外部排序

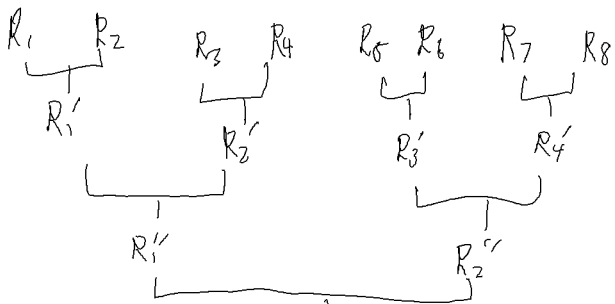
概念

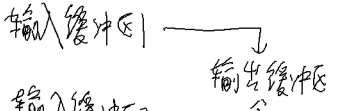
待排记录很多，存储在外存中，排序时将数据逐部分调入内存排序，过程中需要多次内存和外存的交换

外部排序主要时间花费在访问磁盘（I/O次数）

归并排序法
① 根据缓冲区大小，划分文件成长度L的子文件，依次读入内存排序，排好后重新写回外存，排好后的有序子文件称归并段
② 对归并段逐趟归并，使归并段由小到大直至整个文件有序

例





外部排序总时间 = 内部排序时间 + 外存读写时间 + 内部归并时间

增大归并路，可以减少归并趟数，进而减少I/O次数

r个初始归并段，做k路归并，可以用严格k叉树来表示
(三有度为k和度为0)

归并趟数 S = 树高 - 1 = $\lceil \log_k r \rceil$

多路平衡归并和败者树

S趟归并所需的比较次数

$$S(n-1)(k-1) = \lceil \log_2 k \rceil (n-1)(k-1) = \lceil \log_2 r \rceil (n-1) \frac{(k-1)}{\lceil \log_2 k \rceil}$$

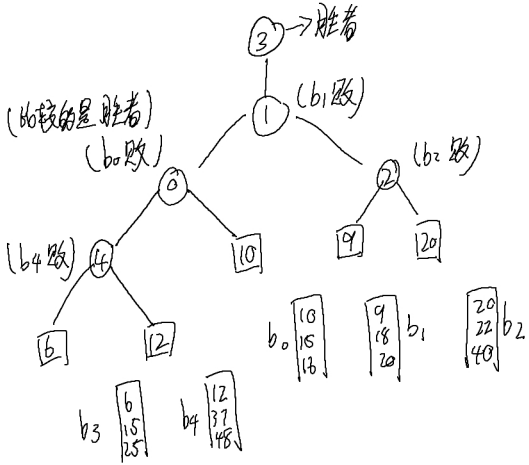
内部归并排序效率受k的影响，因此引入败者树

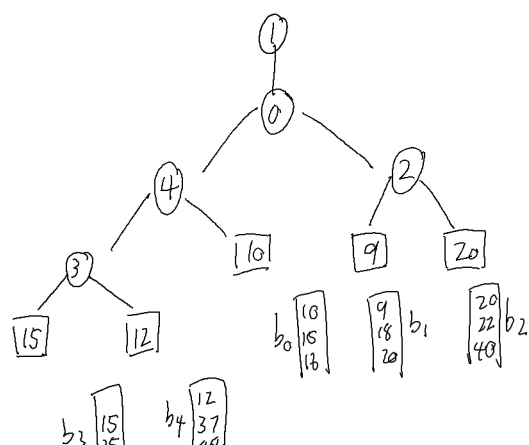
败者树 ⇒ 一棵完全二叉树，k个叶子存放k个归并段中当前的关键字记录，内部结点记录左右子树的失败者，让胜者向上继续比较，直至根。
若大的失败，小的胜利，则指针指向最小数

k路归并败者树深度 $\lceil \log_2 k \rceil$

因此 S趟归并比较次数 $S(n-1)\lceil \log_2 k \rceil = \lceil \log_2 r \rceil (n-1)$

归并路数k不是越大越好，k增大，缓冲区个数也增多


3
→胜者
(b1败)
1
(b0败)
(始终的是胜者)
0
(b2败)
2
(b4败)
4
10
9
20
6
12
b0
10
15
18
b1
9
18
20
b2
20
22
40
b3
6
15
25
b4
12
37
48


1'
0
4
2
3'
10
9
20
15
12
b0
10
15
18
b1
9
18
20
b2
20
22
40
b3
15
25
b4
12
37
48

5路归并败者树

置换-选择排序（生成初始归并段）

归并段长度（记数r　　r=⌈n/l⌉

置换-选择算法可以产生更长的初始归并段，以减少记数r

① 从输入文件FI中输出w个记录到工作区WA中
② 从WA中选出最小的记录，记为MINIMAX
③ MINIMAX输出到输出文件FO中
④ 若FI不空，FI输出下一个记录到WA中
⑤ WA中找出大于MINIMAX的最小值作为新MINIMAX
⑥ 重复③-⑤直至选不出新的MINIMAX，输出一个归并段的结符
使用败者树从WA中选MINIMAX

FO	WA (w=3)	FI
		17,21,05,44,10,12,56,32,29
	17,21,(05)	44,10,12,56,32,29
05	(17),21,44	10,12,56,32,29
05,17	10,(21),44	12,56,32,29
05,17,21	10,12,(44)	56,32,29
05,17,21,44	10,12,(56)	32,29
05,17,21,44,56#	(10),12,32	29
10	29,(12),32	
10,12	(29),32	
10,12,29	(32)	
10,12,29,32#		

最佳归并树

置换-选择排序 得到的归并段长度不等

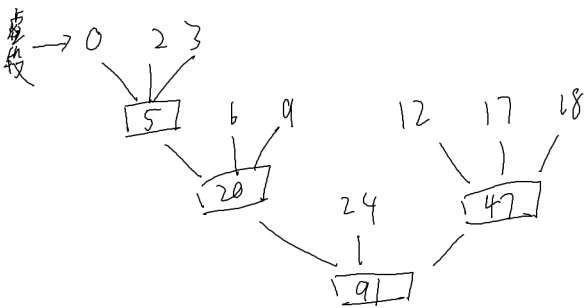
归并长度不等的归并段 使 I/O 记数最少，建立最佳归并树

I/O记数 = 2 × WPL（带权路径长度）（权值为归并段长度）

最佳归并树 → 哈夫曼树推广到 m 叉情况

若初始段个数不足以构成严格 m 叉树，则需要补充长为0的虚段

{2, 3, 6, 9, 24, 12, 17, 18} 不足以构成严格 3 叉树



如何判定要加几个虚段

设度为0的结点 n_0（初始=n）个，度为k的结点 n_k 个

$$\begin{cases} (n_0-1)\%(k-1)=0 & \text{不用加虚段} \\ (n_0-1)\%(k-1)=u\neq 0 & \text{加 } k-u-1 \text{ 个虚段} \end{cases}$$

上述例子 $(8-1)\%(3-1)=7\%2=1$

加 $k-u-1=3-1-1=1$ 个虚段