# lesson9 greedy algorithm

## 1.introduction

### optimization problem

根据一系列的constraints和一个optimization function，能够满足constraints的方法称作feasible solutions,feasible solutions中能使得optimization function得到最优值的称作optimal solution

### greedy method

make best desicion at each stage,under some criterion(标准)，每一个decision都不会被改变，所以每一个decision都要保证feasibility

### 适用:

local optimum=global optimum;

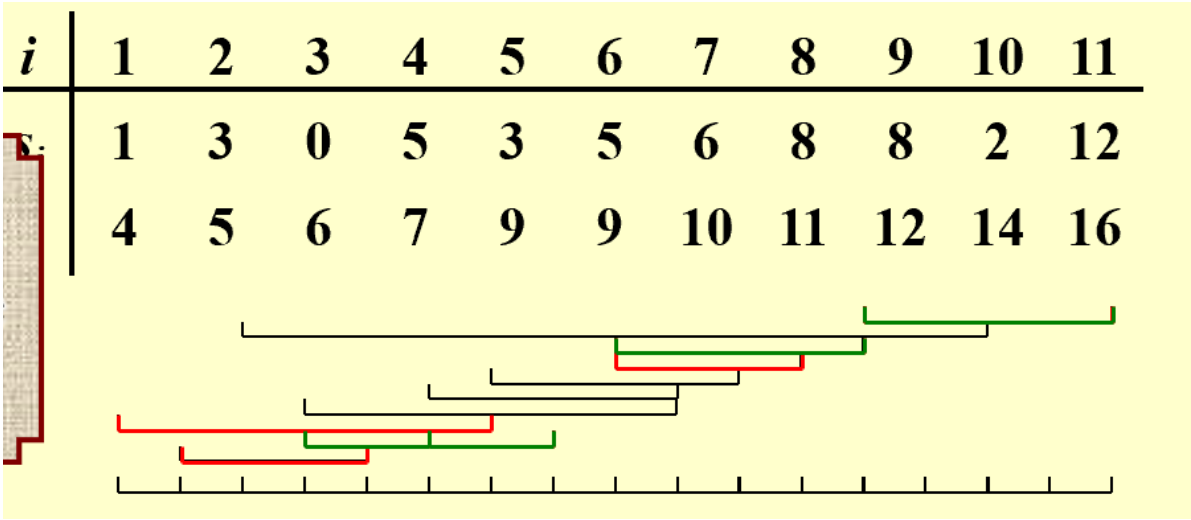
or

not guarantee optimal solutions,but close to the optimal

## 2.eg:activity selection problem

### 1.简介

有{a1,a2...an}个活动，每个活动开始和结束时间为[si,fi)，尽可能多的选择活动，活动之间要相互兼容（时间段不重叠）




### 2.how to be greedy?

method1: DP solution

定义：Sij

()不包括ai,aj，1代表ak ,

A DP Solution $a_1\ a_2\ \dots\ \underbrace{a_i \dots a_k \dots a_j}_{S_{ij}} \dots\ a_n$

$$c_{ij} = \begin{cases} 0 & \text{if } S_{ij} = \Phi \\ \max\limits_{a_k \in S_{ij}} \{ c_{ik} + c_{kj} + 1 \} & \text{if } S_{ij} \neq \Phi \end{cases}$$

$O(N^2)$

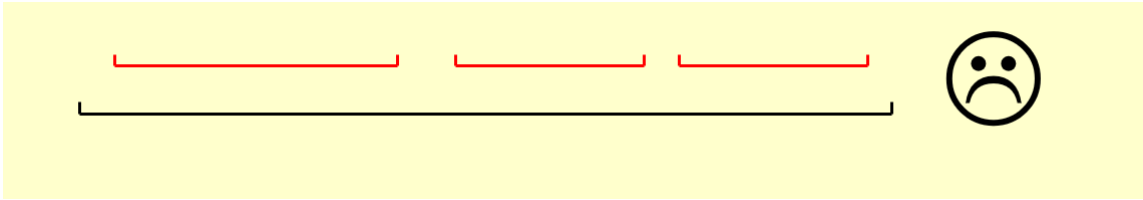
时间复杂度，因为要遍历1~n,k遍历i~j,所以是O(N^2)

method2:

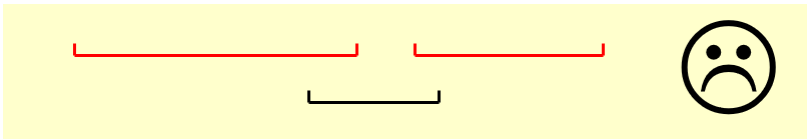
k遍历i~j？？？浪费时间，为什么不能greedy?

贪心策略：

1）找最先开始的?

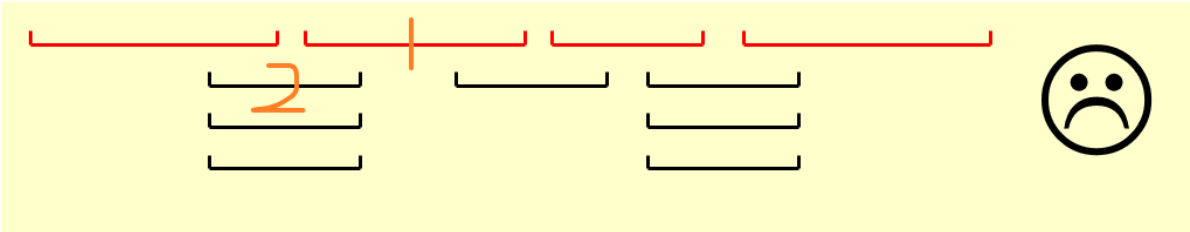


2）每次找最短的?



3）每次找和剩余intervals conflicts最少的?

肯定选1不选2




## 正确：选择最早结束的！！！

## 3.正确性检查

1）该算法不产生over-lapping intervals

\2) 结果是最优的

证明

$S_k$是一个非空子集，$a_m$是$S_k$中结束时间最早的活动，
那么$a_m$一定属于$S_k$中size最大的活动子集(活动相互兼容)

证明：定义$A_k$为最佳解，$a_{ef}$是$A_k$中结束时间最早的活动，

if( $a_m = a_{ef}$ ),得证

else

$a_{ef} = a_m + A_k'$,

$\because f_m \le f_{ef}$ $\therefore A_k'$是另一个最佳解

(递归可证)

因为是tail recusion 所以可以用iteration 证明;

该方法的时间复杂度为O(NlogN) 等同于finish time的排序时间

start latest? yes!!

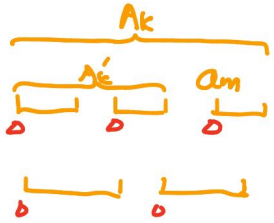
$S_k$是一个非空子集，$a_m$是$S_k$中开始时间最晚的活动，
那么$a_m$一定属于$S_k$中size最大的活动子集(活动相互兼容)

证明：定义$A_k$为最佳解，$a_m$是$A_k$中开始时间最晚的活动，

if( $a_m = a_{ef}$ ),得证

else

$a_{ef} = A_k' + a_m$




$\because S_m > S_{ef}$ $\therefore A_k'$是另一个最佳解

(递归可证)

## 4.再看DP Solution

$$c_{1,j} = \begin{cases} 1 & \text{if } j = 1 \\ \max\{c_{1,j-1}, c_{1,k(j)} + 1\} & \text{if } j > 1 \end{cases}$$

在C i, j-1的基础上，增加了aj,

如果选择aj,前面选择和他兼容的最近的活动；

如果不选择，则保留C 1,j-1；

如果activity 有一个weight ,要求得到最大weight,

$$c_{1,j} = \begin{cases} w_1 & \text{if } j = 1 \\ \max\{ c_{1,j-1}, c_{1,k(j)} + w_j \} & \text{if } j > 1 \end{cases}$$

DP 还是对的；

但是greedy 不一定!!

## 5.elements of greedy strategy

1.将优化问题转换为一个**选择**和一个**子问题**

2.证明**greedy choice**总是对应一个**最优解**

3.证明**子问题的最优解**和**choice**的结合，可以得到**原始问题的最优解**

一个贪心策略，总是对应一个笨拙的DP解

# 3.eg: huffman code

## 1.introduction

for **file compression**

text:

length: 1000

characters：a,u,x,z

1）如果用字节存储 1个字节，8bit ,一共8000bit

2）如果用2bit编码a,u,x,z，需要2000+bit

所以需要 [ logC ] (向上取整） bit来编码一段text ( C是不同的character数量)

高阶： frequency： = number of occurrences of a symbol

用可变长codes来进行编码，高频的character编码长度短

## 2.structure of huffman tree

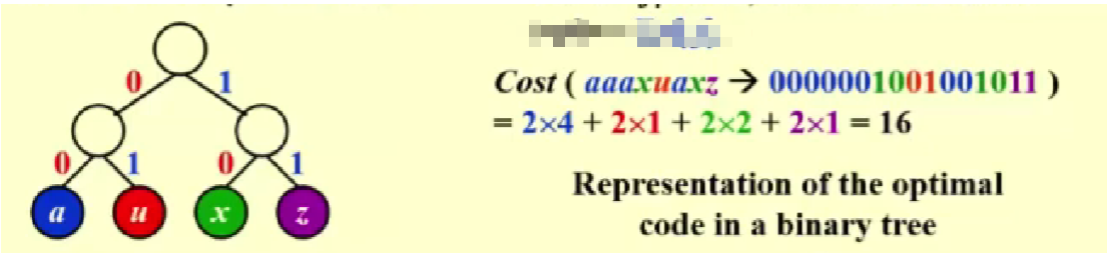
自底向上构建树

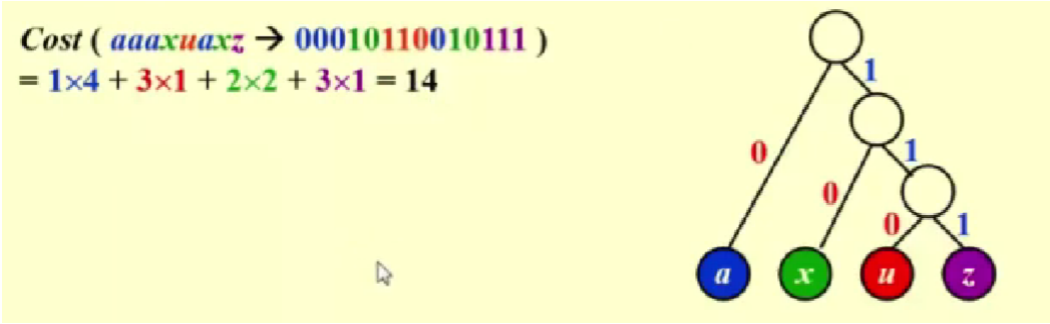
di:depth

fi: frequency

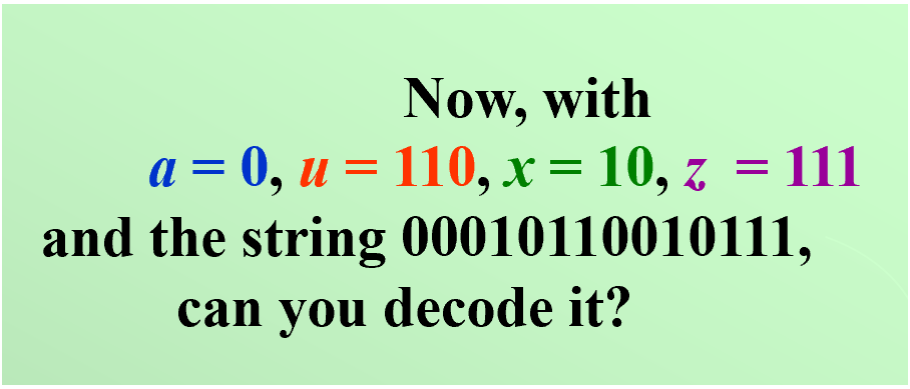
cost of the code = Σdi * fi

树的结构不一样，cost也不一样




解码：




得到：aaaxuaxz

编码规则：没有一个码是另一个码的prefix，也就是说所有的character必须是full tree（要么有两个孩子，要么是叶子结点） 的叶子结点

full tree: 如果存在一个度为1的结点，那么根据最小cost可以删掉这个节点，所以不用存在

这样的code 叫做prefix code;

## how to obtain a full binary tree of min total cost ?

greedy!!

```
void Huffman ( PriorityQueue  heap[ ],  int  C )
{
    consider the C characters as C single node binary trees,
    and initialize them into a min heap;

    for ( i = 1; i < C; i++ ) {
      create a new node;
      /* be greedy here */
      delete root from min heap and attach it to left_child of node;
      delete root from min heap and attach it to right_child of node;
      /*pop两次，返回一个新的结点*/
      weight of node = sum of weights of its children;
      /* weight of a tree = sum of the frequencies of its leaves */
      insert node into min heap;
  }
}
```

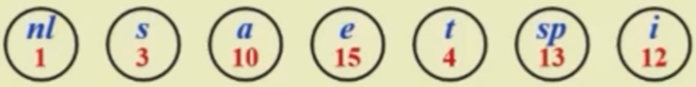
时间复杂度O(ClogC)

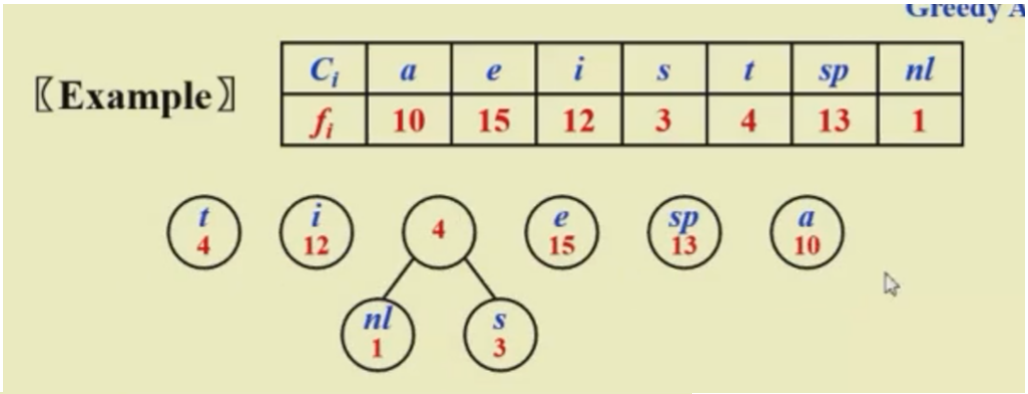
外层遍历C次，内层delete root两次，插入一次，每个操作的时间复杂度都是O（logN)

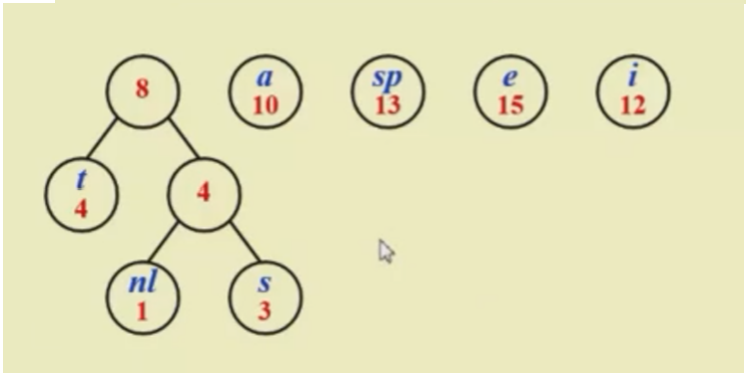
栗子：

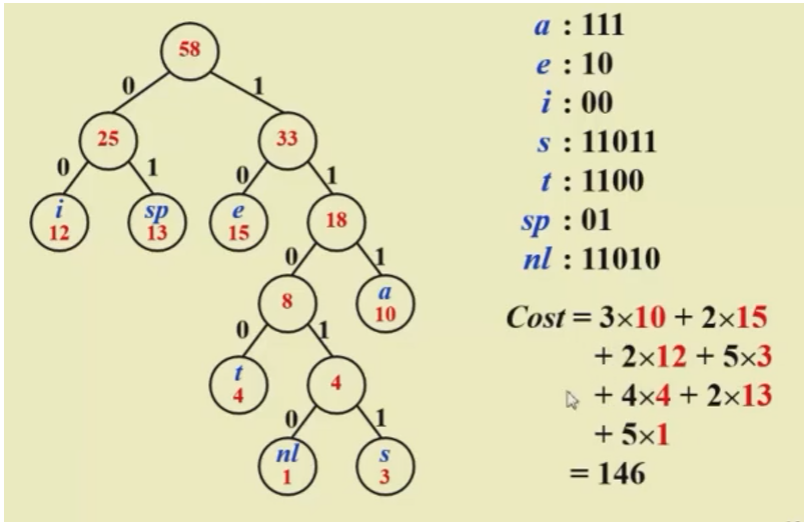
〖Example〗

| $C_i$ | a | e | i | s | t | sp | nl |
|---|---|---|---|---|---|---|---|
| $f_i$ | 10 | 15 | 12 | 3 | 4 | 13 | 1 |





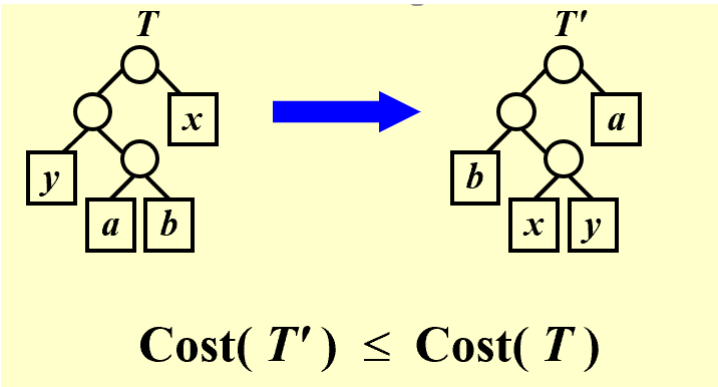




# Proof

## 1.lemma1

Let C be an alphabet in which each character c ∈ C has frequency c.freq. Let x and y be two characters in C having the lowest frequencies. Then there exists an optimal prefix code for C in which the codewords for x and y have the same length and differ only in the last bit.

也就是说x,y是C中frequency最低的，一定存在一颗哈夫曼树T'，x,y为相邻结点，且cost(T') <=cost(T)




## 2.lemma2

如果T'是C'的最优解集，那么T'中一个元素z用x,y替换后的T，是C的最优解集；局部最优解是整体最优解的一个部分