Data Structures and Algorithms (10)

Instructor: Ming Zhang
Textbook Authors: Ming Zhang, Tengjiao Wang and Haiyan Zhao
Higher Education Press, 2008.6 (the "Eleventh Five-Year" national planning textbook)

https://courses.edx.org/courses/PekingX/04830050x/2T2014/

Chapter 10. Search

- 10.1 Search in a linear list
- 10.2 Search in a set
- 10.3 Search in a hash table
- Summary

Search in a Linear List

- 10.1.1 Sequential search
- 10.1.2 Binary search
- 10.1.3 Blocking search

Sequential Search

- Compare the key values of records in a linear list with the given value one by one
  - If the key value of a record is equal to the given value, the search hits;
  - Otherwise the search misses (cannot find the given value in the end)
- Storage: sequential or linked
- Sorting requirements: none

Sequential Search with Sentinel

```
// Return position of the element if hit; otherwise return 0
template  <class Type>
class Item {
private:
    Type  key;                              // key field
                                            // other fields
public:
    Item(Type value):key(value) {}
    Type getKey() {return key;}             // get the key
    void setKey(Type k){ key=k;}            // set the key
};
vector<Item<Type>*>  dataList;
template  <class Type>  int SeqSearch(vector<Item<Type>*>& dataList, int
length, Type k) {
    int i=length;
    dataList[0]->setKey (k);                // set the 0th element as the element
                                            to be searched, set the lookout
    while(dataList[i]->getKey()!=k) i--;
    return i;                               // return the position of the element
}
```

Performance Analysis of the Sequential Search

- Search hits: assume the probability of searching any key value is uniform: $P_i = 1/n$

$$\sum_{i=0}^{n-1} P_i \cdot (n-i) = \frac{1}{n}\sum_{i=0}^{n-1}(n-i)$$

$$= \frac{1}{n}\sum_{i=1}^{n} i = \frac{n+1}{2}$$

- Search misses: assume that n+1 times of comparisons are needed when the search misses (with a sentinel)

Average Search Length of Sequential Search

- Assume the probability of search hit is p, and the probability of search miss is q=(1-p)

$$\begin{aligned} \text{ASL} &= p \cdot \frac{n+1}{2} + q \cdot (n+1) \\ &= p \cdot \frac{n+1}{2} + (1-p)(n+1) \\ &= (n+1)(1-p/2) \end{aligned}$$

- $(n+1)/2 < ASL < (n+1)$

Pros and Cons of Sequential Search

- Pros: insertion in $\Theta(1)$ time
  - We can insert a new element into the tail of list
- Cons: search in $\Theta(n)$ time
  - Too time-consuming

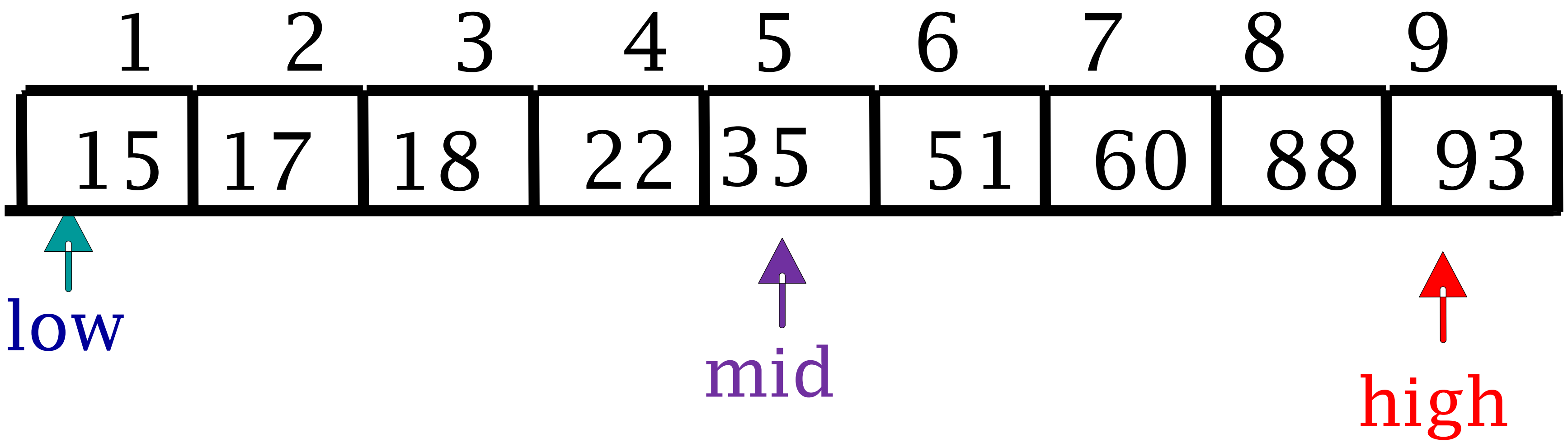
Binary Search

- Compare any element dataList[i].Key with the given value K, there are three situations:
  - (1) Key = K, succeed, return dataList[i]
  - (2) Key > K, the element to find must be before dataList[i] if exists
  - (3) Key < K, the element to find must be after dataList[i] if exists
- Reduce the range of latter search

Binary Search Algorithm

```
template  <class Type> int BinSearch (vector<Item<Type>*>& dataList, int
length, Type k){
    int low=1, high=length, mid;
    while (low<=high)  {
        mid=(low+high)/2;
        if (k<dataList[mid]->getKey())
            high = mid-1;                // drop the right half of the search range
        else if (k>dataList[mid]->getKey())
            low = mid+1;                 // drop the left half of the search range
       else return mid;                  // return if succeeds
    }
    return 0;                            // if fails, return 0
}
```

Key value 18 low=1 high=9



The first time: l=1, h=9, mid=5; array[5]=35>18

The second time: l=1, h=4, mid=2; array[2]=17<18

The third time: l=3, h=4, mid=3; array[3]=18 = 18

Performance Analysis of the Binary Search

- Maximum search length is

$$\lceil \log_2(n+1) \rceil$$

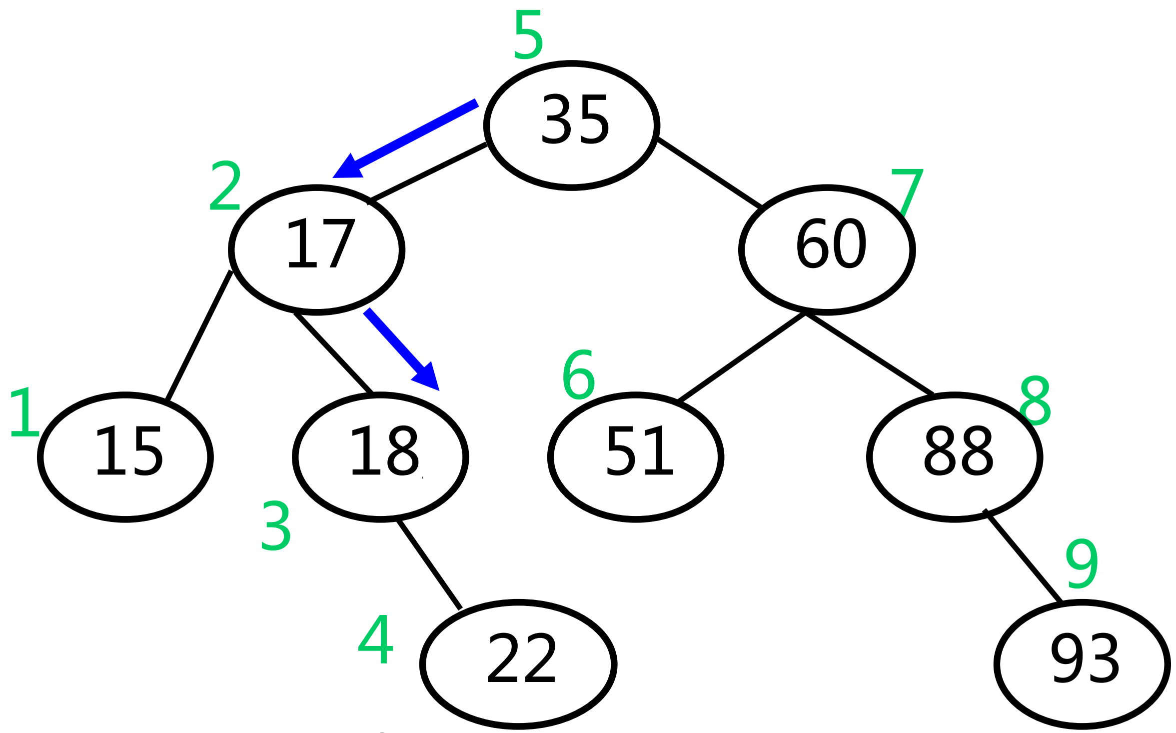
- Failed search length is

$$\lceil \log_2(n+1) \rceil$$

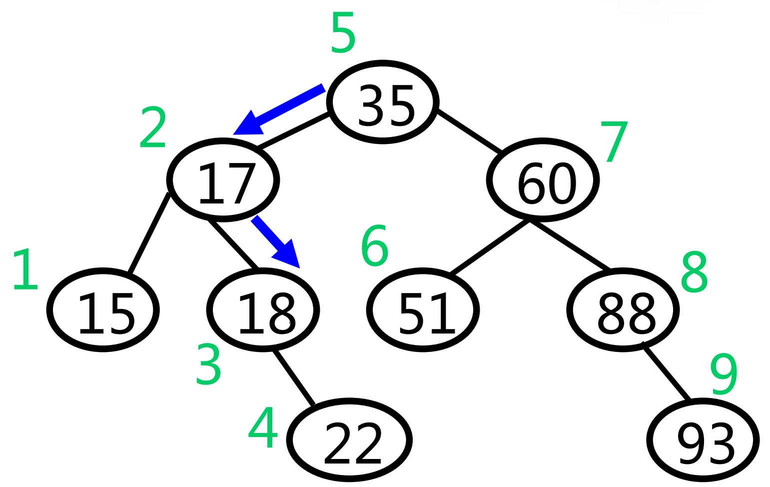
Or

$$\lfloor \log_2(n+1) \rfloor$$

- In the complexity analysis
  - The logarithm base is 2
  - When the log base changes, the order of complexity will not change



Performance Analysis of the Binary Search



- ASL of successful search is:

$$ASL = \frac{1}{n}\left(\sum_{i=1}^{j} i \cdot 2^{i-1}\right)$$

$$= \frac{n+1}{n}\log_2(n+1) - 1$$

$$\approx \log_2(n+1) - 1 \qquad (n > 50)$$

- Pros: the average and maximum search length is in the same order, and the search is very fast
- Cons: need sorting, sequential storage, difficult to update (insertion/deletion)

Ideas of the Blocking Search

- "Ordering between blocks"
  - Assume that the linear list contains n data element, split it into b blocks
  - The maximum element in any block must be smaller than the minimum element in the next block
  - Keys of elements are not always ordered in one block
- Tradeoff between sequential and binary searches
  - Not only fast
  - But also enables flexible update

Blocking Search – Index Sequential Structure

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
22	12	13	9	8		33	42	44	24	48		60	80	74	49	86	53

- Link: starting position of a block
- Key: Maximum key value in the block
- Count: #elements in a block

link:	0	6	12
Key:	22	48	86
count:	5	5	6

Performance Analysis of Blocking Search

- Blocking search is a two-level search
  - First, find the block where the specific element stays at, with ASL_b
  - Second, find the specific element inside that block, with ASL_w

$$ASL = ASL_b + ASL_w$$

$$\approx \log_2 (b+1)-1 + (s+1)/2$$

$$\approx \log_2(1+n/s) + s/2$$

Performance Analysis of Blocking Search

- If we use sequential search in both the index table and the blocks

$$\mathrm{ASL}_b = \frac{b+1}{2} \qquad \mathrm{ASL}_w = \frac{s+1}{2}$$

$$\mathrm{ASL} = \frac{b+1}{2} + \frac{s+1}{2} = \frac{b+s}{2} + 1$$

$$= \frac{n+s^2}{2s} + 1$$

- When s = $\sqrt{n}$, we obtain the minimum ASL:

$$\mathrm{ASL} = \sqrt{n} + 1 \approx \sqrt{n}$$

Performance Analysis of Blocking Search

- When n=10,000,
  - Sequential search takes 5,000 comparisons
  - Binary search takes 14 comparisons
  - Block search takes 100 comparisons

Pros & Cons of Blocking Search

- Pros:
  - Easy to insert and delete
  - Few movement of records
- Cons:
  - Space of a auxiliary array is needed
  - The blocks need to be sorted at the beginning
  - When a large number of insertion/deletion are done, or nodes are distributed unevenly, the efficiency will decrease.

Thinking

- Try comparing the sequential search witch binary search in terms of advantages and disadvantages.

- What are the application scenes of these search methods respectively?

Data Structures and Algorithms

Thanks

the National Elaborate Course (Only available for IPs in China)
http://www.jpk.pku.edu.cn/pkujpk/course/sjjg/

Ming Zhang, Tengjiao Wang and Haiyan Zhao

Higher Education Press, 2008.6 (awarded as the "Eleventh Five-Year" national planning textbook)