OI数据结构之树状数组

GzsJAYOfficial　有为骚年

HI-OIER

August, 2025

关于我

我是2011年出生的究极无敌蒟蒻

天天被5033 ls 和jzc ls单调队列

被chennie ls + karma ls吊打

入坑OI一年半的新玩家

希望混E类省队的究极大菜鸡

联系方式QQ:3224063731(备注B站算法) 否则不认识QAQ

树状数组是什么？

树状数组是一种支持“单点修改，区间查询”/“区间修改，区间查询”

树状数组是什么？

树状数组是一种支持“单点修改，区间查询”/“区间修改，区间查询”

它是通过什么实现的？

树状数组是什么？

树状数组是一种支持“单点修改，区间查询”/“区间修改，区间查询”

它是通过什么实现的？

通过位运算(lowbit)存储位置，实现O(log n)的时间复杂度

树状数组是什么？

树状数组是一种支持“单点修改，区间查询”/“区间修改，区间查询”

它是通过什么实现的？

通过位运算(lowbit)存储位置，实现O(log n)的时间复杂度

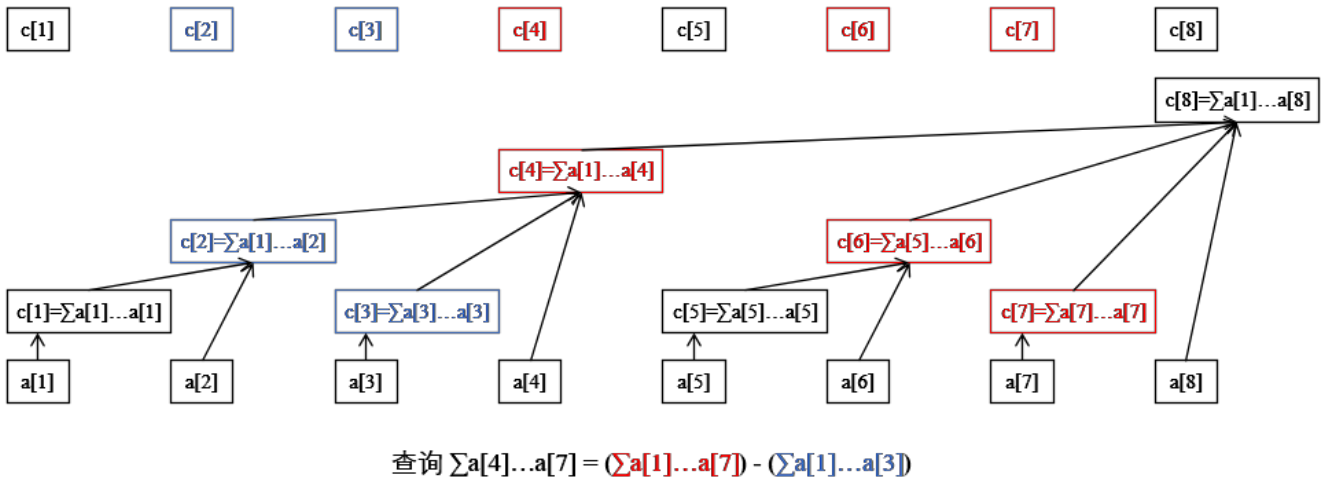

Figure:

树状数组是什么？

树状数组是一种支持“单点修改，区间查询”/“区间修改，区间查询”

它是通过什么实现的？

通过位运算(lowbit)存储位置，实现O(log n)的时间复杂度

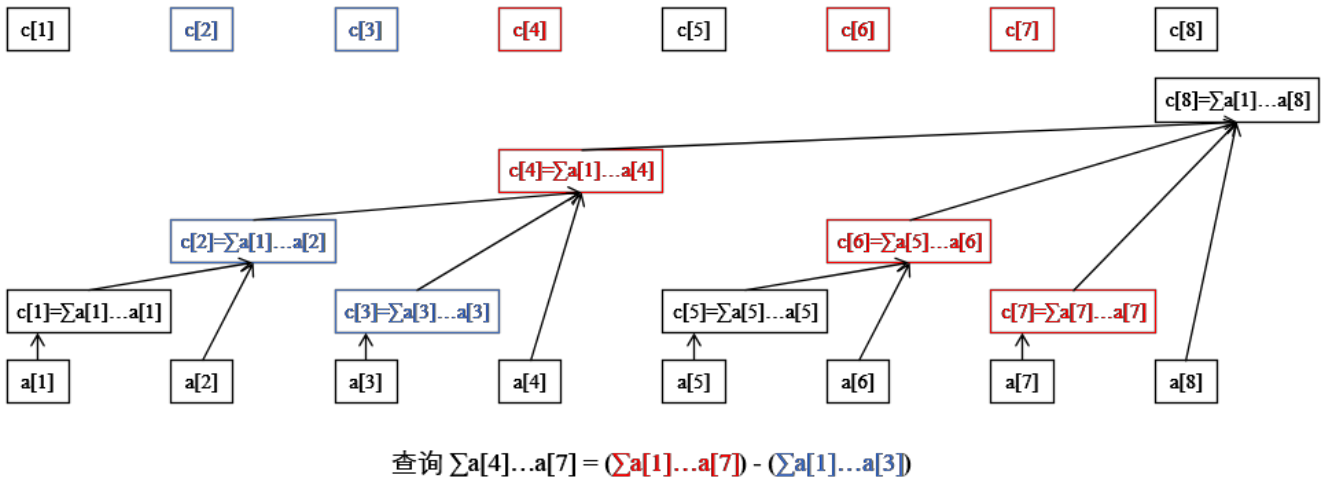

Figure:

树状数组的工作原理

树状数组实际上并不是一棵树

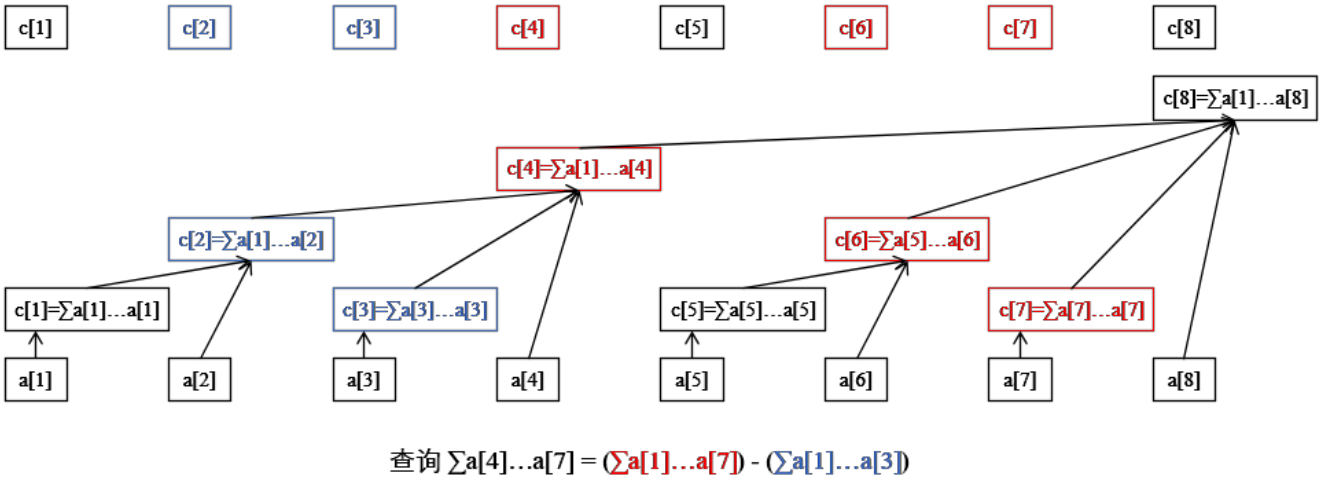

Figure:

树状数组的工作原理

树状数组实际上并不是一棵树

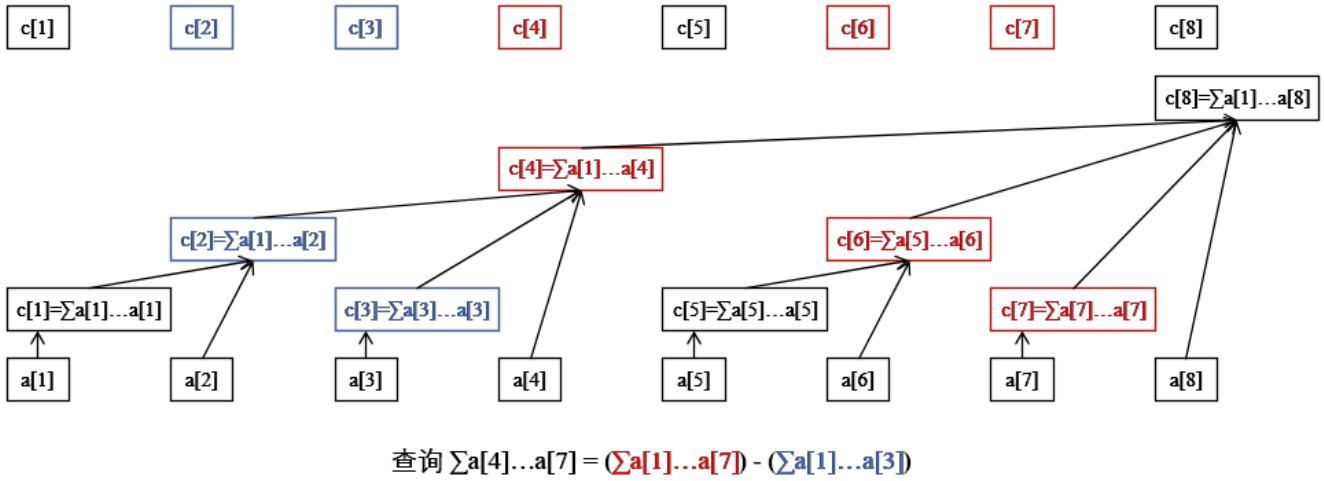

Figure:

从这张图片里我们可以观察到，C_i 为$a_?$ 到a_i 的和。

树状数组的工作原理

树状数组实际上并不是一棵树

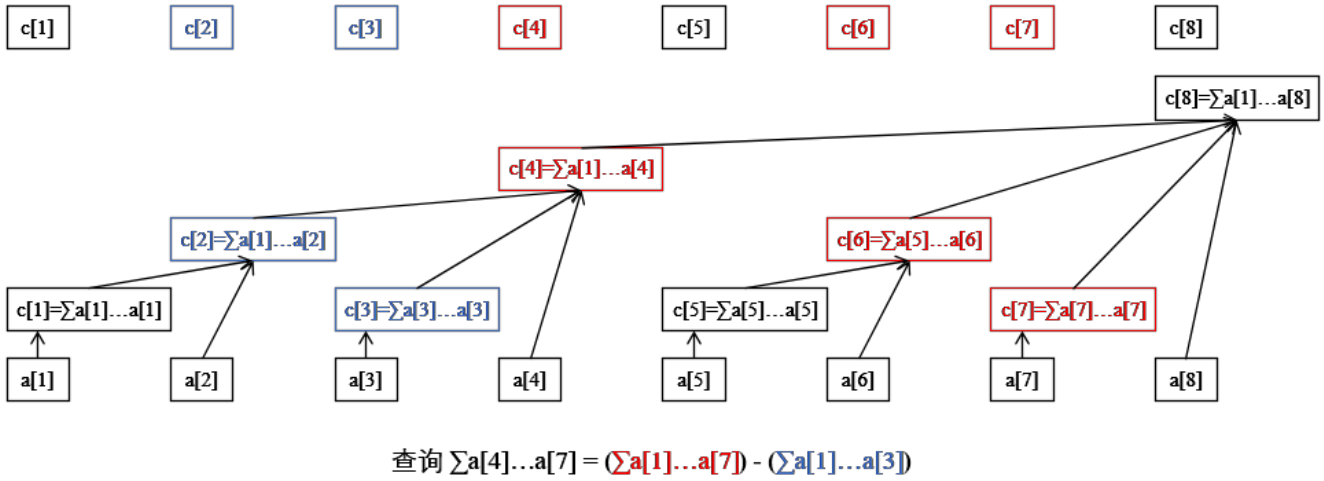

Figure:

从这张图片里我们可以观察到，C_i 为$a_?$ 到a_i 的和。

这时你可能要说了，UP，那我们是不是可以用区间和了？

树状数组的工作原理

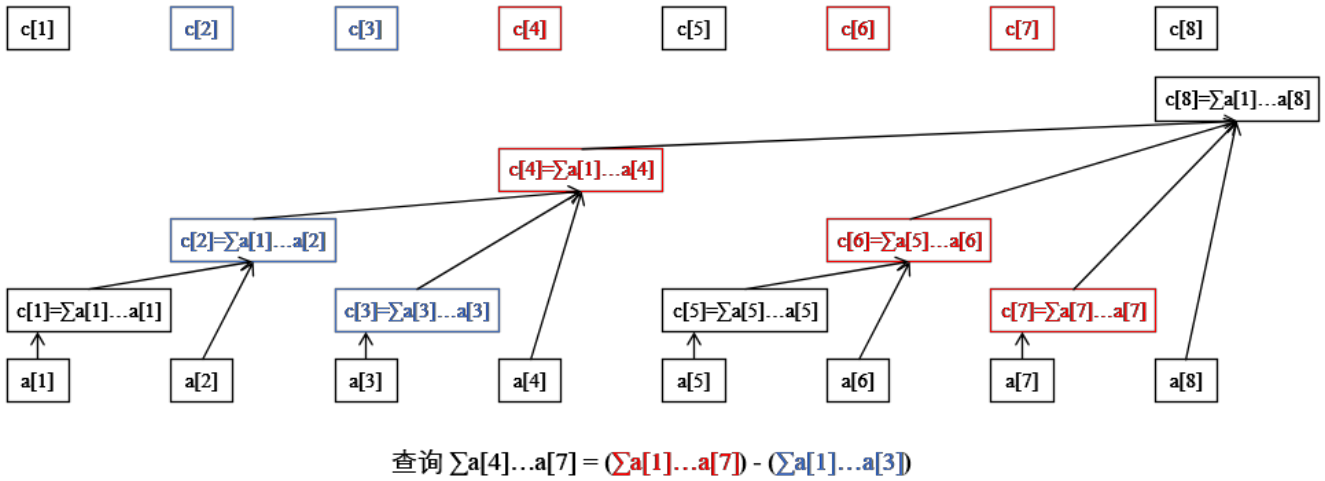

Figure:

没错！但是我们该如何维护数组C呢？

lowbit

这个时候就应该聊到开头的lowbit了，

lowbit

这个时候就应该聊到开头的lowbit了，

lowbit 的实质是将一个数的二进制全部取反即

```
int lowbit(int x) return x & -x;
```

lowbit

这个时候就应该聊到开头的lowbit了，

lowbit 的实质是将一个数的二进制全部取反即

```
int lowbit(int x) return x & -x;
```

这对我们的树状数组有什么用呢？

lowbit

这个时候就应该聊到开头的lowbit了，

lowbit 的实质是将一个数的二进制全部取反即

```
int lowbit(int x) return x & -x;
```

这对我们的树状数组有什么用呢？

很显然，可以用来求h_i 中i 的值

lowbit

这个时候就应该聊到开头的lowbit了，

lowbit 的实质是将一个数的二进制全部取反即

```
int lowbit(int x) return x & -x;
```

这对我们的树状数组有什么用呢？

很显然，可以用来求h_i 中i 的值

联系之前提到的图片，我们可以发现，C_i 管辖的区间为$a[x - lowbit(x) + 1, x]$

实现前缀和

首先实现前缀和

实现前缀和

首先实现前缀和

根据刚刚的结论，即C_i管辖的区间为$a[x - lowbit(x) + 1, x]$

可以得出实现求前缀和的代码

即

Untitled

```
int query_1_n(int n) {
  int ans = 0;
  while(n =/= 0) {
    ans += a[n];
    n -= lowbit(n);
  }
  return ans;
}
```

实现区间和

有了前缀和，区间和也很好求了。

实现区间和

有了前缀和，区间和也很好求了。

就是用A_1 到A_R 的前缀和减去A_1 到A_L 的前缀和（记得L区间-1）

实现区间和

有了前缀和，区间和也很好求了。

就是用A_1 到A_R 的前缀和减去A_1 到A_L 的前缀和（记得L区间-1）实现代码

BIT_Query

```
int query(int l,int r) {
  return query_1_n(r)-query_1_n(l-1);
}
```

Figure:

实现单点修改

既然我们的C数组是像下图一样实现的

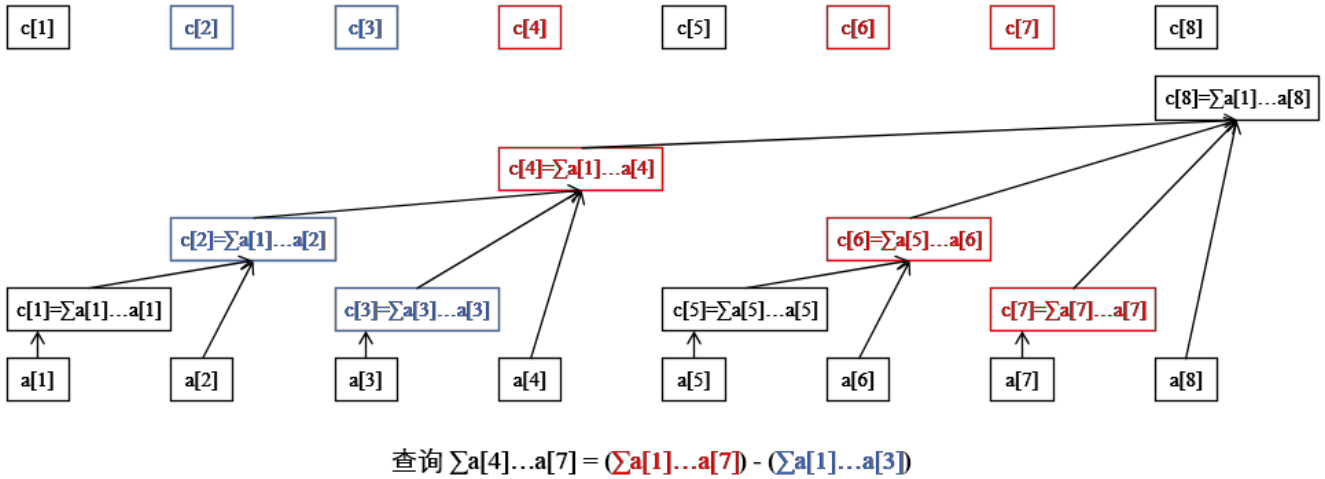

Figure:

那么就可以很自然的知道，修改一个值，要把那个值的父亲节点也给同步更新了

实现也很简单

实现单点修改

既然我们的C数组是像下图一样实现的

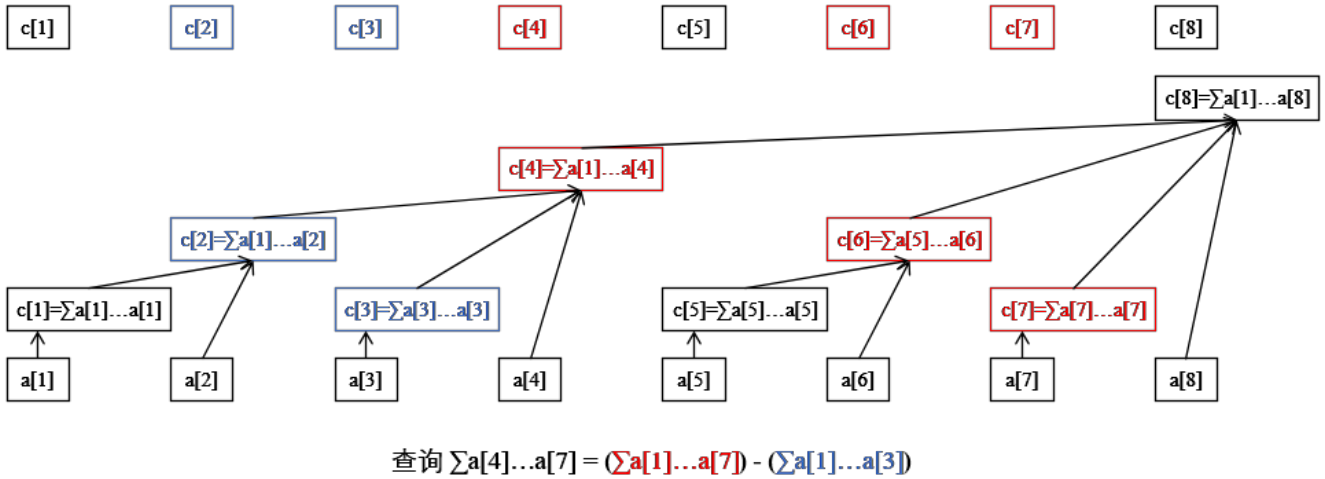

Figure:

那么就可以很自然的知道，修改一个值，要把那个值的父亲节点也给同步更新了

实现也很简单

实现单点修改

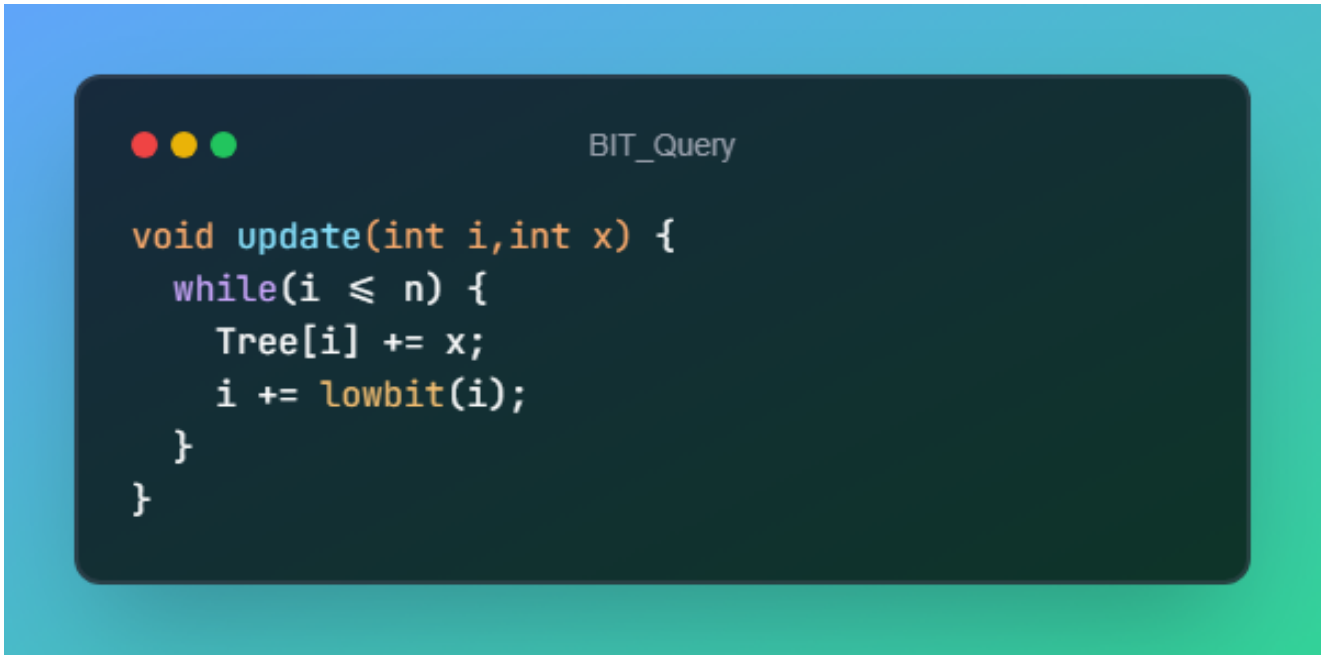

Figure:

最后

最后，我把代码模板和讲义放到了我的GitHUB仓库里，欢迎大家来查看（白嫖）

最后的最后，再给大家推荐几道树状数组的题目

https://www.luogu.com.cn/problem/P3374 板子

https://codeforces.com/contest/2130/problem/D 很经典的逆序对用法