# 集训队作业解题报告

西南大学附属中学校 蒋凌宇

2019 年 11 月 19 日

# 1 Painting Edges

## 1.1 题意

给定一张 $n$ 个点 $m$ 条边的简单无向图。每条边可以不被染色或被染成 $k$ 种颜色中的一种，初始时为未染色。

有 $q$ 次操作，每次操作为改变一条边的颜色。如果一次操作后，存在一种颜色的边形成的图不是而二分图，则该操作不被执行。

输出每次操作是否被执行。

## 1.2 数据范围

$2 \le n \le 5 \cdot 10^5$, $1 \le m, q \le 5 \cdot 10^5$, $1 \le k \le 50$。

## 1.3 解题过程

### 1.3.1 简化版的题目

考虑如下的问题：

> 操作一定被执行，每次操作后询问当前状态是否合法，即是否每种颜色的边组成的图都是二分图。

则我们需要对每种颜色分别进行加边、删边操作，并维护其是否是二分图。

容易发现加边操作可以通过并查集维护到根的距离奇偶性轻松解决，但删边操作很难完成。为此，我们引入线段树分治。

### 1.3.2 线段树分治

我们考虑利用分治的思想避免删边操作。

可以发现，每条边的颜色被划分为了若干个时间区间，且所有边的区间数量和是 $O(q)$ 的。对时间建立线段树，将每个时间区间覆盖在线段树上的 $O(\log q)$ 个节点，然后对线段树进行分治。

在每个节点，我们加入覆盖了这个节点的所有边。如果是叶节点，则我们已经加入该时刻的所有边，直接查询该时刻是否合法即可；否则分别对左右子树递归进行处理。在处理完当前节点后，撤销之前进行的所有操作。

其中撤销操作可使用按秩合并的并查集实现。总时间复杂度 $O(m+nk+q\log q\log n)$。

#### 1.3.3 标准解法

本题的标准解法与简化版的题目类似。问题在于开始时我们无法确定每条边在每个区间的颜色。我们在开始时不进行覆盖，在每次操作后便可以决定它后面一段区间的颜色，此时进行覆盖即可。时间复杂度不变。

# 2 Captain America

## 2.1 题意

给定二维平面上的 $n$ 个点 $(x_i, y_i)$，你需要将每个点涂成红色或蓝色。有 $m$ 个限制，每个限制为一条平行于 $x$ 轴或 $y$ 轴的直线上红色和蓝色点的数量之差的绝对值不超过一个数。将一个点涂红和涂蓝分别花费 $r$ 和 $b$ 元，找到最小化花费的方案，或报告无解。

## 2.2 数据范围

$1 \le n, m \le 10^5$, $1 \le x_i, y_i \le 10^9$。

## 2.3 解题过程

### 2.3.1 简化问题

首先将坐标离散化，同一条直线上若有多个限制，取最紧的一个，若没有限制，则设为该直线上的点数。

由于每个点都需要被染色，不妨设 $r \le b$，则问题转化为最大化红色点数。

### 2.3.2 标准算法

设直线 $x = i$ 上的总点数为 $c_i$，限制为 $d_i$，最终染色方案数中红色点的数量为 $x_i$。则需要满足 $|x_i - (c_i - x_i)| \le d_i$，即 $\lceil \frac{c_i - d_i}{2} \rceil \le x_i \le \lfloor \frac{c_i + d_i}{2} \rfloor$。

我们构建一个网络：

$X$ 集的每个点表示直线 $x=i$，$Y$ 集的每个点表示直线 $y=i$。源点 $S$ 向每个 $X$ 集中的点连下界为 $\lceil\frac{c_i-d_i}{2}\rceil$，上界为 $\lfloor\frac{c_i+d_i}{2}\rfloor$ 的边，$Y$ 集到汇点 $T$ 连边同理。原问题中的每个点转化为 $x_i$ 到 $y_i$ 的一条边，容量为 1，有流量经过表示该点被染成红色。

容易发现，上述网络的一个可行流即为原问题的一组合法染色方案，其最大流即为原问题的最优解。

使用 Dinic 算法求上下界最大流，由于 $X$ 集与 $Y$ 集之间的边为单位边权，Dinic 算法的时间复杂度为 $O(E\sqrt{E})$。总时间复杂度 $O(m+n\sqrt{n})$。

# 3 Sweet Alchemy

## 3.1 题意

给定 $N$ 种食物，食物之间有树形的依赖关系。你有若干原材料，每种食物需要消耗一定数量的材料。你需要制作每种食物若干个，令总数最大，且需要满足若节点 $i$ 依赖 $p_i$，则 $c_{p_i}\le c_i\le c_{p_i}+D$，其中 $D$ 给定，$c_i$ 表示制作食物 $i$ 的数量。

## 3.2 数据范围

$1\le N\le 50, 1\le p_i<i$。

## 3.3 解题过程

### 3.3.1 简化问题

令 $d_i=c_i-c_{p_i}(i\ge 2), d_1=c_1$，则 $0\le d_i\le D(i\ge 2)$。则此时 $d_i$ 表示 $i$ 的子树内的食物各制作一个，则价值为子树大小，费用为子树内原费用的和。

则问题转化为：

> 给定 $N$ 种物品，每个物品有体积 $w_i$、价值 $v_i$ 和个数 $z_i$，其中 $v_i\le N$。
>
> 询问给定的容量能容纳物品的最大价值和。

### 3.3.2 标准算法

上文所述的问题即为经典的多重背包问题，但本题的限制条件中无论是背包容量，还是最终答案，都可能非常大，故无法使用经典算法解决。但本题的物品价值非常小，考虑从这个方面入手。

背包问题的一个较优解可以通过贪心得到，即将物品按照 $\frac{v_i}{w_i}$ 从大到小排序，从前往后选择。但由于物品无法拆分，这种算法无法得到最优解。

结合本题的条件，可以发现若存在物品 $i, j(i < j)$，满足 $z_i - c_i \geq v_j$ 且 $c_j \geq v_i$，即至少有 $v_j$ 个未选的物品 $i$ 和 $v_i$ 个被选的物品 $j$，我们可以不选 $v_i$ 个物品 $j$，改为选 $v_j$ 个物品 $i$，在总价值不变的情况下减少了体积，故不会变劣。

基于上述观察，我们发现，一定存在一组最优解中，只有不超过一个物品满足 $N \leq c_i \leq z_i - N$，在它前面的物品剩余数量都 $< N$，在它后面的物品选择数量都 $< N$。我们只需要确定前面的每个物品剩余的数量，后面的每个物品选择的数量，对剩余的物品执行上述贪心算法就可以得到最优解。

这可以通过对每种物品取不超过 $N$ 个，进行以价值为状态的多重背包 DP 即可，总价值为 $O(N^3)$，所以时间复杂度为 $O(N^4 \log N)$。