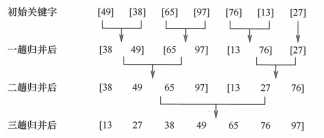
8.5 归并排序和基数排序

归并排序

- 基本思想
  - 每次选定相应的元素分别合成一个新的有序表
    - 2路归并——二合一
    - k路归并——k合一
  - 示意图



- 性能分析
  - 空间复杂度为O(n)
  - 时间复杂度为 $O(n\log_2 n)$
- 稳定排序算法
- 适用于顺序表

基数排序

- 排序思想
  - 最高位优先（MSD）法：按关键字位权重递减依次逐层划分成若干更小的子序列,最后将所有子序列依次连接成一个有序序列
  - 最低位优先（LSD）法：按关键字权重递增依次进行排序,最后形成一个有序序列
- 性能分析
  - 空间效率
    - 一趟排序需要的辅助存储空间为r(r个队列:r个队头指针和r个队尾指针）
    - 基数排序的空间复杂度为O(r）
  - 时间效率
    - 基数排序需要进行d趟分配和收集,一趟分配需要O(n）,一趟收集需要O(r)
    - 基数排序的时间复杂度为O(d(n+r)）　　与序列的初始状态无关
- 稳定排序算法

8.6各种内部排序算法的比较及应用

算法种类	时间复杂度			空间复杂度	是否稳定
	最好情况	平均情况	最坏情况		
直接插入排序	$O(n)$	$O(n^2)$	$O(n^2)$	$O(1)$	是
冒泡排序	$O(n)$	$O(n^2)$	$O(n^2)$	$O(1)$	是
简单选择排序	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(1)$	否
希尔排序				$O(1)$	否
快速排序	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n^2)$	$O(\log_2 n)$	否
堆排序	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(1)$	否
2 路归并排序	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n)$	是
基数排序	$O(d(n+r))$	$O(d(n+r))$	$O(d(n+r))$	$O(r)$	是

排序算法小结

- 若n较小,可采用直接插入排序或简单选择排序
- 当记录本身信息量较大时,用简单选择排序较好
- 若文件的初始状态已按关键字基本有序,则选用直接插入或冒泡排序为宜
- 快速排序被认为是目前基于比较的内部排序方法中最好的方法　　待排序的关键字随机分布时,快速排序的平均时间最短
- 若n较大,则应采用时间复杂度为 $O(n\log_2 n)$ 的排序方法:快速排序、堆排序或归并排序
- 要求排序稳定且时间复杂度为 $O(n\log_2 n)$ 则可选用归并排序
- 若n很大,记录的关键字位数较少且可以分解时,采用基数排序较好
- 当记录本身信息量较大时,为避免耗费大量时间移动记录,可用链表作为存储结构