算法设计与分析
Design and Analysis of Algorithms

主讲人 徐云

Fall 2018, USTC

第8章 线性时间的排序

8.1 排序的下界

8.2 计数排序

8.3 基数排序

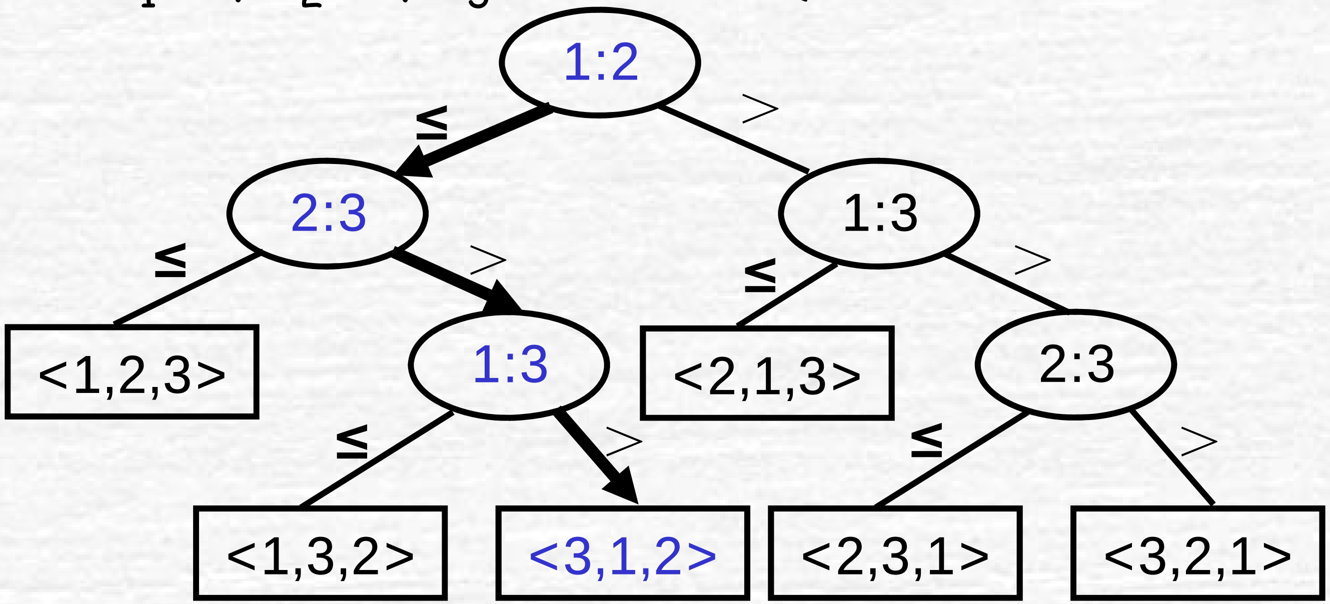
8.4 桶排序

8.1 排序的下界

- 判定树模型
- 最坏情形的下界

判定树模型

- 任何n个元素$a_1,a_2,\ldots,a_n$的基于比较的排序可以用一棵二叉排序树表示，其叶子节点代表一类输入实例的排序结果。
- 示例：$a_1=6, a_2=8, a_3=5$其判定树如下



该实例的比较路径由箭头所示，叶子节点代表的是排序结果。

最坏情形的下界

- 定理8.1 基于比较排序算法在最坏情形下，需要$\Omega(n\log n)$。
- 证明：设h, l分别代表判定树的高度和叶子数

  ∵ 判定树是一棵二叉树

  ∴ $l \le 2^h$ //叶子数不超过2^h

  ∴ $n! \le l \le 2^h$ //$n!$为排序的结果数

  于是有，$h \ge \log n! \ge \Omega(n\log n)$ 证毕！

第8章 线性时间的排序

8.2 计数排序

- 基本思想
- 一种特殊情形的计数排序
- 一般情形的计数排序

基本思想

- 统计小于或等于A[i]的元素数目，将A[i]置入相应的位置，即

  A[i]→B[小于或等于A[i]的元素数目]

- 主要要解决的问题：
  - 计数：统计小于或等于A[i]的元素数目
  - 值相同元素的处理

一种特殊情形的计数排序

- 问题：n个互不相同的整数A[1..n]，1≤A[i]≤n

  i=1~n

- 排序算法：

```
SpecialCountingSort(A,B)
{//B[1..n]为排序结果
    for i←1 to n do
        B[A[i]]←A[i]; //如A[i]=5，就放到B[5]中
}
```

时间：O(n)，无比较

一般情形的计数排序（1）

- 问题：n个可以相同的整数A[1..n]，1≤A[i]≤k，i=1~n，这里k是A[i]的取值范围，不一定为n。
- 基本思想：

  步骤：A[1..n]→计数器C[1..k]→B[1..n]

  - Step 1(值相同元素的计数)：将A中的值为i的元素个数记入C[i]中；
  - Step 2(累计计数)：对C[1..k]进行修改，使得C[i]的值表示为≤i的元素个数；
  - Step 3(放置)：将A[j]依据C[A[j]]，放入正确位置B[C[A[j]]]上，并修改C[A[j]] ← C[A[j]]-1；

一般情形的计数排序（2）

- 算法：

```
CountingSort(A,B,k)
{//B[1..n]为排序结果，C[1..k]为计数数组
    for i←1 to k do  C[i]←0;
    for j←1 to length[A] do    //扫描A，值相同元素计数
      C[A[j]]++;
    for i←2 to k do             //C[i]修改，累计计数
      C[i]←C[i]+C[i-1];
    for j←length[A] downto 1 do
    {  B[C[A[j]]]←A[j];
       C[A[j]]--;
    }
}
```

时间：$T(n,k)=\Theta(n+k)=\Theta(n)$，如果$k=\Theta(n)$时

第8章 线性时间的排序

- 8.1 排序的下界
- 8.2 计数排序
- 8.3 基数排序
- 8.4 桶排序

8.3 基数排序

- 算法
- 一些讨论

基数排序算法

- 假定A[1..n]是非负整数，用k进制表示为不超过d位数。
- 算法：

```
RadixSort(A,d)
{   for i←1 to d do    由低位到高位
        使用稳定的排序算法对A的第i位排序；//如计数排序
}
```

- 时间：$T(n)=\Theta(d(n+k))$ //k为基，d为位数
  $=\Theta(n)$ //如果$k=\Theta(n)$且d是常数
- 示例：Fig. 8.3

一些讨论

- d不为常数，基数排序算法还是线性时间吗？

  设n个整数的取值范围是$0\sim n^c$，c是整常数，$c\geq1$

  对于十进制整数，n^c需要的位数$d=\lfloor\log_{10}n^c\rfloor+1\approx\log_{10}n$

  $\therefore$ $T(n)=\Theta(d(n+k))=\Theta(n\log n)$ //k为10

  因此，不是线性时间排序

- 算法何时为线性时间？
  - Idea: 只要使d变为常数，k变大到与n同阶
  - How to do:

    选基k=n，则n^c的位数为$\log_n n^c=c=d$

    $\therefore$ d=c， k=n

    $\therefore$ $T(n)=\Theta(n)$

第8章 线性时间的排序

8.4 桶排序

- 基本思想
- 示例
- 算法描述

基本思想和示例

- 假定：输入是均匀分布在[0,1)上的实数。
- 基本思想：

① [0,1)划分为[0,1/n),[1/n,2/n),…,[k/n,(k+1)/n),…,[(n-1)/n,1)n个大小相等的子区间，每个子区间看作一个桶；

② 将n个元素分配到桶中；

③ 对每个桶里的元素进行排序，依次连接桶；

- 示例：Fig. 8.4

桶排序算法（1）

- 输入：$0 \le A[1..n] < 1$
- 辅助数组：$B[0..n-1]$是一个指针数组，指向每个桶(链表)
- 关键字映射：

由于$0 \le A[i] < 1$，必须将$A[i]$映射到$0,1,\ldots,n-1$上

$\because$ $[0,1) \rightarrow [0,n)$　//通过函数$nA[i]$

即 $k \le nA[i] < k+1$　//存在k

$\therefore$ 桶号$k = \lfloor nA[i] \rfloor$　//映射函数

桶排序算法（2）

- 算法描述：

```
BucketSort(A)
{  n←length[A];                                          Time
   for i←1 to n do        //扫描A                        θ (n)
      将A[i]插入到链表B[⌊nA[i]⌋]中;
   for i←0 to n-1 do                                     θ (n)*
      用插入排序将B[i]排序;
   将B[0], B[1],…, B[n-1]连接起来;                       θ (n)
}
```

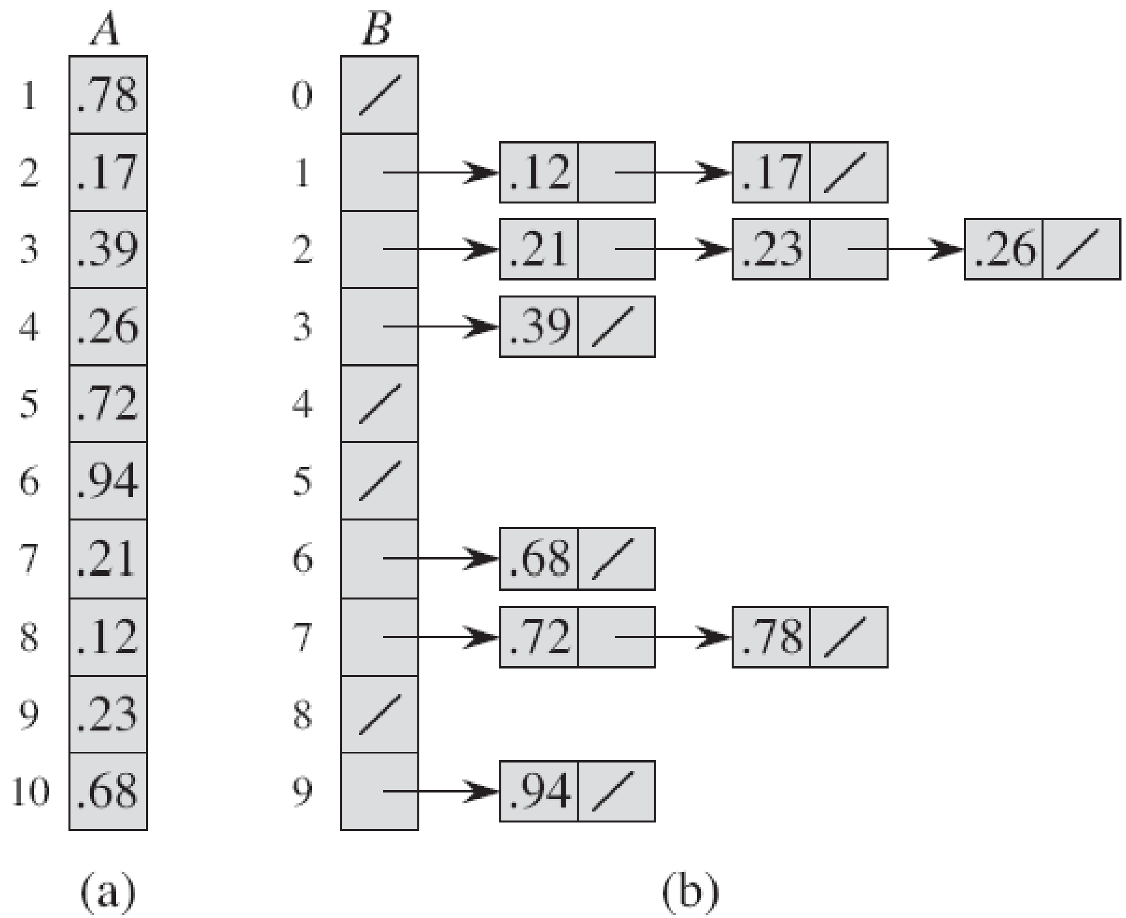
*注：∵ n个数是均匀分布在[0,1)中，

∴ 每个桶中大约只有一个数， ∴ 时间为O(1)

- 期望时间分析：略 θ(n)

桶排序算法（3）

- 示例：


A
B
1 .78
2 .17
3 .39
4 .26
5 .72
6 .94
7 .21
8 .12
9 .23
10 .68
0
1 .12 .17
2 .21 .23 .26
3 .39
4
5
6 .68
7 .72 .78
8
9 .94
(a)
(b)

End of Chap8