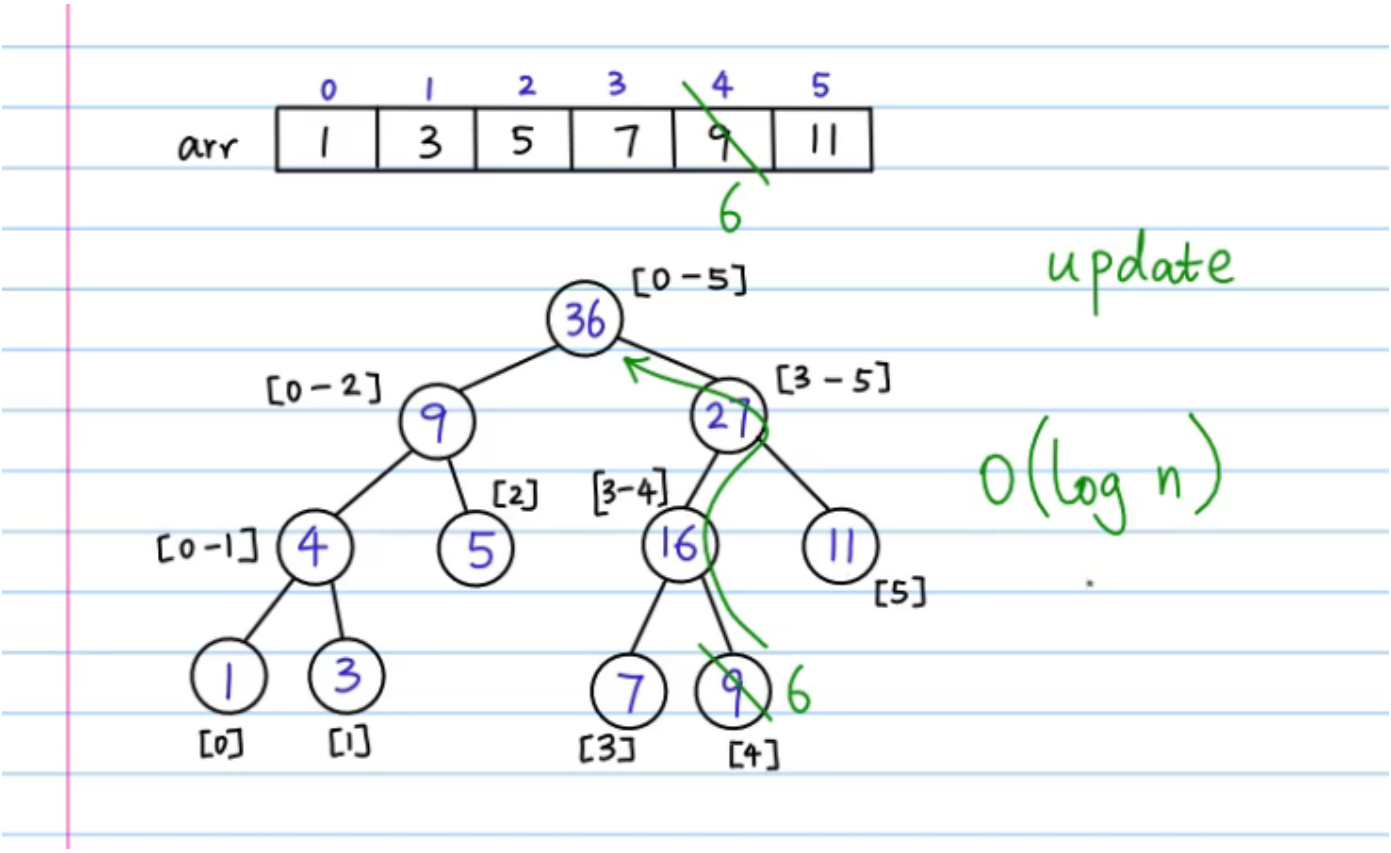
1.问题引入

假设有一个很大的数组，要频繁地进行求区间和(query)及更新操作(update)，最常用的方法是遍历，那么query操作的时间复杂度为O(n)，update时间复杂度为O(1)。

之前提到过前缀和的方法，可以将query操作降为O(1)，而update的时间复杂度为O(n)，因为需要更新前缀和数组。

以上两种办法的总体速度都不快。

2.基本概念



- 利用满二叉树进行存储

```
#include<iostream>
using namespace std;
const int MAX_LEN = 100;
// 构造区间和树
void build_tree(int arr[],int tree[],int node,int start,int end){
    if(start==end) {
        tree[node]=arr[start];
        return;
    }
    int mid = (start+end)/2;
    build_tree(arr, tree, node*2, start, mid);
    build_tree(arr, tree, node*2+1, mid+1, end);
    tree[node]=tree[node*2]+tree[node*2+1];
    return;
```

```
}
// 更新某个数据
void update_tree(int arr[],int tree[],int i,int start,int end){
    int mid=(start+end)/2;
    // 查找arr[i]
    int node = 1;
    while(tree[node*2]!=0){
        if(i<=mid) {
            node = node*2;
            end = mid;
        }else{
            node = node*2+1;
            start = mid + 1;
        }
        mid = (start+end)/2;
    }
    // 回溯
    tree[node] = arr[i];
    node = node/2;
    while(node>=1){
        tree[node] = tree[node*2]+tree[node*2+1];
        node = node/2;
    }
}
// 查找某个区间的和
int query_tree(int tree[],int node,int start,int end,int L,int R){
    if(start==L&&end==R) return tree[node];
    int mid = (start+end)/2;
    if(start<=L&&mid>=R){
        return query_tree(tree,node*2,start, mid, L, R);
    }else if(L>mid&&R<=end){
        return query_tree(tree,node*2+1,mid+1, end, L, R);
    }else{
        return query_tree(tree,node*2,start, mid,
L,mid)+query_tree(tree,node*2+1,mid+1, end, mid+1,R);
    }
}
int main(){
    int arr[] = {1, 3, 5, 7, 9, 11};
    int sizeArr = sizeof(arr)/sizeof(arr[0]);
    int tree[MAX_LEN] = {0};
    build_tree(arr, tree, 1, 0, sizeArr-1);
    for(int i=0;i<20;i++){
        cout<<tree[i]<<'\t';
    }
    cout << endl;
    // 更新值
//    arr[3] = 20;
//    update_tree(arr, tree, 3, 0, sizeArr-1);
```

```
//    for(int i=0;i<20;i++){
//        cout<<tree[i]<<'\t';
//    }
    cout << "区间arr[2..5]的和为" << query_tree(tree, 1, 0, sizeArr-1, 2, 5);
    cout<<endl;
    return 0;
}
```

参考资料

https://www.bilibili.com/video/BV1cb411t7AM?from=search&seid=6781647437020147354&spm_id_from=333.337.0.0