浅析树的划分问题

东北育才学校　　贝小辉

概要

- 树的划分问题：将给定的一棵树划分为若干棵子树，使其能够满足一定的条件或是使得某个特定的函数达到最值。

- 树的最大－最小划分问题。

问题的提出

草莓（ NOI2003 Day 2-2 test6 ～ test9 ）

题目大意：

给出一片草莓中每个草莓的重量以及它们的连接情况。令 sum(i) 表示第 i 块草莓田中所有草莓重量的和 (1≤i≤k) , x = min{ sum(i) | 1≤i≤k } 。你的任务就是要把一块草莓田分割成 k 块，且分割方案需要满足如下的条件：

- 每一块中的草莓必然是直接或者间接的和其他草莓相连接的；
- 这种分割方案所对应的 x 尽可能的大。

最后输出你的分割方案和结果。

问题的提出

这是一道提交答案式的题目，其中 test6 ～ test9 所给的图是一棵树，若不考虑具体的数据情况，我们可以将原问题抽象成如下问题：

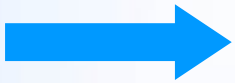
给定一棵树以及树中每个顶点的一个非负权值，将树划分为 k 棵子树，定义： sum(i) 表示第 i 棵子树中所有顶点权值的和， x=min{ sum(i)| 1≤i≤ k }，请求出 x 的最大值并输出一种划分方案。

我们把它称作树的最大－最小划分问题。

算法 1：问题转化

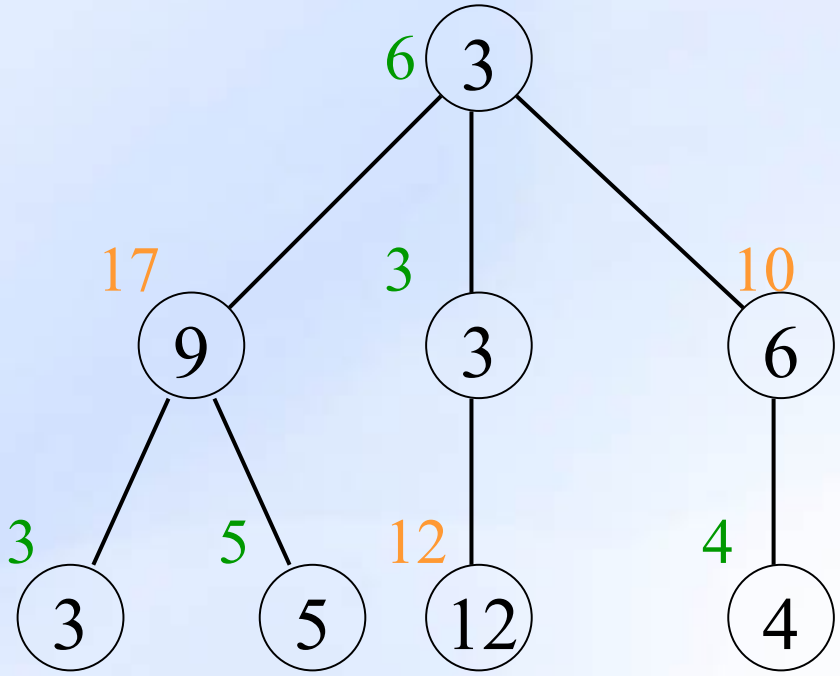
- 考虑新问题：

对于一个确定的下界，最多可将树划分为多少棵子树，使得每棵子树的权值和都不小于此下界？

新问题 + 二分法  原问题

解决新问题

- 新问题的解决只需要一个以贪心思想为基础的扫描算法。



下界：
10

解决原问题

- 时间复杂度： O(N)　　已是理论下界

- 通过二分法来找到最大的下界 x，使得划分的最大子树数目不小于 k， x 即为原问题的解。

小结

- 解决问题的途径：问题转化
- 实现简单，运行效果好

Perfect ?

- 运行时间依赖于节点的权值范围
- 若节点的权值范围很大或者权值是小数甚至无理数……

时间复杂度不依赖于节点权值范围的算法?

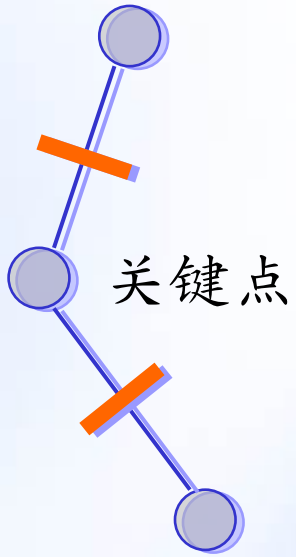
新思路： 割

- 一条边所连接的两个顶点分属不同的子树，则称在这条边上有一个“割”。
- 每个割对应一棵子树 + 根节点所在子树
- 划分 k 棵子树

将 k-1 个割分配到 k-1 个不同的边上

新思路：　移动

- 一次移动被定义为将一个割从一条边移到一条与它相邻的边上，并且保证新的边一定是在原来那条边的下一层。



新算法：　初始状态　＋　移动规则

初始状态

最简单的方法：

- 任选一个度为 1 的顶点为根
- 将所有割都放在与根相连的唯一的边上

可以由初始状态到达任何一个目标状态

关键： 移动规则的制定

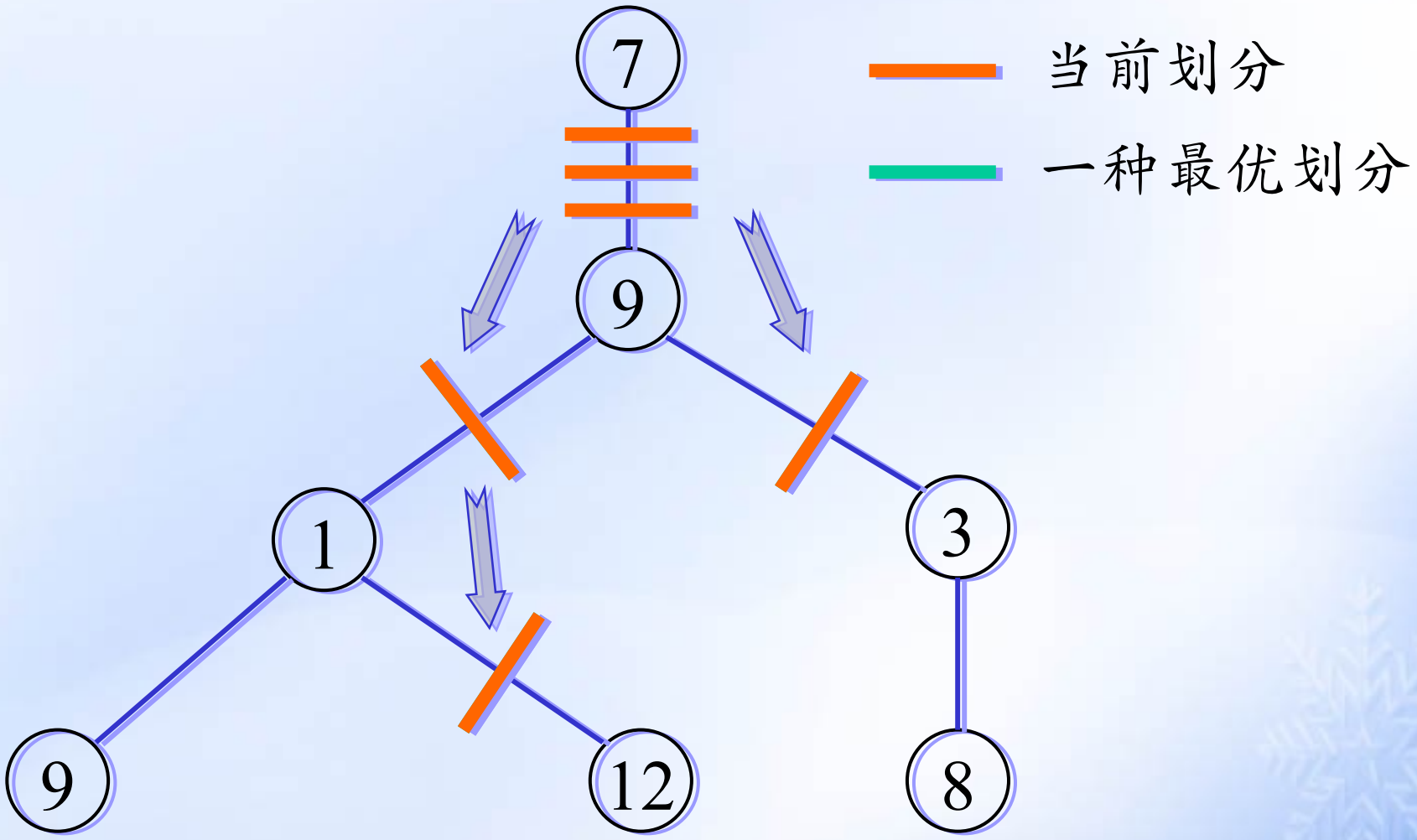
移动规则

- 依据的还是一种贪心的思想：

1. 计算出当前状态下子树权值和的最小值 Wmin
2. 考虑所有可能的移动，找出能使移动后的割所对应的子树权值和 Wnow 最大的那种移动
3. 如果 Wnow ≥ Wmin，那么进行这步移动，并转到步骤 1
4. 算法结束， Wmin 即为所求的最大的最小值，当前划分即为一种最优划分

证明？

例子



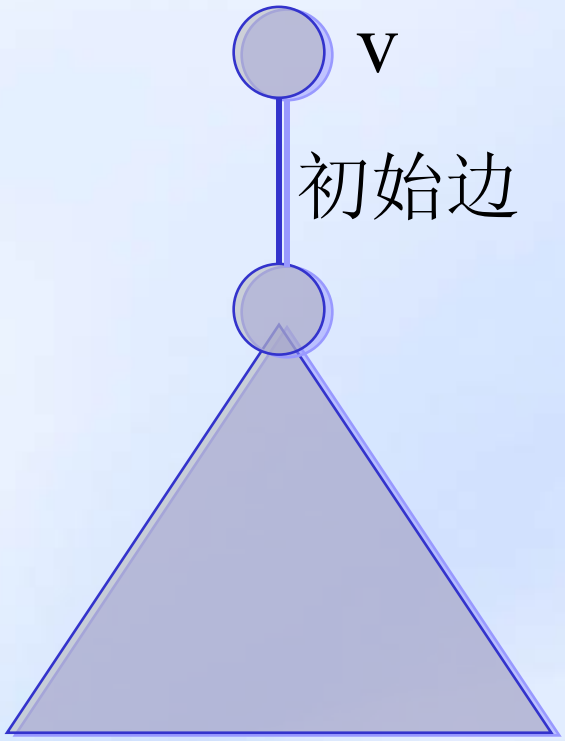
“上方”

- 当前划分总是在某个最优划分的“上方”
- “上方”的定义

划分 A 在划分 A’ 的上方，也就是存在一种 A 的割和 A’ 的割的一一对应，使得每个 A 的割都在它所对应的 A’ 的割的上方。

更加实用的性质 ？

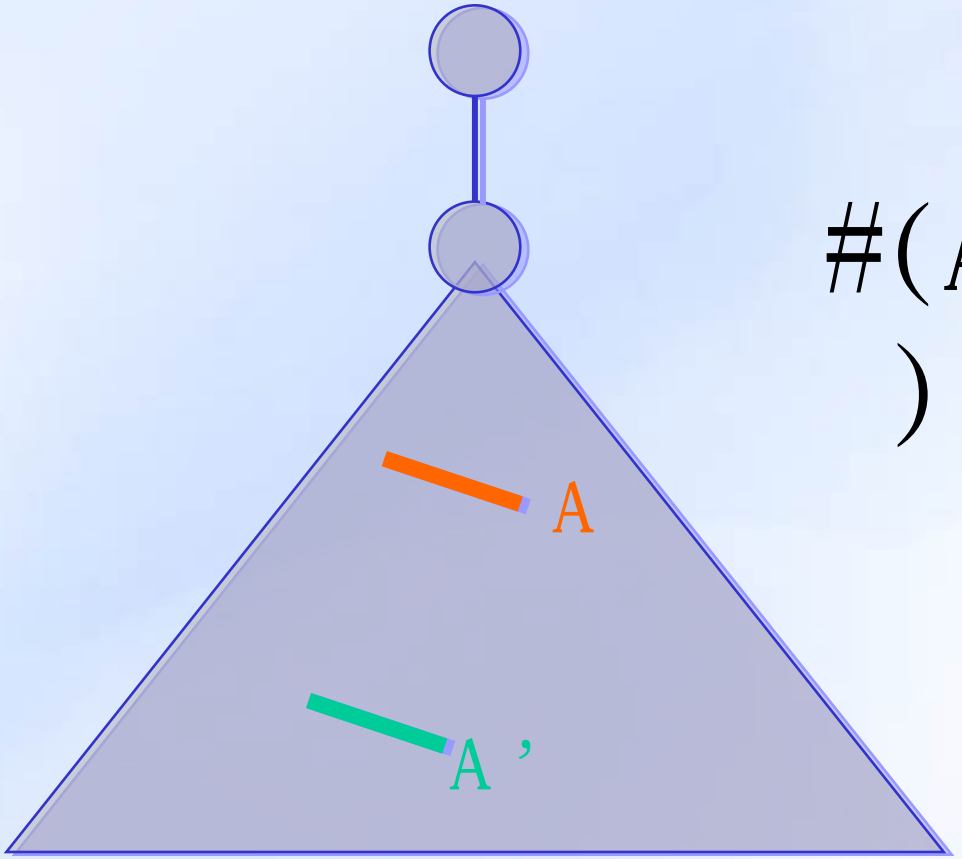
定义：部分子树



- 若一棵树 T 的子树 T’ 包含了顶点 v 连同 v 的某一个儿子以及这个儿子的所有后继，则称 T’ 是 T 在顶点 v 处的一棵部分子树。与 v 相连的唯一一条边被称为 T’ 的初始边。

重要性质

- 划分 A 在划分 A’ 上方



$$\#(A) \leqslant \#(A')$$

证明算法

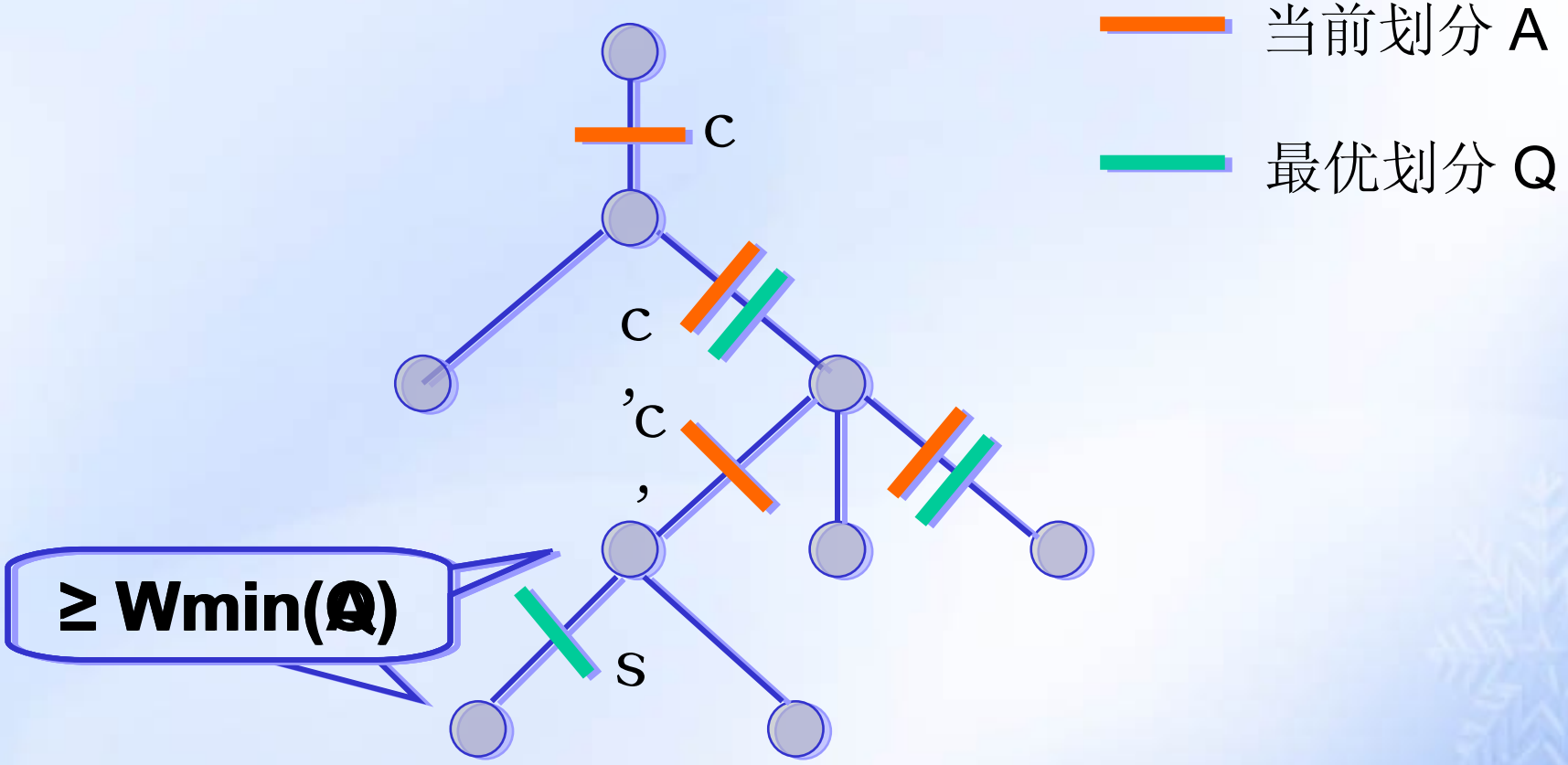
- (1)　在初始状态时的划分 A 是在任何一个最优划分 Q 的上方的。
- (2)　若存在一个最优划分 Q 使得当前的划分 A 是在 Q 的上方，且 A 和 Q 不相等，则算法一定不会终止。
- (3)　设 A 在 Q 的上方且 A 不等于 Q，在算法进行一步后 A 变为 A’，我们一定还能找到一个最优划分 Q’ 使得 A’ 在 Q’ 上方。
- (4)　算法会在有限步内终止，算法终止时的划分一定是一个最优划分。

一些说明

- 字母 A 表示由算法进行而得到的划分。
- 字母 Q 表示一个最优划分，即使得最小子树最大的划分。
- 用 Wmin(A) 表示在划分 A 下的最小子树的权值和
- 对于任意划分 A， $Wmin(A) \leq Wmin(Q)$

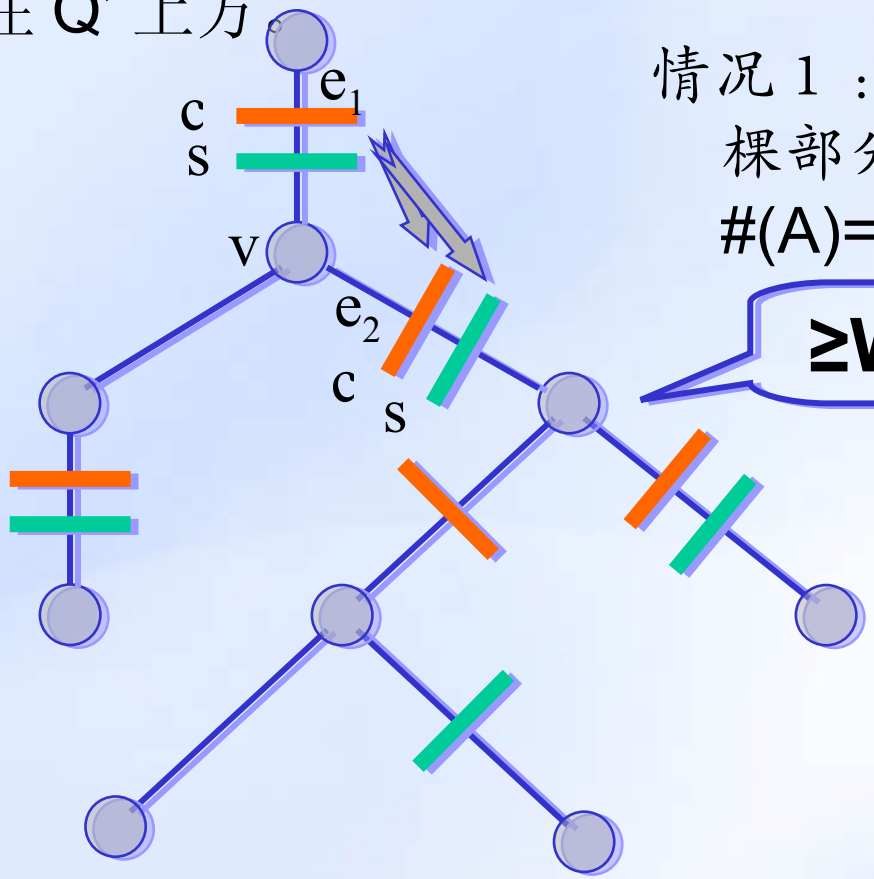
证明算法 (2)

- (2) 若存在一个最优划分 Q 使得当前的划分 A 是在 Q 的上方，且 A 和 Q 不相等，则算法一定不会终止。



证明算法 (3)

- (3) 设 A 在 Q 的上方且 A 不等于 Q，在算法进行一步后 A 变为 A'，我们一定还能找到一个最优划分 Q' 使得 A' 在 Q' 上方。



情况 1：对于在顶点 v 处的每一棵部分子树 T'，都有 #(A)=#(Q)。

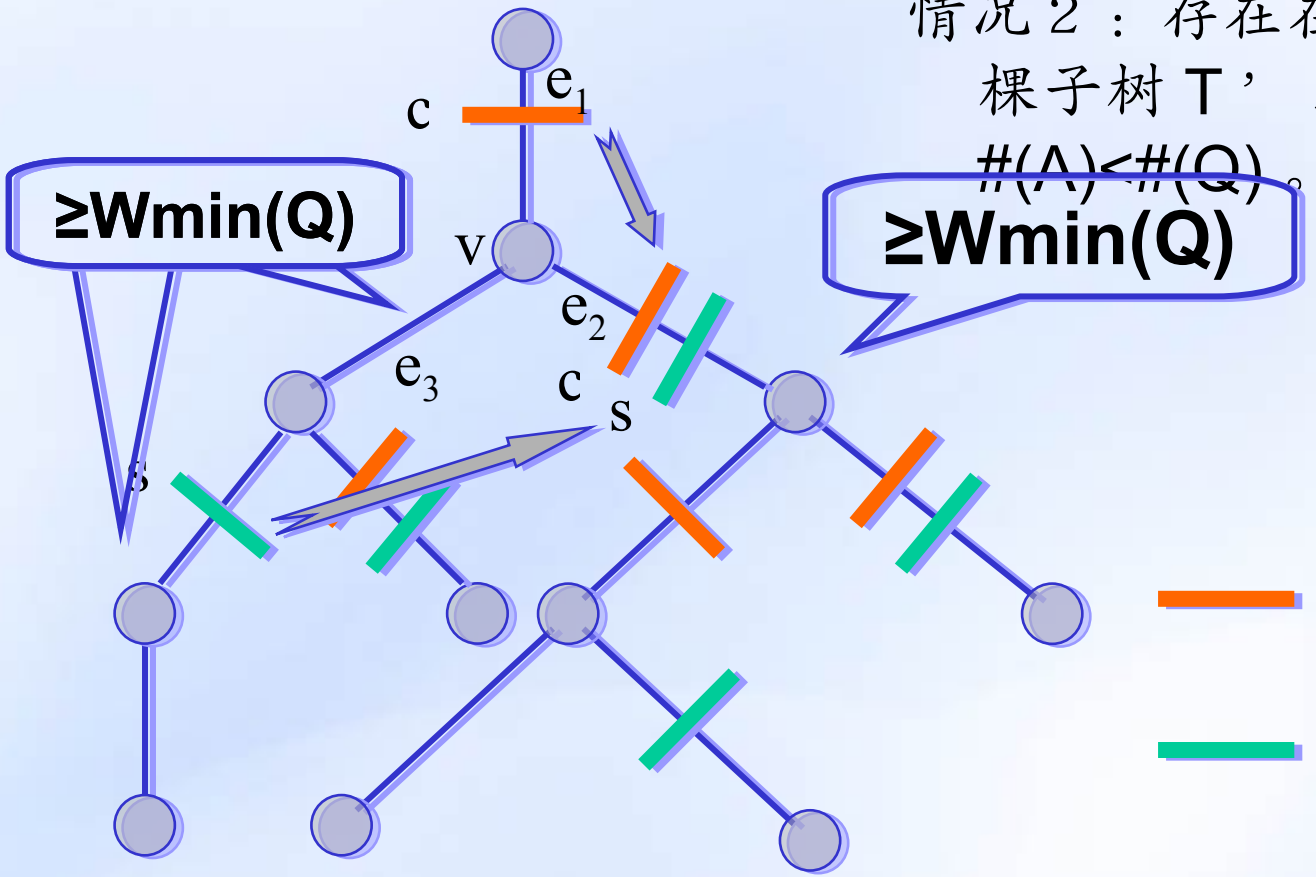
≥Wmin(Q)

当前划分 A

最优划分 Q

证明算法 (3)

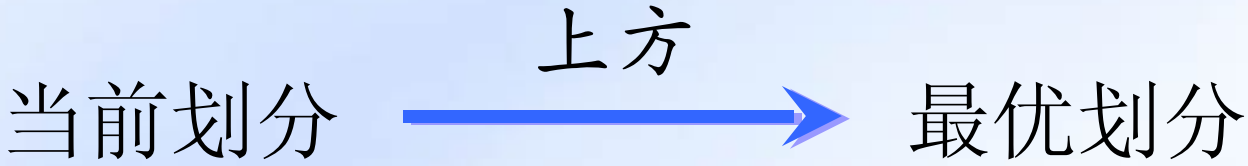
情况２：存在在顶点 v 处的某棵子树 T’，使得 #(A)<#(Q)。



呼，终于证完了……

小结

- 新思想，新方向：割，移动
- 贯穿整个证明过程的思想：上方



“序”的概念

算法的扩展

权函数的扩展

子树中所有节点的权值之和

子树中节点权值最大值？

子树半径？（树的 P 中心问题）

…… ？

权函数的必要条件：

若 T' 是 T 的任意一棵子树，则 W(T) ≥ W(T')，其中 W(T) 表示树 T 的权函数

总结

- 两个算法，它们通过不同的方式思考问题，从而得到了不同的解法

  算法 1

  问题转化

  二分法

  算法 2

  割，移动

  “ 上方”的概念

总结

- 从不同的角度看问题
- 深入思考问题
- 深刻了解问题本质
- 设计出符合题目特点的优秀算法

Thank you all !

时间复杂度

- 只是单纯的按照算法的流程去做时间效率很低
- 需要对算法进行一些必要的优化使得它能够高效的完成流程中的每一步操作

- $O(k^2\ rd(T) + kn)$ 其中 rd(T) 是树 T 的半径