第二部分

树形结构应用

树形结构

- 树形结构：
  - 树、森林
  - 二叉树（满二叉树、完全二叉树、扩充二叉树）
  - K叉树
- 优先队列：
  - 堆、左高树、小大根交替堆
- 竞赛树
- 搜索树：
  - 二叉搜索树、AVL搜索树、B-树、红黑树、伸展树、Treap结构、N-ary Trie、后缀树
- Huffman树

外排序

- 如果数据存放在外存文件中，则需要考虑外存特点，采用外存文件排序技术，简称外排序(external sort)。
- 需要根据内存的大小，将外存中的数据文件划分成若干段，每次把其中一段读入内存并用内排序方法进行排序。这些已排序的段或有序的子文件称为顺串或归并段(run)。

外部排序

- 外部排序法(external sorting method)一般包括两步：
- 1、产生部分排序结果顺串；
- 2、将这些顺串合并在一起得到最终的顺串。

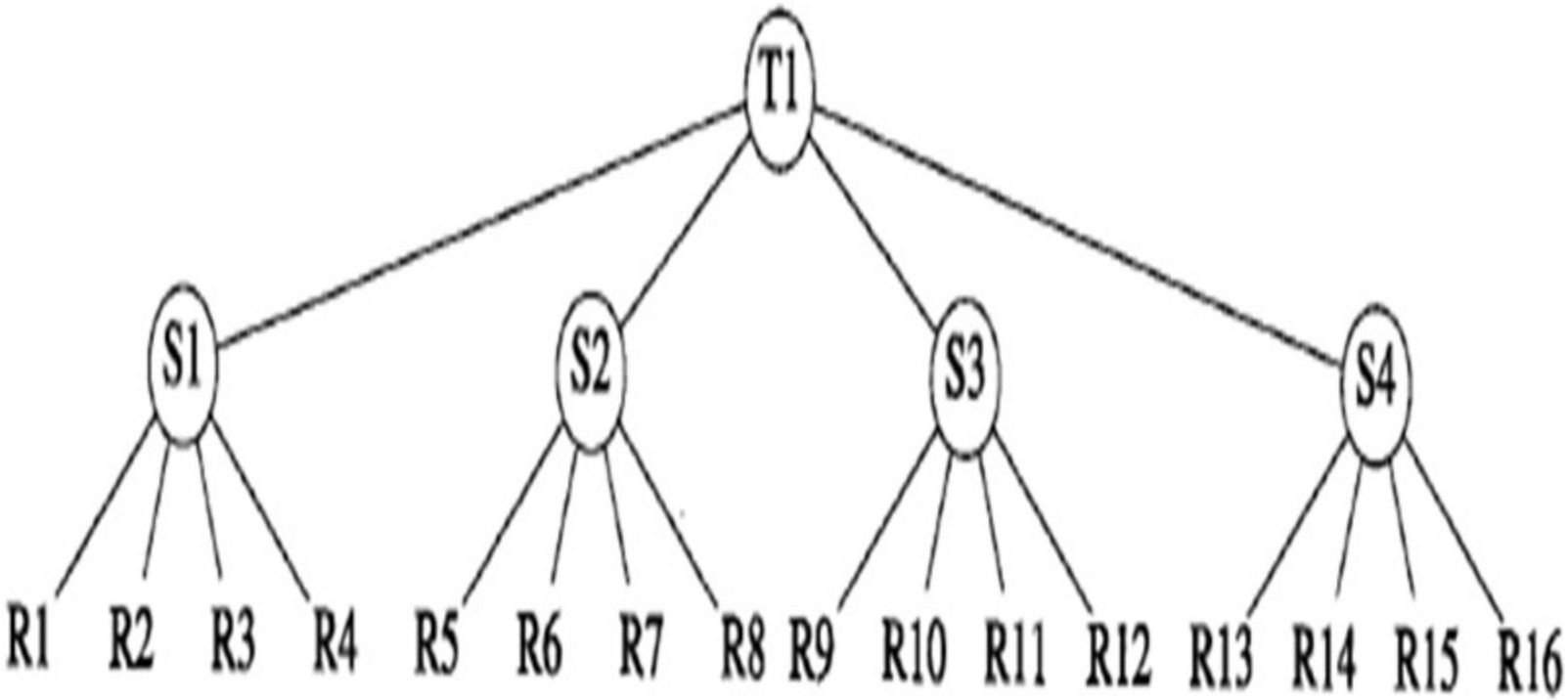
外部排序例

- 假设要为含16000个记录的文件排序，且内存容量为1000个记录。

- 第1步，重复以下步骤16次，可得到16个排序结果(顺串)：
  - 输入1000个记录
  - 用内部排序法对这1000个记录进行排序
  - 输出排序结果顺串

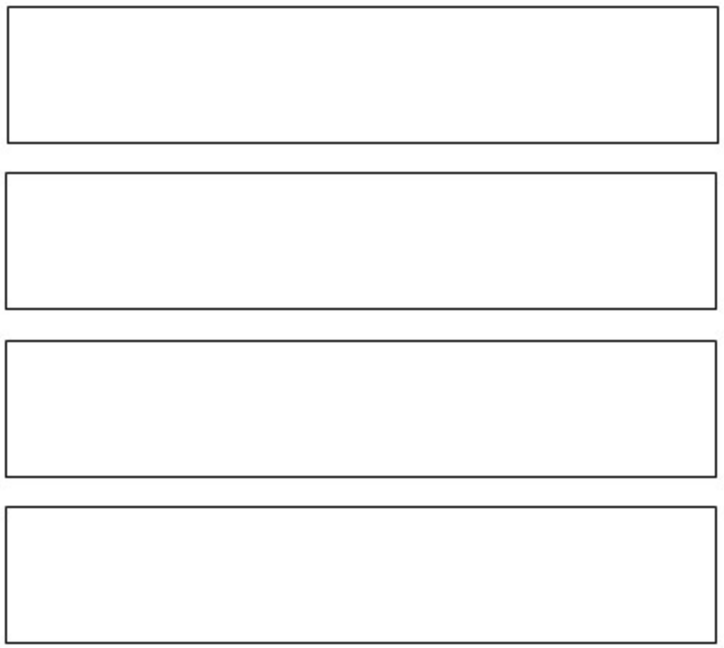
外部排序例

- 第2步，重复地进行k路合并:将k个顺串合并成一个顺串。
- 例：k=4



K路合并的原始方法

- 设内存容量为1000个记录
- 每个缓冲区为200个记录



输入顺串缓冲区

输出顺串缓冲区

K路合并的的原始方法

- 从k个顺串的前面不断地移出值最小的元素，该元素被移至输出顺串中。当所有的元素从k个输入顺串移至输出顺串中时，便完成了合并过程。
- 只要有足够内存保存k个元素值，就可合并任意长度的k个顺串。
- 将每一个元素合并到输出顺串中需O(k)时间，因此产生大小为n的顺串所需要的时间为O(kn)。

外排序

- 外排序所需要的时间由三部分组成：
  - 内部排序所需要的时间
  - 外存信息读写所需要的时间
  - 内部归并所需要的时间
- 减少外存信息的读写次数是提高外部排序效率的关键

外排序时间

- 对同一个文件而言，进行外排序所需读写外存的次数与归并趟数有关系
- 假设有**m**个初始顺串，每次对**k**个顺串进行归并，归并趟数为**$[\log_k m]$**

□为了减少归并趟数，可以从两个方面着手：

– 减少初始顺串的个数**m**

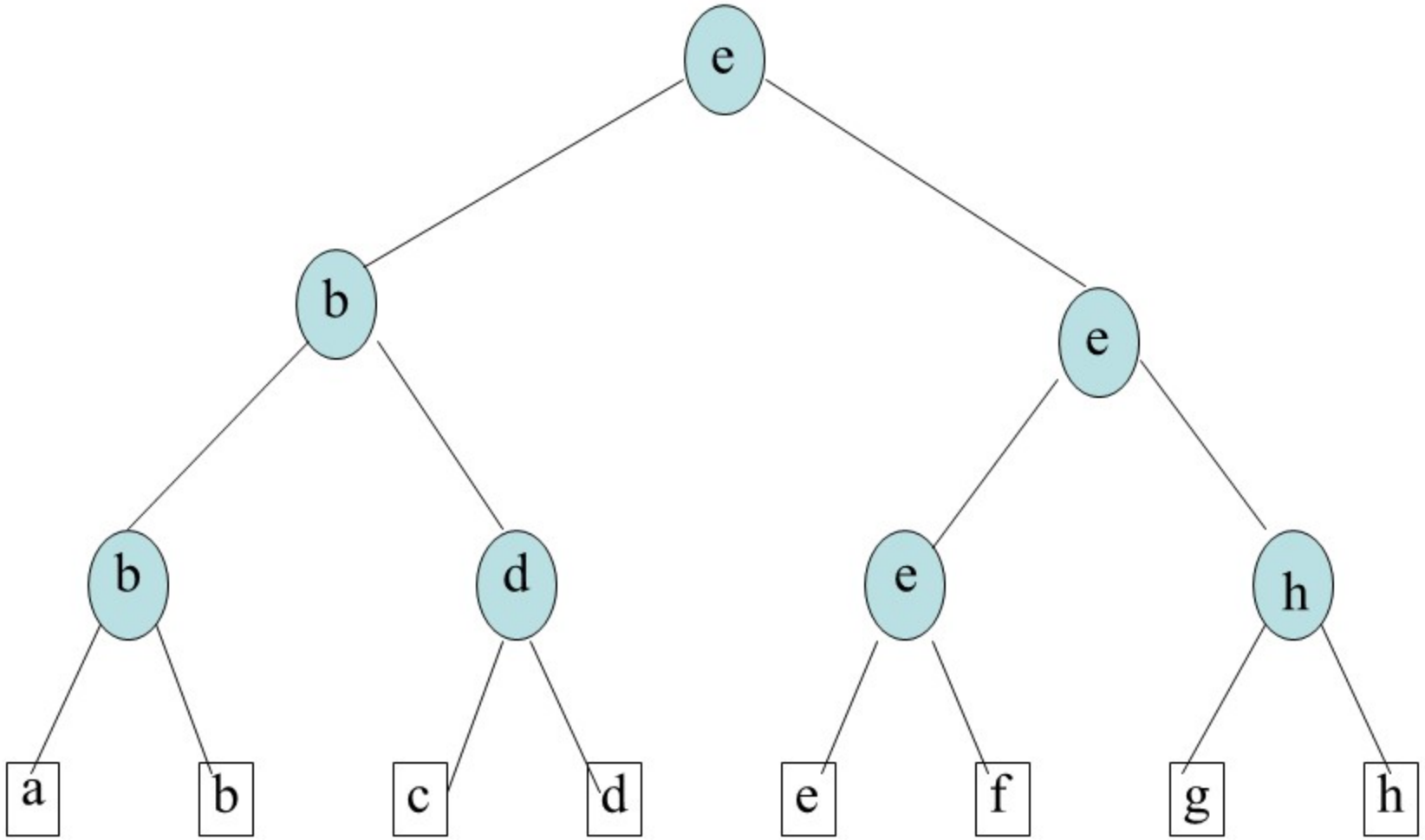
– 增加归并的顺串数量**k**

□减少k路合并时间

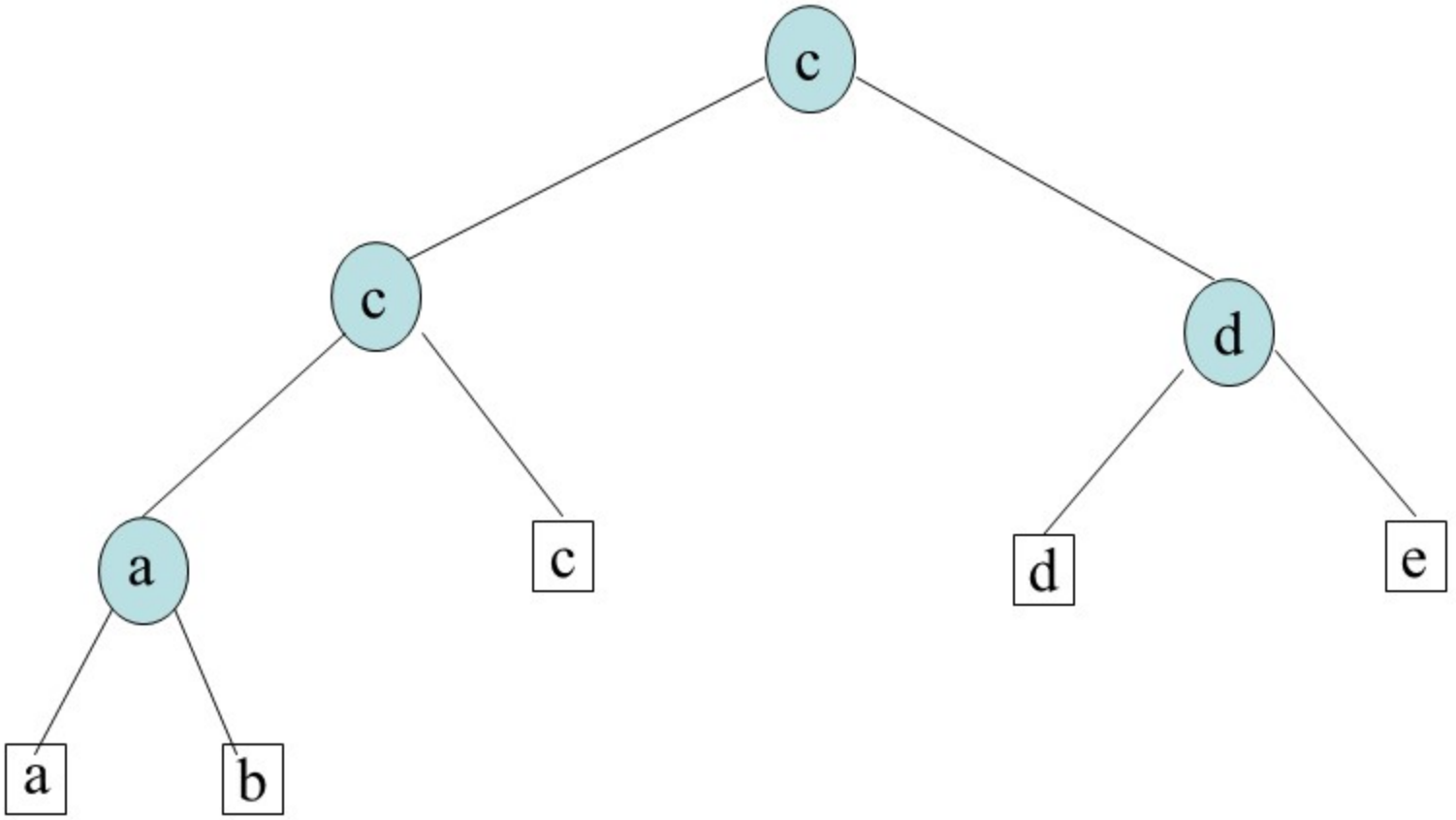
竞赛树

- 竞赛树是完全二叉树，竞赛树的类型：赢者树、输者树
  - 每个外部节点分别代表一名选手
  - 每个内部节点分别代表一场比赛，参加每场比赛的选手是子节点所对应的两名选手。
  - 同一层节点所构成的一轮比赛可以同时进行。
- 竞赛树在某些情况下也被称为选择树（selection tree）。

竞赛树示例



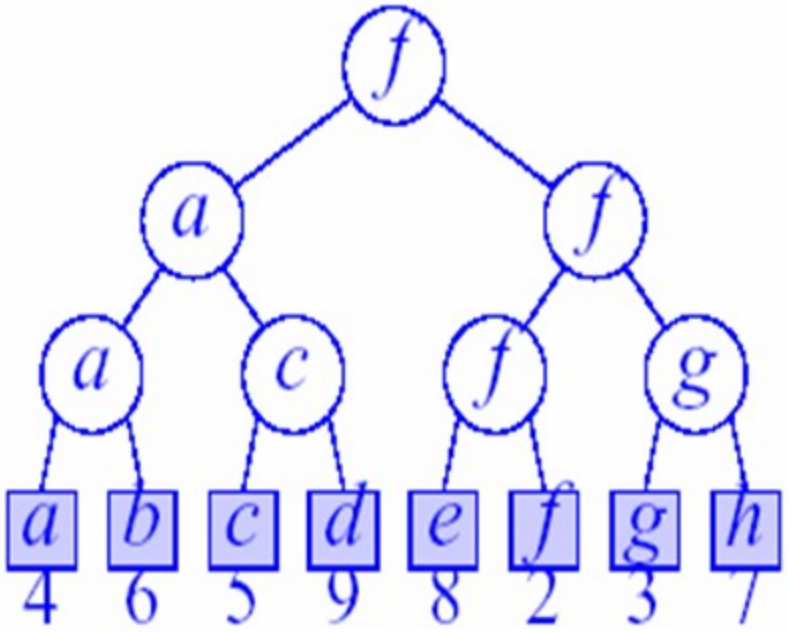
竞赛树示例



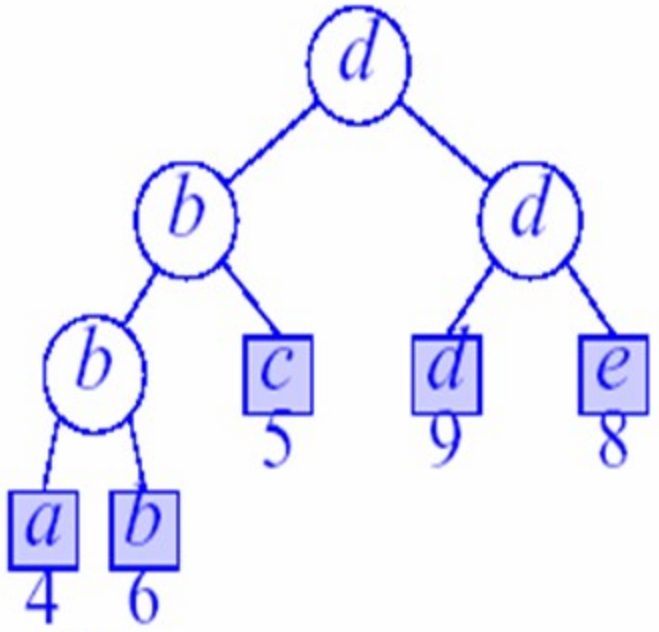
赢者树(Winner Trees)

- 赢者树定义：对于n 名选手，赢者树是一棵含n 个外部节点，$n-1$个内部节点的完全二叉树，其中每个内部节点记录了相应赛局的赢家。
- 为决定一场比赛的赢家，假设每个选手有一得分，且赢者取决于对两选手得分的比较。
- 最小赢者树（min winner tree）：得分小的选手获胜；
- 最大赢者树（ max winner tree）：得分大的选手获胜。

赢者树



(a) 最小赢者树



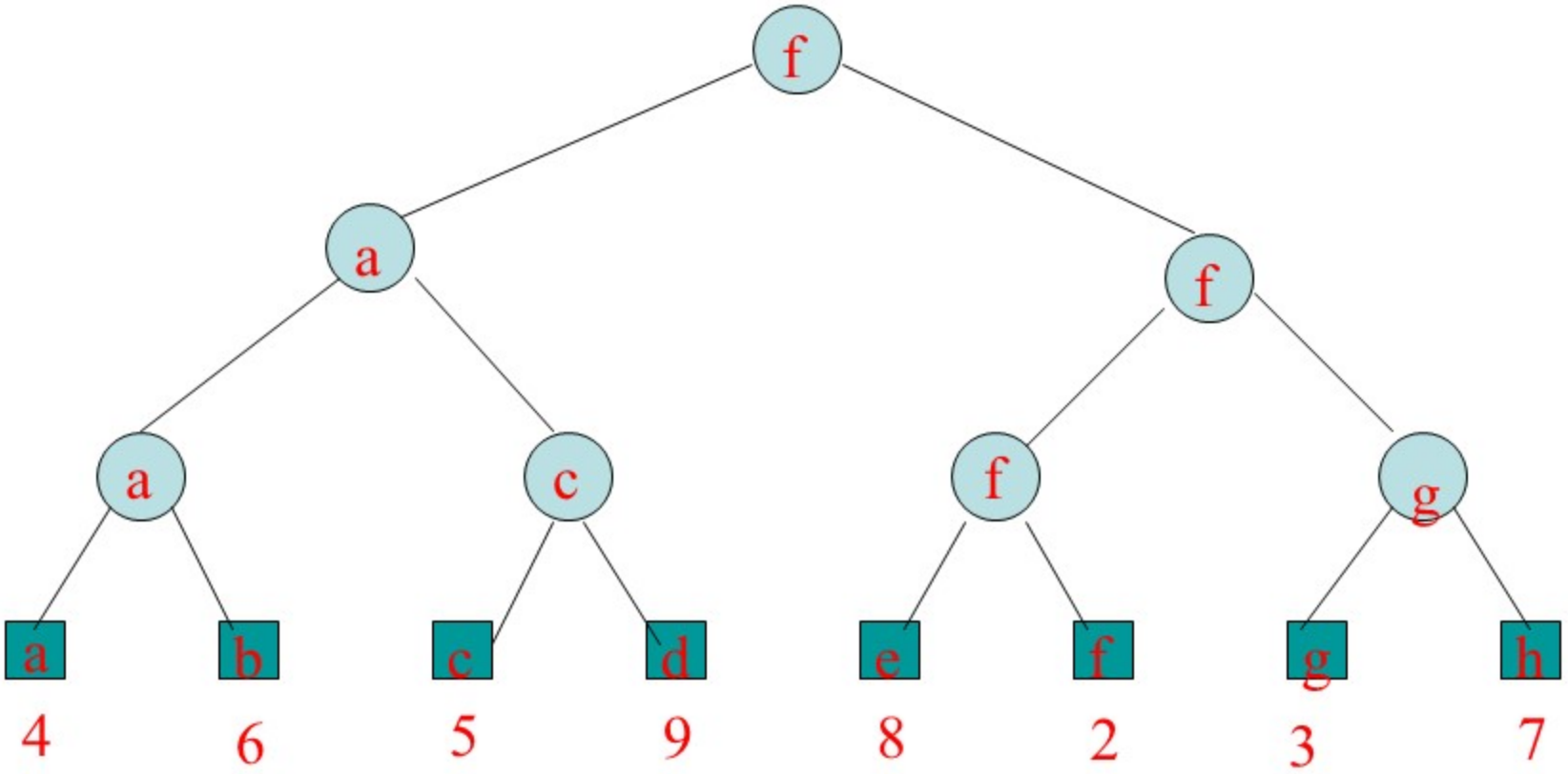
(b) 最大赢者树

高度是：$\lceil \log_2 n \rceil$

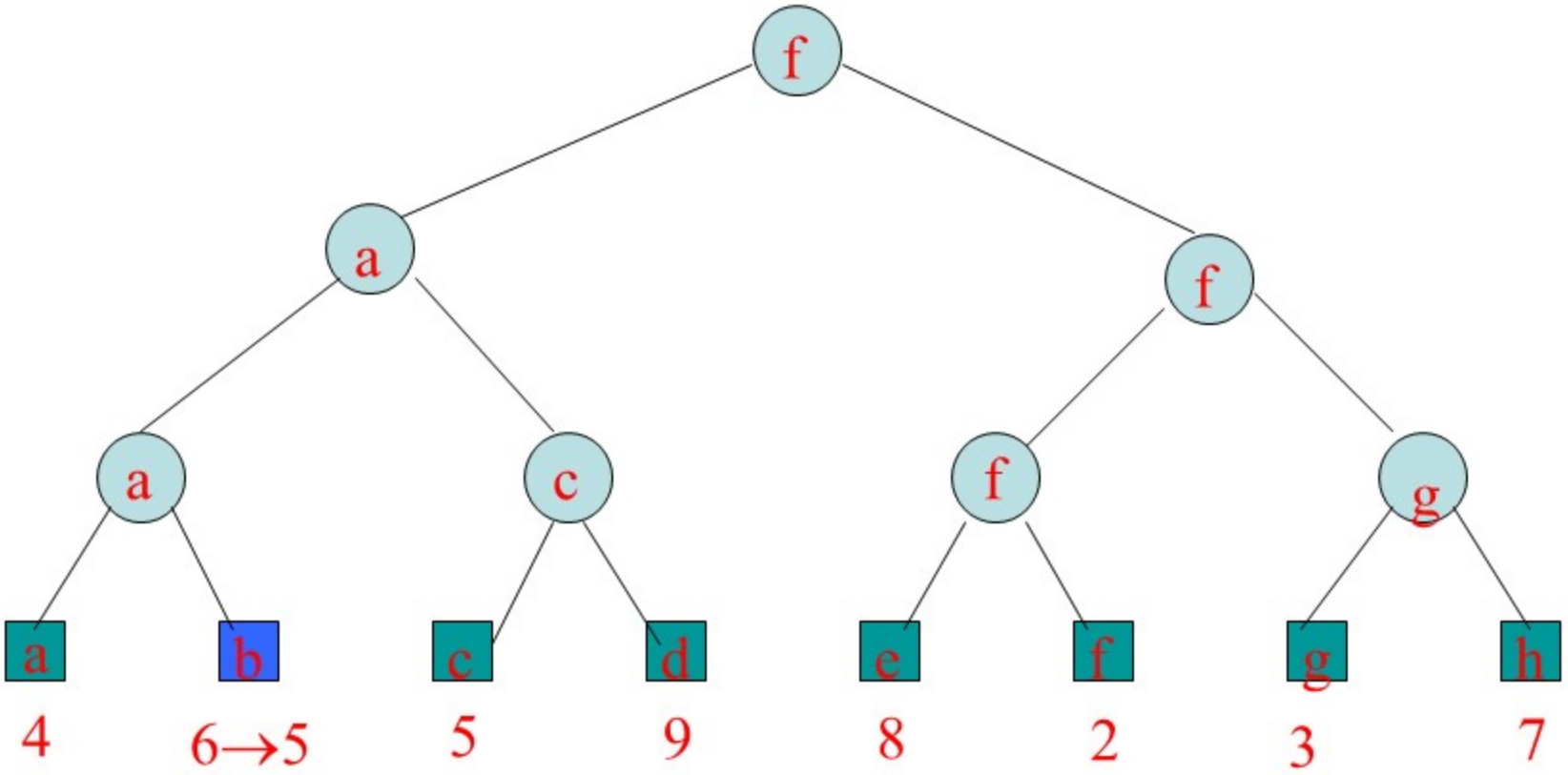
赢者树

- 赢者树的一个优点是：
  - 如果一个选手的分数值改变了，可以很容易地修改这棵赢者树。
  - 只需要沿着从代表该选手的外部结点到根结点的路径，修改二叉树，而不必改变其它比赛的结果。

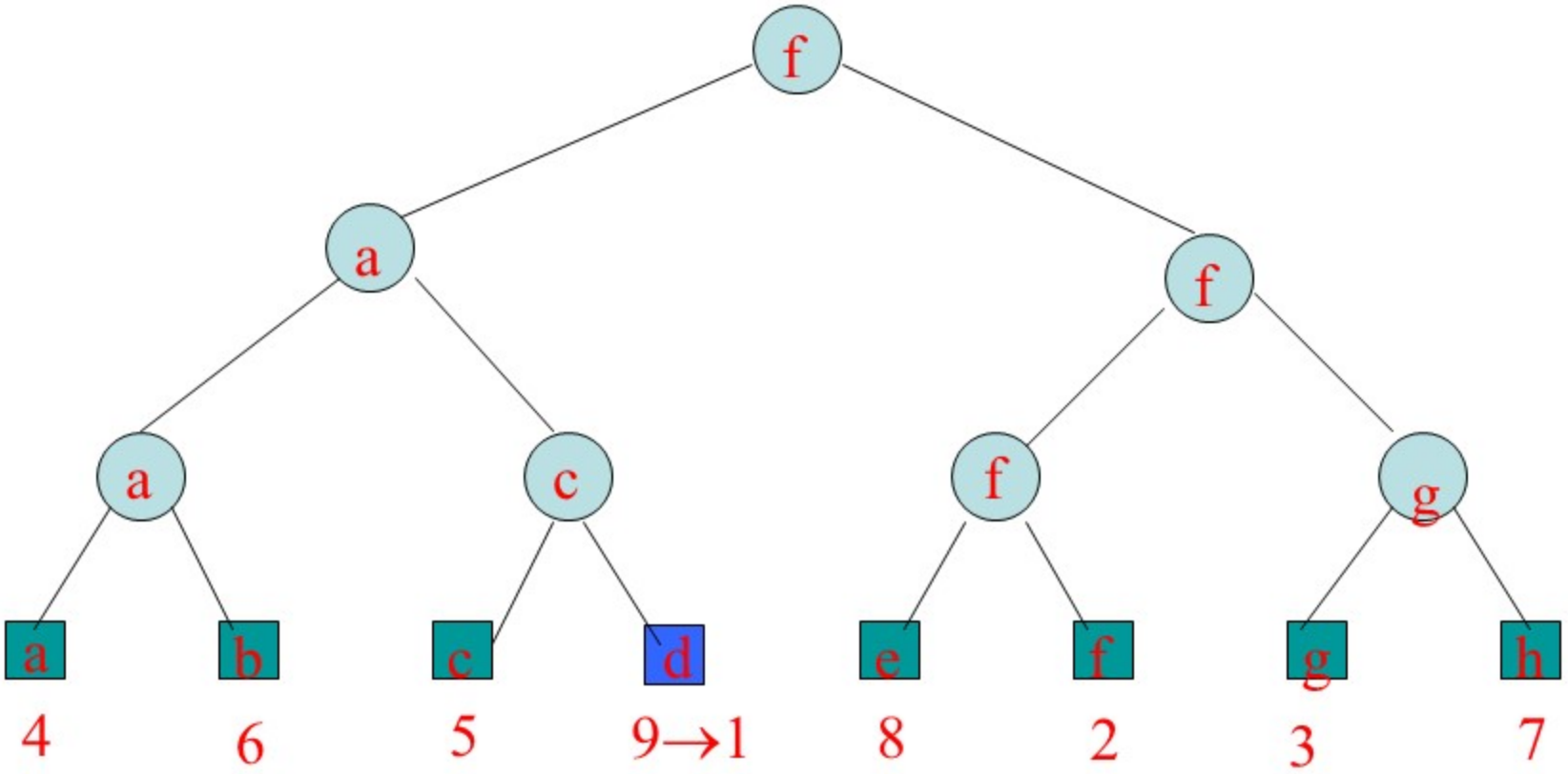
赢者树 操作—重构



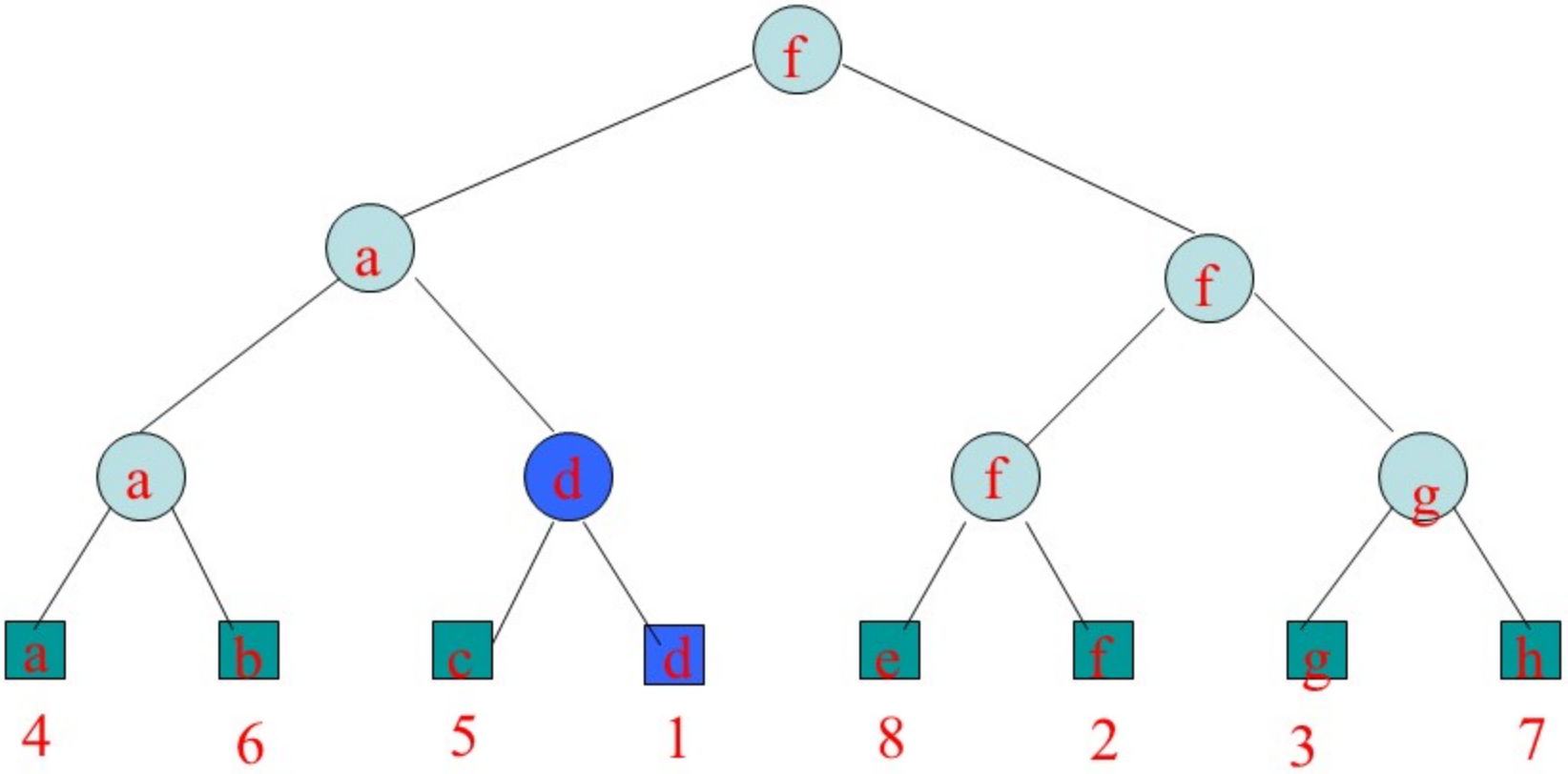
赢者树 操作 — 重构



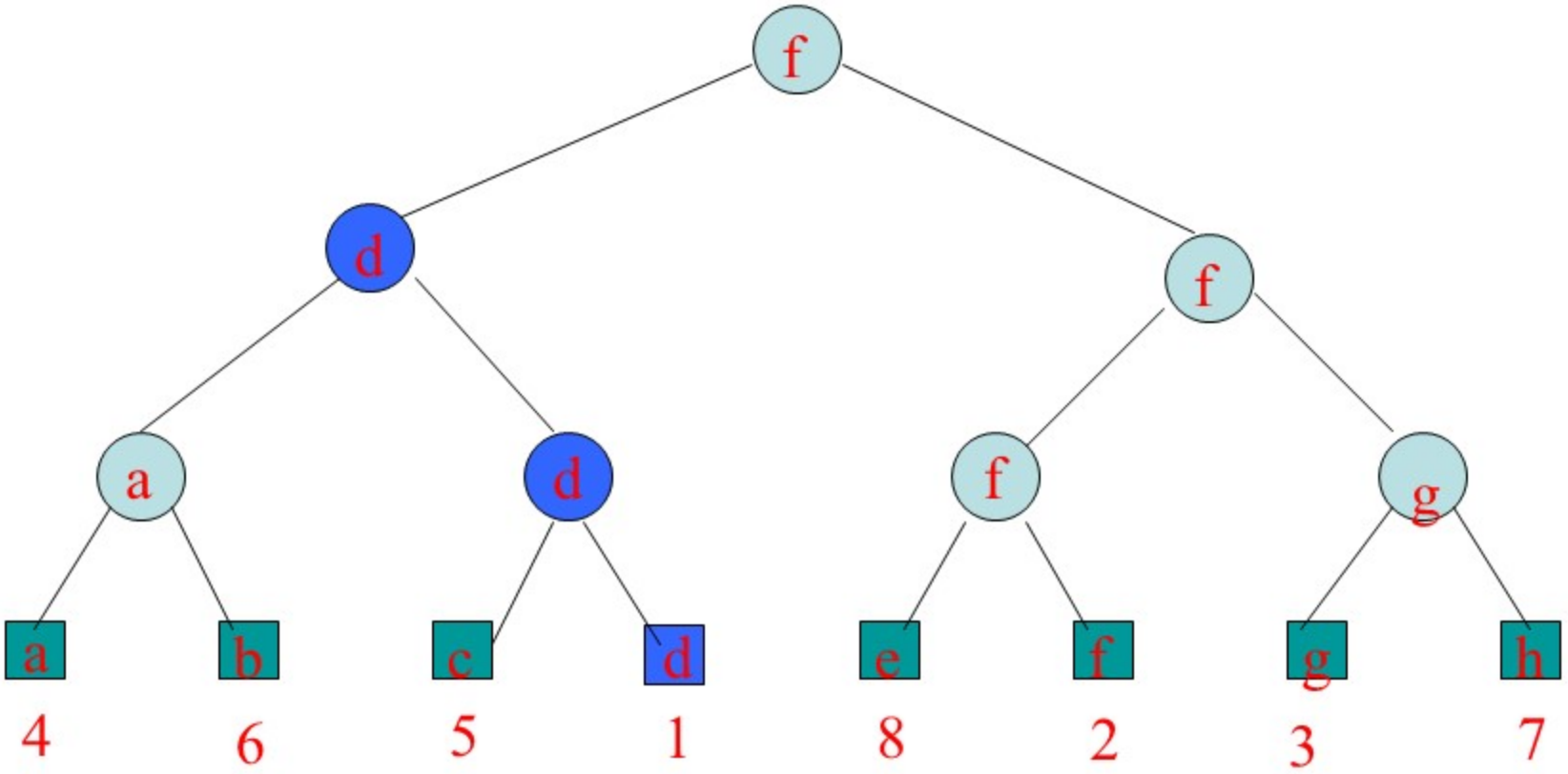
赢者树 操作 — 重构



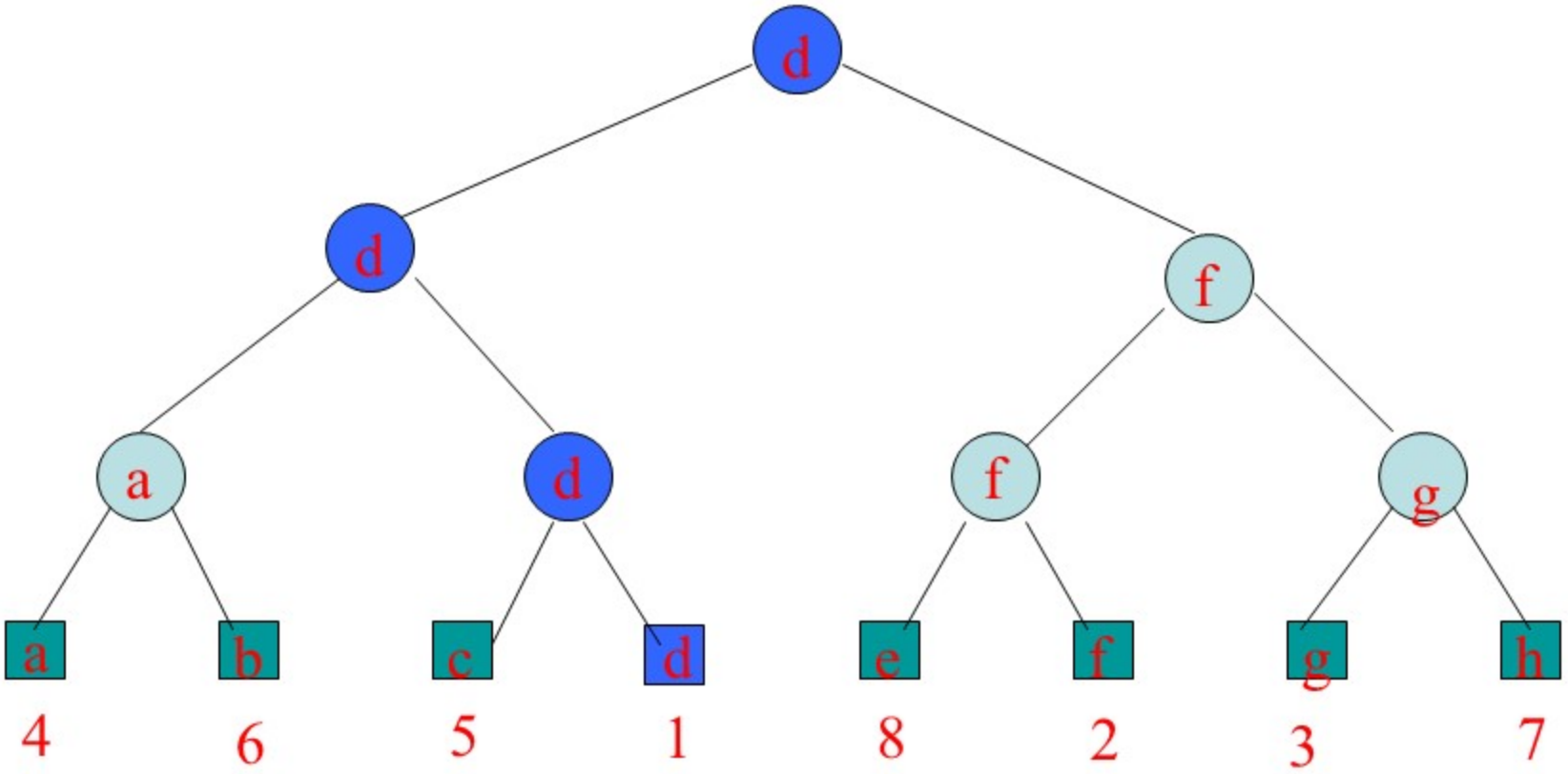
赢者树 操作 — 重构



赢者树 操作 — 重构



赢者树 操作 — 重构



- 时间： O(log n) 次.

缩短k路合并的时间？

- 若使用**赢者树**，进行k路合并，则可将这个时间缩短为$\Theta(k+n\log k)$。
  - 首先用O(k)的时间初始化含k个选手的赢者树。这k个选手都是k个被合并顺串中的头一个元素。
  - 然后将赢者移入输出顺串中并用相应的输入顺串中的下一个元素替代之。如果在该输入顺串中无下一元素，则需用一个key值很大(不妨为∞)的元素替代之。
  - k次移入和替代赢家共需耗时$\Theta(\log k)$。因此采用赢者树进行k路合并的总时间为$\Theta(k+n\log k)$。

减少初始顺串的个数？

- 设内存空间能够容纳的赢者树包含的外部节点(选手)为p个，使用一个含*p*名选手的赢者树，
  - 每个选手对应输入集合中的一个元素。
  - 每个选手有一个值（即对应的元素值）和一个顺串号。
  - 赢者规则：
    - 具有较小顺串号的元素获胜。
    - 顺串号相同，较小元素值的元素获胜。

顺串的生成方法

- 从输入集合(包含全部元素)中输入前p个元素. 初始这p个元素的顺串号均为1。
- 建立这p个选手的**最小赢者树**。
- 重复：
  - 将最终赢者W移入它的顺串号所对应的顺串中；
  - 若输入集合中有下一个输入元素，则N=下一个输入元素，否则N = ∞；
  - 如果 N 的元素值≥W 的值，则 元素N的顺串号=W的顺串号，否则 元素N的顺串号=W的顺串号+1。
  - N代替W，*重构*赢者树
- 直到所有的元素都输出到对应的顺串中。

顺串的生成方法

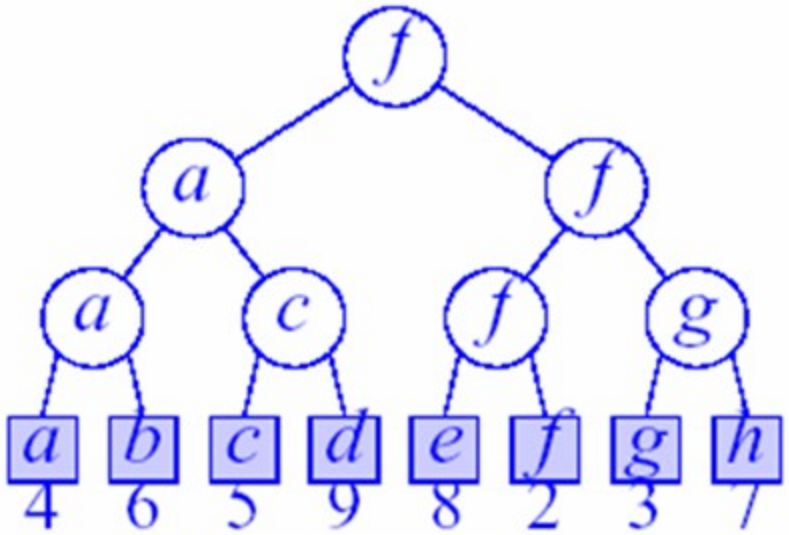
- 例：17, 21, 5, 44, 10, 12, 56, 32, 29
- 使用以上方法生成顺串时，顺串的平均长度约为$2p$。
- 特别是，当输入集合已排好序(或几乎排好序)，效率会比较高。

输者树(Loser Trees)

- 输(败)者树是赢者树的一种变体。对于n名选手，输者树是一棵含n 个外部节点，n-1个内部节点的完全二叉树，其中每个内部节点记录了相应赛局的输者。
- 另外，根结点处加入一个结点来记录整个比赛的胜者。
- 采用输者树是为了简化重构的过程

例 最小输者树

- 在许多情况下，赢者被新选手替换（赢者的值改变）。
- f: 2改变为5

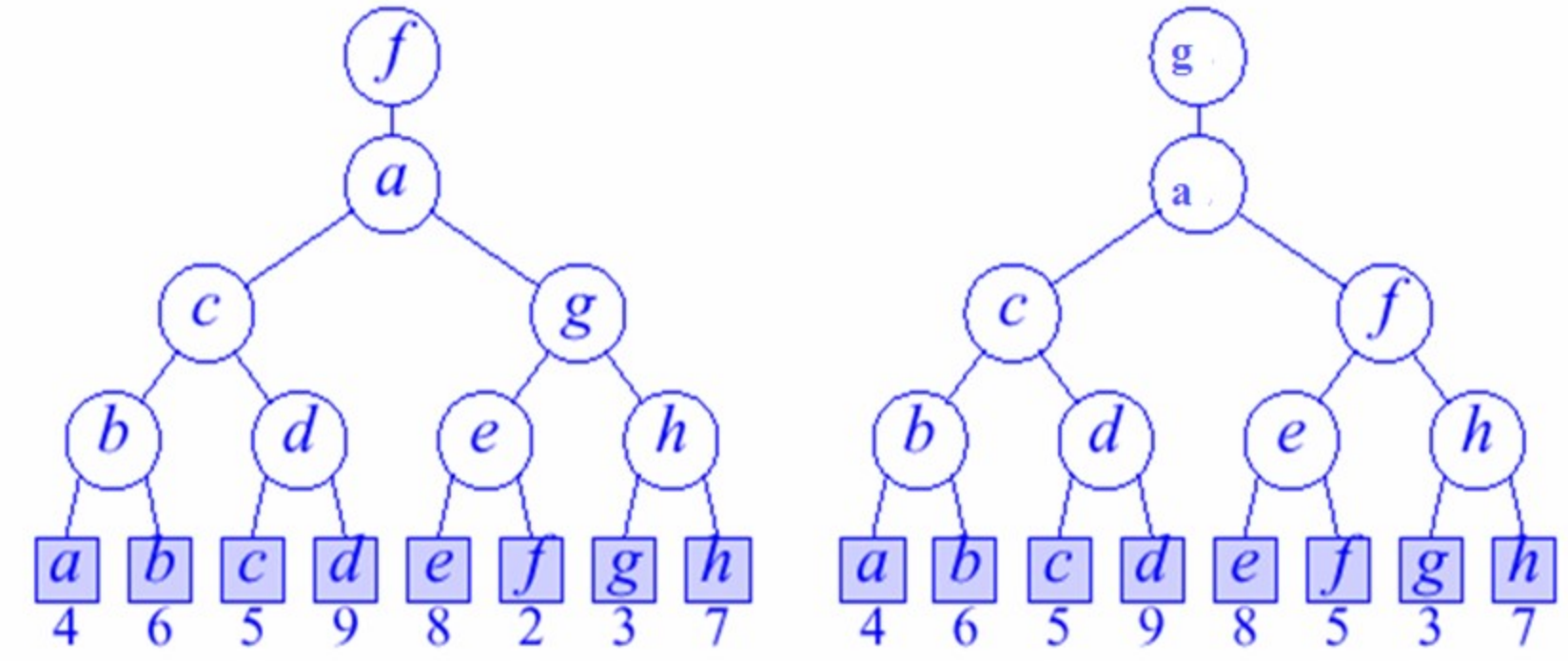


(a) 最小赢者树

f——e, f——g, g——a

e, g,a 对应比赛的输者

例 最小输者树



（a） 初始化 （b） f: 2改变为5

- 使用输者树可以简化**赢者的值发生变化**的情况，但不能简化改变其他选手的值的情况。