《L4-CSP-J 复赛进阶算法与数据结构》

Day1 贪心策略(下)

主讲人：小武老师

可达信奥_小武老师_keda.ac

贪心策略

贪心策略应用：区间覆盖、删数问题、霍夫曼编码

贪心策略

你只能拿5张，怎么才能拿最多的面额？显然每次选择剩下钞票中面值最大的一张，最后你的选择一定是最优的。

贪心策略

基本思路

把求解的问题分成若干个子问题

↓

对每个子问题求解，得到子问题的局部最优解

↓

把子问题的最优解合成原问题的全局最优解

适用问题

前提：局部最优策略能导致产生全局最优解

区间覆盖

假设有n个区间，分别是：$[l1,r1],[l2,r2],[l3,r3],\cdots[ln,rn]$

从这n个区间中选出某些区间，要求这些区间满足两两不相交（端点相交的情况不算相交），最多能选出多少个区间呢？（这个问题有很多应用：如参加活动、任务调度、教师排课等)。最多不相交区间如下图所示：

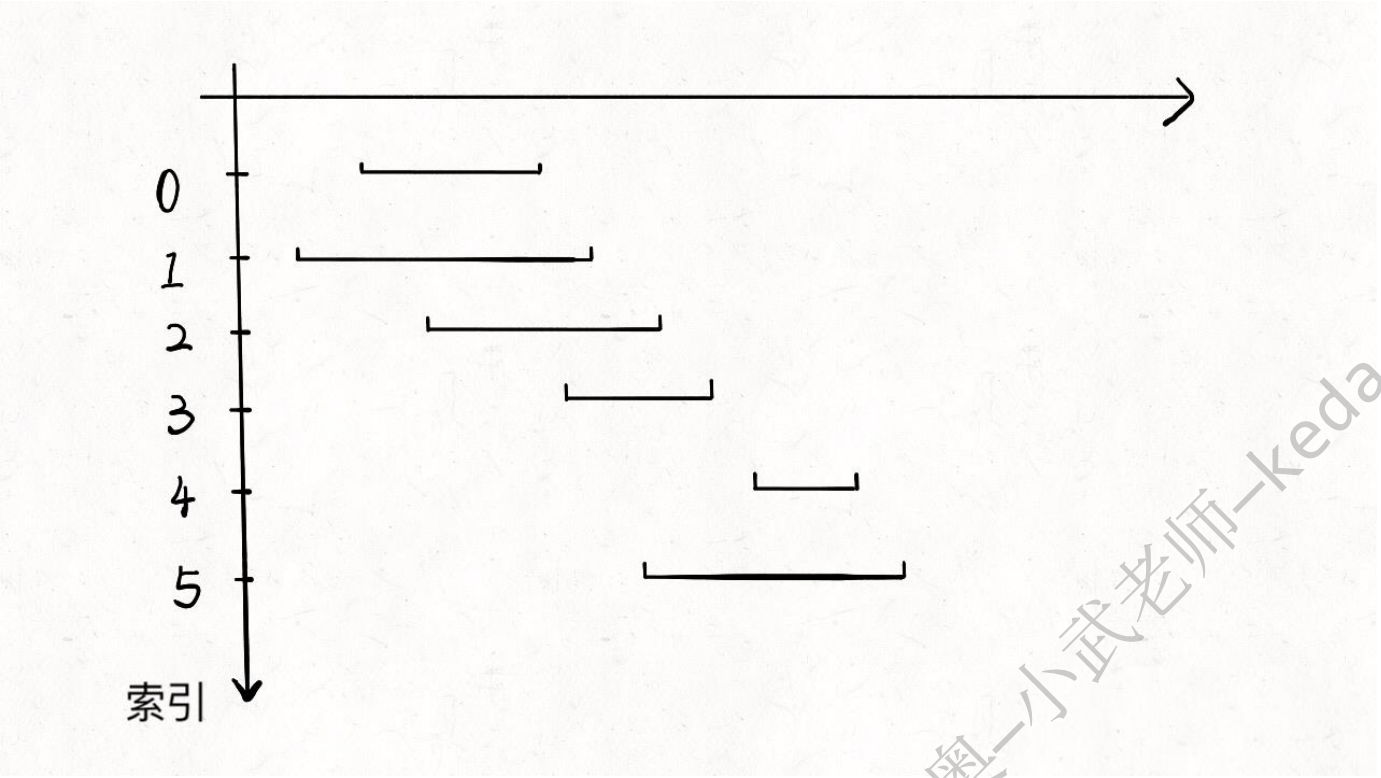
区间：$[6,8],[2,4],[3,5],[1,5],[5,9],[8,10]$

不相交区间：$[2,4],[6,8],[8,10]$

分析：这个问题就相当于选择几个不相交的区间，从左到右将[l_min，r_max]覆盖、可以按照右端点从小到大的顺序对这n个区间排序。每次选择左端点与前面的已经覆盖的区间不重合而右端点又尽量小的区间，这样可以让剩下的未覆盖的区间尽可能的大，就可以放置更多的区间。

区间覆盖

算出这些区间中最多有几个互不相交的区间



三个步骤：

1、按照右端点从小到大排序

2、找到第一个结束最早的区间

3、删除与x相交的区间，重复1-3

所有与 x 相交的区间必然会与 x 的 end 相交；如果一个区间不想与 x 的 end 相交，它的 start 必须要大于（或等于）x 的 end

区间覆盖

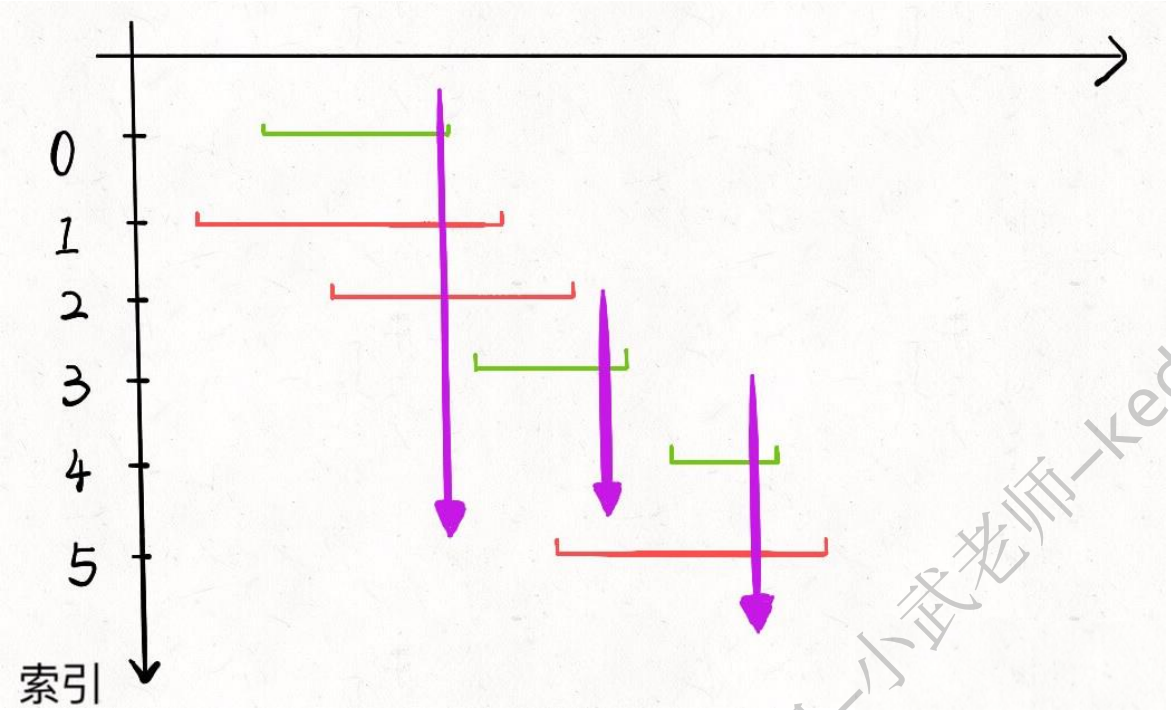
Task 2:用最少的箭头射爆气球

三个步骤：

1、按照右端点从小到大排序

2、找到第一个结束最早的区间

3、删除与x相交的区间，重复1-3



这个问题和区间调度算法一模一样！如果最多有 n 个不重叠的区间，那么就至少需要 n 个箭头穿透所有区间

Task 2:用最少的箭头射爆气球

输入：points = [[10,16],[2,8],[1,6],[7,12]]
输出：2

解释：对于该样例，x = 6 可以射爆 [2,8],[1,6] 两个气球，以及 x = 11 射爆另外两个气球

输入： points = [[1,2],[3,4],[5,6],[7,8]]
输出： 4

输入： points = [[1,2],[2,3],[3,4],[4,5]]
输出： 2

编程实践 Online Judge

P0385 P0393

可达信奥_小武老师_keda.ac

区间覆盖

P0385. 整数区间

```
#include <bits/stdc++.h>
using namespace std;
const int maxn = 10001;
struct range{
    int left;
    int right;
}a[maxn];
bool cmp(range a, range b) {
    return a.right < b.right;
}
```

```
int main() {
    int n, cnt, x_end = 0;
    cin >> n;
    for (int i = 1; i <= n; i++)
        cin >> a[i].left >> a[i].right;
    sort(a + 1, a + 1 + n, cmp);
    x_end = a[1].right;
    cnt = 1;  //至少有一个区间不相交
    for (int i = 2; i <= n; i++) {
        if (a[i].left > x_end) {
            cnt++;
            x_end = a[i].right;
        }
    }
    cout << cnt;
    return 0;
}
```

区间覆盖

P0393. 书架

```cpp
#include <bits/stdc++.h>
using namespace std;
const int maxn = 20001;
int a[maxn];
int main() {
    int n, cnt = 0;
    long long b;
    long sum = 0;
    cin >> n >> b;
    for(int i = 1; i <= n; i++) cin >> a[i];
    sort(a+1, a+n+1);
    while(sum < b){
        cnt++;
        sum += a[n-cnt+1];
    }
    cout << cnt;
    return 0;
}
```

编程实践 Online Judge

P0382

删数问题

在一个非负整数a中，我们希望从中一出k个数字，让剩下的数字值最小，如何选择移除哪k个数字呢？

$$2315678 \rightarrow 3 \rightarrow 215678$$

$$215678 \rightarrow 2 \rightarrow 15678$$

$$15678 \rightarrow k = 4 \rightarrow 156]$$

分析：给定n个数，组成一个最小的数，我们希望高位的数越小越好。因此从左到右应该是递增的，如果删除1个数，让剩下的数变的最小，那么应该从左往右找到第一个下降（变小的）的数。尝试反证法证明。

AC P0382. 删数问题

删数问题

三个坑：

1、注意高精度及字符结束标识

2、注意是字符

3、删除前导0

```cpp
#include <bits/stdc++.h>
using namespace std;
char s[250];
int main() {
    int k;
    scanf("%s %d", s, &k);
    int len = strlen(s);
    while(k--){
        for(int i = 0; i < len; i++){
            if(s[i] > s[i+1]){
                for(int j = i; j < len; j++){
                    s[j] = s[j+1];
                }
                break;
            }
        }
        len--;
    }
    int m = 0;
    while(s[m] == '0') m++;
    for(; s[m] != '\0'; m++) cout << s[m];
    return 0;
}
```

哈夫曼编码

最优前缀码

假设有一个文件，包含1000个字符，每个字符占1Byte(1Byte = 8bit)，则存储这1000个字符要 1000Byte =8000bit，有没有更加节省空间的存储方式呢？假如，这1000个字符只包含6种不同的字符，分别是a、b、c、d、e、f。

编码方式1

既然只有6种字符，可以用3个bit来表示一个字符，则存储这1000个字符只需要3000bit。解压时，按照顺序读取即可。

有无更好的方法？如果出现的字符频率不一样呢？

哈夫曼编码

编码方式2

各个字符编码之间不能出现某个编码是另一个编码的前缀，否则解压过程中会有歧义

字符	出现频率	编码	每个字符总的二进制位数
a	450	1	450
b	350	01	700
c	90	001	270
d	60	0001	240
e	30	00001	150
f	20	00000	100

解压时，按照顺序读取，这1000个字符只需要1910bit存储空间。

哈夫曼编码

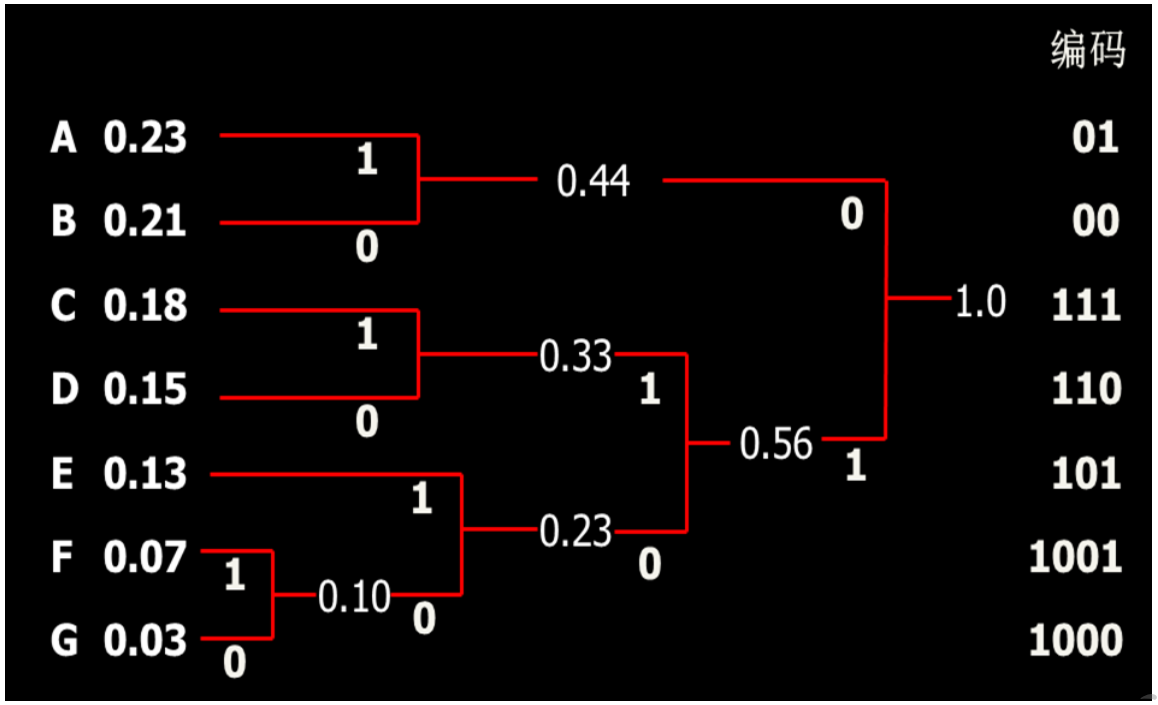
问题：如何根据字符出现的频率，给不同的字符进行不同长度的编码呢?

哈夫曼编码(Huffman Coding)，又称霍夫曼编码，是一种编码方式，哈夫曼编码是可变字长编码(VLC)的一种。 Huffman于1952年提出一种编码方法，该方法完全依据字符出现概率来构造异字头的平均长度最短的码字，有时称之为最佳编码，一般就叫做Huffman编码（有时也称为霍夫曼编码）。

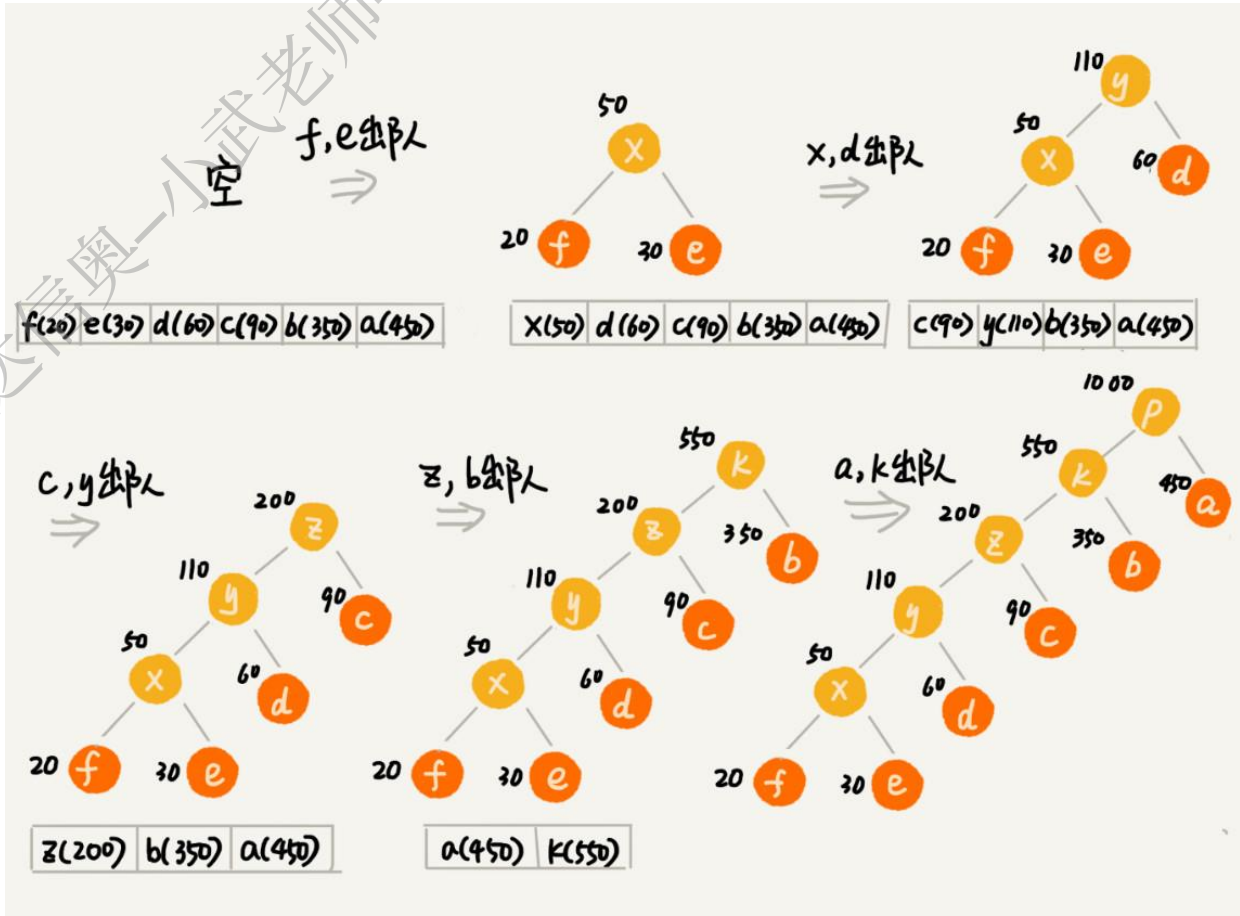
我们可以想一下，假设有一段文本，我们要给里面的每个字符都编码，对于一个出现概率很高的字符，如果我们给它编很长的码元，那么总体编码的平均码长就会变得很长，降低编码效率。你们现在明白为啥大概率字符使用较短码字，小概率字符使用较长码字了吧？就是为了使平均码长尽可能短！所以才说它是一种数据压缩技术嘛。

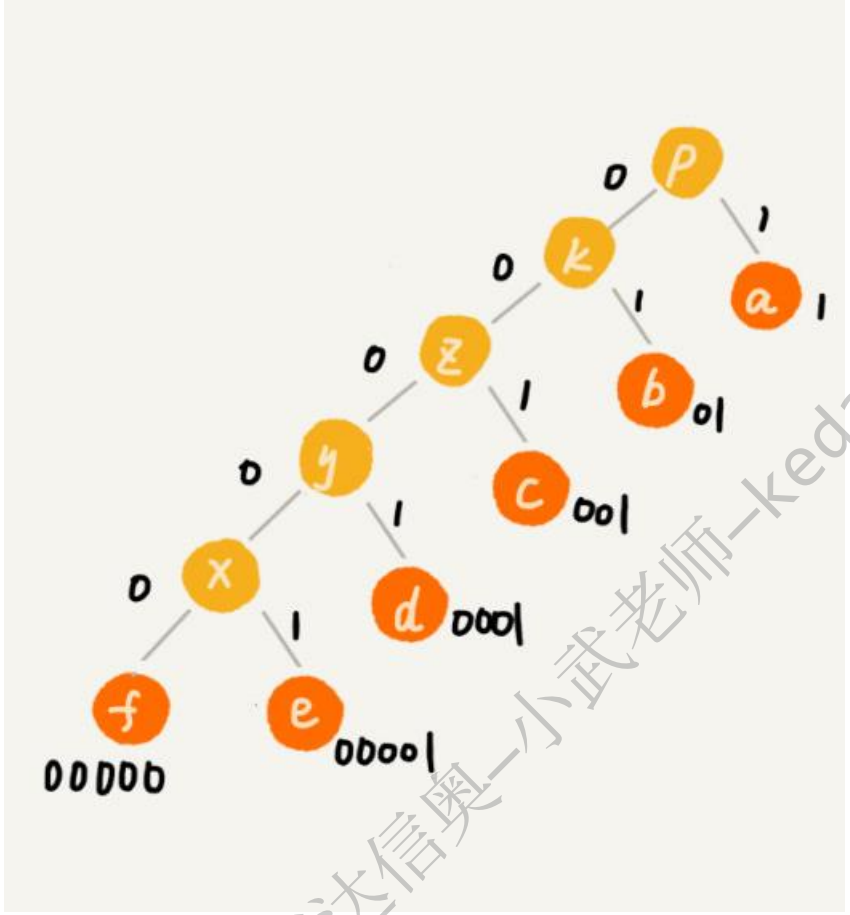
哈夫曼编码

哈夫曼编码(Huffman Coding)，又称霍夫曼编码，是一种编码方式，哈夫曼编码是可变字长编码(VLC)的一种。Huffman于1952年提出一种编码方法，该方法完全依据字符出现概率来构造异字头的平均长度最短的码字，有时称之为最佳编码，一般就叫做Huffman编码（有时也称为霍夫曼编码）。



哈夫曼编码





哈夫曼编码

再看 P0087. 合并果子

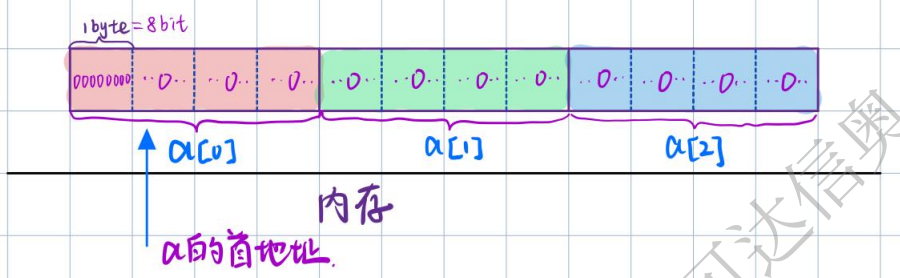
哈夫曼编码

memset 用法

sizeof(a)：返回a的字节数
int n; sizeof(n) → 4
int a[100]; sizeof(a) → 400

memset(a, 0, sizeof(a));
将内存当中从地址为a的地方开始，把共sizeof(a)个字节的内存空间都设置为0.

int a[3];
memset(a, 0, sizeof(a));
=12byte=96bit
内存设置



哈夫曼编码

再看 P0087. 合并果子

数组模拟队列解法

```cpp
#include<bits/stdc++.h>
using namespace std;
int n, n2 = 0, a1[10010],a2[10010],sum = 0;
int main(){
    cin >> n;
    memset(a1, 127, sizeof(a1));
    memset(a2, 127, sizeof(a2));
    for(int i = 0; i < n; i++) cin >> a1[i];
    sort(a1, a1+n);
    int i = 0, j = 0, k, w;
    for(k = 1; k < n; k++){
        w = a1[i] < a2[j] ? a1[i++]:a2[j++];//取最小值
        w += a1[i] < a2[j] ? a1[i++]:a2[j++];//取第二小
        a2[n2++] = w; //
        sum += w;
    }
    cout << sum;
    return 0;
}
```

编程实践 Online Judge

做Day02 贪心专题作业

贪心总结

1、贪心不用刻意去记忆贪心策略的原理，多练习才是最有效的学习方法。

2、对于贪心算法解决问题，最难的是将要解决的问题抽象成贪心模型。

3、一般贪心问题的正确性看起来是显而易见的，但要严谨地证明算法能够得到最优解，并不是一件容易的事情。

4、一般来讲，可以多举几个例子，局部验证一下贪心算法的正确性，enough!

5、区间问题、哈夫曼编码思想、单源最短路、最小生成树等

课后习题与实验

Talk is cheap, show me the code !

下节课见啦！