哈希表相关问题

2017211123 褚逸豪

2018 年 5 月 20 日

1到3节均基于假设：数据是按照原顺序插入散列表的。

1 计算H(x)

源关键字	H(x)	源关键字	H(x)
Jan	4	Jul	4
Feb	2	Aug	0
Mar	6	Sep	9
Apr	0	Oct	7
May	6	Nov	6
Jun	4	Dec	1

2 使用线性探测法处理冲突

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Apr	Aug	Feb	Dec	Jan	Jun	Mar	May	Jul	Sep	Oct	Nov					

3 使用链地址法处理冲突

0	→Apr	→Aug	⋀
1	→Dec	⋀	
2	→Feb	⋀	
3	⋀		
4	→Jan	→Jun	→Jul⋀
5	⋀		
6	→Mar	→May	→Nov⋀
7	→Oct	⋀	
8	⋀		
9	→Sep	⋀	
10	⋀		
11	⋀		
12	⋀		
13	⋀		
14	⋀		
15	⋀		
16	⋀		

4 平均查找长度

4.1 线性探测法

4.1.1 查找成功

$$S_{nl} \approx \frac{1}{2}\left(1+\frac{1}{1-\frac{12}{17}}\right)=\frac{11}{5}=2.2$$

4.1.2 查找失败

$$U_{nl} \approx \frac{1}{2}\left(1+\frac{1}{\left(1-\frac{12}{17}\right)^2}\right)=\frac{157}{25}=6.28$$

4.2 链地址法

4.2.1 查找成功

$$S_{ne} \approx 1+\frac{\frac{12}{17}}{2}=\frac{23}{17}\approx 1.35$$

4.2.2 查找失败

$$U_{ne} \approx \frac{12}{17}+e^{-\frac{12}{17}}\approx 3.42$$

快速排序最坏复杂度

2017211123 褚逸豪

2018 年 5 月 22 日

1 假设

假设我们要将序列排序为非降序序列，而给定序列是一个有序的降序序列，每次选取下标最小的元素作为参照元素

2 证明

第一层$PARTION-SORT$，参照元素要向右移动$n-1$次，递归到下一层时，右侧将直接退出，而左侧余下n-1个元素等待$PARTION-SORT$，参考元素将移动$n-2$次。以此类推，总共将有$\sum_{i=1}^{n-1} i = \frac{n(n-1)}{2}$次元素移动，因此此种快速排序的复杂度将退化至$O(n^2)$

05/22/18 11:32:46 D:\git-repos\data-structure-homework\10\e30.cpp

```
 1  #include <cstdio>
 2  #include <iostream>
 3  #include <cstring>
 4
 5  using namespace std;
 6  struct node {
 7      int data;
 8      node *nxt;
 9  } *hd, buffer[1007], *cur;
10  node *newnode(int data, node *xt) {
11      cur->data = data;
12      cur->nxt = xt;
13      return cur++;
14  }
15  void init() {
16      cur = buffer + 1;
17      hd = buffer;
18      hd->data = 0x7fffffff + 1;
19      hd->nxt = NULL;
20  }
21  void insert(int data) {
22      node *p = hd;
23      for (; p->nxt && p->data <= data && p->nxt->data <= data; p = p->nxt);
24      p->nxt = newnode(data, p->nxt);
25  }
26  int main() {
27      init();
28      int x;
29      while (~scanf("%d", &x))
30          insert(x);
31      for (node *p = hd->nxt; p; p = p->nxt)
32          printf("%d ", p->data);
33  }
34  /**
35   root ▶ … ╱ git-repos ╱ data-structure-homework ╱ 10 ▶ 2 ▶ g++ e30.cpp -g
36   root ▶ … ╱ git-repos ╱ data-structure-homework ╱ 10 ▶ 2 ▶ ./a.out  ◀  master
37  23 14 3 53 52 3214 243 321 245 2 43 242 52
38  2 3 14 23 43 52 52 53 242 243 245 321 3214
39   root ▶ … ╱ git-repos ╱ data-structure-homework ╱ 10 ▶ 2 ▶          ◀  master
40  /*
```

05/23/18 09:17:05 D:\git-repos\data-structure-homework\10\e31.cpp

```cpp
#include <cstdio>
#include <iostream>
#include <cstring>
using namespace std;
int c[1007], a[1007], rag[1007];
int n;
int main() {
    scanf("%d", &n);
    for (int i = 0; i < n; ++i)
        scanf("%d", a + i);
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
            if (i == j) continue;
            if (a[j] < a[i]) ++c[i];
        }
    }
    for (int i = 0; i < n; ++i)
        rag[c[i]] = a[i];
    for (int i = 0; i < n; ++i)
        printf("%d ", rag[i]);
}
/**
 root ▶ … ﹥ git-repos ﹥ data-structure-homework ﹥ 10 ▶ g++ e31.cpp ◀  master
 root ▶ … ﹥ git-repos ﹥ data-structure-homework ﹥ 10 ▶ ./a.out ◀  master
5
43 23 12 3 65
3 12 23 43 65 #
 root ▶ … ﹥ git-repos ﹥ data-structure-homework ﹥ 10 ▶ ./a.out ◀  master
10
9 8 7 6 5 4 3 2 1 0
0 1 2 3 4 5 6 7 8 9 #
 root ▶ … ﹥ git-repos ﹥ data-structure-homework ﹥ 10 ▶ ◀  master
```

最大值最小值算法

2017211123 褚逸豪

2018 年 5 月 23 日

1 记号

我们将待排序序列记作$a_{1\ldots n}$，约定序列长度不低于2，最小值为A，最大值为B

2 算法

1. 比较a_1与a_2，将较大值赋给B，较小值赋给A，进入步骤2
2. 置计数器i为3，进入步骤3
3. 若$B < a_i$，那么令$B := a_i$并进入步骤5，否则进入步骤4
4. 若$A > a_i$，那么令$A := a_i$，进入步骤5
5. 令计数器i自加1，如果$i > n$，那么进入步骤6，否则进入步骤3
6. 输出A和B，分别为数列最大值和最小值，算法结束

3 算法分析

3.1 比较次数

步骤1进行1次比较，步骤2到6最多进行$2(n-2)$次比较，合计最多进行$2n-3$次比较

3.2 最坏情况

若数列最大值出现在位置1或位置2，那么显然，每次都会进行两次判断，此时达到最坏情况，共进行$2n-3$次比较