8.高级搜索树

(b3) B-树：查找

高至天低至深海

每寸搜索着这天下

寻觅着那个"它"

邓俊辉

deng@tsinghua.edu.cn

算法

❖ 将根节点作为当前节点 //常驻RAM

只要当前节点非外部节点

在当前节点中顺序查找 //RAM内部

若找到目标关键码，则

返回查找成功

否则 //止于某一对下层引用

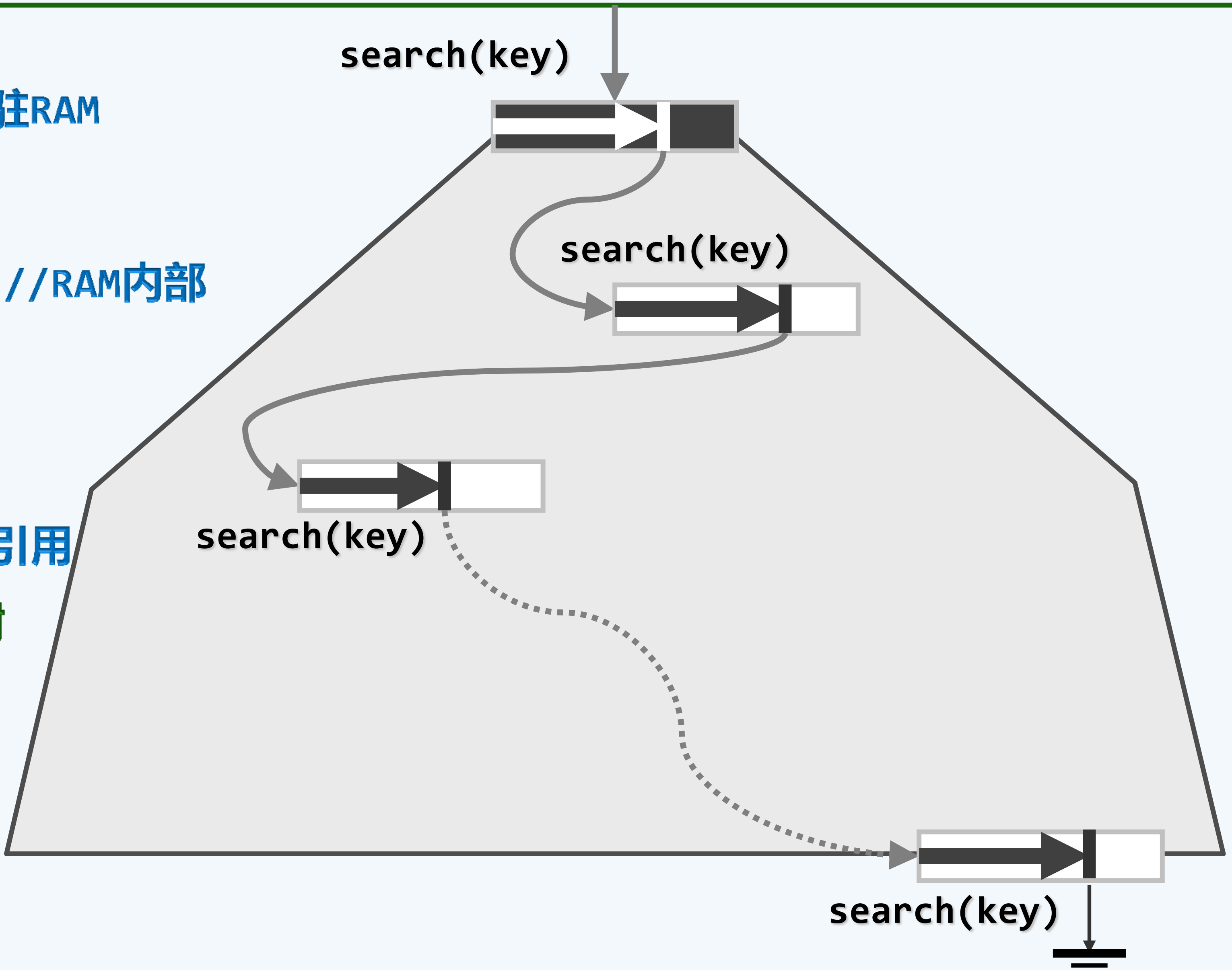
沿引用，转至对应子树

将其根节点读入内存

//I/O，最为耗时

更新当前节点

返回查找失败

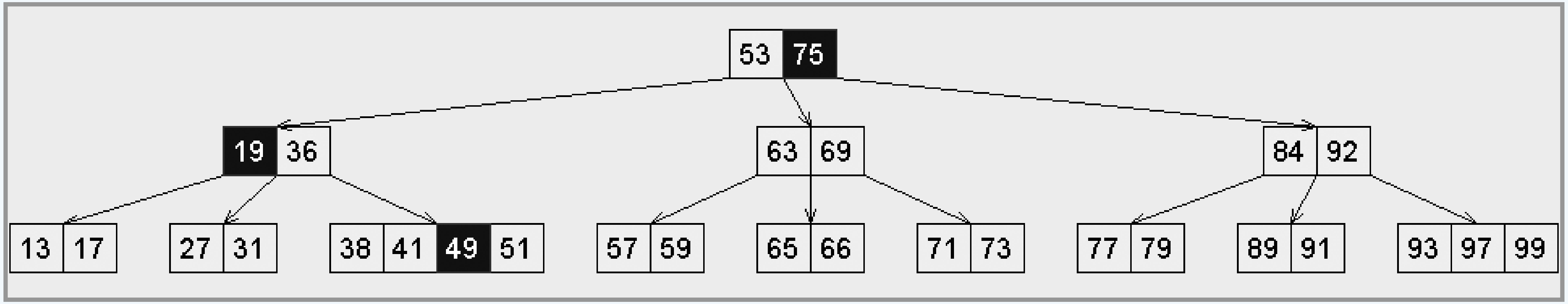


实例

❖ (3,5)-树：53 97 36 89 41 75 19 84 77 79 51 57 99 91
92 93 17 73 13 66 59 49 63 65 71 69 27 31 38

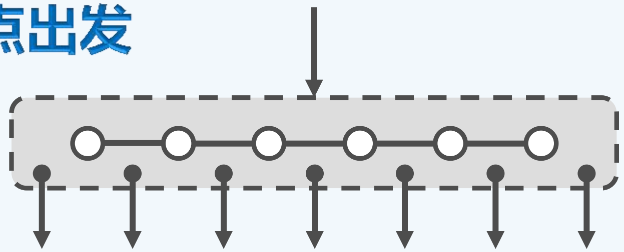
成功查找：75, 19, 49

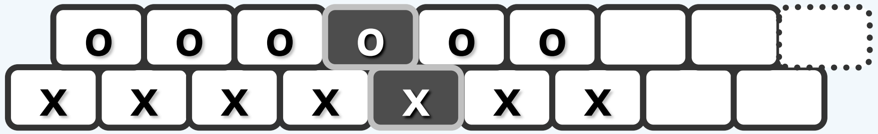
失败查找：5, 45



实现

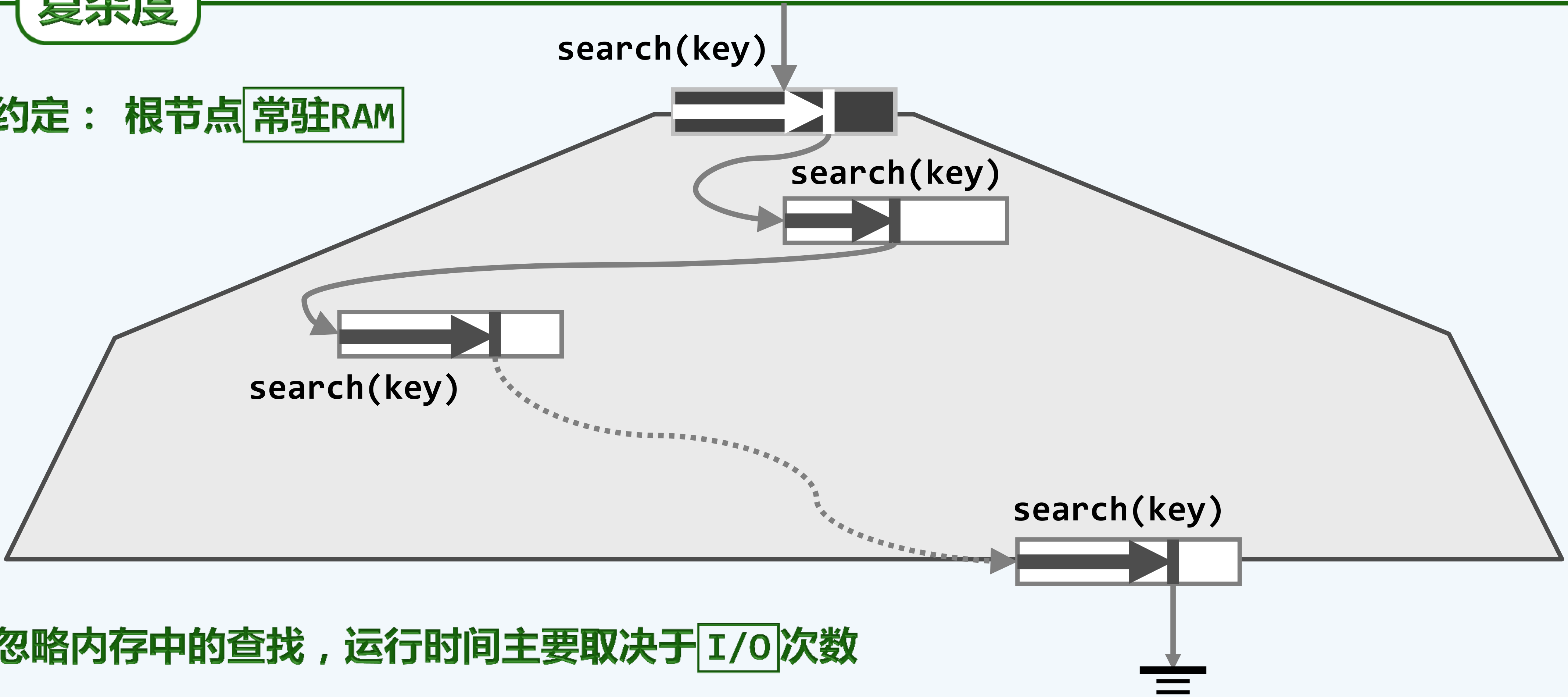
```
❖ template <typename T> BTNodePosi(T) BTree<T>::search( const T & e ) {
     BTNodePosi(T) v = _root; _hot = NULL; //从根节点出发
     while ( v ) { //逐层查找
        Rank r = v->key.search( e ); //在当前节点对应的向量中顺序查找
        if ( 0 <= r && e == v->key[ r ] ) return v; //若成功，则返回；否则...
        _hot = v; v = v->child[ r + 1 ]; //沿引用转至对应的下层子树，并载入其根I/O
     } //若因!v而退出，则意味着抵达外部节点
     return NULL; //失败
  }
```





复杂度

❖ 约定： 根节点 $\boxed{\text{常驻RAM}}$



❖ 忽略内存中的查找，运行时间主要取决于 $\boxed{\text{I/O}}$ 次数

❖ 在每一深度 $\boxed{\text{至多一次}}$ I/O

❖ 故运行时间 = Θ(终止节点的深度) = $\boxed{\mathcal{O}(h)}$

最大树高

- 含 $\boxed{N}$ 个关键码的 $\boxed{m}$ 阶B-树，$\boxed{\text{最大高度} = ?}$

- 为此，内部节点应尽可能"瘦"，各层节点数依次为

$$n_0 = 1,\ n_1 = 2,\ n_2 = 2 \times \lceil m/2 \rceil,\ \ldots$$

$$n_k = 2 \times \lceil m/2 \rceil^{k-1}$$

- 考查 $\boxed{\text{外部节点}}$ 所在层

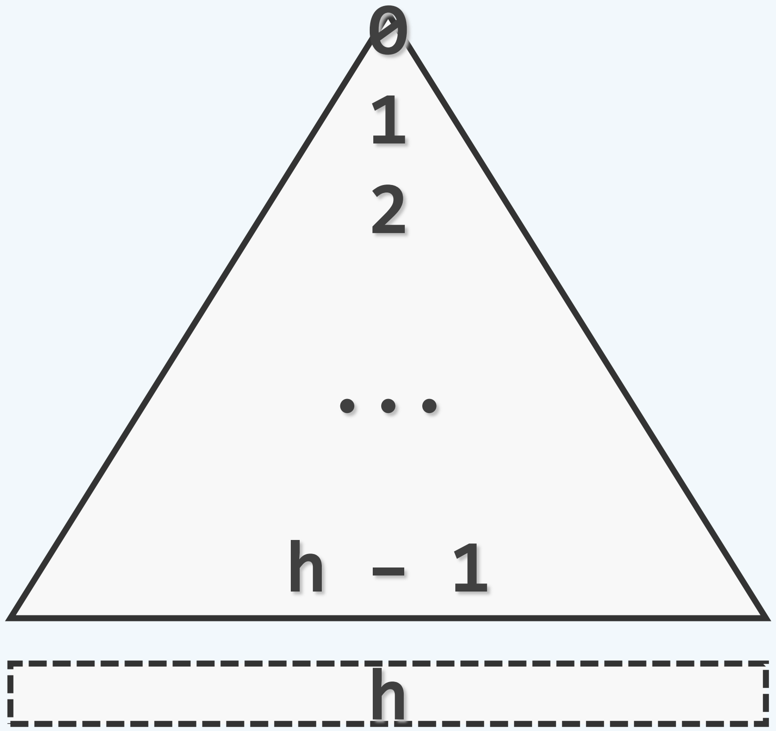
$$\boxed{N + 1 = n_h} \geq 2 \times (\lceil m/2 \rceil)^{h-1}$$

$$h \leq 1 + \log_{\lceil m/2 \rceil} \lfloor (N + 1)/2 \rfloor = \mathcal{O}(\log_m N)$$

- 相对于BBST：

$$\log_{\lceil m/2 \rceil}(N/2) \,/\, \log_2 N = 1/(\log_2 m - 1)$$

若取m = 256，树高（I/O次数）约降低至 $\boxed{1/7}$



最小树高

- 含$\boxed{N}$个关键码的$\boxed{m}$阶B-树，$\boxed{\text{最小高度} = ?}$
- 为此，内部节点应尽可能“胖”

  各层节点数依次为

  $n_0 = 1,\ n_1 = m,\ n_2 = m^2$

  $n_3 = m^3,\ \ldots,\ n_{h-1} = m^{h-1},\ \boxed{n_h = m^h}$

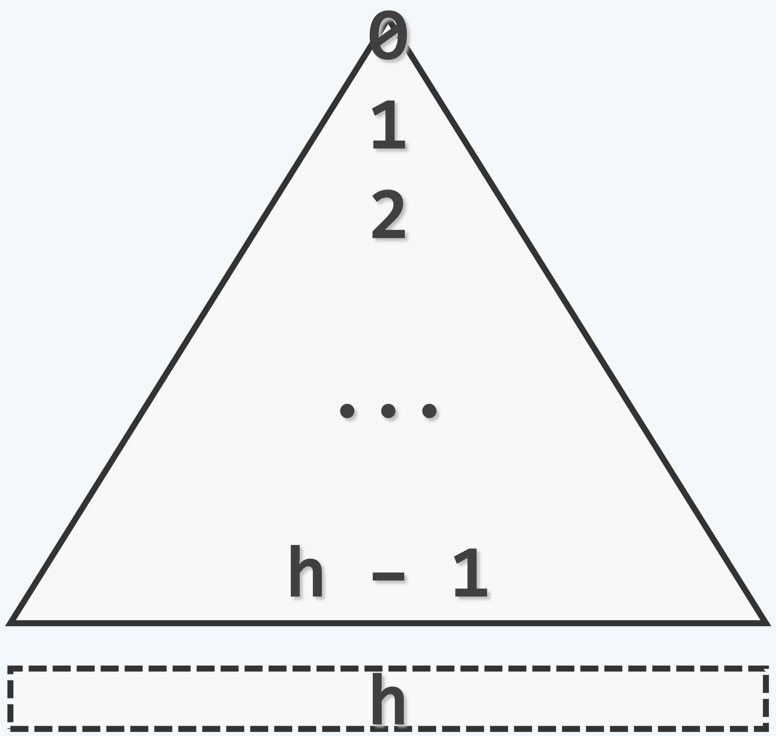
- 考查$\boxed{\text{外部节点}}$所在层：

  $\boxed{N + 1 = n_h} \le m^h$

  $h \ge \log_m(N + 1) = \Omega(\log_m N)$

- 相对于BBST：$(\log_m N - 1)/\log_2 N = \log_m 2 - \log_N 2 \approx 1/\log_2 m$

  若取$m = 256$，树高（I/O次数）约降低至$\boxed{1/8}$



意义与价值

- ❖ BBST，究竟可能多高？
- ❖ 考查高度为h的BBST（如AVL）可包含节点的数目f

```
h =  33 时，f = 2 ^ 033 = 8.6 * 10 ^ 09    //全球人口
h =  77 时，f = 2 ^ 077 = 1.5 * 10 ^ 23    //全球人口的体细胞总数
h = 133 时，f = 2 ^ 133 = 1.1 * 10 ^ 40    //国际象棋可能的局面总数
h = 260 时，f = 2 ^ 260 = 1.8 * 10 ^ 78    //目前可观测宇宙中基本粒子总数
```

- ❖ 由此可见，B-树的价值，的确更多地体现在实用方面

  通过选取适当的节点规模（m），弥合存储层级之间巨大的速度差异

- ❖ 另外，在算法方面，B-树也有其独特的价值与地位...