

向量

有序向量：二分查找（版本A）

02-D2

自从爷爷去后，这山被二郎菩萨点上火，烧杀了大半。我们蹲在井里，钻在涧内，藏于铁板桥下，得了性命。及至火灭烟消，出来时，又没花果养赡，难以存活，别处又去了一半。我们这一半，捱苦的住在山中，这两年，又被些打猎的抢了一半去也。

邓俊辉

deng@tsinghua.edu.cn

统一接口

```
template <typename T> //查找算法统一接口, 0 <= lo < hi <= _size

Rank Vector<T>::search( T const & e, Rank lo, Rank hi ) const {

    return ( rand() % 2 ) ? //按各50%的概率随机选用

        binSearch( _elem, e, lo, hi ) //二分查找算法, 或者

        : fibSearch( _elem, e, lo, hi ); //Fibonacci查找算法

}
```



有序向量中，每个元素都是轴点

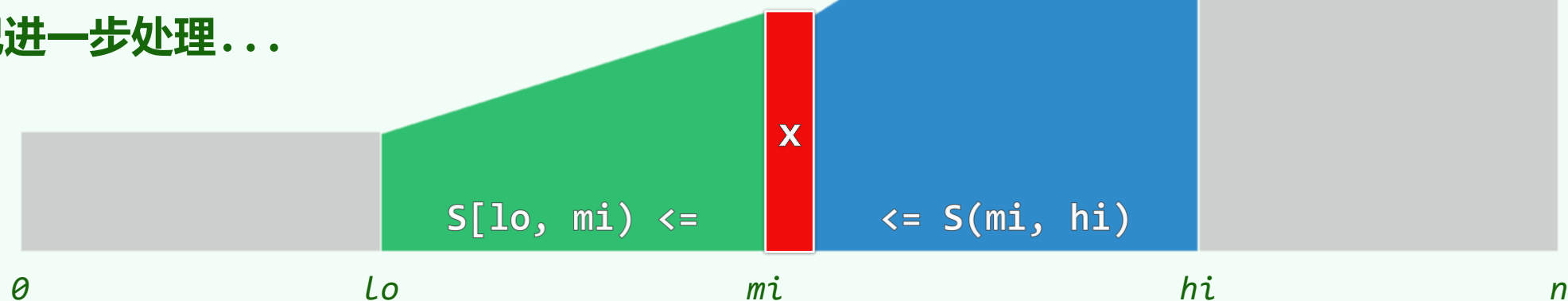
❖ 以任一元素 $x = S[mi]$ 为界，都可将待查找区间 $[lo, hi)$ 分为三部分

$$S[lo, mi) \leq S[mi] \leq S(mi, hi)$$

因此...

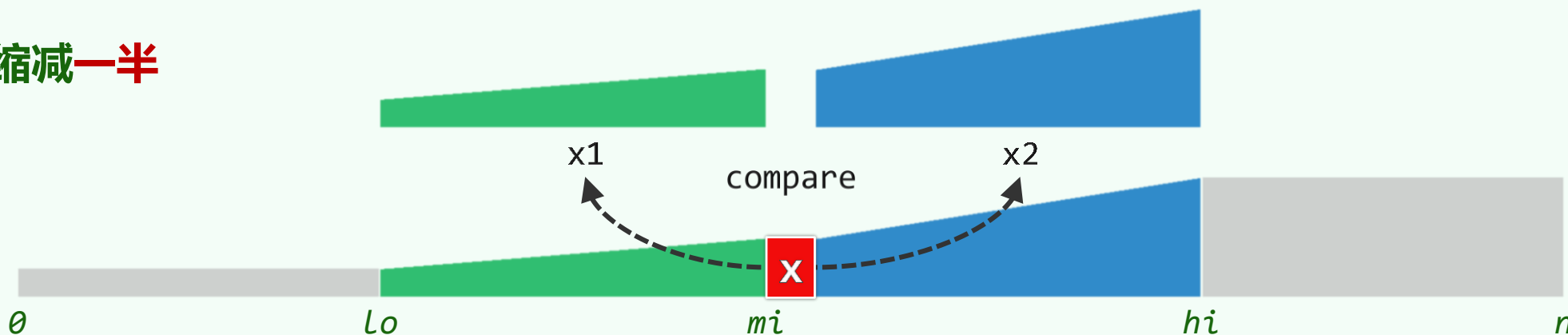
❖ 只需将目标元素 e 与 x 做一次比较

即可分三种情况进一步处理...



减而治之

- ❖ $e < x$: 则 e 若存在必属于左侧子区间, 故可 (减除 $S[mi, hi)$ 并) 递归深入 $S[lo, mi)$
- $x < e$: 则 e 若存在必属于右侧子区间, 亦可 (减除 $S[lo, mi]$ 并) 递归深入 $S(mi, hi)$
- $e = x$: 已在此处命中, 可随即返回 //若有多个, 返回何者?
- ❖ 若轴点 mi 取作中点, 则每经过至多两次比较
- ❖ 或者能够命中, 或者
- 将问题规模缩减一半



实现

```
template <typename T> //在有序向量区间[lo, hi)内查找元素e
static Rank binSearch( T * S, T const & e, Rank lo, Rank hi ) {
    while ( lo < hi ) { //每步迭代可能要做两次比较判断，有三个分支
        Rank mi = ( lo + hi ) >> 1; //轴点居中（区间宽度折半，等效于其数值右移一位）
        if ( e < S[mi] ) hi = mi; //深入前半段[lo, mi)
        else if ( S[mi] < e ) lo = mi + 1; //深入后半段(mi, hi)
        else return mi; //命中
    }
    return -1; //失败
}
```

实例 + 复杂度

S.search(8, 0, 7):

经 $2 + 1 + 2 = 5$ 次比较, 在S[4]命中

	0	1	2	3	4	5	6	7
(1)	2	4	5	7	8	9	12	
(2)					8	9	12	
(3)					8			

S.search(3, 0, 7):

经 $1 + 1 + 2 = 4$ 次比较, 在S[1]失败

	0	1	2	3	4	5	6	7
(1)	2	4	5	7	8	9	12	
(2)	2	4	5					
(3)	2							
(4)								

❖ 线性“递归”： $T(n) = T(n/2) + \mathcal{O}(1) = \mathcal{O}(\log n)$ ，大大优于顺序查找

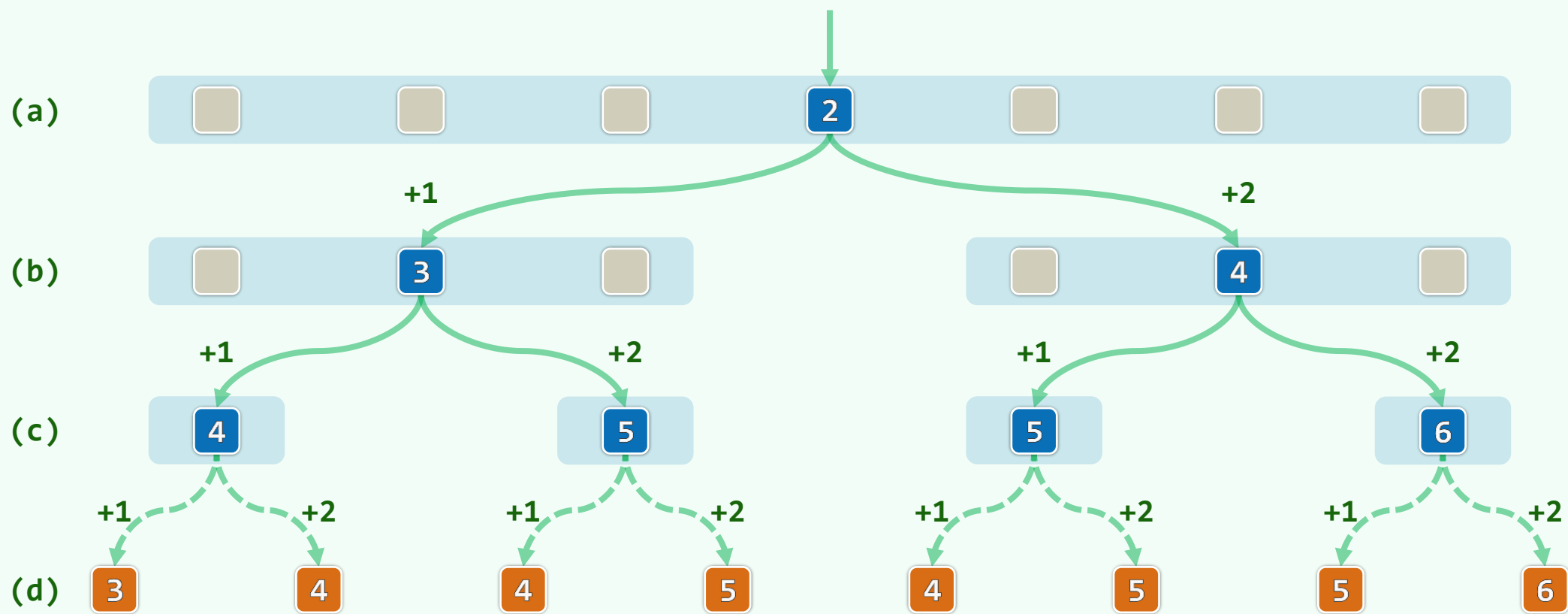
“递归”跟踪：轴点总能取到中点，递归深度 $\mathcal{O}(\log n)$ ；各递归实例仅耗时 $\mathcal{O}(1)$

关键码的比较次数 ~ 查找长度 (search length)

❖ 通常，需分别针对**成功**与**失败**查找，从**最好**、**最坏**、**平均**等角度评估

❖ 比如，成功、失败时的平均查找长度均大致为 $O(1.50 \cdot \log n)$

// 详见教材、习题解析



查找长度实例：n = 7

❖ **成功**情况共**7**种，查找长度分别为

{ 4, 3, 5, 2, 5, 4, 6 }

等概率情况下，平均 = $29 / 7 = 4.14$

❖ **失败**情况共**8**种，查找长度分别为

{ 3, 4, 4, 5, 4, 5, 5, 6 }

等概率情况下，平均 = $36 / 8 = 4.50$

