 k ，那么这就意味着选择一条边，就要额外付出 k 的代价。
- 于是我们可以去掉 subtask2 中的第二维数组，直接在 dp 过程中记录选择边数即可。
- 复杂度 $O(n \cdot d^2 \cdot \log k)$ 。



subtask4

- 注意到一个重要性质，即如果我们选择了一条公路或铁路，那么在它上面的公路或铁路也必须选。这个贪心思想显然是对的。
- 于是我们继续 **subtask3** 的思路。显然，如果一个子树内没有一条边，那么这个子树内的所有乡村选的公路和铁路都是一样多的。
- 那么我们先预处理出 $\text{sum}[x][a][b]$ 表示 x 的子树内选了 a 条公路， b 条铁路的总代价和。 $\text{depa}[x]$ 表示根到 x 经过的公路数量， $\text{depb}[x]$ 表示根到 x 经过的铁路数量。

- 设 $f[x][a][b]$ 表示 x 的子树中，公路状态为 a ，铁路状态为 b 的最小代价。
- a 为 0 表示 x 到根的所有公路都选，否则表示 x 的子树内所有公路都不选，且已经选了 $a-1$ 条公路。 b 的意义同理。
- 可以发现，当 a, b 都不为 0 时，我们可以 $O(1)$ 得到答案。所以这个 dp 的状态数是 $O(n*d)$ 的，转移是 $O(1)$ 的。
- 于是我们便解决了这题，总复杂度 $O(n*d*\log(k)+n*d^2)$ 。要特别注意空间复杂度的优化。
- 其实如果去掉凸优化，时间复杂度在 `subtask3` 中也是可以承受的。但是空间复杂度太大，理论上是没有办法卡过去的。

S

- 题意：略。
- 来源： `codeforces 321D` 。
- 好像是几年前的一道集训队作业题。
- 因为完全是蒯的，所以出题人不知道怎么出部分分，在此深表歉意。
- tag：枚举，结论。



subtask1

- 爆搜，复杂度 $O(2^{(x \times x)} \times n^2)$ 。



subtask2

- 退火或爬山。
- 出题人写的爬山离最优解只差一点点。
- 介于出题人根本没有任何乱搞能力，所以我相信大家一定能用近似算法通过这个 subtask 甚至是后面的 subtask !



subtask3

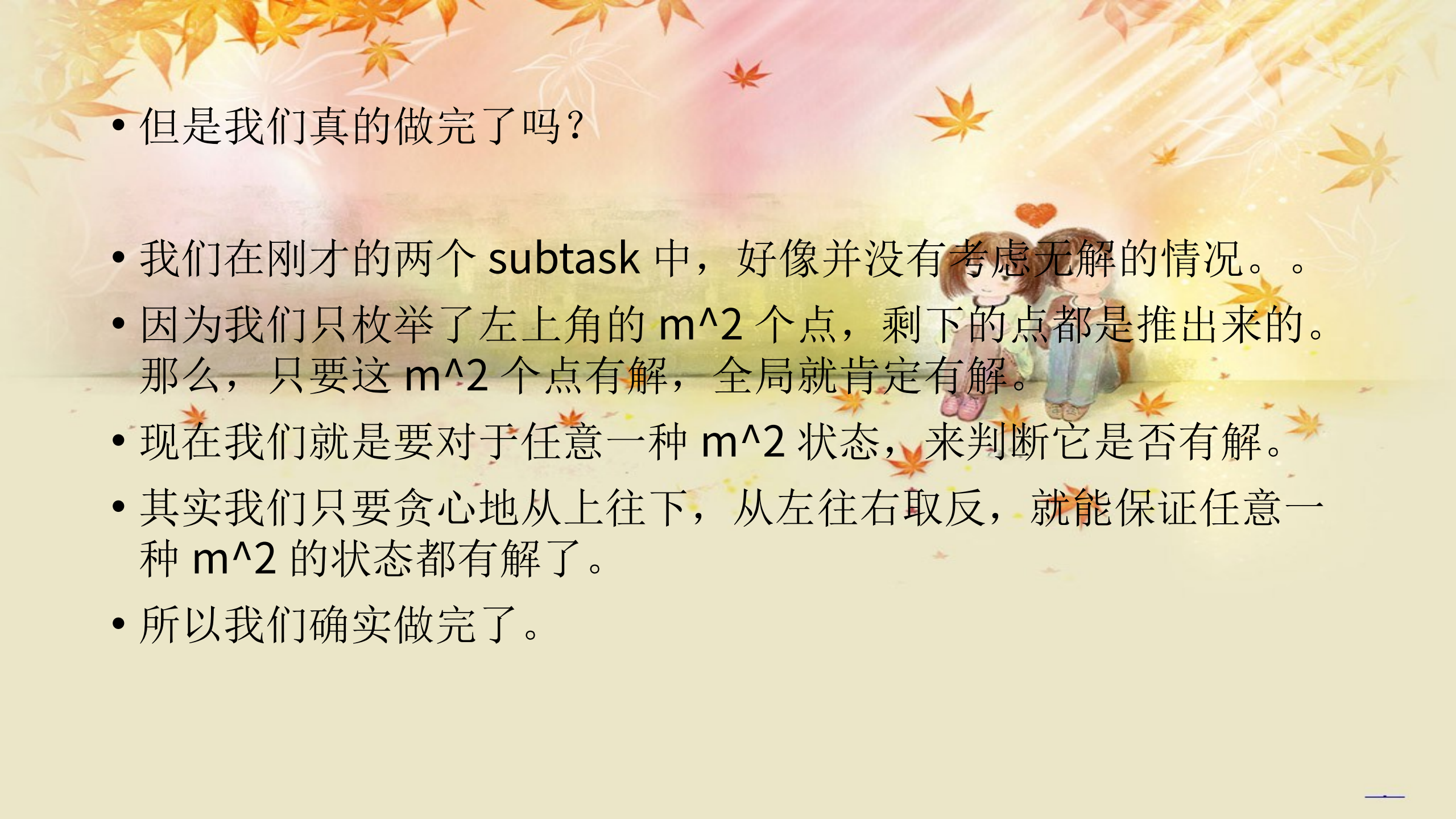
- 看来部分分确实少了，这里已经与正解相关了。。
- 我们转变一下思路，把每个矩形有没有变号改成每个点有没有变号。
- 这里有一个很容易证明的结论： $X_{i,j} \oplus X_{m,j} \oplus X_{i+m,j} = 0$ 。
- m 就是题中的 x ，列同理。
- 很显然每次取反两个矩形，上式的三个变量要么变化 0 个，要么变换 2 个，于是结论得证。
- 那么我们可以枚举 $(1 \sim m, m)$ 和 $(m, 1 \sim m)$ 的 X ，并推出两条中线的其他点。
- 然后我们枚举左上角的 $(m-1) \times (m-1)$ 的矩阵的每个点的 X ，这时我们可以发现，其他三个角的点的 X 都能算出来了。复杂度 $O(2^n \cdot n^2)$ 。



subtask4

- 当我们枚举完 $(1 \sim m, m)$ 后，我们可以发现， $(m, 1 \sim m-1)$ 的每个点都是独立的了。
- 所以我们只要枚举 $(1 \sim m, m)$ 就行了！
- 推出 $(1 \sim n, m)$ 以后，我们再分别枚举 $(m, 1 \sim m-1)$ 的所有点的 X 。
- 然后对于 (m, i) ，我们再枚举 $(1 \sim m-1, i)$ ，同时推出其他三个角的对应点。
- 复杂度 $O(2^m * n^2)$ 。



The background of the slide features a soft, warm-toned illustration. At the top, there are clusters of vibrant orange and yellow autumn leaves. In the center-right, two young children, a girl and a boy, are depicted. The girl has brown hair and is wearing a pink dress, while the boy has brown hair and is wearing a blue shirt and green pants. They are standing close together, and a small red heart floats above them. The ground is covered with a layer of fallen orange leaves. The overall atmosphere is gentle and nostalgic, with a light pink and yellow color palette.

- 但是我们真的做完了吗？

- 我们在刚才的两个 **subtask** 中，好像并没有考虑无解的情况。。
- 因为我们只枚举了左上角的 m^2 个点，剩下的点都是推出来的。那么，只要这 m^2 个点有解，全局就肯定有解。
- 现在我们就是要对于任意一种 m^2 状态，来判断它是否有解。
- 其实我们只要贪心地从上往下，从左往右取反，就能保证任意一种 m^2 的状态都有解了。
- 所以我们确实做完了。

f

- 题意：略。
- 来源：各种模板拼起来原创。
- 这道题改了无数回，因为我每次写完都发现非标算的一些代码量较小的做法能过。
- tag：动态 dp，虚树，链分治，lct。



subtask1

- 写两个 dp 就行了，复杂度 $O(n^2)$ 。



subtask2

- 对于链的情况，我们可以使用线段树来维护。
- 第一问转化为求一个集合的最大连续子段和。
- 第二问转化为求一个集合的最大连通连续子段和。
- 注意链不是 $1 \sim n$ 的，复杂度 $O(n \log n)$ 。



subtask3

- 只涉及修改点权，且询问只有直径，那么我们可以考虑使用点分治或边分治。
- 我们给分治每一层的树用 **dfs** 序标号。我们可以发现，每次单点修改转化到每一层上就是子树修改。
- 如果是点分治那么查询的时候要开一个堆来维护重心的每个儿子的最长链。
- 如果是边分治注意会被菊花树卡，要把树重构成二叉树。
- 复杂度 $O(n\log^2 n)$ 。



subtask4

- 因为 $k \leq 2$ ，所以我们可以直接开两棵树出来。
- 我们使用链分治来解决这个问题。
- 首先我们对整棵树树链剖分，然后我们把重儿子和轻儿子分开来算贡献。
- 对于轻儿子，我们直接把它的最长链丢到一个堆里，把 >0 的连通块加进来即可。
- 对于重儿子，我们使用线段树来维护最大连续子段和，分情况转移就行了。
- 复杂度 $O(n \log^2 n)$ 。



subtask5

- 在 subtask4 的基础上离线建虚树。
- 注意建虚树以后，不连通的点会连通。为了避免这种情况，我们在把点 x 加入集合 y 时，把 x 的父亲也加入集合 y 。



subtask6

- 我们现在要在线构虚树了。
- 同时我们发现，树的形态会改变，所以我们把树链剖分换成 **lct**。
- 首先对于每个集合，用 **set** 维护 **dfs** 序。
- 我们加入 x 点时，找到它 **dfs** 序上的前驱，设为 u ，如果没有就不管。
- 找到以后，我们求出这两个点的 **lca**。如果 **lca** 在树上，那么直接把 x 连向 **lca** 就行。
- 否则我们要在树上二分出 u 的祖先中深度大于 **lca** 的最小深度祖先 g 。这个我们可以在 **splay** 上二分得到。
- 找到 g 和 g 的父亲 f 以后，断开 f 和 g ，连接 **lca** 和 g, f ，并连接 u 和 **lca**。
- 找 x 的后继同理。



- 然后我们考虑怎么用 **lct** 来实现树链剖分的操作。
- 虚儿子就对应着树链剖分的轻儿子，实儿子就对应着树链剖分的重儿子。
- 虚实边转化时，我们把新的实儿子的贡献减去，新的虚儿子的贡献加上即可。
- 维护实儿子贡献与线段树类似，只是换成了 **splay** 而已。
- 这里有一个 **trick**：这道题中的 **lct** 不需要用换根操作，可以节省常数并减少代码量然而还是有 **7.7kb**。
- 总复杂度 $O(n \log^2 n)$ ，有没有人通过这道题呢？



结语

- 祝大家在 NOI 中不要看错题，不要写挂。
- 祝大家都能在 NOI 中取得一个好成绩。
- 祝省队爷们都能进集训队。
- 感谢各位的聆听。

