

θ8-B4

高级搜索树

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"

...按照模型的运算量，用现有的最高计算能力模拟百分之一秒的聚变过程，就需大约二十年时间。而研究过程中的模拟需要反复进行，这使得模型的实际应用成为不可能

邓俊辉

deng@tsinghua.edu.cn

算法

从（常驻RAM的）根节点开始

只要当前节点不是外部节点

在当前节点中**顺序查找** //RAM内部

若找到目标关键码，则

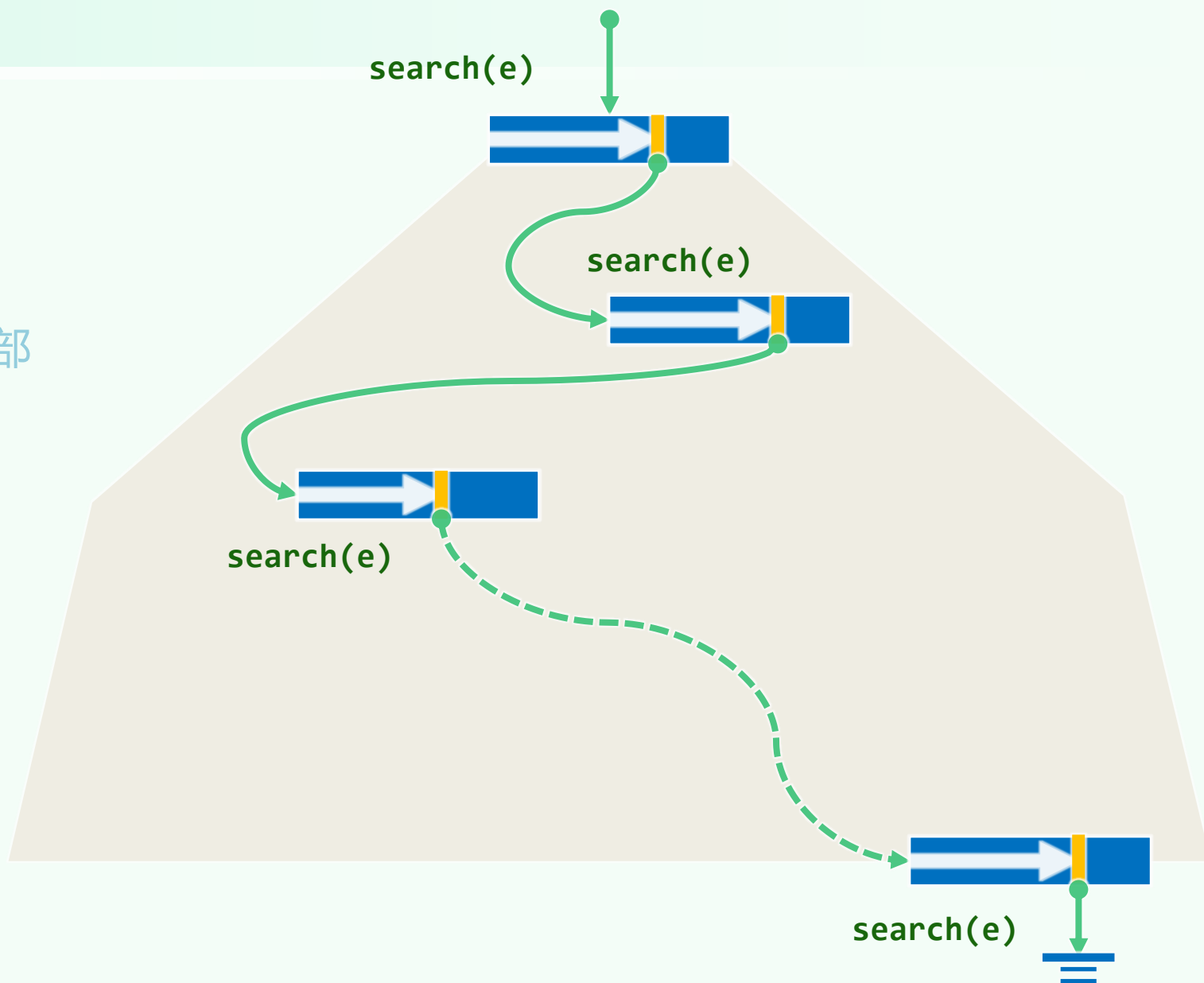
返回查找成功

否则 //止于某一向下的引用

沿引用找到孩子节点

将其**读入内存** //I/O耗时

返回查找失败

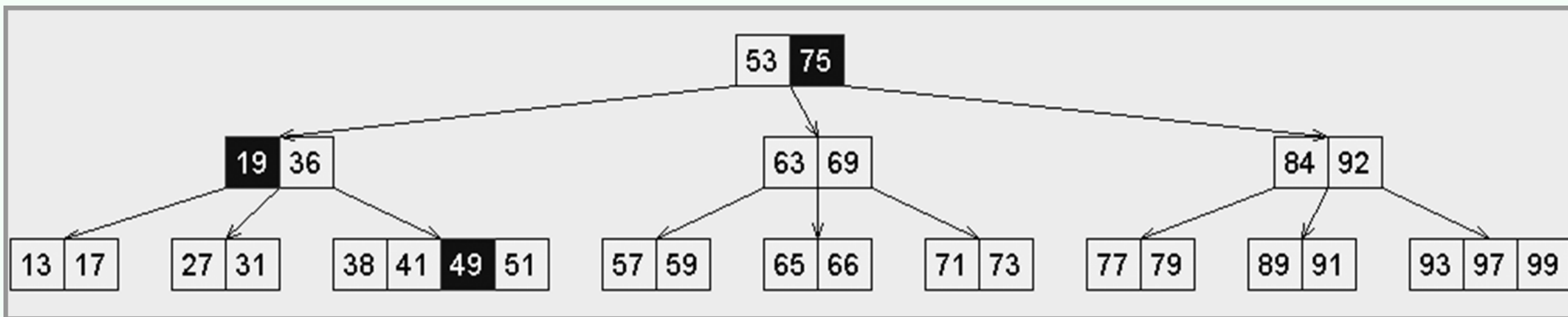


实例

❖ (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 ) {  
    _hot = NULL;  
  
    for ( BTNodePosi<T> v = _root; v; ) { //从根节点出发, 逐层深入地  
        Rank r = v->key.search( e ); //在当前节点中, 找到不大于e的最大关键码  
        if ( (-1 != r) && (e == v->key[r]) ) return v; //成功; 否则...  
        _hot = v; v = v->child[ r + 1 ]; //沿引用转至对应的孩子节点 (I/O)  
    } //这里在向量key内是二分查找, 但对通常的_m, 改为顺序查找效率反而更高  
    return NULL; //失败  
}
```

Diagram illustrating the search process in a B-tree node. The node contains a vector of keys (0, 0, 0, 0, 0, 0, 0, ...) and a vector of child pointers (X, X, X, X, X, X, X, ...). The search process involves finding the largest key less than or equal to the target value 'e'. In the diagram, the target value 'e' is represented by a red dot on the horizontal line, and the corresponding child pointer is highlighted in red.

性能

❖ 约定：根节点常驻RAM

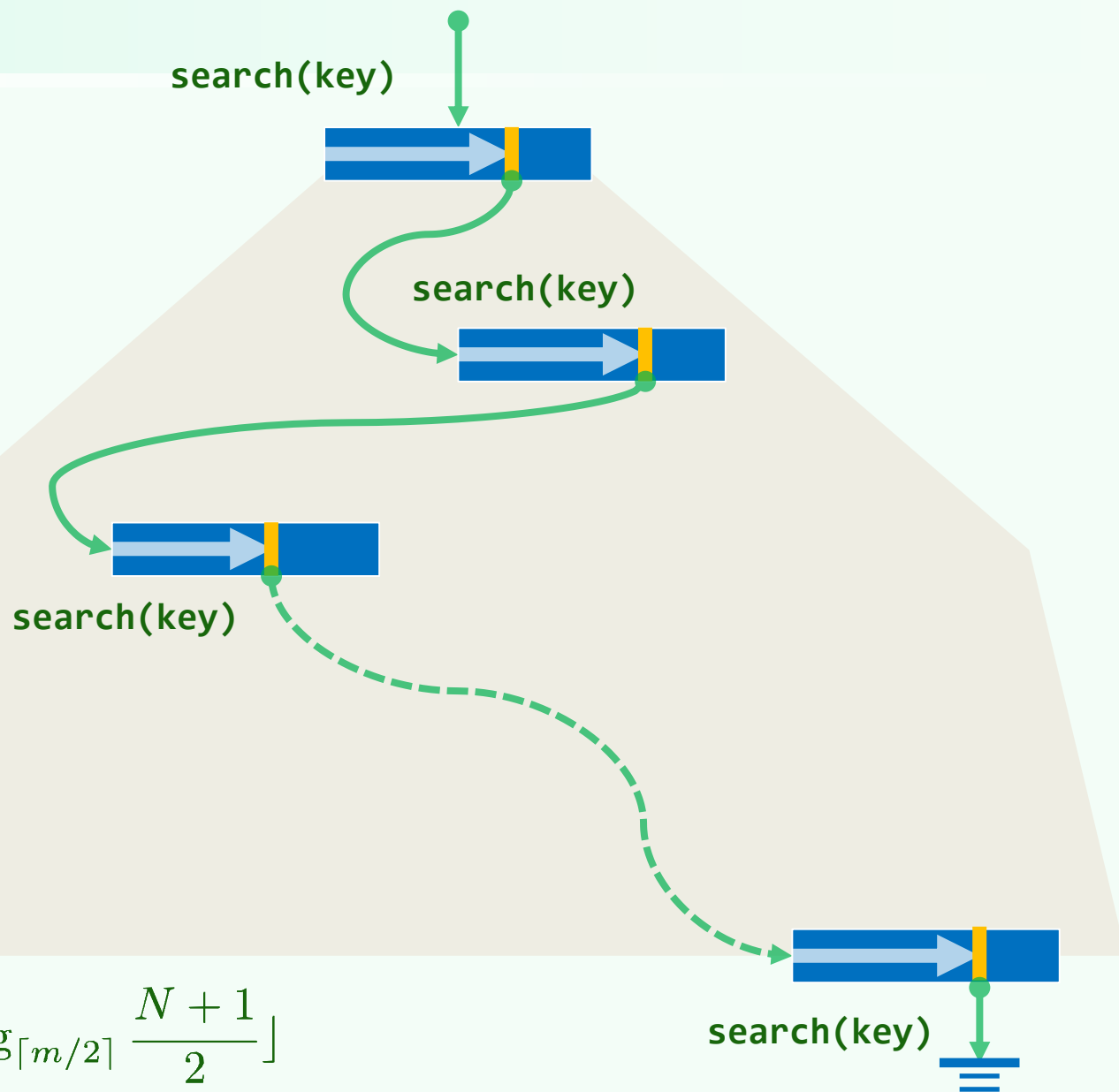
❖ 忽略内存中的查找

运行时间主要取决于I/O次数

❖ 在每一深度至多一次I/O

❖ 故运行时间为 $\mathcal{O}(\log n)$

❖ 可以证明： $\log_m (N + 1) \leq h \leq 1 + \lfloor \log_{\lceil m/2 \rceil} \frac{N + 1}{2} \rfloor$



含N个关键码的m阶B-树，可能有多高？

❖ 为此，内部节点应尽可能地“瘦”

$$n_k \geq 2 \times \lceil m/2 \rceil^{k-1}, \quad \forall k > 0$$

❖ 考查外部节点所在的那层：

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

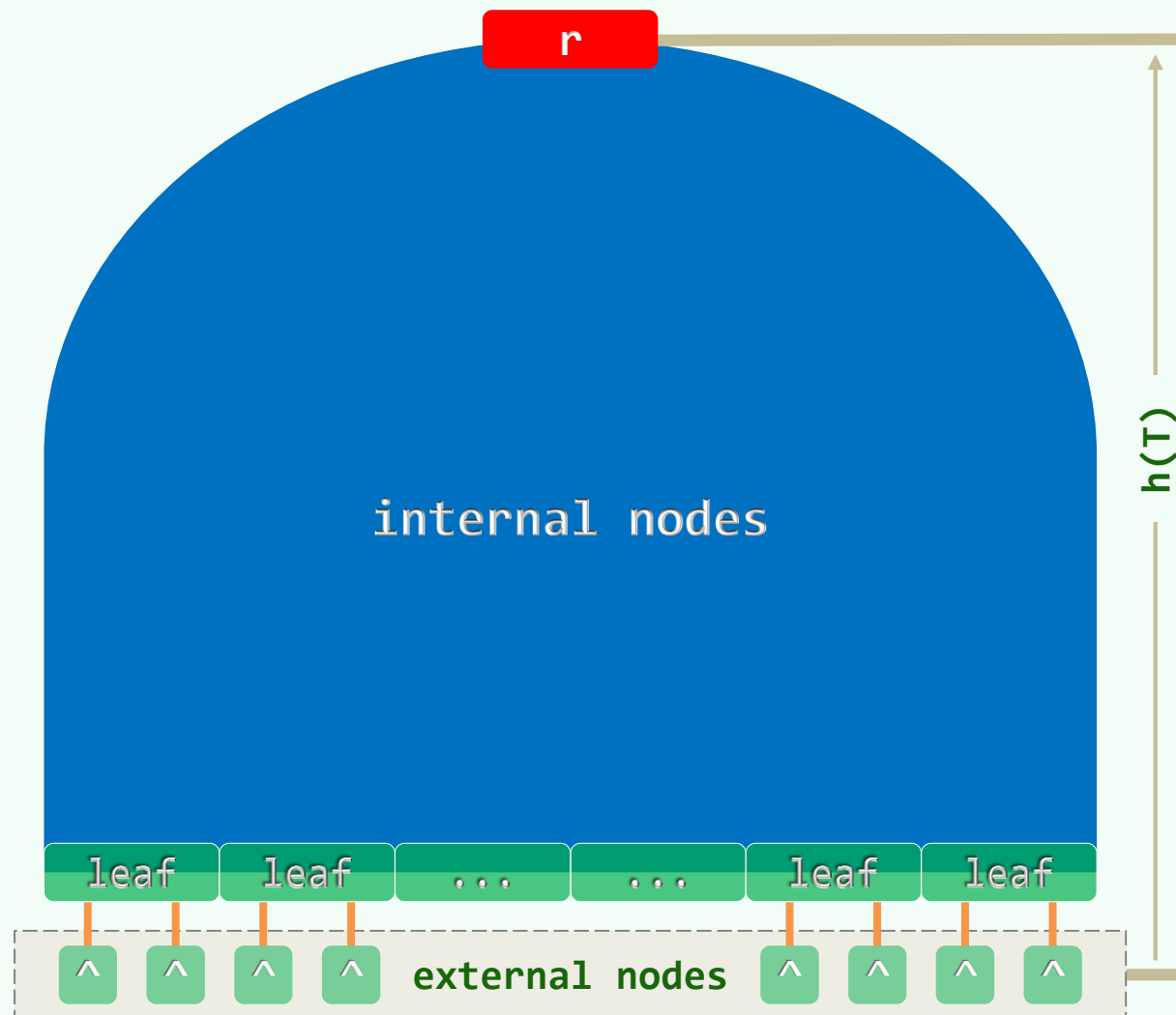
$$h \leq 1 + \lfloor \log_{\lceil \frac{m}{2} \rceil} \frac{N+1}{2} \rfloor = \mathcal{O}(\log_m N)$$

❖ 相对于BBST：

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

若取 $m = 256$ ，树高约降低至1/7

...用4年上完大学，还是28年？



含N个关键码的m阶B-树，可能有多矮？

❖ 为此，内部节点应尽可能地“胖”

$$n_k \leq m^k, \quad \forall k \geq 0$$

❖ 依然，考查外部节点所在的那层

$$N + 1 = n_h \leq m^h$$

$$h \geq \lceil \log_m (N + 1) \rceil = \Omega(\log_m N)$$

❖ 相对于BBST：

$$\begin{aligned} & (\log_m N - 1) / \log_2 N \\ &= \log_m 2 - \log_N 2 \approx 1 / \log_2 m \end{aligned}$$

若取 $m = 256$ ，树高约降低至 **1/8**

