# CSP-S 数据结构总结

gyh20

# 目录

# 树状数组

## Fenwick Tree

树状数组是一个简单，常数小，实用的数据结构，但其维护的信息比较简单，结构不如线段树完全。

# 树状数组

## Fenwick Tree

小技巧：
1. 将修改和查询的函数交换（+lowbit 与 -lowbit 互换），可以看做维护后缀和。
2. 树状数组的区间加区间和是比正常写法的线段树块很多的，具体的写法可以看做两个树状数组，一个维护原数组，一个维护前缀和。
3. 树状数组的一个小卡常，例如要在 $x$ 处 $+z$，$y$ 处 $-z$，可以 $x, y$ 一起跳，每次跳小的一边，两个跳到同一位置就立刻退出，实测很有效。

# 线段树

## Segment Tree

线段树是一个很优秀的树结构，较简单，功能多，可以维护复杂信息。

可以动态开点，可以打懒标记。

# 线段树

## Segment Tree

小技巧：
1. 线段树如果标记永久化常数会小很多。
2. 假如将线段树的大小开成 $2^k$ 的形式，可以很简单的算出一个区间在线段树上的编号，于是可以做到非递归进行线段树的操作，常数大幅减小，这就是 ZKW 线段树。
3. 线段树可以维护的信息种类太多了，在之后单独放一栏来讲。

# 并查集

## Disjoint Set Union

并查集支持合并，加上按秩合并和路径压缩可以打到 $O(n\alpha n)$ 的优秀复杂度。

一些技巧：

1. 可以用来遍历数组并删除某一位置，例如【BZOJ3211】花神游历各国。
2. 可以用来维护颜色合并信息，比如每个点初始有一个颜色，把其中一个颜色全部变为其它的某个颜色。
3. 启发式合并。这个在之后单独来。

# ST 表

## Sparse Table

正常的写法是预处理 $O(n \log n)$ 查询 $O(1)$ 的，用四毛子可以做到 $O(n) - O(1)$，不过 CSP-S 难度的比赛应该不会出现。
ST 表可以配合欧拉环游序求 LCA 的方法做到 $O(n \log n) - O(1)$ 甚至 $O(n) - O(1)$ 的 LCA。

# 堆

## Heap

堆是一个常数很小的支持 $O(\log n)$ 插入，$O(1)$ 求最大值的数据结构，可以用系统的 priority_queue，效率很高。
使用两个堆可以做到删除，常数远小于 set。

# 二分

## Binary Search

普通的二分即为二分答案，然后用数据结构检查当前值是否可行。或者在很多具有单调性的题中，可以用二分找到合适的位置。

# 二分

## Binary Search

普通的二分即为二分答案，然后用数据结构检查当前值是否可行。或者在很多具有单调性的题中，可以用二分找到合适的位置。

当然也可以扩展到数据结构上二分，在之后讨论。

# 倍增

## Binary Lifting

从大到小枚举 $2^i$，每次判断是否可以取到，如果可以取到就取，若有信息则维护。

# 倍增

## Binary Lifting

从大到小枚举 $2^i$，每次判断是否可以取到，如果可以取到就取，若有信息则维护。
下面给出一些例题，大家可以感受一下倍增的具体过程。

# 倍增

## Binary Lifting

**例题 1**

给定一棵树，多次询问一个点的 $k$ 级祖先。

# 倍增

## Binary Lifting

**例题 1**

给定一棵树，多次询问一个点的 $k$ 级祖先。

**变形**

给定一棵树，多次询问一个点第一个子树大小 $\geq k$ 的祖先。

# 倍增

## Binary Lifting

**例题 1**

给定一棵树，多次询问一个点的 $k$ 级祖先。

**变形**

给定一棵树，多次询问一个点第一个子树大小 $\geq k$ 的祖先。

二者都是用同样的倍增，但可以发现第二个倍增的终止条件与当前跳到的点有关，这个时候的倍增需要注意信息是否有单调性。

# 倍增

Binary Lifting

**例题 2：ARC060C Tak and Hotels**

在数轴上有 $n$ 个旅店，第 $i$ 个旅店的坐标为 $x_i$，$x_i$ 递增，Takahashi 每天可以走不超过 $L$ 的路程，且每天必须在一个旅店休息。
$q$ 次询问，求 Takahashi 从第 $a_i$ 个旅店出发，走多少天能走到第 $b_i$ 个旅店。
$n, q \leq 10^5$。

# 倍增

## Binary Lifting

可以尝试找一个倍增结构。若某一天在坐标为 $x_i$ 的旅店，则第二天会走到最大的坐标不超过 $x_i + L$ 的旅店，也就是，只知道当天的位置可以知道下一天的位置，于是可以倍增，求从 $i$ 出发，走 $2^k$ 天走到的位置，每次询问从 $a_i$ 开始倍增即可。

# 倍增

Binary Lifting

**例题 3:【APIO2009】会议中心**

给定 $n$ 条线段，你需要选出最多的不相交的线段，如果有多种方案取其中字典序最小的一种，$n \leq 10^5$。

# 倍增

## Binary Lifting

求最多经典贪心，将线段按 $r$ 排序，每次能选就选。

# 倍增

## Binary Lifting

求最多经典贪心，将线段按 $r$ 排序，每次能选就选。
对字典序的要求可以看做按照原顺序从前向后贪心，每次能选就选，此时的能选就选代表选了之后不违背条件，且不会使最终的答案变小。例如第一次就是假如选择线段 $[l,r]$，然后需要保证的是 $(-inf,l)$ 和 $(r,inf)$ 两个区间内分别可选的最多线段加 1 为原来的答案，实际上之后的东西也可以看作类似的结构。

# 倍增

## Binary Lifting

求最多经典贪心，将线段按 $r$ 排序，每次能选就选。
对字典序的要求可以看做按照原顺序从前向后贪心，每次能选就选，此时的能选就选代表选了之后不违背条件，且不会使最终的答案变小。例如第一次就是假如选择线段 $[l, r]$，然后需要保证的是 $(-inf, l)$ 和 $(r, inf)$ 两个区间内分别可选的最多线段加 1 为原来的答案，实际上之后的东西也可以看作类似的结构。
考虑贪心的一个小性质：若选择了线段 $[l, r]$，则下一个选择的一定是 $L > r$ 的 $R$ 最小的线段 $[L, R]$，也就是若选了一条线段，下一个被选择的线段是固定的！于是我们得到了倍增结构，求若某条线段选择，下 $2^k$ 条被选择的会是什么线段，这样我们可以快速找到一个区间内部的答案，字典序贪心就是简单的。

# 倍增

## Binary Lifting

**例题 4：CF1523H Hopping Around the Array**

有一个序列 $a$，第 $i$ 个点一步可以跳到 $[i, i+a_i]$ 中的任何一个点。

多次询问 $l, r, k$，你可以提前删掉 $k$ 个位置，剩下的位置会依次向前补齐并重标号，求走到 $r$ 的最小步数是多少。

$n, q \leq 20000, k \leq 30$。

可以先思考 $k = 0$。

# 倍增

## Binary Lifting

先考虑 $k=0$ 的情况，此时出现了一个问题，走到某一个点时下一个走到的点不确定。有可能一次走到 $r$，或者走到区间中 $i+a_i$ 最大的点这样来满足下一次走的更远。但容易发现，除了最后一次我们会一次走到 $r$，其他时候都是走到 $i+a_i$ 最大的点，于是仍然可以倍增，即倍增走到第一次 $\geq r$ 的时候，此时要特判答案为 $0,1$。

# 倍增

## Binary Lifting

先考虑 $k=0$ 的情况，此时出现了一个问题，走到某一个点时下一个走到的点不确定。有可能一次走到 $r$，或者走到区间中 $i+a_i$ 最大的点这样来满足下一次走的更远。但容易发现，除了最后一次我们会一次走到 $r$，其他时候都是走到 $i+a_i$ 最大的点，于是仍然可以倍增，即倍增走到第一次 $\geq r$ 的时候，此时要特判答案为 $0,1$。

$k>0$ 时并没有简单的贪心，考虑 DP，之前提到倍增可以信息合并，于是我们预处理 $f_{i,j,k}$ 表示从 $i$ 出发，走 $2^k$ 步，且删掉了 $j$ 个位置能走到的 $i+a_i$ 最大的点，这个信息是可以合并的，做到 $O(nk^2\log n)$ 是容易的。

# 分治

## Divide and Conquer

最经典的分治就是处理区间 $l \sim r$ 的信息时先求出所有过 $mid$ 的答案，然后递归处理 $(l, mid-1)$ 和 $(mid+1, r)$。
点分治和边分治大致同理只不过每次找的重心，另外点分治是多个子树合并。
CDQ 分治可以看成用分治消掉某些维度的影响，转化为前一半对后一半的影响，然后递归下去。

# 分治

## Divide and Conquer

**例题 1：CF526F Pudding Monsters**

给定长度为 $n$ 的排列，求其中有多少个区间排序之后值域是连续的。

$n \leq 3 \times 10^5$

尝试用分治的方法解决这个问题。

分治，处理过 mid 的区间的区间信息。先处理出一个点到 mid 的最大值最小值。
然后分类讨论，分最大值和最小值是否在 mid 的同侧，在或不在都可以得到一些限制，可以双指针在另一边找到合法位置的个数。

# 分治

## Divide and Conquer

分治结构是可以支持区间询问的，如果离线的话会很简单。
用一道例题辅助理解。

**【2020统一省选B】消息传递**

给定一棵树。
$q$ 次询问 $x, k$，请求出与 $x$ 距离恰好为 $k$ 的点数。
$n, q \leq 10^5$。

# 分治

## Divide and Conquer

分治结构也是可以记录下来的，在序列上这个结构是猫树，在树结构中这个结构是点分树。
用一道例题辅助理解：

**CodeChef-SEGPROD**

有一个长为 $n$ 的序列，$q$ 次询问区间乘积，对 $P$（不保证是素数）取模。
强制在线。
$n \le 10^6, q \le 2 \times 10^7$。

# 分块

## Brute Force

一般分块题在 CF 上的 tag 都是 brute force。
分块很多时候是用 $\sqrt{n}$ 维护块信息，支持整块操作和重构。
当然也很多时候用来平衡复杂度，把线段树/树状数组的 $O(\log) - O(\log)$ 变为 $O(1) - O(\sqrt{})$ 一类。
一些简单的应用可以去看 LOJ 上的数列分块入门，困难的应用可以去洛谷看 Ynoi，不过后者在 CSP 出现的概率较小，这里暂时不作展开。

# 线段树的其他应用

## Segment Tree

除了区间和，线段树还可以维护最大子段和，区间直径之类的信息。

由于线段树的结构非常优美，其可以支持更复杂的一些操作。

# 线段树的其他应用

## Segment Tree

线段树维护矩阵乘法，也就是说，将线段树上的信息，包括懒标记，换成矩阵。

### 例题

给定一个 01 串，$q$ 次询问一个区间 01 交错子序列的个数。

这个是简单的将 DP 方程看成矩阵，用线段树维护这个矩阵。

线段树区间加维护历史版本和也可以看成矩阵乘法，矩阵大小为 3，分别代表常数项（区间长度），区间和，区间历史版本和，贡献全部可以写成矩阵。

# 线段树的其他应用

## Segment Tree

线段树也可以维护非 $O(1)$ 量的信息，一个常用的做法是维护 $O($ 区间长度 $)$ 的信息。

**一个经典的例子**

有 $n$ 个形如 $f(x)=kx+b$ 的函数，$q$ 次询问一个区间和一个 $x$，求 $x$ 在区间内的 $f(x)$ 的最大值。

另外一个例题是 CF1172F Nauuo and Bug。
线段树由于结构优美还有很多应用，这里不再展开。

# 扫描线

## Sweepline

**例题**

平面上有 $n$ 个矩形，求所有矩形的面积并。

当然，矩形的来源并不一定是输入的矩形，很多时候可以是 dfs 序上的区间，矩形则对应 dfs 序上的路径。

# 扫描线

## Sweepline

对于一些形如长度为 $n$ 的数列，$q$ 次询问区间之类的题，也可以通过简单找找性质转化为二维数点问题来扫描线。

### P8543 「Wdoi-2」纯粹的复仇女神

给定 $n$ 个二元组 $(c_i, a_i)$，$q$ 次询问，每次给出一个区间 $[l, r]$，令 $S$ 表示区间内出现过的 $c_i$ 的集合，定义一个 $x$ 的价值为 $\min_{l \le i \le r, c_i = x} a_i$，求 $\max_{x \in S} f(x)$。

$n \le 2 \times 10^5, q \le 10^6$

# 扫描线

## Sweepline

考虑最终作为最小值的一个数，想办法统计这个数的贡献。
假设 $a_x$ 是最终的最小值，那么找到左边第一个比 $a_x$ 小的位置 $y$，右边第一个比 $a_x$ 大的位置 $z$，只要最终的询问区间满足 $y \le l \le x \le r \le z$，那么 $a_x$ 便可以被统计进入答案，于是相当与把 $n$ 个数转化为 $n$ 个矩形取 max。
矩形 max 不能差分，但仍可以利用线段树结构，用线段树维护可删堆代替。

# 莫队

## Mo's Algorithm

如果题目允许离线，并且可以支持在原有结构上快速加入或删除一个数可以考虑莫队，大致结构是相同的，有时需要根据具体情况调整内部的算法。

**【AHOI2013】作业（的子问题）**

给定一个颜色序列，$q$ 次询问区间中有多少种 $\geq a$ 且 $\leq b$ 的颜色。

$n, q \leq 10^5$

# 莫队

## Mo's Algorithm

加入删除数是容易的，统计一下每个颜色是否出现即可，询问相当于询问区间和。
但这样的复杂度是 $O(n\sqrt{m}\log n + m\log n)$ 的，可以尝试平衡复杂度。
用分块代替树状数组，$O(1)$ 修改 $O(\sqrt{n})$ 查询，总复杂度 $O(n\sqrt{m} + m\sqrt{n})$。

# 容斥

## Inclusion and Exclusion

有些时候一些东西维护起来非常麻烦，甚至不能维护，这个时候需要转化统计的对象，可以大幅减小运算量。

# 容斥

## Inclusion and Exclusion

有些时候一些东西维护起来非常麻烦，甚至不能维护，这个时候需要转化统计的对象，可以大幅减小运算量。
比如差分就是一个最简单的容斥，一些复杂一点的东西有树上连通块个数 = 边 − 点 +1，所有数之和可以看做对于每一个正整数 $k$，求 $\geq k$ 的数的个数之和。

# 特殊树结构

## Structures

笛卡尔树：序列的区间最大值为两点 LCA。
Kruskal 重构树：路径最大值为两点 LCA。
两个树结构在与路径最大值相关的问题中非常有用，并且容易实现。
很奇妙的另一个树结构：并查集启发式合并得到的树。

### 例题

有一个无向有边权图，多次询问两个点，求两点之间的最优路径，一条路径比另一条更优当且仅当其中次大边权更小。

这题做法很多，这里给一个最简单的。
如果是最大边最小那显然是最小生成树。
次大边最小可以这样考虑：按照边权从小到大枚举每条边链接，若两点所在连通块之间有一条未被枚举到的边则当前枚举的边为答案。
考虑按秩合并并查集，得到一棵高度为 log 的树，然后怎么做都行了。

# 其它

## Others

很多数据结构题会存在一些性质，来让某些枚举量变成 $\log n$ 或者 $\sqrt{n}$ 级别，有些题会保证数据随机一类的性质，这里提供一些常见性质。

1. $n$ 个数，出现次数 $\sqrt{n}$ 的数不超过 $\sqrt{n}$ 个，这个性质常用于根号分治。

2. 如果询问和二进制有关，可以考虑按位处理，和因数有关，可以考虑枚举倍数，两者都是 $O(n\log n)$ 的。

3. 随机序列的前缀最大值个数期望有 $O(\log n)$ 个，随机序列的 LIS 期望为 $O(\sqrt{n})$，随机序列的前缀和的前缀最大值个数为 $O(\sqrt{n})$。

4. 父亲在 $[1, i-1]$ 随机的树树高为 $O(\log n)$，用 Prufer 序列随机的树树高为 $O(\sqrt{n})$。

还有很多。这里只是举一些常见的例子。

看一道有趣的题。

**【北大集训2021】出题高手**

给定一个序列 $a$，保证随机生成，且值域在 $[-1000, 1000]$ 之中。

定义一个区间 $[l, r]$ 的价值为 $\frac{(\sum_{i=l}^{r} a_i)^2}{r-l+1}$。

$q$ 次询问一个区间 $[ql, qr]$，求满足 $ql \le l \le r \le qr$ 的区间 $[l, r]$ 中，$f(l, r)$ 最大的一个。

$n \le 5 \times 10^5, q = 1$ 或 $n \le 10^5, q \le 3 \times 10^5$。

首先差分转为 $(s_r - s_{l-1})^2$，类似于这样 $[ql, qr]$ 这样的询问可以用之前提到的方法转为二维数点扫描线。
容易发现，对于固定的 $r$，其对应最优的 $l$ 一定是后缀最大值或后缀最小值，否则调整显然更优，由于 $s$ 是随机序列的前缀和，这样直接做会得到 $O(n\sqrt{n})$ 个区间。
考虑优化，采用分治，每次强行这个区间过 $mid$，那么左边和右边选的位置分别需要是前缀最小值/最大值，这样两边的位置都只有 $O(\sqrt{n})$ 个，单次分治的复杂度就是 $O(n)$，总共得到的区间个数就是 $O(n\log n)$。

1. 数据结构知识总结
2. 做题方法总结
3. 真题

最后放一些往年 CSP/NOIP 的真题，以及 NOI 中一些较简单的题目。
如果还想找某一个类型的题做可以找我。

**[NOIP2015 提高组] 运输计划**

给定一棵带非负边权的树和树上的 $m$ 条路径，你需要将树上的一条边边权变为 0，最小化最终路径长度的最大值。

$n, m \leq 3 \times 10^5$

看到最小化最大值，考虑二分答案。
此时有一些路径已经合法，有一些路径还未合法。
最终被修改的边一定在所有未合法的路径上，也就是在他们的交上，所以判断交上是否存在一条边权足够大的边即可。

**[NOIP2016 提高组] 天天爱跑步**

给定一棵带非负边权的树和树上的 $m$ 个人，每个人会沿最短路以 1 的速度在 0 时刻出发，从 $s_i$ 走到 $t_i$，同时树上的每个点也有一个人，其想知道在 $w_j$ 时刻有多少人恰好在这个位置出现。
$n, m \leq 3 \times 10^5$

看到路径，可以考虑将其拆成从 $x$ 到 lca 和从 lca 到 $y$ 两部分。每一部分都可以看作 $x$ 走到 $y$ 的性质，也就是在 $x$ 处加入，在 $y$ 处删除，最后查询权值为某个值的位置个数，是容易解决的。

**[NOI2010] 超级钢琴**

有一个数列 $a$，可能存在负数，定义 $l, r$ 的价值为区间内所有数之和。
求满足长度在 $[L, R]$ 之间的区间中前价值前 $k$ 大的区间的价值和。
$n, k \leq 5 \times 10^5$

首先可以先差分变成 $s_r - s_l$。
假如是最大值那么就可以看做对于每一个 $i$，找到 $[i+l, i+r]$ 中最大的一个 $s_x$ 然后用 $s_x - s_i$ 来和答案产生贡献。
那求 $k$ 大值可以维护一些三元组 $(x, l, r)$ 表示 $x$，当前可以在 $[x+l, x+r]$ 之中选择。
每次取出最大的一个，根据最大值的位置分裂，做 $k$ 次即可，用堆维护。

## [NOI2016] 区间

数轴上有 $n$ 个区间 $[l_i, r_i]$，你需要选出 $m$ 个区间，使得这些区间有交。一个方案的代价是选择线段中长度的最大值减去长度的最小值。求最小代价，无解输出 $-1$。

$n \leq 5 \times 10^5, m \leq 2 \times 10^5$。

由于和长度有关，可以先按照长度排序。
然后双指针用线段树维护每个位置被覆盖的次数，存在一个位置被覆盖超过 $m$ 次即为合法。

### [NOI2018] 归程

有一个 $n$ 点 $m$ 边无向图，每条边有海拔和长度两个权值。
$q$ 次询问，给定 $x,k$，求从一个点出发，先走一些海拔超过 $k$ 的路径，然后步行走到 1 号点，求步行的最小距离。
$n \le 2\times 10^5, m, q \le 4\times 10^5$。

先考虑能走到的位置有哪些。
明显和最大生成树有关，不妨建一棵 Kruskal 重构树，此时能走到的点就是这个树的一个子树。
预处理 1 到所有点的最短路，求一个子树最小值即可。

## [NOI2021] 轻重边

有一棵树，你需要支持将一条链变成重边，并将与这条链相邻的其它边变成轻边。
然后询问路径重边条数。
$n, m \leq 10^5$。

直接维护轻重切换可以做到 $O(n\log^2 n)$。
更优美的做法：一条边是重边当且仅当这条边两个端点最后一次没操作的时间相同。那最后就是线段树维护颜色连续段。

其它一些没讲的题，可以下来自己看一下。
[NOIP2017 提高组] 列队
[NOIP2018 提高组] 赛道修建
[NOIP2018 提高组] 保卫王国
[CSP-S2019] 树的重心
[CSP-S2021] 廊桥分配
[NOI2015] 软件包管理器
[NOI2017] 蚯蚓排队

**Thanks!**