计数排序

- 计数排序的基本思想就是对每一个输入元素 x ，确定出小于 x 的元素个数。有了这一信息，就可以把 x 直接放到它在最终输出数组上的位置上。

1. 假设 n 个输入元素中的每一个都是介于0到 k 之间的整数，此处 k 为某个整数。当 $k=O(n)$时，计数排序的运行时间为 $\Theta(n)$。
2. 设待排序的记录$a_1,a_2,\cdots,a_n$的关键字的值各不相同，采用计数方式进行排序，根据关键字值比a_i（$1\le i\le n$）小的记录个数确定a_i（$1\le i\le n$）排序后的位置。用类C语言设计算法实现该计数排序。

4.11基数排序

- 不通过待排序数据元素之间的比较
- 根据关键字本身的性质进行排序

  分配排序，桶排序，基数排序
- 例1：名字(英文)按照字典序排序
- 例2：1—m之间的整数排序
  - 关键字的结构或取值范围

基数排序

- 例3：一年的全国高考考生人数为500万，分数100--900，没有小数，把这500万考生按照分数排序。
  - 可能的分数为801个，创建801个“桶”，每个桶对应一个分值
  - 依次查看每个学生的成绩，将其放入对应其分值的桶
  - 从900分对应的那个桶开始依次收集，即完成排序
- 如果待排序的数据元素的取值是从0到2亿？
  - 那就要分配2亿个桶了，这是“不可能（太浪费）”的，所以桶排序有其局限性，适合元素值集合并不大的情况。

桶排序要求：关键字的取值范围或结构已知

桶排序

- 桶排序的基本操作：
  - 分配关键字：根据每个数据元素关键字的值将其分配到相应的桶中 O(n)
  - 每个“桶”内排序 数据元素均匀分布，则每个桶中数据元素个数均匀
  - 收集桶
- 不能象比较排序那样以统一的数据元素之间的“比较”次数衡量“工作量”
- 分析每一步骤地工作进行时间复杂度的分析
- 桶排序的时间取决于每个桶内的排序方法

$$k(cn_i \log n_i) = k(cn/k \log(n/k)) = cn\log(n/k)$$

桶排序

$$k(cn_i \log n_i) = k(cn/k \log(n/k)) = cn \log(n/k)$$

- $k=n/10 \rightarrow cn\log 10 \rightarrow O(n)$
- 每个“桶”内排序方法的性能影响桶排序的性能
- 对每个桶再递归进行桶排序？
- 桶的数量，收集工作
- 能否存在有效的方式分配和收集“一趟统一”进行？

基数排序

- 一张牌有两个关键字组成：花色(♠<♥ <♣<♦)+面值(2<3<4<...<A)。假如一张牌的大小首先被花色决定，同花色的牌由数字决定。
- 52张牌排序有两种算法：

  (1)首先按照面值对所有牌进行排序，然后按照花色再次对所有牌进行排序。

  (2)首先按照花色将所有牌分成4组。然后同组的牌(同花色)再按照面值进行排序。

基数排序—方法1

(1)首先按照面值对所有牌进行排序，然后按照花色再次对所有牌进行排序。

牌的面值13个，准备13个空桶，分别对应13个面值

依次查看52张牌，根据其面值将其放入对应的桶中

基数排序—方法1

(1)首先按照面值对所有牌进行排序，然后按照花色再次对所有牌进行排序。

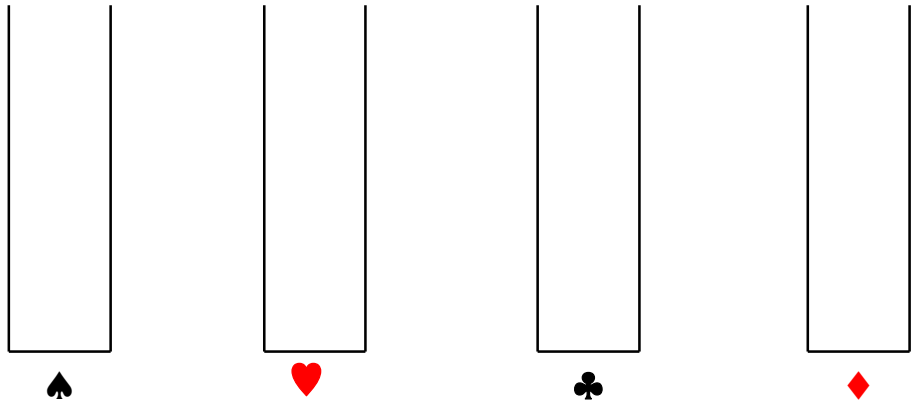
牌的面值13个，准备13个空桶，分别对应13个面值

依次查看52张牌，根据其面值将其放入对应的桶中----第一次分配

从“2”号开始依次收集，将每个桶中的牌顺次倒出，52张牌现在的顺序是4张2，4张3，4张4，……，4张K,4张A----第一次收集

基数排序—方法1

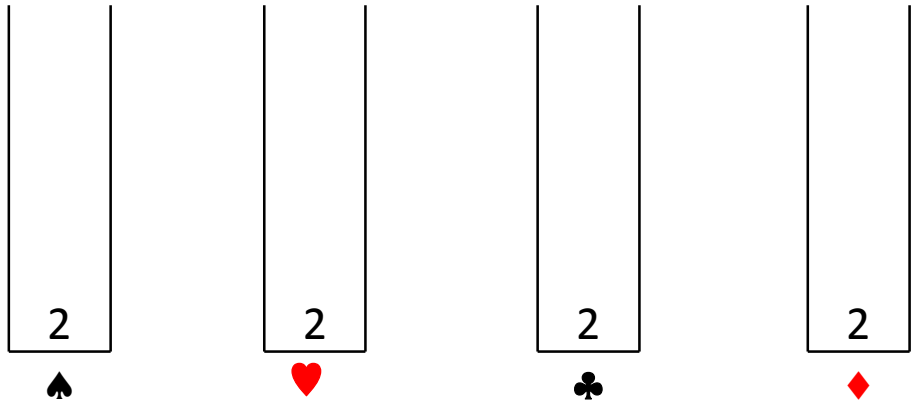
根据第一次收集的结果然后按照花色再次对所有牌进行排序。花色有4个，准备4个空桶，分别对应4个花色



依次查看第一次收集得到52张牌，根据花色将其放入对应的桶中----第二次分配

基数排序—方法1

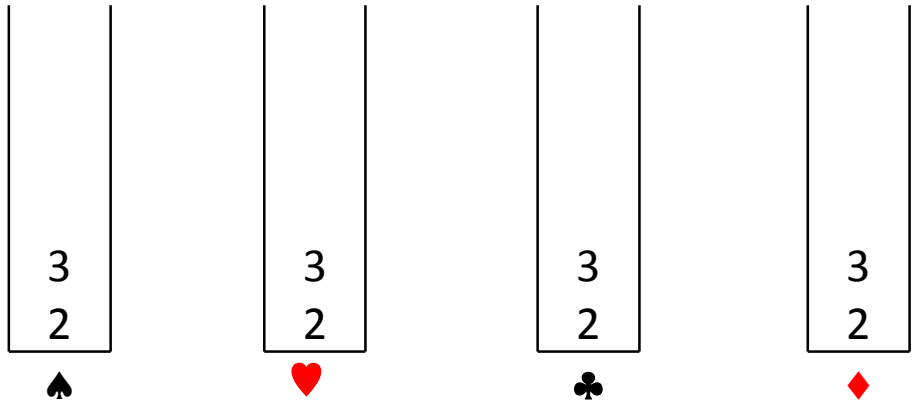
根据第一次收集的结果然后按照花色再次对所有牌进行排序。花色有4个，准备4个空桶，分别对应4个花色



依次查看第一次收集得到52张牌，根据花色将其放入对应的桶中----第二次分配

基数排序—方法1

根据第一次收集的结果然后按照花色再次对所有牌进行排序。花色有4个，准备4个空桶，分别对应4个花色



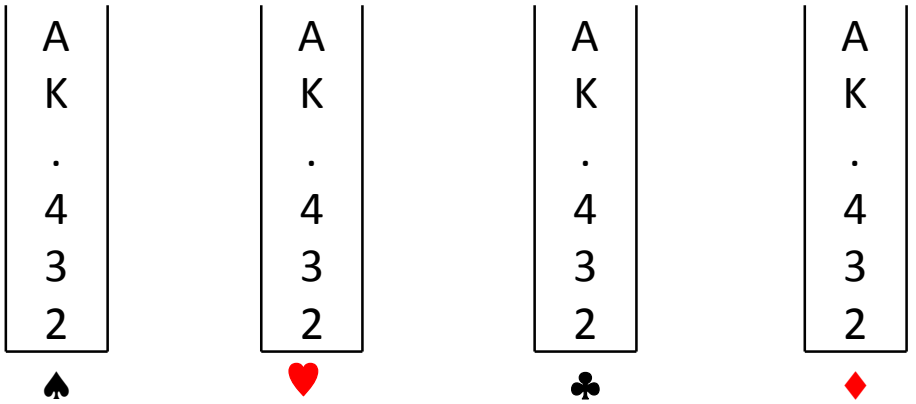
依次查看第一次收集得到52张牌，根据花色将其放入对应的桶中----第二次分配

收集桶的方式：最先进入一个桶的“2”最先倒出----实现时考虑用队列模拟桶

基数排序—方法1

根据第一次收集的结果然后按照花色再次对所有牌进行排序。花色有4个，准备4个空桶，分别对应4个花色

扑克牌的排序是根据其2个值决定的：花色和面值，且花色的优先级高于面值



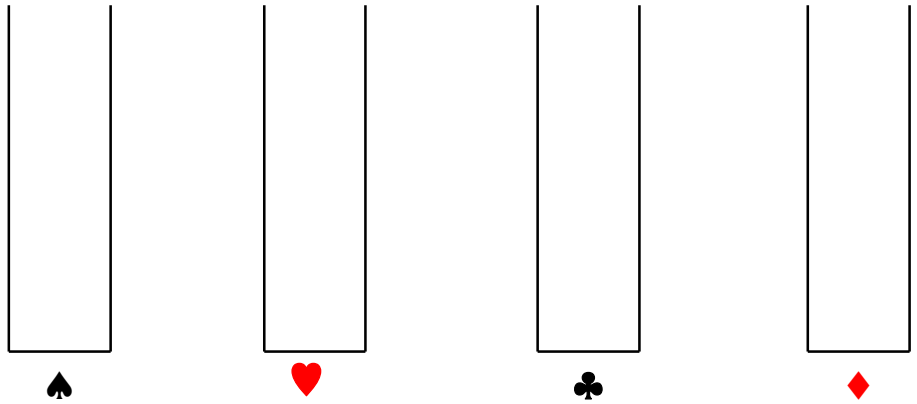
排序过程：先根据面值，后根据花色。二次分配二次收集。没有进行2张扑克牌的比较

依次查看第一次收集得到52张牌，根据花色将其放入对应的桶中----第二次分配

从“♠”号桶开始依次收集，将每个桶桶口封死，桶底打开，顺次倒出每一个桶中的牌，52张牌就排好序了：13张♠从2到A，13张♥从2到A，13张♣从2到A，13张♦从2到A ----第二次收集

基数排序—方法2

(2)首先按照花色将所有牌分成4组。然后同组的牌(同花色)再按照面值进行排序。花色有4个，准备4个空桶，分别对应4个花色

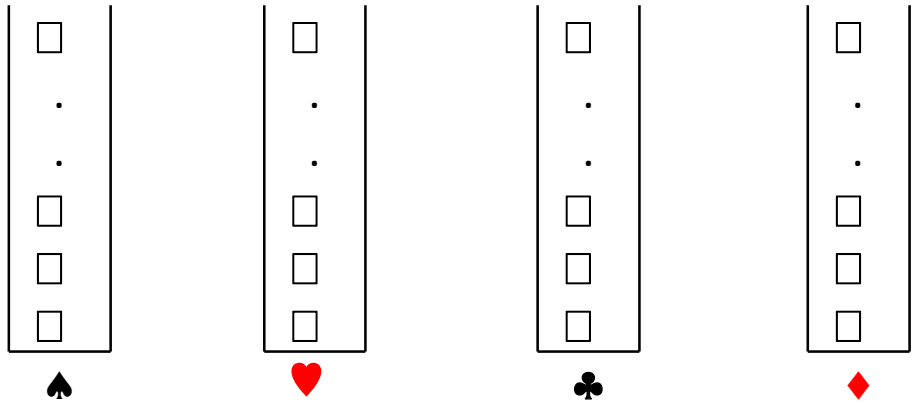


依次查看52张牌，根据花色将其放入对应的桶中----第一次分配

基数排序—方法2

(2)首先按照花色将所有牌分成4组。然后同组的牌(同花色)再按照面值进行排序。花色有4个，准备4个空桶，分别对应4个花色

花色相同的牌放入同一个桶中，每个桶中有13张牌



依次查看52张牌，根据花色将其放入对应的桶中----第一次分配

第一次收集？能否和前面的方法一样将所有桶依次收集在一起？

基数排序—方法2

若4个花色的桶依次一起收集，则第一次收集的52张牌为：13张♠，13张♥，13张♣，13张♦。牌的面值13个，准备13个空桶，分别对应13个面值

依次查看第一次收集得到的52张牌，根据其面值将其放入对应的桶中

基数排序—方法2

若4个花色的桶依次一起收集，则第一次收集的52张牌为：13张♠，13张♥，13张♣，13张♦。牌的面值13个，准备13个空桶，分别对应13个面值

依次查看第一次收集得到的52张牌，根据其面值将其放入对应的桶中

基数排序—方法2

若4个花色的桶依次一起收集，则第一次收集的52张牌为：13张♠，13张♥，13张♣，13张♦。牌的面值13个，准备13个空桶，分别对应13个面值

依次查看第一次收集得到的52张牌，根据其面值将其放入对应的桶中

基数排序—方法2

若4个花色的桶依次一起收集，则第一次收集的52张牌为：13张♠，13张♥，13张♣，13张♦。牌的面值13个，准备13个空桶，分别对应13个面值

依次查看第一次收集得到的52张牌，根据其面值将其放入对应的桶中

基数排序—方法2

若4个花色的桶依次一起收集，则第一次收集的52张牌为：13张♠，13张♥，13张♣，13张♦。牌的面值13个，准备13个空桶，分别对应13个面值

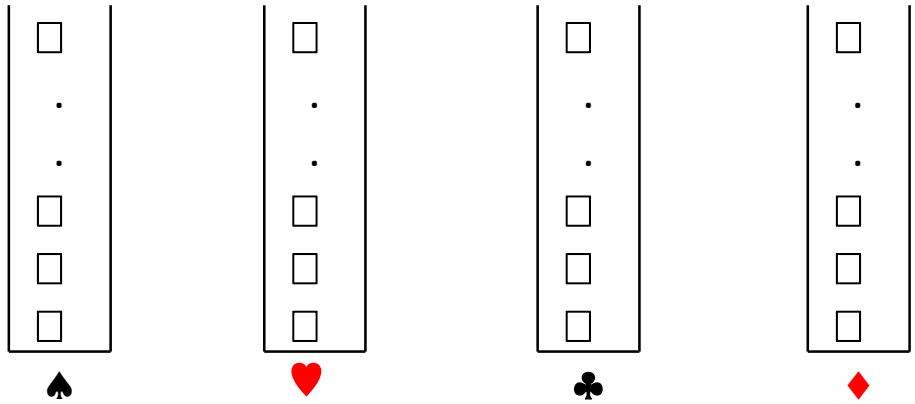
依次查看第一次收集得到的52张牌，根据其面值将其放入对应的桶中

第二次收集？　一起收集得不到正确的排序结果。

基数排序----正确的方法一

(2)首先按照花色将所有牌分成4组。然后同组的牌(同花色)再按照面值进行排序。花色有4个，准备4个空桶，分别对应4个花色

花色相同的牌放入同一个桶中，每个桶中有13张牌



依次查看52张牌，根据花色将其放入对应的桶中----第一次分配

第一次收集？

不一起收集。先收集♠桶，对收集的结果按照面值进行桶排序；再收集♥桶，对收集的结果按照面值进行桶排序；……

基数排序

- 基数排序是一种借助“多关键字排序”的思想来实现“单关键字排序”的内部排序算法。
- n 个记录的序列 $\{R_1, R_2, \cdots, R_n\}$,对关键字 $(K_i^1, \cdots, K_i^d)$ 有序是指：对于序列中任意两个记录 R_i 和 $R_j(1 \leqslant i < j \leqslant n)$ 都满足下列(词典)有序关系：

  $(K_i^1, \cdots, K_i^d) < (K_j^1, \cdots, K_j^d)$

  K^1 被称为 “最主”位关键字

  K^d 被称为 “最次”位关键字

基数排序----2种方法

- **最高位优先MSD法：**

✓ MSD——必须将序列逐层分割成若干子序列，再对各子序列分别排序。

- **最低位优先LSD法：** 先对K^d进行桶排序排序，再对K^{d-1}进行桶排序，……， 依次类推，直至最后对最高位关键字桶排序完成为止。

✓ LSD——不必逐层分割成子序列，并且可不通过关键字比较，而通过若干次分配与收集实现排序。

- 通常采用低位优先。
- 以下以低位优先为例介绍基数排序

基数排序

可看作有3个关键字：
百位、十位、个位

- 例 {369，367，167，239，237，138，230，139}低位优先基数排序：
  - 首先按其“个位数”取值分配到“0, 1, …, 9”10 个桶，之后按从 0 至 9 的顺序将它们“收集”在一起；
  - 然后按其“十位数”取值分配到“0, 1, …, 9”10个桶，之后再按从 0 至 9 的顺序将它们“收集”在一起；
  - 最后按其“百位数”重复一遍上述操作。

❁对于多关键字的记录序列，通常利用LSD法借助分配-收集”进行排序。

❁对于数字型或字符型的单逻辑关键字，可以看成由多个数位或多个字符构成的多关键字，也可以采用“分配-收集”的排序方法。

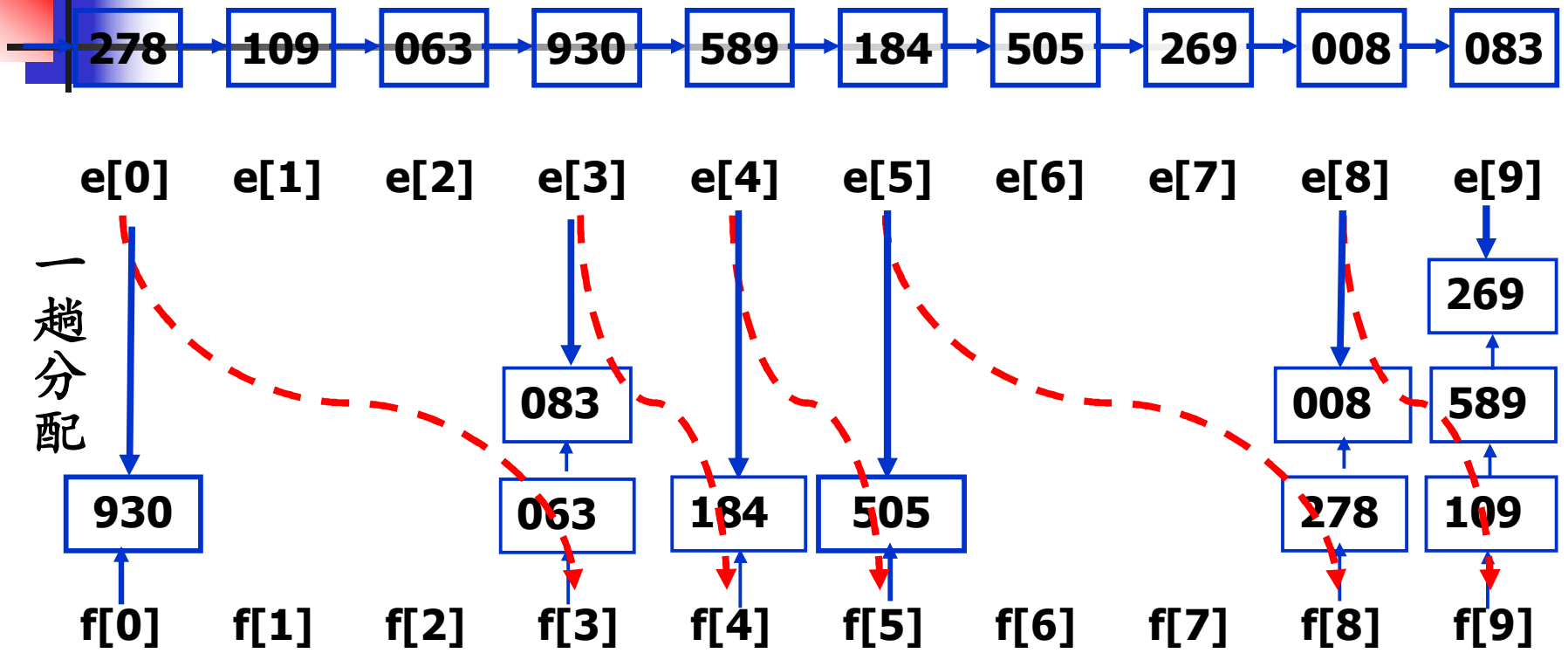
基数排序的实现

- 采用链表作存储结构，即链式基数排序，具体作法为：

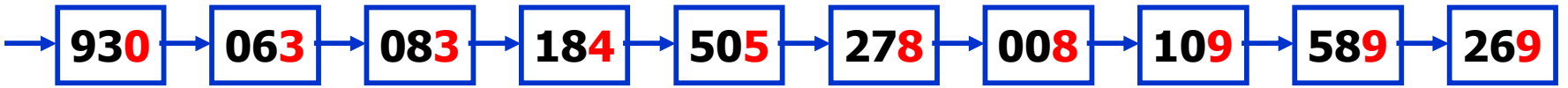
1. 待排序记录以指针相链，构成一个链表；
2. “分配” 时，按当前“关键字位”所取值，将记录分配到不同的 “链队列”（桶）中，每个队列（桶）中记录的 “关键字位” 相同；
3. “收集”时，按当前关键字位取值从小到大将各队列（桶）首尾相链成一个链表；
4. 对每个关键字位均重复 2 和 3 两步。

链式基数排序 Linked Radix Sort

初始状态：

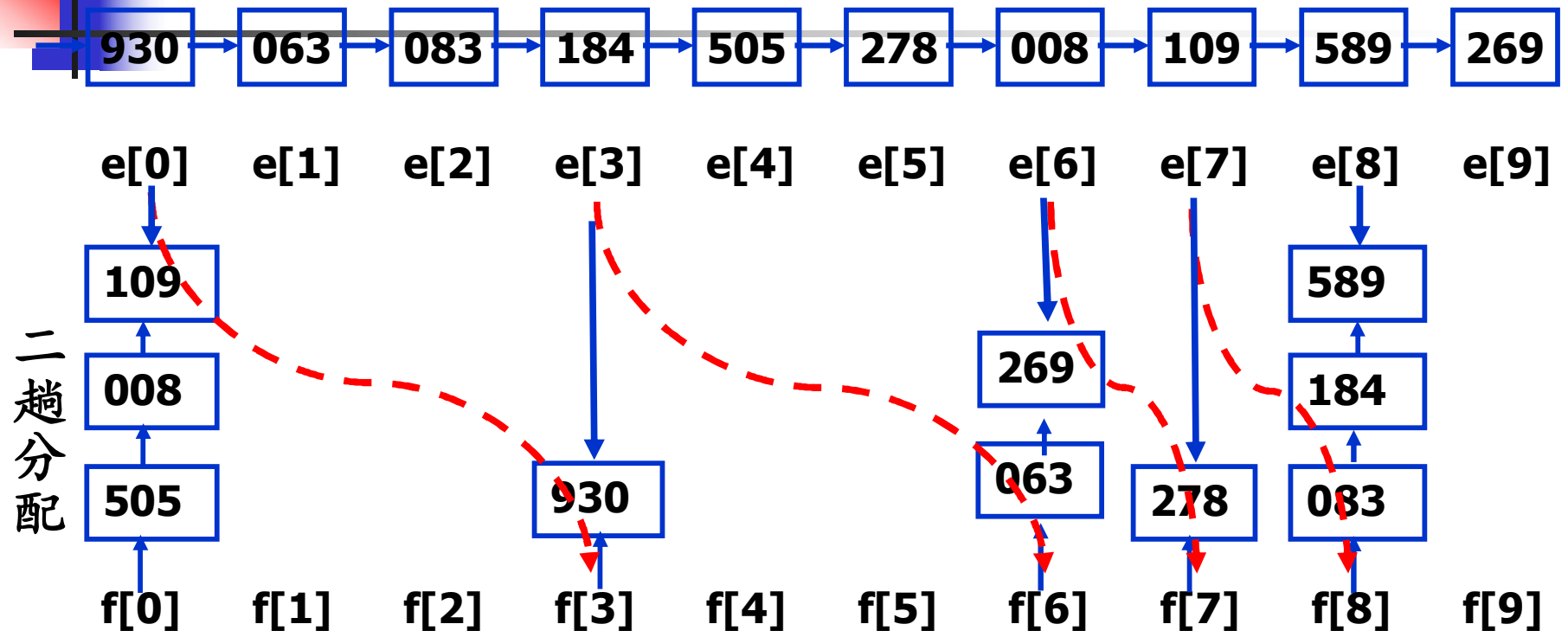


一趟收集：

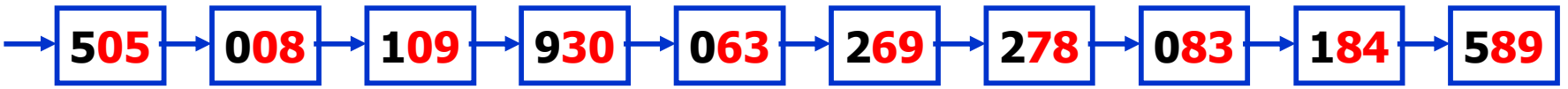


链式基数排序 Linked Radix Sort

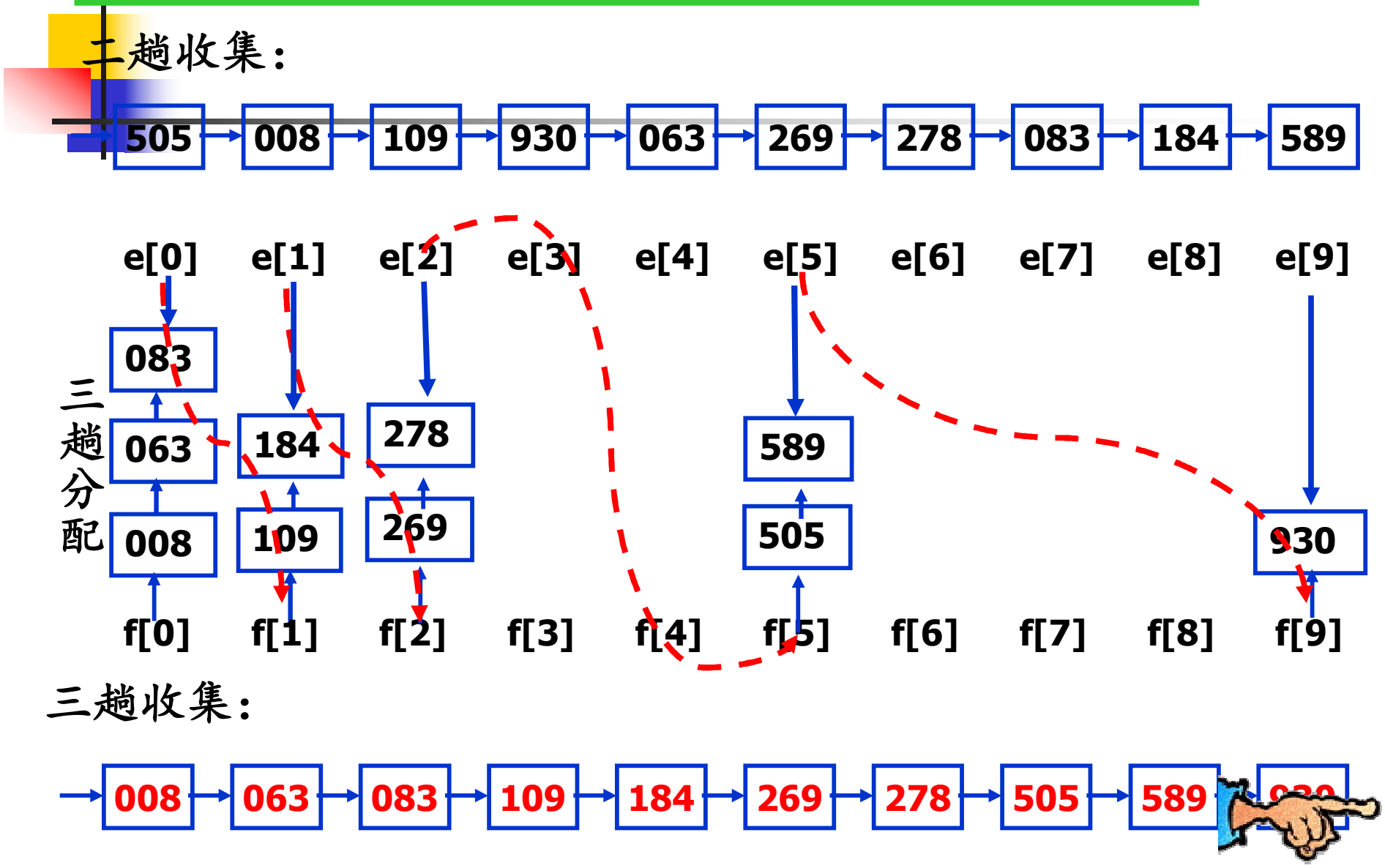
一趟收集：



二趟收集：



链式基数排序 Linked Radix Sort



基数排序

- “分配”和“收集”的实际操作仅为修改链表中的指针和设置队列的头、尾指针；
- 为查找使用，该链表尚需应用算法Arrange 将它调整为有序表。
- 时间复杂度为O(d(n+rd))。其中：
  - 分配为O(n)，收集为O(rd)(rd为“基”)，d为“分配-收集”的趟数
- 空间复杂度为O(rd)