

# Summer Training Week 5

## 比赛简记

### Max.D.

#### 专题

本周暂无

#### 比赛

百度之星复赛（凉凉，只写了两题没拿到衣服）

一场cf div2 rating小涨

#### 题目

暂无

### Hardict

#### 专题

无

#### 比赛

百度之星一场

#### 题目

暂无

### MountVoom

#### 专题

无

## 比赛

求求来点正常cf div1

遇见类似原题的题不要被轻易影响

## 题目

无

## 个人总结

### 陈铭煊 Max.D.

补题+学习

### 龙鹏宇 Hardict

### 肖思炀 MountVoom

该补点难题了

## 本周推荐

### 陈铭煊 Max.D.

来源：

[牛客2020多校训练第九场 C Groundhog and Gaming Time](#)

标签：

区间，动态规划，线段树

**题意：**

给出  $n$  个区间  $[L_i, R_i]$  等概率取其中一个子集  $a$  得到的贡献为  $|[L_{a_1}, R_{a_1}] \cap [L_{a_2}, R_{a_2}] \cap \dots \cap [L_{a_m}, R_{a_m}]|^2$  求贡献的期望值（模  $998244353$  意义下）。

$1 \leq n \leq 5 \times 10^5, 0 \leq L_i \leq R_i \leq 10^9$

**题解：**

这里给出两个方法，一种是大家用的比较多的方法；另一种是  $dp$  通解，也就是不仅可以做平方贡献，随便和区间有关的贡献都可以。

问题首先转化为所有子集的贡献和。

将区间交的平方，看作是区间交里面元素两两配对的个数，进一步，是左点。那么问题就是每一对元素，要求其被区间交覆盖的总次数。我们将所有区间按左端点从小到大排序，按顺序加入区间  $i$  这样我们看每个离散化后的原子区间，被区间覆盖的次数，就是左到右不递增了。那么必包含区间  $i$  时，每个原子区间的每个配对右点在区间交中带来的贡献数的和，就是  $(\text{原子区间长度} \times \text{左边的点的个数的和} \times 2 + \text{原子区间长度}^2 \times \text{覆盖数} - 1)$ 。

这些信息，包括次数和乘积等，可以用线段树加以维护，每一次区间更新，然后区间查询即可。

第二种方法是题解的方法：

1. 线段的交取决于最大的左端点以及最小的右端点，同时维护两个东西比较困难。
2. 所以我们先按照线段左端点从大到小排序，那么排序后的线段的交取决于最小的右端点，以及第一个被选择的线段的左端点。
3. 考虑到直接维护右端点比较麻烦，所以考虑在一开始就钦定一个点  $X$  作为最小的右端点。
4. 所有右端点大于等于  $X$  的线段都可以选择，反之不能选择。
5. 其次被选择的线段中至少有一个线段的右端点等于  $X$  那么这个方案就是合法的。

所以可以写出一个  $O(n^2)$  的  $dp$

1.  $dp[i][j][0/1]$  代表前  $i$  个线段中钦定的  $X$  为  $j$  是/否有一个线段的右端点为  $X$
2.  $dp[i][j][0/1] = dp[i-1][j][0/1] \ (j < L[i] \vee j > R[i])$
3.  $dp[i][j][0/1] = dp[i-1][j][0/1] * 2 + (j - L[i])^2 \ (L[i] \leq j < R[i])$
4.  $dp[i][R[i]][0] = dp[i-1][R[i]][0]$
5.  $dp[i][R[i]][1] = dp[i-1][R[i]][0] + dp[i-1][R[i]][1] * 2 + (R[i] - L[i])^2$

最后使用线段树或其他数据结构就可以将该  $dp$  优化到  $O(n \log n)$

**评论：**

赛场上想的方法也是用线段树，很麻烦，第一种平方的处理很有意义。

好题，两种方法的思考都很有技巧，尤其是第二种，普适性似乎很广，当做一种套路记住好了。

## 龙鹏宇 Hardict

来源：

标签：

题意：

题解：

评论：

## 肖思炀 MountVoom

来源：

[牛客第十场 J. Identical Trees](#)

标签：

树形dp 二分图最大权匹配，树哈希

题意：

给定两棵同构的树，需要找到一个对应关系使得相同的标号尽可能多。

题解：


树形dp  $dp[i][j]$ 表示把第一棵树的i结点和第2棵树的j节点对应起来所需要的最小花费。

转移的时候对它们的子树做一个二分图最大权匹配即可，这样总的复杂度仍然是 $O(n^3)$

评论：

cmx鸽鸽写的时候树哈希被卡了，需要注意

From:  
<https://wiki.cvbbacm.com/> - **CVBB ACM Team**

Permanent link:  
[https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:alchemist:weekly\\_digest\\_10&rev=1597398247](https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:alchemist:weekly_digest_10&rev=1597398247) 

Last update: **2020/08/14 17:44**