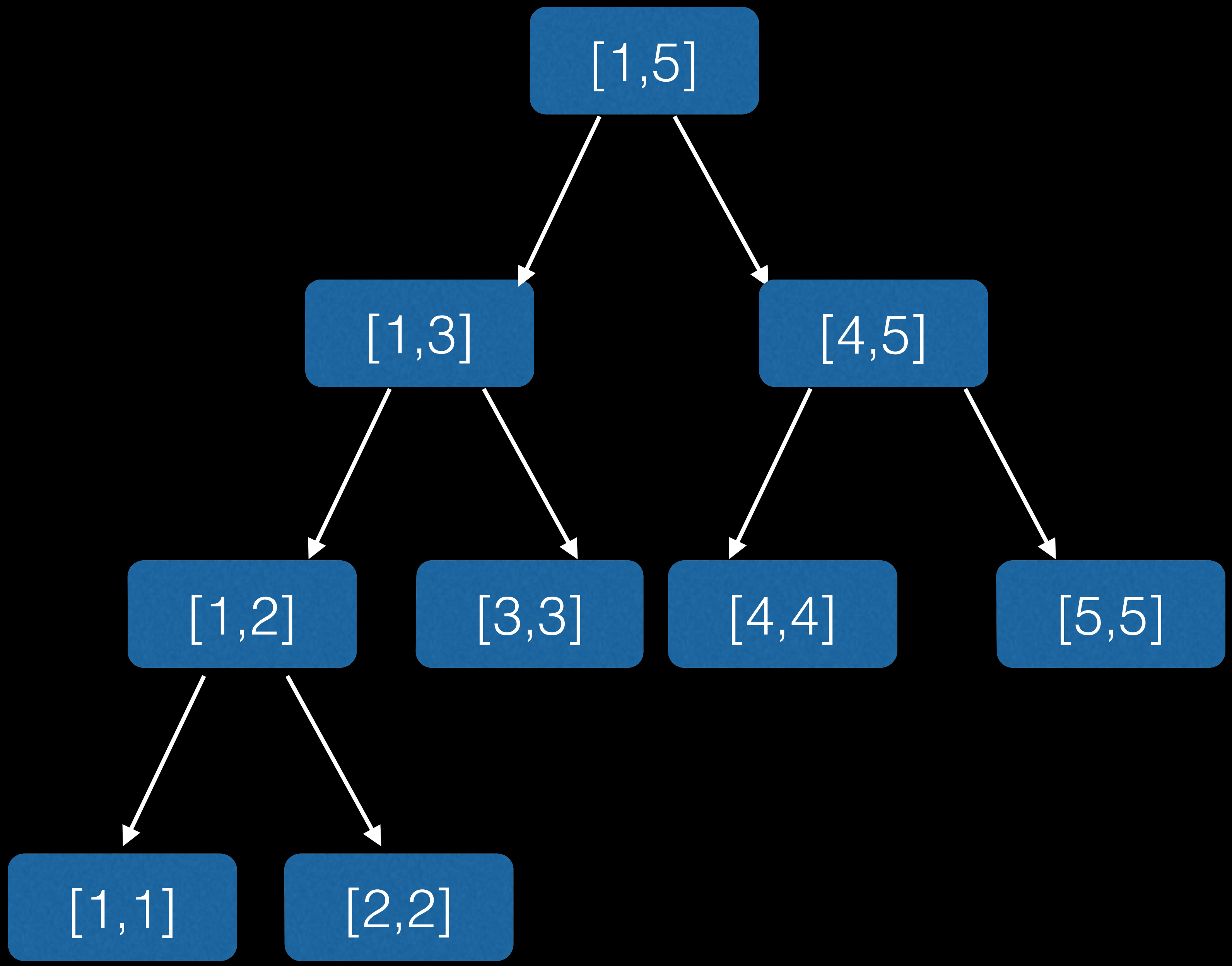
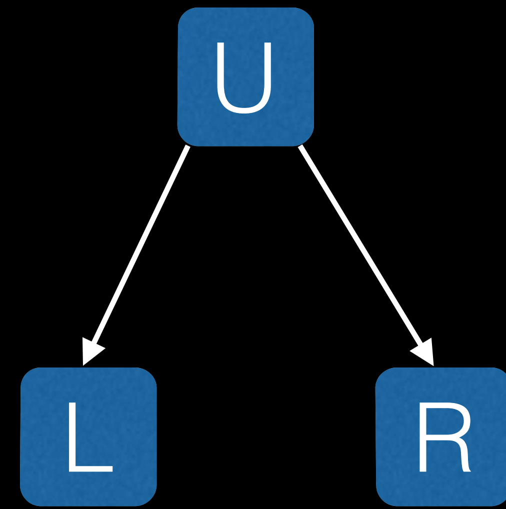
线段树

软工1104 林安泽

线段树的基本结构	线段树建立 build(l, r)
单点查询	单点询问 query(x)
区间查询	区间询问 query(l, r)
单点修改/加减	modify(x, v)/add(x, v) update/pushup操作
区间加减	add(l, r, v) lazy标记 pushdown操作

给出n个数的序列，要求支持两种操作	
modify(x,t)	修改第x个数，变为t
query(l,r)	询问l到r的区间和


[1,5]
[1,3]
[4,5]
[1,2]
[3,3]
[4,4]
[5,5]
[1,1]
[2,2]

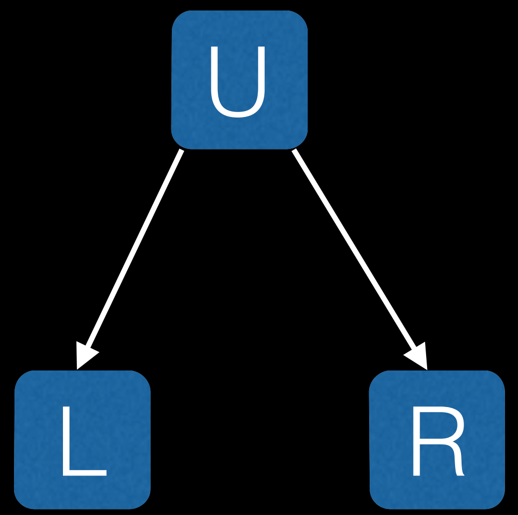


U
range:
[l, r]

L
range:
[l,mid]

R
range:
(mid, r]

struct	
[l, r]	记录的是[l,r]这一段的信息
sum	区间和



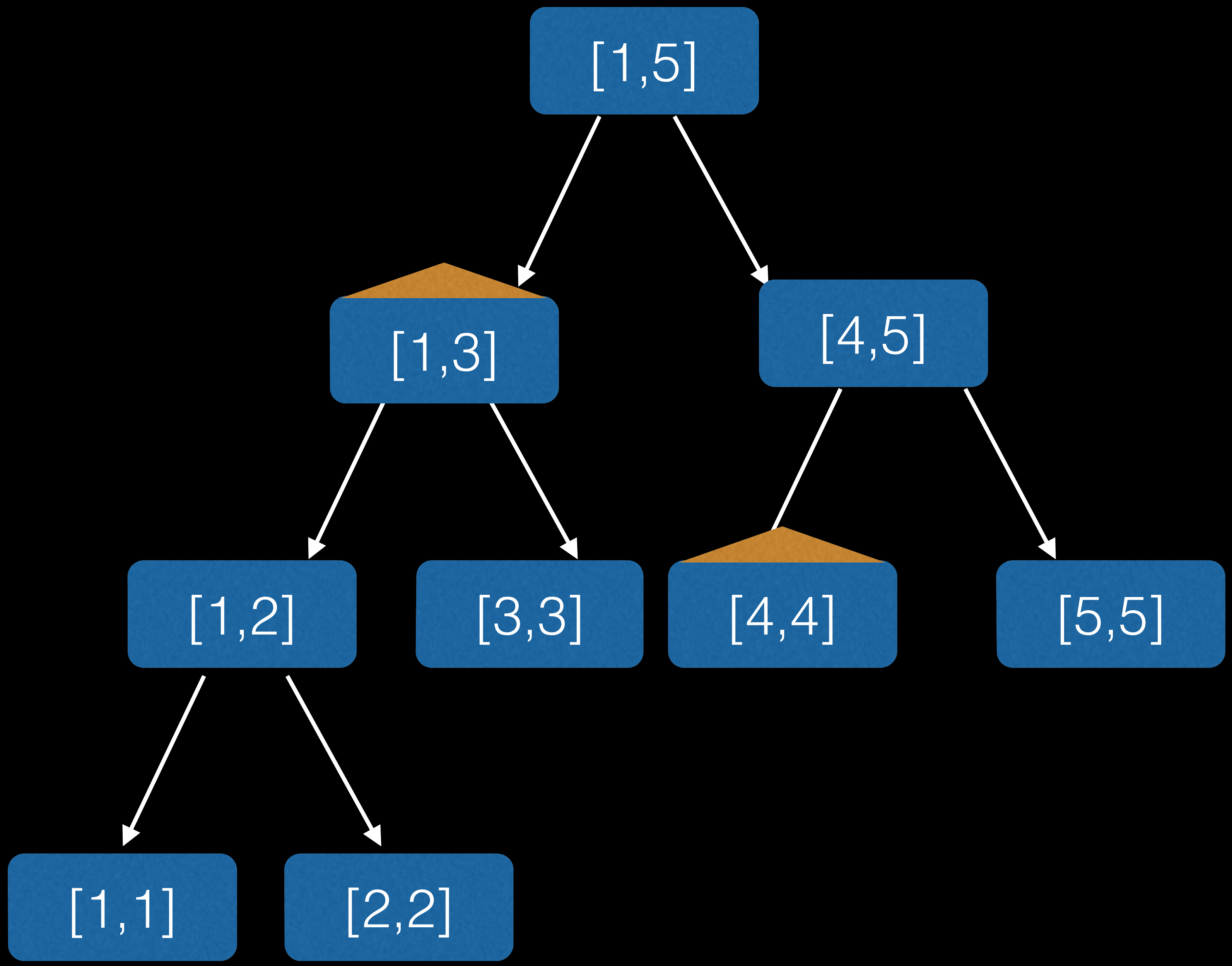
U	L	R
range: [1,2]	range: [1, 1]	range: [2,2]
value: ?	value: 3	value: 6

struct	
[l, r]	记录的是[l,r]这一段的信息
sum	区间和

update
sumU = sumL + sumR

给出n个数的序列，要求支持两种操作	
add(l, r, v)	区间l到r加v
query(l,r)	询问l到r的区间和

add(1, 4, 3)



struct	
[l, r]	记录的是[l,r]这一段的信息
sum	区间和
lazy_sum	lazy标记，表示这段区间数值加lazy_sum，尚未传递到子树

pushdown	
L	lazy_sumL += lazy_sumU sumL += (mid-l+1)*lazy_sumU
R	lazy_sumR += lazy_sumU sumL += (r-mid)*lazy_sumU
U	lazy_sumU = 0

update
sumU = sumL + sumR

只要能写出这三个模块，
线段树问题基本上就解决了。

线段树进阶

给出n个数的序列，要求支持三种操作	
add(l, r, v)	区间l到r加v
modify(x, v)	修改Ax为v
query(l,r)	sigma(l,r)[aAi^2+bAi+c]

struct	
[l, r]	记录的是[l,r]这一段的信息
sum	sigma(l,r)[aAi²+bAi+c]
lazy_sum	lazy标记，表示这段区间数值加lazy_sum，尚未传递到子树

update
sumU = sumL + sumR

pushdown	
L	lazy_sumL += lazy_sumU sumL += (mid-l+1)*lazy_sumU
R	lazy_sumR += lazy_sumU sumL += (r-mid)*lazy_sumU
U	lazy_sumU = 0

struct	
[l, r]	记录的是[l,r]这一段的信息
sum	sigma(l,r)[aAi²+bAi+c]
lazy_sum	lazy标记，表示这段区间数值加lazy_sum，尚未传递到子树

update
sumU = sumL + sumR

pushdown	
L	~~la[illegible]sumL += lazy_sumU~~ ~~sumL[illegible](mid-l+1)*la[illegible]_sumU~~
R	~~lazy_sumR +[illegible]y_sumU~~ ~~sumL += ([illegible]d)[illegible] sumU~~
U	~~la[illegible]umU = 0~~

add(U, x)

$$\text{sigma}(l,r)[a(A_i+x)^2+b(A_i+x)+c]$$

$$=\text{sigma}(l,r)[ax^2+(2aA_i+b)x+aA_i^2+bA_i+c]$$

$$=a*(r-l+1)*x^2+(2*a*\text{sigma}(A_i)+b)x$$
$$+a*\text{sigma}(A_i^2)+b*\text{sigma}(A_i)+c$$

struct	
[l, r]	记录的是[l,r]这一段的信息
sum	sigma(l,r)[aAi²+bAi+c]
sum_Ai	sigma(l,r)[Ai]
sum_Ai2	sigma(l,r)[Ai²]
lazy_sum	lazy标记，表示这段区间数值加lazy_sum，尚未传递到子树

pushdown	
L	lazy_sumL += lazy_sumU add(L, lazy_sumU)
R	lazy_sumR += lazy_sumU add(R, lazy_sumU)
U	lazy_sumU = 0

update
sumU = sumL + sumR
sum_Ai = sumAiL + sumAiR
sum_Ai2 = sumAi2L + sumAi2R

给出n个数的序列，要求支持三种操作	
add(l, r, v)	区间l到r加v
multiply(l, r, v)	区间l到r乘v
query(l,r)	sigma(l,r)[Ai]

Solution

对于没有乘操作的时候，我们只需要打一个标记a，
表示这段区间 [x => x+a]

现在我们需要打两个标记a, b,
表示这段[x => ax+b]，初始a=1, b=0

[ax+b] 经过一次乘操作 => (ax+b)*v = avx+bv
=> a->av b->bv
经过一次加操作 => (ax+b) + v = ax + b+v
=> a->a b->b+v

给出n个数的序列，要求支持一种操作	
query(x, y)	max{a[i]+a[i+1]+…+a[j], x<=i<=j<=y}

struct	
[l, r]	记录的是[l,r]这一段的信息
sum	sigma(l,r)[Ai]
left_max	max{a[l]+…a[x], l<=x<=r}
right_max	max{a[x]+…+a[r], l<=x<=r}
max_ans	max{a[x]+…+a[y], l<=x<=y<=r}

update
left_max = max(left_maxL, sumL + left_maxR)
right_max = max(right_maxR, sumR + right_maxL)
max_ans = max(max_ansL, max_ansR, right_maxL+left_maxR)

给出n个数的序列，要求支持两种操作	
sqrt(l, r)	l到r区间内，Ai->sqrt(Ai)
query(l,r)	区间和

Solution

对于一个数N，经过logN次开根号操作，就会变成1.
对于一个区间，如果区间里面的数都是1， 就不需要再对这个区间进行操作。

均摊之后时间复杂度 O(nlgn)

题目推荐

SPOJ GSS2

UESTC1546 Bracket Sequence

矩形面积并、周长并等经典扫描线问题

Thanks