根号数据结构 & 根号分治

Isaunoya

2024-07-10

1. 根号数据结构

序列分块

- 首先，这个是传统意义上的分块。
-
-
-
-

序列分块

- 首先，这个是传统意义上的分块。
- 疑似有人后面要讲分块，所以这部分可以简单一些！
-
-
-

序列分块

- 首先，这个是传统意义上的分块。
- 疑似有人后面要讲分块，所以这部分可以简单一些！
- 分块是一个块状结构。
-
-

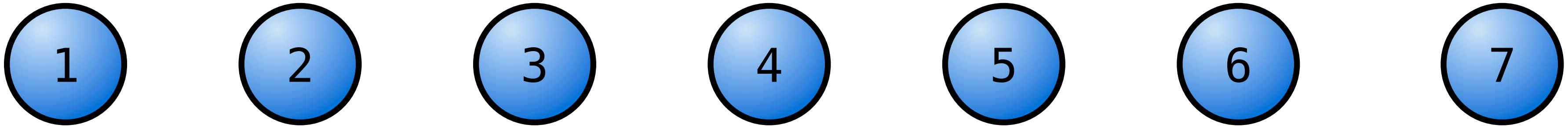
序列分块

- 首先，这个是传统意义上的分块。
- 疑似有人后面要讲分块，所以这部分可以简单一些！
- 分块是一个块状结构。
- 可以做一些简单的区间修改，区间查询的操作。
- 比如说，区间加法，区间查询和。

图解

假设一共有 7 个元素，我们设定块长为 2。（初始全都为 0。）

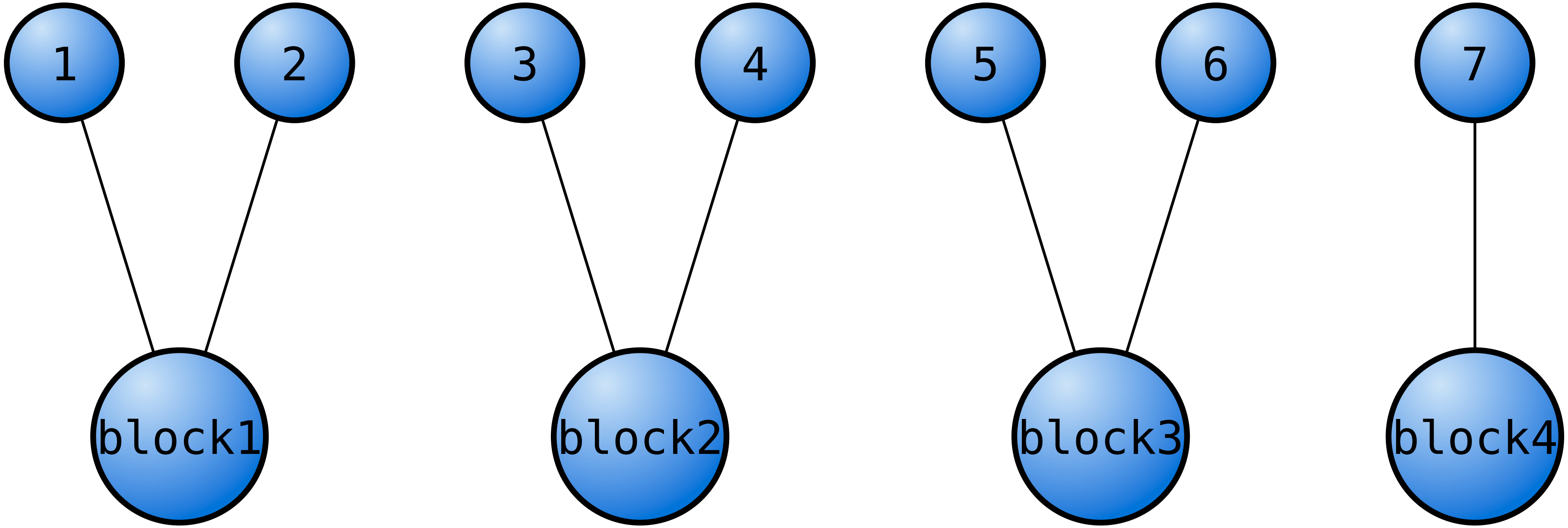
我们可以做如下划分。



图解

假设一共有 7 个元素，我们设定块长为 2。（初始全都为 0。）

我们可以做如下划分。

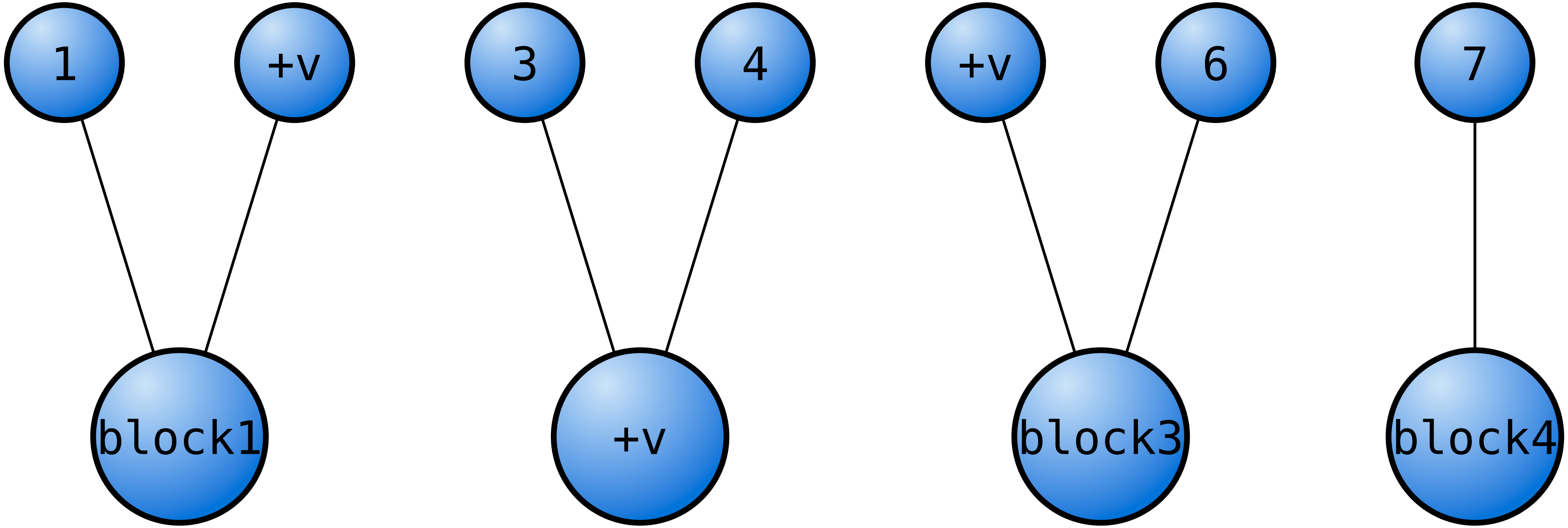


图解

假设我们要将 [2,5] 区间整体加 v，我们发现 [2,5] 区间完整的包含了 block2，以及两个散块，散块不考虑整体处理，而是直接暴力将其加 v（只要将 (2) 和 (5) 直接加），整块（只有 block2）直接加上 v。

图解

假设我们要将 [2,5] 区间整体加 v，我们发现 [2,5] 区间完整的包含了 block2，以及两个散块，散块不考虑整体处理，而是直接暴力将其加 v（只要将 (2) 和 (5) 直接加），整块（只有 block2）直接加上 v。



总结

所以其实是类似线段树的懒标记。

在做这一问题的时候可以简单的认为这就是一个根号叉的线段树。

假设有 n 个点，块大小是 B，那么我们修改的复杂度是 $O(\frac{n}{B}+B)$ 的，查询的复杂度也是 $O(\frac{n}{B}+B)$ 的。

通过均值不等式，我们可以知道复杂度最优是在 $B=\sqrt{n}$ 的时候。

试试看！

- 你有一个长度是 2^n 的序列 a，下标从 1 到 2^n.（$n \le 18$，$-10^9 \le a_i \le 10^9$）.
- 你需要对这个序列进行 q（$q \le 2 \cdot 10^5$）次操作，每次操作你都会得到一个整数 k（$0 \le k \le n-1$）. 你需要进行如下操作：
- 对于 $\forall i \in [1, 2^n - 2^k]$，如果 a_i 在本次操作中已经被交换过了，那么忽略它；否则，将 a_i 和 a_{i+2^k} 进行交换.
- 在这之后，输出序列最大子段和. 本题中，最大子段可以一个数都不选.

注意，每次操作之后不会撤销.

经典应用

- 也许你想到了一个 log 做法，但是我还是想讲根号做法。
-
-
-

经典应用

- 也许你想到了一个 log 做法，但是我还是想讲根号做法。
- 按照以往经验，得到最大子段和只需要，前缀最大值，后缀最大值，中间的最大值，这些信息合并起来就可以得到区间的最大子段和。
-
-

经典应用

- 也许你想到了一个 log 做法，但是我还是想讲根号做法。
- 按照以往经验，得到最大子段和只需要，前缀最大值，后缀最大值，中间的最大值，这些信息合并起来就可以得到区间的最大子段和。
- 其实这题也不例外，我们可以维护一个每个块的前缀，后缀，中间的最大子段和。
-

经典应用

- 也许你想到了一个 log 做法，但是我还是想讲根号做法。
- 按照以往经验，得到最大子段和只需要，前缀最大值，后缀最大值，中间的最大值，这些信息合并起来就可以得到区间的最大子段和。
- 其实这题也不例外，我们可以维护一个每个块的前缀，后缀，中间的最大子段和。
- 现在这个交换看起来非常棘手，但是你发现他是 2^k，所以那其实非常好做，因为我们可以很自然想到根号分治，将 $k = 9$ 为界，分成两个部分，也就是我取 $B = 2^9$.

第一个部分

- 我们先假设交换的都是 $k \geq 9$ 的部分。
-
-

第一个部分

- 我们先假设交换的都是 $k \geq 9$ 的部分。
- 那其实是相当于我有 $\frac{N}{B}$ 个块，需要打乱相对顺序。
-

第一个部分

- 我们先假设交换的都是 $k \geq 9$ 的部分。
- 那其实是相当于我有 $\frac{N}{B}$ 个块，需要打乱相对顺序。
- 而我每次查询的时候只需要 $O(\frac{N}{B})$ 个信息再合并即可。

第二个部分

- 第二个部分是 $k < 9$ 的部分。
-
-
-
-

第二个部分

- 第二个部分是 $k < 9$ 的部分。
- 这等价于块内交换。
-
-
-

第二个部分

- 第二个部分是 $k < 9$ 的部分。
- 这等价于块内交换。
- 但是无论你再怎么交换，块内的排布也顶多是 B 种。
-
-

第二个部分

- 第二个部分是 $k < 9$ 的部分。
- 这等价于块内交换。
- 但是无论你再怎么交换，块内的排布也顶多是 B 种。
- 也就是我可以事先预处理好，块内的排布对于所有的情况是什么样子的。
- 然后结合第一个部分就可以通过这题了。

参考代码：https://codeforces.com/contest/1716/submission/262397977

值域分块

这里是一个经典的问题。

初始有一个空集合，有两类操作：

- 插入或者删除一个自然数
- 查询集合中的 mex 是多少（mex 即为集合中最小未出现的自然数）

假设第一类操作的个数是 q_1 次。

假设第二类操作的个数是 q_2 次。

这个问题显然可以通过树状数组修改+树状数组上二分来实现。

（只有删空或者第一次添加的时候需要对树状数组做操作）

这样做的复杂度是 $O((q_1 + q_2) \log n)$。

根号做法。

- 这题显然的也会有一个根号的做法。
-
-
-
-
-

根号做法。

- 这题显然的也会有一个根号的做法。
- 注意到我只需要单点修改。
- 单点修改在块里面只需要 $O(1)$ 的时间复杂度。
- 全局查询 mex 只需要 $O(\sqrt{n})$ 的复杂度，因为我需要查看每个块是否是满的，和二分的方式是类似的。
-
-

根号做法。

- 这题显然的也会有一个根号的做法。
- 注意到我只需要单点修改。
- 单点修改在块里面只需要 $O(1)$ 的时间复杂度。
- 全局查询 mex 只需要 $O(\sqrt{n})$ 的复杂度，因为我需要查看每个块是否是满的，和二分的方式是类似的。
- 那么我们容易注意到这个做法其实是 $O(q_1 + q_2\sqrt{n})$ 的。
- 在对于 q_1 很大而 q_2 较小的时候，我们可以采用这个根号平衡的做法。

这题就是 luogu P4137 的部分内容。会莫队的同学可以做一下！

HDU7337

多组数据，给一个长度为 n 的排列，多次查询区间 $[l,r]$ 有多少个满足 $l \le i < j \le r$ 的 (i,j) 满足 $a_i + a_j$ 是一个完全平方数。

数据范围：$T \le 5, n \le 10^5, q \le 10^5$。

做法

- 考虑到能产生贡献的只有至多 $n\sqrt{n}$ 组，因为是得是完全平方数，对于单个 a_i，不可能有超过 $\sqrt{n}$ 组和它相加是一个完全平方数。
-
-
-
-

做法

- 考虑到能产生贡献的只有至多 $n\sqrt{n}$ 组，因为是得是完全平方数，对于单个 a_i，不可能有超过 $\sqrt{n}$ 组和它相加是一个完全平方数。
- 假设 $i < j$，我们可以枚举右端点，类似 HH 的项链一样。
-
-
-

做法

- 考虑到能产生贡献的只有至多 $n\sqrt{n}$ 组，因为是得是完全平方数，对于单个 a_i，不可能有超过 $\sqrt{n}$ 组和它相加是一个完全平方数。
- 假设 $i < j$，我们可以枚举右端点，类似 HH 的项链一样。
- 假设我枚举到 r，我们计算了 $1 \leq i < j \leq r$ 的所有配对。
-
-

做法

- 考虑到能产生贡献的只有至多 $n\sqrt{n}$ 组，因为是得是完全平方数，对于单个 a_i，不可能有超过 $\sqrt{n}$ 组和它相加是一个完全平方数。
- 假设 $i < j$，我们可以枚举右端点，类似 HH 的项链一样。
- 假设我枚举到 r，我们计算了 $1 \leq i < j \leq r$ 的所有配对。
- 我们只需要将所有符合条件的点对 (i, j) 的贡献加在 i 这个点上，也就是说，如果我此时想查询 $[l, r]$ 有多少个配对，我只需要查询 $[l, r]$ 的区间和就可以。
-

做法

- 考虑到能产生贡献的只有至多 $n\sqrt{n}$ 组，因为是得是完全平方数，对于单个 a_i，不可能有超过 $\sqrt{n}$ 组和它相加是一个完全平方数。
- 假设 $i < j$，我们可以枚举右端点，类似 HH 的项链一样。
- 假设我枚举到 r，我们计算了 $1 \leq i < j \leq r$ 的所有配对。
- 我们只需要将所有符合条件的点对 (i, j) 的贡献加在 i 这个点上，也就是说，如果我此时想查询 $[l, r]$ 有多少个配对，我只需要查询 $[l, r]$ 的区间和就可以。
- 同样的，这个操作可以用 $n\sqrt{n}$ 次树状数组解决，这样复杂度是 $O(n\sqrt{n}\log(n))$，难以通过本题。

根号做法好啊！

- 但是我们可以通过 $O(n\sqrt{n})$ 次 $O(1)$ 修改，以及 $O(q)$ 次 $O(\sqrt{n})$ 查询，就可以把复杂度维持在根号了！
- 就完美的去掉了一个 $\log(n)$ 的复杂度。
-

根号做法好啊！

- 但是我们可以通过 $O(n\sqrt{n})$ 次 $O(1)$ 修改，以及 $O(q)$ 次 $O(\sqrt{n})$ 查询，就可以把复杂度维持在根号了！
- 就完美的去掉了一个 $\log(n)$ 的复杂度。
- 所以在复杂度不均衡的时候，通常可以采用这种"根号平衡"的技巧，否则你可能需要采用"莫队二次离线"之类的科技来实现这个东西。

不删除莫队的引入

- 这里想讲一个不删除莫队。（莫队能做的事情大概是这个的子集，并且疑似后续有人也要讲这部分。）
-
-
-

不删除莫队的引入

- 这里想讲一个不删除莫队。（莫队能做的事情大概是这个的子集，并且疑似后续有人也要讲这部分。）

引出一个例题：AT_joisc2014_c

- 给一个数组 A，给 q 次询问，每个询问查询 $[l,r]$ 中 $(c \cdot A_i)$ 的最大值。（其中 c 为 A_i 在 $[l,r]$ 出现的次数。）
-
-

不删除莫队的引入

- 这里想讲一个不删除莫队。（莫队能做的事情大概是这个的子集，并且疑似后续有人也要讲这部分。）

引出一个例题：AT_joisc2014_c

- 给一个数组 A，给 q 次询问，每个询问查询 $[l,r]$ 中 $(c \cdot A_i)$ 的最大值。（其中 c 为 A_i 在 $[l,r]$ 出现的次数。）
- 显然这个操作不可以$O(1)$撤销，如果一定要强行撤销的话，需要配合上 multiset 等工具做到$O(\log)$的撤销。
-

不删除莫队的引入

- 这里想讲一个不删除莫队。（莫队能做的事情大概是这个的子集，并且疑似后续有人也要讲这部分。）

引出一个例题：AT_joisc2014_c

- 给一个数组 A，给 q 次询问，每个询问查询 $[l,r]$ 中 $(c \cdot A_i)$ 的最大值。（其中 c 为 A_i 在 $[l,r]$ 出现的次数。）
- 显然这个操作不可以$O(1)$撤销，如果一定要强行撤销的话，需要配合上 multiset 等工具做到$O(\log)$的撤销。
- 下面我们引入回滚莫队这一工具来帮我们解决这个问题。

不删除莫队的具体写法

- 定义 q_i 是询问，回滚莫队的做法大概就是:
- 分类讨论一下,如果左右指针共处一个块内,直接暴力,就是根号级别的,这种就不用放进 q_i 里面。
- 否则把询问丢到左端点的块里,在同一个块里的,按右端点升序排序，假设块是 $[L, R]$。
-
-
-

不删除莫队的具体写法

- 定义 q_i 是询问，回滚莫队的做法大概就是:
- 分类讨论一下,如果左右指针共处一个块内,直接暴力,就是根号级别的,这种就不用放进 q_i 里面。
- 否则把询问丢到左端点的块里,在同一个块里的,按右端点升序排序，假设块是 $[L, R]$。
- 然后对于每个块求解，由于右端点是递增的，考虑移动右端点。并同时记录 $[R+1, r]$ 的答案。
- 左边的贡献直接从 $[q[i].l, R]$，暴力就行了，这样就能得到 q_i 的答案。
- 由于不能删除，每次在询问之前记录状态，然后复制一遍，再操作，最后回退到状态 $[R+1, r]$。

不删除莫队的复杂度

- 考虑首先你有 $\frac{N}{B}$ 个块，那么对于每个块，都要移动到最右侧（这是最坏情况）。
-
-

不删除莫队的复杂度

- 考虑首先你有 $\frac{N}{B}$ 个块，那么对于每个块，都要移动到最右侧（这是最坏情况）。
- 然后，我们要考虑每个询问，一个询问只需要特殊处理块内的部分，这部分是 QB
-

不删除莫队的复杂度

- 考虑首先你有 $\frac{N}{B}$ 个块，那么对于每个块，都要移动到最右侧（这是最坏情况）。
- 然后，我们要考虑每个询问，一个询问只需要特殊处理块内的部分，这部分是 QB
- 所以分析出来总复杂度是 $O\left(\frac{N^2}{B}+QB\right)$ 的。

取块大小为 $\sqrt{\frac{N^2}{Q}}$ 理论最优，实际上要看常数。

CF736E

- 给你 n 个点，已知 m 对关系 $[u,v](|u-v| \le k)$，k 给出，询问 q 次，每次问你 $[l,r]$ 有多少个连通块。

$n,q \le 10^5, k \le 5, m \le 5 \cdot 10^5$

做法

- 其实我感觉是裸题。（狗头）
-
-
-

做法

- 其实我感觉是裸题。（狗头）
- 感觉这题的问题在于复原部分，就是需要一个可撤销并查集。
- 而这个只是并查集需要按秩合并/不路径压缩就可以完成的事情。
-

做法

- 其实我感觉是裸题。（狗头）
- 感觉这题的问题在于复原部分，就是需要一个可撤销并查集。
- 而这个只是并查集需要按秩合并/不路径压缩就可以完成的事情。
- 记录一下改了哪些点，然后把块内暴力的部分回退即可。

不删除莫队题目合集

- https://www.luogu.com.cn/problem/P5906
- https://www.luogu.com.cn/problem/AT_joisc2014_c
- https://www.luogu.com.cn/problem/CF763E

2. 根号分治

根号分治引入（CF797E）

- 给定长度为 n 的序列 a，q 次询问。
- 每次询问给出 p, k。您要不断地执行操作 $p \leftarrow p + a_p + k$，直到 $p > n$ 为止。询问的答案为操作次数。
- $1 \leq n, q \leq 10^5$，$1 \leq a_i \leq n$，$1 \leq p, k \leq n$。

如何用根号分治解决本题

- 注意到这是一个比较显然的根号分治。
-
-

如何用根号分治解决本题

- 注意到这是一个比较显然的根号分治。
- 因为当 $k \geq \sqrt{n}$ 的时候，无论 a_i 具体是多少，答案至多是 $\frac{n}{k} \leq \sqrt{n}$
-

如何用根号分治解决本题

- 注意到这是一个比较显然的根号分治。
- 因为当 $k \geq \sqrt{n}$ 的时候，无论 a_i 具体是多少，答案至多是 $\frac{n}{k} \leq \sqrt{n}$
- 当 $k < \sqrt{n}$ 的时候，这样的 k 只会有 $\sqrt{n}$ 种，直接预处理出来每个位置的答案即可（从后面转移到前面即可。）

代码：https://codeforces.com/contest/797/submission/97404205

除此之外有什么作用呢？（ABC359G）

给你一棵有 N 个顶点的树。i 这条边双向连接顶点 u_i 和 v_i 。

此外，还给出了一个整数序列 $A = (A_1, ..., A_N)$ 。

定义 $f(i, j)$ 如下：如果是 $A_i = A_j$ ，那么 $f(i, j)$ 就是从顶点 i 移动到顶点 j 所需的最小边数。如果是 $A_i \neq A_j$ ，那么就是 $f(i, j) = 0$ 。

计算下面表达式的值：

$\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} f(i, j)$

- $2 \leq N \leq 2 \cdot 10^5, 1 \leq u_i, v_i \leq N, 1 \leq A_i \leq N$

简单的虚树做法

- 首先注意到，如果 A_i 全部相同，那么可以直接计算每条边的贡献，也就是 $size_i \cdot (n - size_i)$，$size_i$ 表示 i 的子树大小。
-
-
-

简单的虚树做法

- 首先注意到，如果 A_i 全部相同，那么可以直接计算每条边的贡献，也就是 $size_i \cdot (n - size_i)$，$size_i$ 表示 i 的子树大小。
- 然后我们只需要计算颜色为 c 的部分，也就是说，我只要把颜色都为 c 的一些点，建成一棵树，然后加入一些虚点（虚点是为了让那些点联通），组成一颗虚树。
-
-

简单的虚树做法

- 首先注意到，如果 A_i 全部相同，那么可以直接计算每条边的贡献，也就是 $size_i \cdot (n - size_i)$，$size_i$ 表示 i 的子树大小。
- 然后我们只需要计算颜色为 c 的部分，也就是说，我只要把颜色都为 c 的一些点，建成一棵树，然后加入一些虚点（虚点是为了让那些点联通），组成一颗虚树。
- 虚树的边 (u, v)，需要加权，也就是原树上的 $dist(u, v)$，因为这条虚边实际上代表了 $dist(u, v)$ 条边，但是贡献系数都相同，可以一起拿来计算。然后现在的 $size_i$ 其实是表示颜色 c 在 i 子树出现多少次。（也就是不能计算虚点进去）
-

简单的虚树做法

- 首先注意到，如果 A_i 全部相同，那么可以直接计算每条边的贡献，也就是 $size_i \cdot (n - size_i)$，$size_i$ 表示 i 的子树大小。
- 然后我们只需要计算颜色为 c 的部分，也就是说，我只要把颜色都为 c 的一些点，建成一棵树，然后加入一些虚点（虚点是为了让那些点联通），组成一颗虚树。
- 虚树的边 (u, v)，需要加权，也就是原树上的 $dist(u, v)$，因为这条虚边实际上代表了 $dist(u, v)$ 条边，但是贡献系数都相同，可以一起拿来计算。然后现在的 $size_i$ 其实是表示颜色 c 在 i 子树出现多少次。（也就是不能计算虚点进去）
- 按照上面的做法，我们已经做完了这个问题。只需要一个虚树。

但是能不能更简单呢！

可以的！

-
-
-

但是能不能更简单呢！

可以的！

- 首先考虑设定一个阈值 B。
-
-

但是能不能更简单呢！

可以的！

- 首先考虑设定一个阈值 B。
- 对于颜色出现次数 $cnt \le B$ 的，我们可以采用 cnt^2 的做法，这一部分的复杂度是 $\sum cnt^2 \le \sum cnt \cdot B \le n \cdot B$
-

但是能不能更简单呢！

可以的！

- 首先考虑设定一个阈值 B。
- 对于颜色出现次数 $cnt \le B$ 的，我们可以采用 cnt^2 的做法，这一部分的复杂度是 $\sum cnt^2 \le \sum cnt \cdot B \le n \cdot B$
- 剩下的是颜色 c 出现次数 $cnt > B$ 的，这种颜色不会超过 $\frac{n}{B}$ 个，对于这种，我们可以采用整棵树都遍历一遍，也就是，对于考虑 $size_i$ 定义成 i 子树内 c 的出现次数，然后遍历一整棵树，计算 $\sum size_i \times (size_1 - size_i)$ 即可。

代码链接

- 根号分治：https://atcoder.jp/contests/abc359/submissions/54856187
- 虚树：https://atcoder.jp/contests/abc359/submissions/54829502

例题 1

https://www.codechef.com/problems/MONSTER

- 大厨正在玩一个打怪兽的小游戏。游戏中初始时有 n 只怪兽排成一排，从左到右编号为
- 0~$n-1$。第 i 只怪兽的初始血量为 h_i，当怪兽的血量小于等于 0 时，这只怪兽就挂了。
- 大厨要进行 q 次操作。每次操作中，大厨会选择两个整数 x 和 y，并向下标 k 满足 $k\&x=k$ 的怪兽开炮（此处 & 代表按位与操作）。被炮弹打到的怪兽会掉 y 点血。
- 请告诉大厨，在他每次操作后，还有多少怪兽活着。

做法

- 首先问题可以转化成，每个怪兽什么时候死掉，然后做一个前缀和就可以解决这个问题。
-
-
-

做法

- 首先问题可以转化成，每个怪兽什么时候死掉，然后做一个前缀和就可以解决这个问题。
- 如何知道一个怪兽什么时候死掉呢。
-
-

做法

- 首先问题可以转化成，每个怪兽什么时候死掉，然后做一个前缀和就可以解决这个问题。
- 如何知道一个怪兽什么时候死掉呢。
- 引入另一个问题：给一个序列 A_i，要找到第一个前缀和大于等于 S 的位置。（使用分块）
-

做法

- 首先问题可以转化成，每个怪兽什么时候死掉，然后做一个前缀和就可以解决这个问题。
- 如何知道一个怪兽什么时候死掉呢。
- 引入另一个问题：给一个序列 A_i，要找到第一个前缀和大于等于 S 的位置。（使用分块）
- 这个要如何处理，就是考虑在第 i 个块内能不能达到 S 这个值，如果达到了，再去块里面找到更精确的位置。

那这个题到底应该咋做？

- 其实和刚才引入的问题类似。
-
-
-
-

那这个题到底应该咋做？

- 其实和刚才引入的问题类似。
- 考虑每个怪物在一个块里面会受到多少伤害（这个可以通过高维前缀和解决）。
- 然后如果某个怪物 i 在这个块里面死了，我们直接暴力算这个 i 在具体什么时候死掉的。
-
-

那这个题到底应该咋做？

- 其实和刚才引入的问题类似。
- 考虑每个怪物在一个块里面会受到多少伤害（这个可以通过高维前缀和解决）。
- 然后如果某个怪物 i 在这个块里面死了，我们直接暴力算这个 i 在具体什么时候死掉的。
- 这样对于每个块，我们计算的复杂度是 $O(n\log n + q)$
-

那这个题到底应该咋做？

- 其实和刚才引入的问题类似。
- 考虑每个怪物在一个块里面会受到多少伤害（这个可以通过高维前缀和解决）。
- 然后如果某个怪物 i 在这个块里面死了，我们直接暴力算这个 i 在具体什么时候死掉的。
- 这样对于每个块，我们计算的复杂度是 $O(n \log n + q)$
- 对于每个怪物，我们计算的复杂度其实是 $O(B)$

那这个题到底应该咋做？

- 其实和刚才引入的问题类似。
- 考虑每个怪物在一个块里面会受到多少伤害（这个可以通过高维前缀和解决）。
- 然后如果某个怪物 i 在这个块里面死了，我们直接暴力算这个 i 在具体什么时候死掉的。
- 这样对于每个块，我们计算的复杂度是 $O(n\log n+q)$
- 对于每个怪物，我们计算的复杂度其实是 $O(B)$

那么最终复杂度是 $O\left(\frac{n}{B}(n\log n+q)+nB\right)$

例题 2

百度之星决赛 2023

每个位置上有很多颜色，多次询问 x,y，问 (x,y) 这两个颜色有多少共同位置。

$N \leq 2 \times 10^5$，并且保证位置颜色数总和 $\leq 2 \times 10^5$

做法

- 有一个很显然的做法是这样的，可以直接 bitset，f_x 表示 x 有那些位置，最后答案就是 $(f_x \& f_y).count()$，这个复杂度是 $O(q \times \frac{n}{w})$ 的，赛时卡了这个做法。
-
-
-
-

做法

- 有一个很显然的做法是这样的，可以直接 bitset，f_x 表示 x 有那些位置，最后答案就是 $(f_x \& f_y).count()$，这个复杂度是 $O(q \times \frac{n}{w})$ 的，赛时卡了这个做法。
- 然后就是考虑根号分治。
-
-
-

做法

- 有一个很显然的做法是这样的，可以直接 bitset，f_x 表示 x 有那些位置，最后答案就是 $(f_x \& f_y).count()$，这个复杂度是 $O(q \times \frac{n}{w})$ 的，赛时卡了这个做法。
- 然后就是考虑根号分治。
- 如果其中一个颜色出现了 $\leq B$ 次，那么我可以计算出来他和所有颜色重合位置数，暴力可以 $O(B)$ 单次询问，这部分是 $O(qB)$ 的。
-
-

做法

- 有一个很显然的做法是这样的，可以直接 bitset，f_x 表示 x 有那些位置，最后答案就是 $(f_x \& f_y).count()$，这个复杂度是 $O(q \times \frac{n}{w})$ 的，赛时卡了这个做法。
- 然后就是考虑根号分治。
- 如果其中一个颜色出现了 $\leq B$ 次，那么我可以计算出来他和所有颜色重合位置数，暴力可以 $O(B)$ 单次询问，这部分是 $O(qB)$ 的。
- 否则两个颜色都出现 $> B$ 次，考虑颜色个数是 $\frac{n}{B}$ 的。
- 可以预处理出来 $ans_{x,y}$ 表示 x 和 y 的相交个数，x 的预处理，我最多需要枚举 $\sum_y cnt_y = n$ 次，复杂度 $O(\frac{N}{B} \times N)$ 的预处理

例题 3

给你一个长为 n 的序列 a，有 m 次查询，每次查询区间 $[l,r]$ 模 k 意义下的最小值。

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

小编也很惊讶到底怎么做

- 可以先考虑把 k 根号分治。
-
-
-

小编也很惊讶到底怎么做

- 可以先考虑把 k 根号分治。
- 对于 $k \le B$ 的部分，直接预处理出来，然后带一个 rmq，就能解决这个问题。
-
-

小编也很惊讶到底怎么做

- 可以先考虑把 k 根号分治。
- 对于 $k \leq B$ 的部分，直接预处理出来，然后带一个 rmq，就能解决这个问题。
- 对于 $k > B$ 的部分可以考虑 $a \bmod k = a - c \times k$，枚举 c，寻找 $a_i \geq c \times k$ 并且 $a_i - c \times k$ 最小。
- 然后这样一个询问就会变成，$O(\frac{V}{B})$ 个询问，然后我们考虑离线解决这个问题，将元素降序排序，将询问也降序排序，依次将 $\geq k$ 的元素插入到序列，再进行 rmq 即可。

如何平衡复杂度呢？

- 考虑你一共只有 $O(n)$ 次插入，有 $O\left(m \times \frac{V}{B}\right)$ 次查询，要考虑别的方向。
-
-
-
-

如何平衡复杂度呢？

- 考虑你一共只有 $O(n)$ 次插入，有 $O(m \times \frac{V}{B})$ 次查询，要考虑别的方向。
- 猫树分治可以做到一个事情是分成 $\log$ 层，占用 $O(n \log n)$ 的空间，然后可以做到 $O(1)$ 查询 rmq。（没有用到 ST 表的那种区间可重复。）
-
-
-

如何平衡复杂度呢？

- 考虑你一共只有 $O(n)$ 次插入，有 $O(m \times \frac{V}{B})$ 次查询，要考虑别的方向。
- 猫树分治可以做到一个事情是分成 $\log$ 层，占用 $O(n \log n)$ 的空间，然后可以做到 $O(1)$ 查询 rmq。（没有用到 ST 表的那种区间可重复。）
- 但是猫树有一个弱点，是插入一个数字需要 $O(n)$ 的时间复杂度。
- 所以我们考虑先分块，维护一个前缀 min 和后缀 min，就可以做到一个 $O(\sqrt{n})$ 修改，$O(1)$ 查询的复杂度。
- 总复杂度是 $O(n\sqrt{n} + m\frac{V}{B})$，其中 V 是值域大小。

例题 4

CF1039D 有一棵 n 个节点的树。

其中一个简单路径的集合被称为 k 合法当且仅当：

树的每个节点至多属于其中一条路径，且每条路径恰好包含 k 个点。

对于 $k \in [1, n]$，求出 k 合法路径集合的最多路径数 即：设 k 合法路径集合为 S，求最大的 $|S|$。

$n \leq 10^5$。

暴力！

你需要知道的：

- NOIp2018 D1T3: https://www.luogu.com.cn/problem/P5021
- 不详细展开如何做这个题
- 只需要按照上面那个题贪心求解就可以做到对于单个 k 的问题来做到 $O(n)$.

做法！

- 观察：$ans_{i+1} \le ans_i$ 且 $ans_i \le \frac{n}{i}$，这其实类似整除分块的性质。
-
-
-

做法！

- 观察：$ans_{i+1} \le ans_i$ 且 $ans_i \le \frac{n}{i}$，这其实类似整除分块的性质。
- 考虑根号分治，一部分暴力，一部分按根号的性质来。
-
-

做法！

- 观察：$ans_{i+1} \le ans_i$ 且 $ans_i \le \frac{n}{i}$，这其实类似整除分块的性质。
- 考虑根号分治，一部分暴力，一部分按根号的性质来。
- 考虑到块取成 q，一部分暴力的复杂度是 $O(nq)$。
-

做法！

- 观察：$ans_{i+1} \leq ans_i$ 且 $ans_i \leq \frac{n}{i}$，这其实类似整除分块的性质。
- 考虑根号分治，一部分暴力，一部分按根号的性质来。
- 考虑到块取成 q，一部分暴力的复杂度是 $O(nq)$。
- 然后你发现剩下的值域仅仅是 $\left[0, \frac{n}{q}\right]$，由于这个部分你需要一个二分，所以复杂度带个 log，考虑到每个块，然后剩下的部分复杂度就是 $\frac{n}{q} n \log n$，所以把两部分搞起来，理论复杂度是 $O\left(nq + \frac{n}{q} n \log n\right)$

例题 5

CF587F

- 给定 n 个字符串 $s_{\{1...n\}}$。
- q 次询问 $s_{\{l...r\}}$ 在 s_k 中出现了多少次。
- $n, q, \sum_{i=1}^{n} |s_i| \leq 10^5$。

做法

（如果不会 AC 自动机可以先忽略一点 AC 自动机的部分，因为 AC 自动机去掉之后就是一个树上的问题）

- 我们考虑到 AC 自动机的本质，可以先建出一个 fail 树。
- k 的子树里都是包含 s_k 的串。
- 查询 k 的子树和就相当于查询 s_k 出现了几次。
-
-

做法

（如果不会 AC 自动机可以先忽略一点 AC 自动机的部分，因为 AC 自动机去掉之后就是一个树上的问题）

- 我们考虑到 AC 自动机的本质，可以先建出一个 fail 树。
- k 的子树里都是包含 s_k 的串。
- 查询 k 的子树和就相当于查询 s_k 出现了几次。
- 如果暴力做，是把 s_k 对应的点设置成 1，然后 $i \in [l, r]$ 的 s_i 对应节点都查询一遍子树和。
-

做法

（如果不会 AC 自动机可以先忽略一点 AC 自动机的部分，因为 AC 自动机去掉之后就是一个树上的问题）

- 我们考虑到 AC 自动机的本质，可以先建出一个 fail 树。
- k 的子树里都是包含 s_k 的串。
- 查询 k 的子树和就相当于查询 s_k 出现了几次。
- 如果暴力做，是把 s_k 对应的点设置成 1，然后 $i \in [l, r]$ 的 s_i 对应节点都查询一遍子树和。
- 然后如果再简化一点呢？是不是可以前缀和相减！

根号分治！

- 这个显然可以根号分治，$> \sqrt{\sum len_i}$ 的和 $\le \sqrt{\sum len_i}$ 的分别处理。
- 为方便书写，下文规定 $L = \sum len_i$
-
-

根号分治！

- 这个显然可以根号分治，$> \sqrt{\sum len_i}$ 的和 $\leq \sqrt{\sum len_i}$ 的分别处理。
- 为方便书写，下文规定 $L = \sum len_i$
- 首先我们对于 $|s_k| > \sqrt{L}$ 的，我们可以相对暴力一点，可以考虑直接采用上文的暴力做法，单点加 1，然后询问直接做前缀相减。
-

根号分治!

- 这个显然可以根号分治，$> \sqrt{\sum len_i}$ 的和 $\leq \sqrt{\sum len_i}$ 的分别处理。
- 为方便书写，下文规定 $L = \sum len_i$
- 首先我们对于 $|s_k| > \sqrt{L}$ 的，我们可以相对暴力一点，可以考虑直接采用上文的暴力做法，单点加 1，然后询问直接做前缀相减。
- 如果对于 $|s_k| \leq \sqrt{L}$ 的，可以采用反过来的方式，考虑让 $i \in [l, r]$ 对应的 s_i 节点子树加 1，最后查询 s_k 对应节点的值。（这个可以通过 dfn 序加上树状数组差分实现）

如果你没有听懂怎么反过来的也没关系！

我觉得和下面这个例子很像！（可以理解一下）

- 单点 u 加 v。
- 查询子树 a 的和。
-
-
-

如果你没有听懂怎么反过来的也没关系！

我觉得和下面这个例子很像！（可以理解一下）

- 单点 u 加 v。
- 查询子树 a 的和。
- 这个问题其实可以转化成：
- root $\rightarrow u$ 加 v。（链加）
- 查询 a 处的值。

例题 6

P5309 [Ynoi2011] 初始化

给定一个长度为 n 的序列 $a_1, a_2, \ldots, a_n$。q 次操作，操作有两种：

1. 对于所有 $1 \le i \le n, i \equiv y \pmod x$，将 a_i 加上 z。
2. 求 $a_l, \ldots, a_r$ 的和。

答案对 $10^9 + 7$ 取模。$n, q \le 2 \times 10^5$。

做法

- 考虑对 x 根号分治。
- 对于 $x \geq \sqrt{n}$，分块加法即可！
- 对于 $x < \sqrt{n}$，发现本质不同的 x, y 只有 $O(n)$ 种类，直接记录每一项的贡献，相当于打上懒标记
-
-

做法

- 考虑对 x 根号分治。
- 对于 $x \geq \sqrt{n}$，分块加法即可！
- 对于 $x < \sqrt{n}$，发现本质不同的 x, y 只有 $O(n)$ 种类，直接记录每一项的贡献，相当于打上懒标记
- 查询时，第一类可以直接分块查询！
- 第二类可以通过遍历每个 x 来查询答案！

综上，复杂度 $O(n\sqrt{n})$，可以通过本题。

例题 7

「CodeChef COUNTARI」Arithmetic Progressions

- 给定一个序列 $a_1, \ldots a_n$。
- 求满足下列条件的三元组 (i, j, k) 的数量：
  - $1 \le i < j < k \le n$
  - $a_j - a_i = a_k - a_j$
- $3 \le n \le 10^5, 1 \le a_i \le 3 \times 10^4$

做法！

- 先将式子写成 $2a_j = a_i + a_k$。
- 考虑分块，假设块大小为 B，值域为 m，分若干情况讨论。
-
-
-
-

做法！

- 先将式子写成 $2a_j = a_i + a_k$。
- 考虑分块，假设块大小为 B，值域为 m，分若干情况讨论。
- 假设 (i, j, k) 属于不同块，枚举 j 所在的块，两边可以直接 NTT，然后直接统计贡献就可以，时间复杂度是 $O\left(\frac{nm\log m}{B}\right)$
-
-
-

做法！

- 先将式子写成 $2a_j = a_i + a_k$。
- 考虑分块，假设块大小为 B，值域为 m，分若干情况讨论。
- 假设 (i, j, k) 属于不同块，枚举 j 所在的块，两边可以直接 NTT，然后直接统计贡献就可以，时间复杂度是 $O\left(\frac{nm\log m}{B}\right)$
- j, k 在同一块，在枚举 j 的时候同时维护 j 前面每个数的出现次数，并枚举 k 统计贡献即可，复杂度是 $O(nB)$ 的。
-
-

做法！

- 先将式子写成 $2a_j = a_i + a_k$。
- 考虑分块，假设块大小为 B，值域为 m，分若干情况讨论。
- 假设 (i, j, k) 属于不同块，枚举 j 所在的块，两边可以直接 NTT，然后直接统计贡献就可以，时间复杂度是 $O\left(\frac{nm\log m}{B}\right)$
- j, k 在同一块，在枚举 j 的时候同时维护 j 前面每个数的出现次数，并枚举 k 统计贡献即可，复杂度是 $O(nB)$ 的。
- i, j 在一个块，但是 j, k 不在同一个块：类似，时间复杂度 $O(nB)$。
- 可以知道 $B = \sqrt{m\log m}$ 最优，时间复杂度 $O(n\sqrt{m\log m})$

例题 8

https://www.hackerrank.com/contests/countercode/challenges/subset/problem

有三种操作：

- 插入 x
- 删除 x
- 给定 s,查询 $a\&s = a$ 的个数

$x, s < 2^{16}, q \leq 200000$

做法 & 代码

- 1 操作和 2 操作本质一样，只有 +1/-1 系数的区别
- 这一个东西出现在根号分治里面
- 我可以同样的把前八位和后八位分治（按照二进制位）
-

做法 & 代码

- 1 操作和 2 操作本质一样，只有 +1/-1 系数的区别
- 这一个东西出现在根号分治里面
- 我可以同样的把前八位和后八位分治（按照二进制位）
- 记录一个 $a_{x,y}$ 表示前 8 位为 x 的超集，后 8 位为 y 的个数。

做法 & 代码

- 1 操作和 2 操作本质一样，只有 +1/-1 系数的区别
- 这一个东西出现在根号分治里面
- 我可以同样的把前八位和后八位分治（按照二进制位）
- 记录一个 $a_{x,y}$ 表示前 8 位为 x 的超集，后 8 位为 y 的个数。

那我查询的时候只需要去做一个很容易的事情，就是假设查询的是 $x \times 2^8 + y$，我只需要查询 $\sum_{z \subset y} a_{x,z}$

https://www.hackerrank.com/contests/countercode/challenges/subset/submissions/code/178351650

例题 9 P4062 [Code+1] Yazid 的新生舞会

https://www.luogu.com.cn/problem/P4062

对于任意一个子区间 $[l, r]$，如果该子区间内的众数在该子区间的出现次数严格大于 $\frac{r-l+1}{2}$（即该子区间长度的一半），那么 Yazid 就说这个子区间是“新生舞会的”。

所谓众数，即为该子区间内出现次数最多的数。特别地，如果出现次数最多的数有多个，我们规定值最小的数为众数。

现在，Yazid 想知道，共有多少个子区间是“新生舞会的”。

$n \leq 5 \times 10^5$

做法！

- 对于颜色出现总数 $\leq \sqrt{n}$ 的，区间长度一定不会超过 $2\sqrt{n}$，那么我们就只计算区间长度 $\leq 2\sqrt{n}$ 的，这样的复杂度是 $O(n\sqrt{n})$的。
- 对于颜色出现总数 $> \sqrt{n}$ 的，我们可以直接 $O(n)$ 扫一遍，就可以知道答案了，这个的复杂度也是 $O(n\sqrt{n})$ 的。

综上，复杂度是 $O(n\sqrt{n})$ 的。

3. 按时间分治

引入例题 1

http://zijian-lv.com/problem/45

定义 $\hat{A}_i = \sum_{j=0}^{n-1} n^{ij} A_j$ 其中 $n^k = e^{\frac{2\pi i k}{n}}$

- i x 表示 A_i 加上 x。
- i 表示查询 $\hat{A}_i$ 的值。

注意：数组是从 0 开始计数的，输入的都是整数。

假设你和我一样菜不够深刻的理解法法塔

- 你现在手里有个黑盒，可以用 $O(n \log n)$ 的复杂度让你从 A 得到一个 $\hat{A}$。
- 你还有一个操作是，假设你现在一次性修改了 K 个元素，你想得到一个 $\hat{A}_i$，你可以只通过 $O(K)$ 复杂度来完成。
-
-

假设你和我一样菜不够深刻的理解法法塔

- 你现在手里有个黑盒，可以用 $O(n\log n)$ 的复杂度让你从 A 得到一个 $\hat{A}$。
- 你还有一个操作是，假设你现在一次性修改了 K 个元素，你想得到一个 $\hat{A}_i$，你可以只通过 $O(K)$ 复杂度来完成。
- 那么这个事情其实还是比较显然的，我可以每修改 B 个元素，重新做一次法法塔，得到一个新的 $\hat{A}$，然后保证每次“一次性”修改的元素不会超过 B 个，这样就会得到一个 $O(\frac{q}{B}n\log n+qB)$ 的复杂度。
- 此时 B 取到 $\sqrt{n\log n}$ 最优。

例题 2

P5443 [APIO2019] 桥梁

给定一张 N 个点，M 条边的无向带权图。每次询问给定一个二元组 (x, y)，从 x 号节点开始出发，只允许通过边权$\geq y$的边。问能够到达的联通块最大的大小。要求动态修改边权。

对于全部数据，$1 \leq n \leq 5 \times 10^4$，$0 \leq m \leq 10^5$，$1 \leq q \leq 10^5$。

补充

- 这题需要对克鲁斯卡尔重构树可能有一定的理解（?）
-
-

补充

- 这题需要对克鲁斯卡尔重构树可能有一定的理解（?）
- 所以简单的提一嘴最小/最大生成树的克鲁斯卡尔重构树。
- 只允许通过边权 $\geq y$ 的边，相当于克鲁斯卡尔重构树倍增跳到一个点，他的子树就是可以到的所有地方。

做法

- 通过例题 1，你已经理解了按时间分块是做什么用的了！
- 那么这个题我们也可以很自然想到可以按时间分块来做到这个事情！
-
-
-
-

做法

- 通过例题 1，你已经理解了按时间分块是做什么用的了！
- 那么这个题我们也可以很自然想到可以按时间分块来做到这个事情！
- 我们显然是对于一个块，只能重构一次，并且每个询问都要在 $O(B)$ 的复杂度中解决。
- 又限制了 $\geq$ 的条件，所以我们把两个东西全都降序排序，也就是我可以一边加入边，一边解决这个问题！
- 在解决询问的时候，我需要按秩合并的并查集！
- 然后再解决完询问的时候，加入的边，需要再撤销一次！

复杂度是 $O(\frac{q}{B}(m\log n + B \times B\log n)) = O(\frac{qm}{B}\log n + qB\log n)$

例题 3

https://qoj.ac/problem/7181

有三种操作

- 加入一个点到集合 S
- 从集合 S 删除一个点
- 查询有没有一条边 (u, v) 满足 u 在 S 中，而 v 不在 S 中的，然后把这条边删掉。

我可能用了一个麻烦的根号分治做法

- 当时 vp 这场的时候我想出来了一个逆天做法。
- 但还是觉得要不简单讲讲。
-
-
-

我可能用了一个麻烦的根号分治做法

- 当时 vp 这场的时候我想出来了一个逆天做法。
- 但还是觉得要不简单讲讲。
- 初始 checklist 是一个全都合法的边集。
- 对于加入点或者删除点，重新把它的边加入到一个 checklist 里面。（这是一个很显然的暴力做法）
- 然后每次查询的时候直接进入 checklist 里面查找。

时间分块的部分

- 就是考虑时间分块，每一块的大小为 B，然后能注意到的事情是：在一个块内你至多改变 B 个点的状态。
- 可以预见的是，如果你每次加入 $2B$ 条改变状态之后符合条件的边，那么一定会有 B 条是有效的，这已经大于这个块内的询问数量了。
- 所以我们可以用两次根号分治。就可以解决本题。

icpc 网络赛 Multiply Then Plus

有 n 个二元组 (a_i, b_i)，q 个操作。

每个操作形如：

- 1 k a b 表示把第 k 个二元组改成 (a_i, b_i)
- 2 x l r 表示查询 $\max_{l \leq i \leq r}(a_i x + b_i)$

前置知识

- 李超树，李超树上面可以存很多一次函数，每个节点存一条优势直线。
- 可以用来解决全局的 $\max(a_i x + b_i)$。
- 假设你要求最大值的李超树，那么 $[l, r]$ 对应的优势直线就是在 $f(\mathtt{mid})$ 最大的那个直线。
- 非优势直线下传，被严格劣了就丢掉，复杂度 $O(n \log V)$，V 是 x 的范围。

怎么做？

这个题题解给的是 $O(n\log^2 n)$ 的，场上通过的数量不多。

- 考虑按时间分块。
-
-
-
-

怎么做？

这个题题解给的是 $O(n\log^2 n)$ 的，场上通过的数量不多。

- 考虑按时间分块。
- 那么我们可以知道最多单点修改 B 次。
- 我们可以在时间分块的最开始，把这些会被修改的点全都挖掉。
- 然后我们可以写一个静态分块，块里面使用李超树，这样的时间复杂度是 $O(n\log V)$ 的。
-

怎么做？

这个题题解给的是 $O(n\log^2 n)$ 的，场上通过的数量不多。

- 考虑按时间分块。
- 那么我们可以知道最多单点修改 B 次。
- 我们可以在时间分块的最开始，把这些会被修改的点全都挖掉。
- 然后我们可以写一个静态分块，块里面使用李超树，这样的时间复杂度是 $O(n\log V)$ 的。
- 对于每次询问，我们的复杂度是 $O(\frac{n}{B}\log V + B)$。

综上我们可以把这个题做到复杂度 $O(\frac{q}{B}(n\log V) + q(\frac{n}{B}\log V + B))$。

4. 常见根号题目解析

P8330 [ZJOI2022] 众数

九条可怜是一个有超能力的女孩子，但她的超能力只能作用于一些奇怪的事情上。

有一天，可怜得到了一个序列 $a_1, a_2, ..., a_n$，她可以对这个序列使用一次超能力: 选择一个区 间 $[l, r]$（$1 \leq l \leq r \leq n$）和一个整数 $k \in [-10^9, 10^9]$，将区间内的所有数 $a_l, a_{l+1}, ..., a_r$ 加上 k。

九条可怜很喜欢长得比较一致的序列，因此她希望最终的序列众数的出现次数尽可能多。给出序列 a，你需要输出最终序列的众数出现次数的最大值，并输出这个众数的所有可能取值。注意对于一个序列，众数的取值可能不止一个。 保证一个数列中的数不全相等

$\sum n \leq 5 \times 10^5, 1 \leq a_i \leq 10^9.$

做法！

- 是选择一段区间 $[l, r]$ 使得 $[l, r]$ 内的众数次数与除去此段区间外的众数次数的和最大，此时众数答案是 $[l, r]$ 外的任何一个众数。
-
-
-
-

做法！

- 是选择一段区间 $[l, r]$ 使得 $[l, r]$ 内的众数次数与除去此段区间外的众数次数的和最大，此时众数答案是 $[l, r]$ 外的任何一个众数。
- 一般“出现次数”都与 根号分治 挂钩，因为出现次数少的数可以直接考虑出现次数，出现次数多的数可以直接考虑每个数，这样就出现了根号。
-
-
-

做法！

- 是选择一段区间 $[l,r]$ 使得 $[l,r]$ 内的众数次数与除去此段区间外的众数次数的和最大，此时众数答案是 $[l,r]$ 外的任何一个众数。
- 一般“出现次数”都与 根号分治 挂钩，因为出现次数少的数可以直接考虑出现次数，出现次数多的数可以直接考虑每个数，这样就出现了根号。
- 规定 $\geq B$ 的为大数，否则为小数。
- 对于大数来说，对大数做一个前缀和，可以对另一个数直接扫描，就可以知道这个翻转操作带来的增量是多少。
- 注意到可以处理每个位置 i，使得其中某一个数出现 K 次的最小右端点，时间是 $O(nB)$ 的，这样我们只需要枚举去掉哪个区间，去掉这个区间能多出几个数就可以了。

细节部分

细节部分可以自行查看这篇题解的代码。

https://www.luogu.com.cn/article/p9ipplv2

ucup stage 2 Shenzhen G

https://contest.ucup.ac/contest/1540/problem/8334

- 定义 $f(s,t) = \sum s_i \neq t_i$
- 给出 N 个字符串， $s_1 \ldots s_n$，和一个常量 K，每个字符串恰好有 M 个字符组成。
- 有 q 次询问，询问 $\sum f(s_i, t) \leq K$

$N, Q \leq 300, M \leq 6 \times 10^5, K \leq 10$

做法

- 首先发现 K 只有 10，这是很好的事情，因为超过 10 就可以直接扔掉了。
-
-
-
-
-

做法

- 首先发现 K 只有 10，这是很好的事情，因为超过 10 就可以直接扔掉了。
- 然后这引导我们可以分块判定。
-
-
-
-

做法

- 首先发现 K 只有 10，这是很好的事情，因为超过 10 就可以直接扔掉了。
- 然后这引导我们可以分块判定。
- 也就是说我们可以把他分成 B 段，每一段哈希，如果不一样的话我再进去找有几个不一样的。
-
-
-

做法

- 首先发现 K 只有 10，这是很好的事情，因为超过 10 就可以直接扔掉了。
- 然后这引导我们可以分块判定。
- 也就是说我们可以把他分成 B 段，每一段哈希，如果不一样的话我再进去找有几个不一样的。
- 这样至多进去找 K 个块。
- 单次复杂度是 $O\left(N\left(KB + \frac{M}{B}\right)\right)$。
- 综上，复杂度是 $O\left(NQ\left(KB + \frac{M}{B}\right)\right)$ 的，取 $B = \sqrt{\frac{M}{K}}$ 得到最优复杂度。

P7738 [NOI2021] 量子通信

https://www.luogu.com.cn/problem/P7738

题意简述：给出 n 个 256 位二进制数（随机生成）a_i。对于 m 个询问，每次给出一个 256 位二进制数 x 和一个十进制数 k，求是否存在 a_i 使得 x, a_i 不同位数个数不大于 k。

$n \le 4 \times 10^5, 1 \le m \le 1.2 \times 10^5, 0 \le k_i \le 15.$

观察性质

观察性质

- 256 位数，至多 15 个不同。
-

观察性质

- 256 位数，至多 15 个不同。
- 那么如果我把他分成 16×16 的话，至少有一个块是相同的。

做法

- 数据随机的情况下，一块完全相同的概率是 $\frac{1}{2^{16}}$，这已经是一个极大的剪枝了。
- 这意味着，期望只有 $\frac{n}{2^{16}}$ 个需要检测。
-
-
-

做法

- 数据随机的情况下，一块完全相同的概率是 $\frac{1}{2^{16}}$，这已经是一个极大的剪枝了。
- 这意味着，期望只有 $\frac{n}{2^{16}}$ 个需要检测。
- 如何检测两个有多少位不同呢。
- 考虑使用 bitset，$(x \oplus y).count()$ 就是有多少位不同。
- 综上，复杂度是 $O(q\frac{n}{2^{16}}\frac{256}{w})$，可以通过本题。

[ABC259Ex] Yet Another Path Counting

- 有 N 行 N 列的网格图，只能向下或向右走，合法路径的开端和结尾的格子上数字一样
- 找到合法路径条数，对 998244353 取模

$1 \leq N \leq 400, 1 \leq a_{i,j} \leq 400$

做法！

- 对于单种颜色 c 来说，如果 cnt_c 较小，那么此时有一种做法是直接暴力计算，$\sum cnt_c^2 \le Bn^2$，复杂度是 $O(Bn^2)$ 的。
- 对于单种颜色 c 来说，如果 cnt_c 较大，那么有一种做法是直接考虑颜色 (i,j) 的位置加 1，然后直接做类似组合数的事情，这样复杂度是直接 $O\left(\frac{n^2}{B} \times n^2\right)$ 的。
- 综上，我们只需要把 B 取到 n，就可以做到很优秀的复杂度了。

如果你不清楚什么叫做类似组合数的事情。

https://www.luogu.com.cn/problem/AT_agc001_e

这题就是要求 $\sum_{i=1}^{n}\sum_{j=i+1}^{n}\binom{a_i+a_j+b_i+b_j}{a_i+a_j}$

- $n \leq 200000, 1 \leq a_i, b_i \leq 2000$
-
-
-

如果你不清楚什么叫做类似组合数的事情。

https://www.luogu.com.cn/problem/AT_agc001_e

这题就是要求 $\sum_{i=1}^{n} \sum_{j=i+1}^{n} \binom{a_i+a_j+b_i+b_j}{a_i+a_j}$

- $n \leq 200000, 1 \leq a_i, b_i \leq 2000$
- $\binom{a_i+a_j+b_i+b_j}{a_i+a_j}$的意义等同于从 $(-a_j, -b_j)$ 走到 (a_i, b_i)的方案数

所以只需要在起点处全部加上 1 的贡献，然后查看终点处的值即可。

然后这题还多出来一步，是

- $\sum_{i=1}^{n} \sum_{j=i+1}^{n} \binom{a_i+a_j+b_i+b_j}{a_i+a_j}$
- $= \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} \binom{a_i+a_j+b_i+b_j}{a_i+a_j} - \sum_{i=1}^{n} \binom{a_i+a_i}{b_i+b_i}}{2}$

本课件未提到的题目，但是可以去做的

https://www.luogu.com.cn/problem/CF1207F

https://www.luogu.com.cn/problem/CF914F

https://www.luogu.com.cn/problem/CF840E

谢谢大家！