分块思想与数据结构

浙江省余姚中学 李昊

分块思想有何应用？

一. 直接分块
二. 平衡两种针对性不同的算法
三. 定期重构
四. 莫队算法

直接分块

- 信息学竞赛中经常会出现类似于：
- 操作 L R:对第L至第R个数进行...
- 询问 L R:询问第L至第R个数...
- 往往它是需要复杂数据结构来维护，或者难以实现两个区间信息的合并，以致数据结构做不了的

直接分块

- 让我们来看看分块是怎么搞的
- 将连续x个元素分为一个块
- 对于每一个操作[L,R]，一定是中间包含了若干完整块，两边剩余不超过2x个位置，即“[L,K*X],[K*X+1,(K+1)*X]…[K2*Z+1,R]”这种形式
- 对于完整块可以统一处理，如果可以做到O(1),时间复杂度O(n/x)
- 对于剩余的2x个点，可以暴力处理O(x)
- 当x＝sqrt(n)时，每次操作的时间复杂度降到最低O(sqrt(n))
- 若上面两种情况的任意一种需要多一个log的复杂度，那么可以通过调整块的大小来使得每次操作的复杂度变为O(sqrt(nlogn))

看一道简单的例题

- n个数，q个操作
- 操作1:将一段区间内每个数增加x
- 操作2:询问一段区间内数的总和
- 数据范围：n, q<=100000

直接分块的做法

- 按照前面所说的，将序列每x个分一块
- 修改操作：
  - 对于完整的块，每个块有一个标记t，表示该块内每个元素的值都需要加上t，相当于线段树上的lazy标记
  - 对于多余的2x个位置，直接暴力修改值
- 询问操作：
  - 对于完整的块，返回块内元素总和＋t*块大小
  - 对于多余的2x个位置，直接暴力累加
- 由于两种情况都可以sqrt(n)完成，所以x设为sqrt(n)，总时间复杂度O(qsqrt(n))

平衡两种针对性不同的算法

- 信息学竞赛有这样一类题目，当题目中的参数比较小时，可以通过某种姿势暴力，当参数比较大时，又有另一种暴力，这时我们可以通过设置一个阀值，然后分两种情况做
- 这一类题目比较灵活多变，我们直接来看一道例题

TC SRM589 Flipping Bits

- 给一个长度为n的01字符串，再给一个正整数M。
- 每次操作可以将一个位置取反，或者将一个长度为M的倍数的前缀取反。问最少多少次操作，才能使字符串成为一个循环节为M的循环串。

- 数据范围：n<=300

Solution

- 每m个划分为一段
- 有两种操作，单点翻转和整段翻转，如何平衡这两种操作?

Solution

- 当m比较小时可以直接枚举出答案，那么当单点修改完后，每一段为最后答案或者每一位都与最后答案相反，剩下的只是整段翻转的事了
- 但并不知道每段变成哪种情况会使得答案最优
- 所以从后往前对每一段dp一下，记录目前翻转次数的奇偶性
- 那么对于当前段只需数出与答案不相同的个数即可
- 时间复杂度O(2^m*n/m)

Solution

- 当m比较大的时候，发现段数并不怎么多，枚举每一段是否需要翻转
- 对于i，i+m，i+m*2......这些位置，发现它们可以划为一类，因为他们对其他位置没有影响，其他位置对他们也没有影响，只需要保证每一类里最后都是一样的就行，所以对于每一类数一下0多还是1多，把少的修改掉即可
- 时间复杂度O(n*2^sqrt(n/m))

定期重构

- 对于有修改有询问的题，有一种叫“定期重构”的做法
- 每修改sqrt(m)次，就把需要的信息统一处理一次，那么每次询问时，最多就sqrt(m)次修改对本次询问的影响还没处理，暴力处理即可
- 还是直接看题感受一下吧

定期重构

- n个数，q个操作
- 操作1:将一段区间内每个数增加x
- 操作2:询问一段区间内数的总和
- 数据范围：n, q<=100000

定期重构

- 每sqrt(q)次修改操作，我们就把a数组重新算一遍
- 每次询问考虑剩余的sqrt(q)个修改操作对当前询问的影响
- 总时间复杂度O(nsqrt(q))

莫队算法

- 对于无修改，且如果知道区间［l,r］的答案就可以通过O(1)或者O(logn)的时间算出算出[l,r+1]、[l,r-1]、[l+1,r]、[l-1,r]的答案的题目，可以套用这一通用解法
- 先将序列分成sqrt(n)分块，然后将所有询问做双关键字排序，第一关键字为询问的左端点所在的块，第二关键字为询问的右端点
- 那么两个询问[l1,r1],[l1,r2]之间转移的时间为(|l1-l2|+|r1-r2|)*O(移动一步的复杂度)，考虑总的端点挪动次数

莫队算法

- 先将所有询问按照第一关键字分成sqrt(n)类
- 对于左端点，在同一类内的转移，一次不超过sqrt(n)，在不同类之间转移不超过sqrt(n)次
- 对于右端点，在同一类内，右端点是单调不降的，所以同一类内最多转移n次，在不同类之间转移不超过sqrt(n)次
- 那么总转移次数就是nsqrt(n)级别的了

小z的袜子

- 有n个数，m个询问，每次询问从第l个到第r个数中拿出两个数，这两个数相等的概率
- 数据范围：n, m<=50000

莫队算法

- 其实就是要统计区间内相等的对数
- 按照前面所说的将询问先排个序，考虑如何转移
- 开一个数组s，表示目前区间内有s[i]个i
- 若新增一个数x进区间，答案增加s[x]，s[x]+1
- 删除一个数x，s[x]-1，答案减少s[x]
- 时间复杂度O(nsqrt(n))

CodeChef NOV 14 Chef and Churu

- 有一个长度为n的数组A，有n个函数，第i个函数的值为

$$\sum_{j=l[i]}^{r[i]} a[j]$$

- 有两种操作：
- 修改A[i]
- 询问第l~r个函数值的和。

- 数据范围:n<=100000

Solution

- 考虑询问操作的l＝r该怎么做
- 直接对序列分块，维护块的总和即可，询问像最前面那道例题一样做即可，O(sqrt(n))
- 询问其实还有更快的做法
- 修改操作，维护
  - 前i个块的总和
  - 每个数到它所在块左端点所有数的总和
- 询问操作
  - 可以O(1)查询前i个数的总和
  - 前r个数－前l-1个数就是答案了

Solution

- 既然要查询的函数，那么我们把函数也分块
- 维护每个函数块的答案
- 每次询问时统计完整的被包含的块的和
- 统计剩余未被考虑的2*sqrt(n)个函数的值
- 对于每个函数可以通过前i块和、块内前缀和O(1)查询其值。
- 对于每次修改，需要统计出对每个函数块的影响
- 这个可以通过预处理每个函数块中包含点i个函数有几个函数轻松解决
- 时间复杂度O(nsqrt(n))

51nod 双马尾机器人

- 从1~n中选出尽可能少的数，使得剩下的每个数至少和一个被选择的数不互质，k不能被选择

- 数据范围：n, k<=1000

Solution

- 题目相当于被选择的数分解质因数后包含所有质因数
- 每个数分解质因数之后最多只含有一个大于sqrt(n)的质因子
- 以这个质因子为依据将所有数进行分类
- 同时记录质因子2、3、5、7、11、13、17、19、23、29、31有没有出现过
- dp[i][j]表示前i类数j中的质因子已经出现过最少需要选择几个数(j为11个质因子有没有出现过的二进制状态)，那么只需保证每一类中至少选择了一个数方案就是合法的

CodeChef OCT 14 Children Trips

- 有一棵边权为1或2的n个点的树，每次询问从u走到v每天最多走k步，且每天结束必须在节点上，最少要几天。

- 数据范围:n、询问次数<=100000

Solution

- 对k分情况进行讨论
- k>=sqrt(n)
- 最多走sqrt(n)步，每一步二分能走到哪即可
- k<=sqrt(n)
- 对于每个k一起处理，倍增，预处理2^i步能走到哪里
- 时间复杂度O(nsqrt(n)logn)

Solution

- 怎么把log去掉？
- （1）k<=sqrt(n)
  - 还是k相同的询问一起处理
  - 找出每个点向上走长度k走到哪
  - 根据这个新构造一棵树
  - dfs整棵新树，记录根到目前节点的路径
  - 访问到某个点时处理这个点的所有询问
  - 直接在记录的路径上二分即可
  - 时间复杂度O(nsqrt(n)+Qlogn)

Solution

- （2）k>sqrt(n)
  - 每个点u记录向上走长度i能走到哪(i<=sqrt(n))，记为f[u][i]
  - 每次走的时候如果剩余长度>sqrt(n)就用f[u][sqrt(n)]更新，剩余长度减去实际走了多远
  - 否则走到f[u][剩余长度]
  - 因为每步要么走sqrt(n)远，要么走完一次，所以时间复杂度O(sqrt(n))

CodeChef MAY 15 Chef and Balanced Strings

- 一个字符串如果他的字符都出现了偶数次则称为平衡串。询问一个字符串l~r内所有平衡子串长度的k次方和。

- 数据范围:字符串长度n,询问次数m<=100000,1<=k<=2

Solution

- 如何快速判断一个子串是否是平衡串？
- 把前缀和根据每个字母出现次数的奇偶进行hash，s[r]=s[l-1]即为平衡串
- 两个相邻的平衡串合并之后仍是平衡串。
- 对于每个i，找到最小的a[i](a[i]>i)，使得[i,a[i]]为平衡子串。则[i,a[i]]、[i,a[a[i]+1]]、[i,a[a[a[i]+1]+1]]…都是平衡串。
- 将这些位置划为一类

Solution

- 分块预处理第i个块到第j个块的答案。每次询问的时候只需考虑添加2*sqrt(n)个字符后答案的变化。
- 由于只有每一类内的两个位置配对才能组成平衡串，所以考虑每一类内如何维护
- sigma{(wi-wj)^2}=sigma{wi^2}*(个数－1)-sigma{wi}^2+sigma{wi^2}
- 具体的维护方法可以把式子展开，然后维护wi个数，wi总和，wi平方和
- O(nsqrt(n))
- 或者套用莫队算法，维护方法类似

Codeforces Round #221 (Div. 1) Tree and Queries

- 给出一个n个点的树，每个点都有颜色，Q次询问，每次问vi的子树中至少出现ki次的颜色有几种

- 数据范围：n,Q<=10^5

Solution

- 相当于在序列上询问区间中出现次数超过ki的数有几种
- 莫队算法模板题

- 强制在线？

Solution

- 若ki>sqrt(n)
- 可能的颜色最多sqrt(n)种，对每一个颜色判断一下即可
- 如何做到O(1)回答某种颜色在某一区间内的出现次数？
- sum1[i][j]表示在前i个块j出现了几次
- sum2[i]表示在第i个元素所属的块中在第i个元素之前且与第i个元素颜色相同的位置有多少个
- 若ki<=sqrt(n)
- 对序列进行分块，记录从第i个块到第j个块出现超过k(k<=sqrt(n))次的颜色有几种，对于多余的2*sqrt(n)个位置的颜色判断一下即可

Codeforces Round #270 Design Tutorial: Increase the Constraints

- 给出两个01序列A和B，Q次询问，每次询问A的一个子串和B的一个子串异或之后的结果包含多少个1（两个子串长度相同）
- 数据范围：|A|,|B|<=2*10^5,Q<=4*10^5

Solution

- 先考虑简化版本：如果询问的是A和B的子串(保证长度相同)？

- FFT

- 那么原题再把A分块就行了
- 当块的大小为sqrt(nlogn)时时间复杂度最优
- 时间复杂度O(nsqrt(nlogn))

CodeChef JUNE 14 Sereja and Arcs

- 给定一个长度为n的序列A，对于每一对(i,j)(i<j)，如果满足Ai=Aj，那么就在坐标系中画上一个颜色为Ai的以(i,0)(j,0) 为直径的圆。问有多少对颜色不同的圆存在交点。

- 数据范围：1<=N,Ai<=100000

Solution

- 相交对数=总数-不相交对数。
- 不相交对数有两种类型:aabb和baab。
- 第一种比较好求，扫一遍就可以求出来。
- 现在来考虑第二种情况。

Solution

- 设一阀值X，然后分类讨论：
- (1)如果相交的两种颜色中没有出现次数超过X的颜色：
- 枚举这种颜色所有点对，统计每一点对包含多少点对
- 树状数组维护，此类点对最多nX对
- 时间复杂度O（nXlogn）

Solution

- 如果相交的两种颜色中有出现次数超过X的颜色:(假设这种颜色为a,出现z次)
- 对任意同一颜色的两个位置，设前者之前有x次a出现，后者之前有y次a出现，那么这个圆的贡献为(y-x)*(y-x-1)/2+x*(z-y)
- 将此式分解然后线性扫一遍求和即可。
- 时间复杂度O（n^2/X）

- 结合上一种情况，当X=sqrt(nlogn)时，总时间复杂度取到最优值O(n*sqrt(nlogn))

XJOI 神奇的矩阵

- 有一个神奇的矩形。它的第一行每一个元素a(1,i)都是给定的。对于每一个元素a(x,y)(x>1)，它的值都是a(x-1,y)在a(x-1,1)…a(x-1,y)中出现过的次数。但由于这个矩阵很大，人们并不开心这么慢吞吞地计算整个矩阵的值，因此他们找到了你，并要求你快速知道某一个位置的值。有时这个矩阵的第一行还会被修改，你当然也要考虑这些修改的因素。

- 数据范围：n,m,k<=10^5

提示:

- 除了第一行，每两行为一循环周期

Solution

- $x=2$?

- $x=3$?

解法一

- a[2][i]表示a[1][i]在前i个数中出现了几次
- a[3][i]表示有多少个数在前i个数中出现次数大于等于a[1][i]
- a[2]很好维护
- 考虑如何维护a[3]

解法一

- sum[i][j]表示第i块中a[2][x]＝j的x有几个
- 只维护j<=sqrt(n)的
- 修改时，在每个块中其实最多只会影响2个sum值，所以直接维护即可
- 询问时，先求出a[2][i]
  - 若a[2][i]>sqrt(n)，只考虑序列中出现超过sqrt(n)次的数，每个数O(1)验证
  - 否则，先求出完整块中出现a[2][i]的出现次数，再验证序列中出现超过sqrt(n)次的数，和i所在块中剩余的排在i前面的数
- 时间复杂度O(nsqrt(n))

解法二

- 对操作分块
- 每sqrt(n)操作就暴力重构一次，询问时只需考虑sqrt(n)次操作对当前询问影响即可

Manthan, Codefest 16 Fibonacci-ish II

- 给你一个长度为n的数组，Q次询问，每次将第l个到第r个数取出排序去重，求最小的数*fib1+次小的数*fib2+第三小的数*fib3……的总和。

- 数据范围：n,Q<=30000

Solution

- 最外面是一个莫队算法
- 如何维护转移？
- 若加入或删除后数字种类不变则不需要管
- 否则，用权值线段树维护
- 线段树的每个叶子上维护这个数*fib[i],这个数*fib[i+1]
- 当插入一个数的时候，给排在他后面的数统一乘上一个转移矩阵
- 0 1
- 1 1
- 即可

Educational Codeforces Round 6 Xors on Segments

- 给你一个长度为n的数组A，m次询问，每次询问给出(l, r)
- 定义f(u, v)=u xor （u+1） xor……xor v
- 对于每一组询问输出f(A[x], A[y])的最大值(l<=x, y<=r, A[x]<=A[y])

- 数据范围1<=n<=5*10^4, 1<=m<=5*10^3, A[i]<=10^6

Solution

- b[i]=1 xor 2 xor 3…xor i，题目相当于求b[a[x]-1] xor b[a[y]]
- 对数组分块
- 预处理块到块的答案
- 询问时直接用完整块的答案，多余的2sqrt(n)个在可持久化trie上暴力找答案

2015北京网络赛 J Clarke and puzzle

- 给出n位同学的五门成绩，Q次询问，每次给出一位同学的成绩，询问有多少位同学每一门成绩都比该同学差。

- 数据范围：n, Q<=50000

Solution

- 答案={第一门科目成绩比该同学差的人}and……and{第五门科目成绩比该同学差的人}
- 怎么快速实现集合并？
- bitset
- 那么如何把{第一门科目成绩比该同学差的人}快速求出？
- 对成绩分块，预处理
- b[i][j]表示第i门科目成绩小于等于j*sqrt(n)的人的集合
- 那么再加上剩余未被分进完整块的sqrt(n)个人就是{第一门科目成绩比该同学差的人}了
- 所需时间复杂度O(sqrt(n))

Solution

- 最后再把这五个得到的bitset并起来即可
- 时间复杂度O(Qsqrt(n)*5)

集训队互测2015 Robot

- 初始有n个机器人在一条数轴上，给定初始位置(所有机器人都做匀速运动)
- m次操作，每次操作给出当前时间（时间递增），有两种类型
- 改变一个机器人的运动速度
- 询问当前离原点最远的机器人

- 数据范围：n，m<=10^5

Solution

- 超哥线段树维护线段

CodeChef APRIL 15 Black-white Board Game

- 有一个n*n的矩形，每行的黑色格子都是连续的一段区间(l[i],r[i])。两人玩游戏，第一个人选择一个逆序对数为偶数的排列a，且(i,a[i])这个格子为黑色，
- 第二个人类似，但排列的逆序对数需为奇数。出现过的排列不能再出现。给出排列多的人胜。
- 数据范围:n<=100000

Solution

- 相当于求该矩形的行列式
- 先把所有行排序，设l[i]为第i行黑色格子左端点，r[i]为第i行黑色格子右端点
- 以l[i]为第一关键字,r[i]为第二关键字将行排序
- 然后我们来模拟高斯消元的过程
- 将所有l[i]相同的行分为一类
- 我们从小到大枚举每一类，对于某一类，我们取出r[i]最小的一行，然后用这一行去消同一类中其余所有的行
- 那么这一类中的其它行的黑色格子左端点都变成了r[i]+1，将他们都划到第r[i]+1类即可
- 也就是说我们需要一个可以取最小值和可合并的数据结构
- 左偏树是一个不错的选择，时间复杂度O(nlogn)

XJOI 不可视境界线

- 有n个敌人，第i个敌人战斗力为Ai，打败该第i个敌人至少需要Bi战斗力，与第i个敌人需要受到sqrt(a^2+(Ai-C)^2)-a点伤害(a为之前所受伤害，C为当前战斗力)，战斗后战斗力会变成Ai，初始C=0
- 与第i个敌人战斗后就不能再和1~i号敌人战斗了，问战胜n号敌人后至少受多少伤害。

- 数据范围：n<=10^5

Solution

- 第i个点能从第j个点转移(j<i && A[j]>=B[i])
- 战斗后受到的总伤害为sqrt(f[j]^2+(A[j]-A[i])^2)
- 欧几里得距离!
- 最近点对!
- KD tree!

Codeforce k-Maximum Subsequence Sum

- 给一列数，要求支持操作： 1.修改某个数的值 2.读入l,r,k，询问在[l,r]内选不相交的不超过k个子段，最大的和是多少。

- 数据范围：n，m<=10^5，k<=20

Solution

- K=1？
- 经典线段树
- K=20
- 维护20个东西 K^2合并
- 每次复杂度O(K^2logn)

Solution

- 先来看一下网络流该怎么做
- 源点向每个点连边，容量为1费用为0，每个点拆点，i向i’连边容量为1，费用为ai，然后i’向i+1连边，再限制流量不超过k
- 这个网络流显然是对的
- 在增广的过程中，要么找一个目前最大子段和，要么走反向边
- 用线段树来模拟这一过程：
  - 找出目前区间最大子段和，累加进答案，将这一子段*-1
  - 重复上一步骤k次
- 时间复杂度O(nlognk)

Codeforces Round #189 (Div. 1) Ping-Pong

- 有一些区间，能从（a, b）走到（c, d）的条件是$c<a<d$或者$c<b<d$，有以下两种操作：
- 新加一个区间（后加入的区间保证比新加入的区间长）
- 询问能否从第a个加入的区间走到第b个加入的区间
- 初始区间个数为0
- 数据范围：操作数$<=10^5$

Solution

- 如果后加入的区间能走到先加入的区间，则这两个区间能互相走到
- 若存在这样的区间就把他们合并成一个区间
- 为什么是对的？
- 若之后有一条线段能和合并过的线段合并，合并线段中必然包含一条能和该线段合并的线段。
- 这样就把互相能走到的线段都合并起来了，剩下的就是只能单向走的，判断下就可以。