数据结构

北京理工大学 林恺

2021 年 7 月 11 日

目录

单调队列与单调栈

队列

- 普通队列
- 双端队列
- 单调队列

定义

队列内元素具有单调性的队列称为单调队列

栈

- 栈
- 单调栈

定义

栈内元素具有单调性的栈称为单调栈

模板题

n 位依次进入一个初始为空的队伍中，有以下两种入队方式：

- 站到队尾，如果前面的人比自己高，则将前面的同学移出队伍，直到前面的同学比自己矮或前面没有同学了
- 站到队头，如果后面的人比自己矮，则将后面的同学移出队伍，直到后面的同学比自己高或后面没有同学了

$n \leq 10^5$

小结

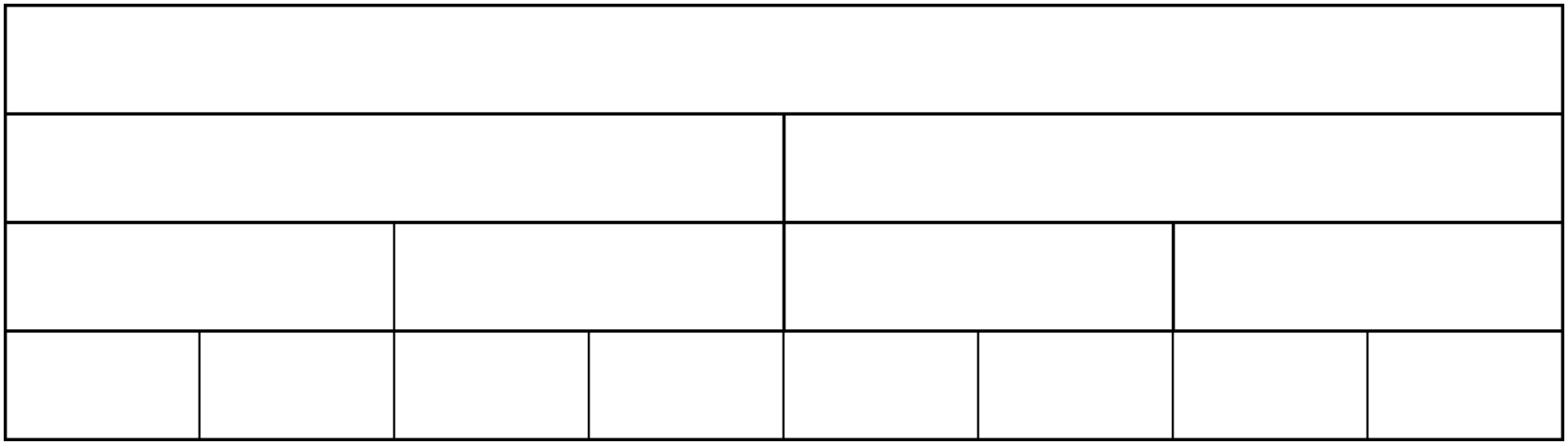
很简单对吧?

单调队列与单调栈其实是一种思想，它在 DP 的优化中起着重要的作用。

倍增

什么是倍增

狭义的说，如果一个数据结构只记录 2^i 等级的信息，就称其为一个倍增数据结构



题目

没有题目

小结

倍增同样作为一种思想，在各种数据结构中发挥着重要的作用。比如接下来要讲的树状数组、线段树、RMQ。

线段树

问题的引入

先来看下面这个问题

给定一个长度为 n 的数组 a，初始全为 0，你需要完成以下两个操作共 q 次：

- 单点加
- 区间求和

$n, q \leq 10^5$

线段树

暴力求和？复杂度 $O(nq)$，难以让人接受

考虑一下刚刚所讲的倍增

我们对其稍加修改，得到这样一个数据结构

<table>
<tr><td colspan="8">1~8</td></tr>
<tr><td colspan="4">1~4</td><td colspan="4">5~8</td></tr>
<tr><td colspan="2">1~2</td><td colspan="2">3~4</td><td colspan="2">5~6</td><td colspan="2">7~8</td></tr>
<tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td></tr>
</table>

线段树

这样，在对任何一个区间求和时，都只需要用到长度为 2^i 的区间各至多两个

查询复杂度降到了 $O(\log n)$

同时，在单点加的过程中，需要修改长度为 2^i 的区间各一个

修改复杂度上升到了 $O(\log n)$

通过提升简单操作的复杂度，降低困难操作的复杂度，这种思想在数据结构中的应用十分广泛。

线段树

但这，仅仅只是线段树的最基础部分

现如今，各大毒瘤出题人已经魔改出了各种各样支持各种复杂操作的线段树来（区间加、区间乘、区间平方、区间取 max、区间取 min、……）

技能：lazytag

再来看下面一个问题

给定一个长度为 n 的数组 a，初始全为 0，你需要完成以下两个操作共 q 次：

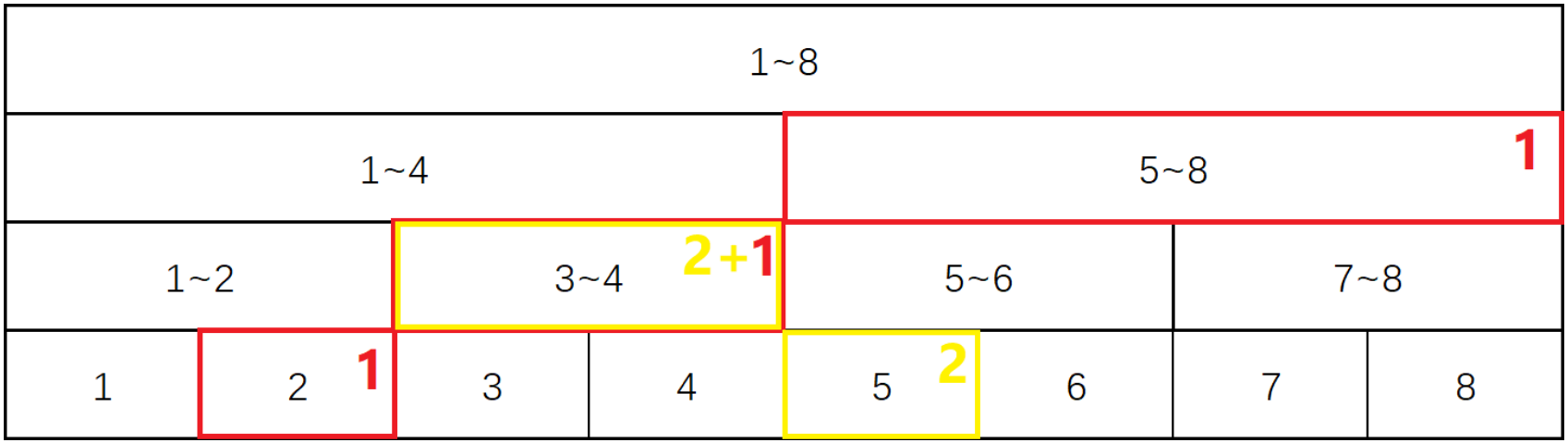
- **区间**加
- 区间求和

$n, q \leq 10^5$

技能：lazytag

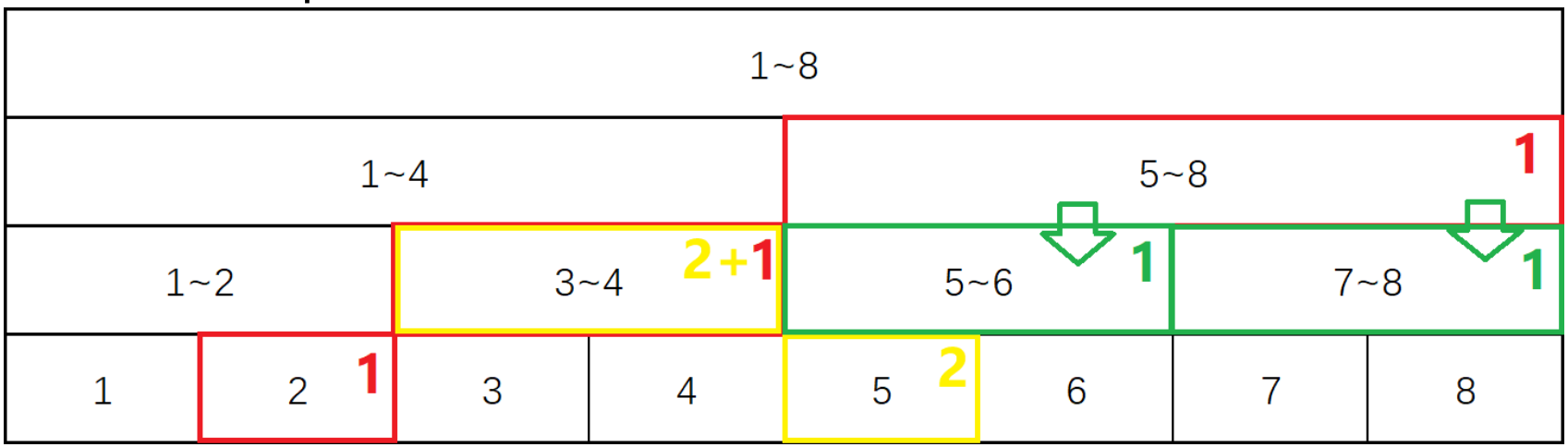
如何处理区间加?

考虑在线段树上的每个节点额外记录一个值，称之为lazytag，其值的含义为当前节点所代表的区间被**完整的加**的值。



技能：lazytag

上面这种标记方式被称为标记永久化，在实际应用中，我们还需要一个 pushdown 操作以方便我们写代码。



技能：lazytag

lazytag 的应用空间十分广阔，你可以通过自己定义各式各样的 lazytag 实现各种复杂的功能

更加复杂的线段树，例如李超线段树、吉老师线段树等都是基于各种 lazytag 实现了某些看似不可能实现的功能

下面的题目都需要你开发各种各样的 lazytag

区间乘法

给定一个长度为 n 的数组 a，初始全为 0，你需要完成以下三种操作共 q 次：

- 区间加
- 区间乘
- 区间求和

$n, q \leq 10^5$

区间乘法

设计 sum,mult 两个 tag，分别表示区间加的量与区间乘的量即可

注意在区间乘的过程中，你需要将 sum 这个 lazytag 也乘上对应的值

区间平方

给定一个长度为 n 的数组 a，初始全为 0，你需要完成以下三种操作共 q 次：

- 区间加
- 区间求和
- 区间求平方和（每个数的平方的和）

$n, q \leq 10^5$

区间平方

假设某个位置原有的值为 a，被加上了 b，那么新的平方值，即

$$(a+b)^2 = a^2 + 2ab + b^2$$

对一个区间加 b 时，区间平方和的增量为

$$(r-l+1) * b^2 + 2b * \sum_{i=l}^{r} a_i$$

树状数组

问题的引入

先来看下面这个问题

给定一个长度为 n 的数组 a，初始全为 0，你需要完成以下两个操作共 q 次：

- 单点修改
- 区间求和

$n, q \leq 5 * 10^6$

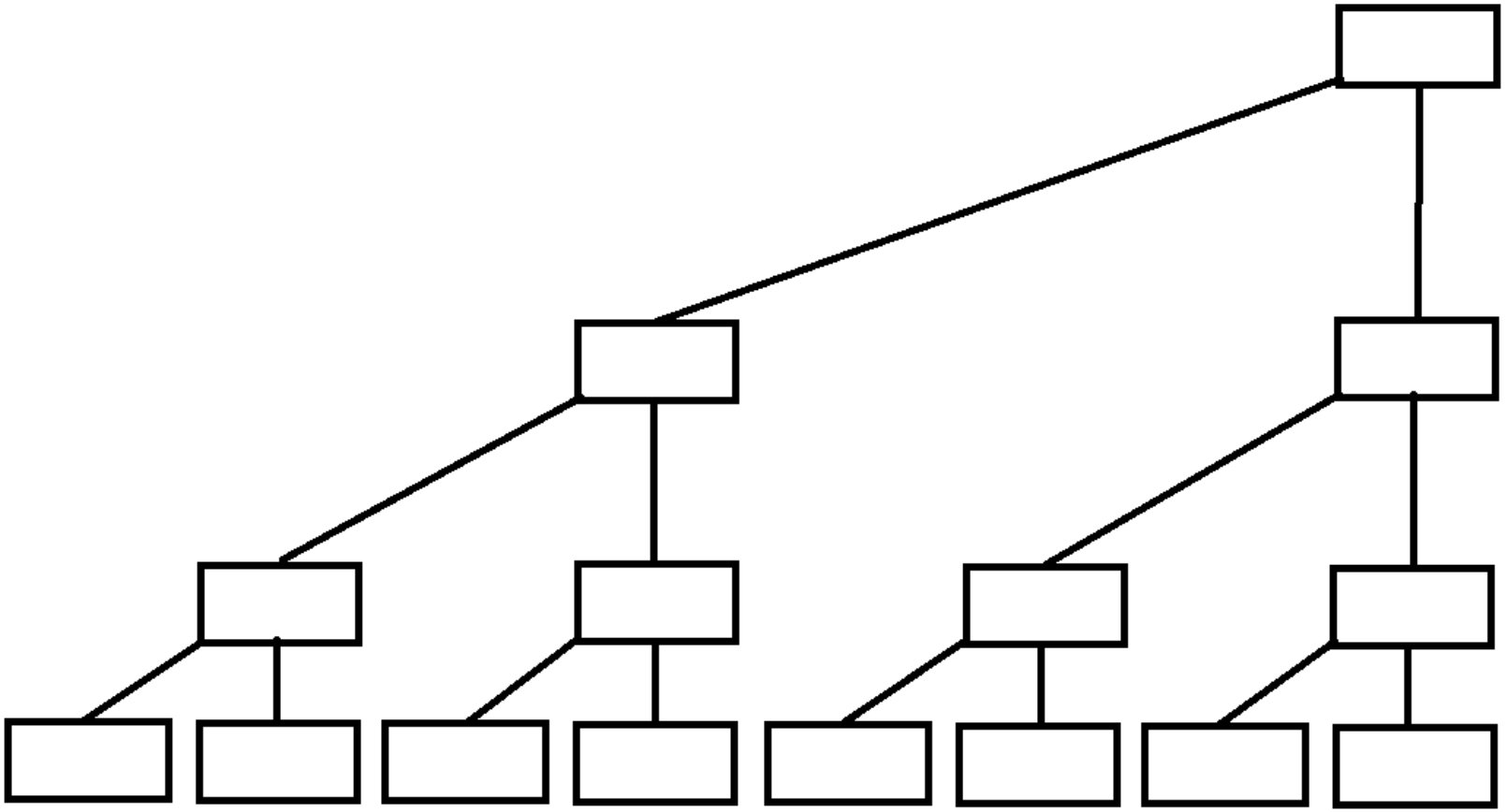
树状数组

数据范围大了不少

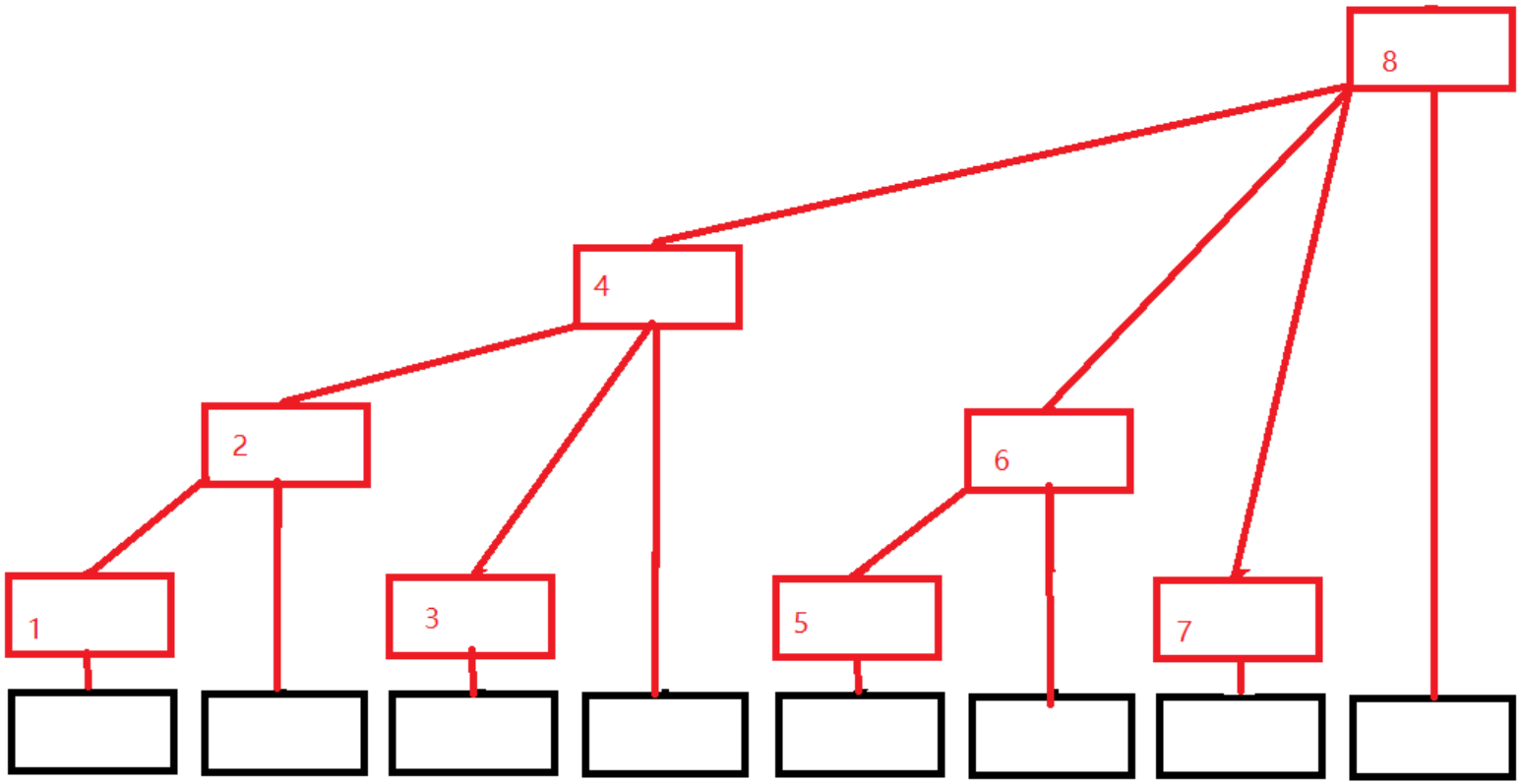
众所周知递归的常数很大，不太能跑

于是树状数组他来了

树状数组



树状数组



树状数组

对于数组中的某个位置 x，若其二进制末尾有 i 个 0，则其在第 i 级，代表了一个长度为 2^i 的区间的和

例如对于标号为 12 的区间，其二进制为 1100，则其代表了 $1001 \sim 1100$ 这段区间的和

在区间求和时，我们首先把问题转化为两个前缀和相减。而后对于前 x 个数的和，可以按 x 的二进制拆分，将其转化为 $\log x$ 个已知和的区间的和

例如前 11 个数的和，其二进制为 1011，则转化为 $0001 \sim 1000$（存储于节点 1000），$1001 \sim 1010$（存储于节点 1010），$1011 \sim 1011$（存储于节点 1011）这三个区间的和

树状数组

很抽象？

没关系，树状数组可以靠背！

```
int lowbit(int x){return x&(-x);}
void add(int x,int y){for(;x<=n;x+=lowbit(x))f[x]+=y;}
int sum(int x){int ans=0;for (;x;x-=lowbit(x))ans+=f[x];return ans;}
```

树状数组

事实上，线段树是可以完美代替树状数组的

树状数组的优势仅在于常数小，码量小

题目

没有题目

RMQ

问题的引入

先来看下面这个问题

给定一个长度为 n 的数组 a，你需要完成以下两个操作共 q 次：

- 区间求最大值
- 区间求最小值

$n \leq 10^5$

$q \leq 10^7$

RMQ

RMQ 是一种静态的数据结构，也就是说它不支持修改

既然不支持修改了，查询肯定要变快

RMQ 的查询是 $O(1)$ 的!

RMQ

RMQ 维护了所有长度为 2^i 的区间的信息，通常使用 ST 表

任何一个长度的区间总能被拆分成两个可能相交的长度为 2^i 的区间，因而把这两个区间的信息合并一下即可得到目标区间的信息

RMQ

例如我们要查询 $5 \sim 233$ 这个区间

首先计算出它的长度为 229，再得知覆盖长度为 229 的区间需要两个长度为 128 的区间，最后查询 $5 \sim 132$ 和 $106 \sim 233$ 这两个区间的最大值，即可得到 $5 \sim 233$ 这个区间的最大值

为什么复杂度是 $O(1)$ 的?

拓展内容：利用 RMQ 求 LCA

众所周知，LCA 的复杂度是 $O(\log n)$ 的

利用 RMQ ，可以将这个问题的复杂度降低到 $O(1)$

预备知识：dfs 序

定义

dfs 序即以 dfs 的方法遍历一棵树时，节点**首次**访问的顺序

预备知识：欧拉序

定义

欧拉序即以 dfs 的方法遍历一棵树时，节点访问的顺序

注意：在欧拉序中，一个节点可以出现多次

预备知识：RMQ 求区间最值所在的位置

在预处理 ST 表的过程中，同时再维护一个最值所在的下标即可

拓展内容：利用 RMQ 求 LCA

在欧拉序上，点 u 和点 v 中间深度最小的点就是他们的 LCA

因此对欧拉序建 RMQ，查询区间最小值即可

完结撒花

祝大家下午切题愉快！

谢谢大家！