09-D1

9.词典

桶排序

算法

Don't put all your eggs

in one basket.

邓俊辉

deng@tsinghua.edu.cn

简单情况

❖ 给定[0, m)内的n个互异整数，如何高效地排序？ //必有n ≤ m

❖ 借助散列表E[0, m) //各元素仅需1个bit

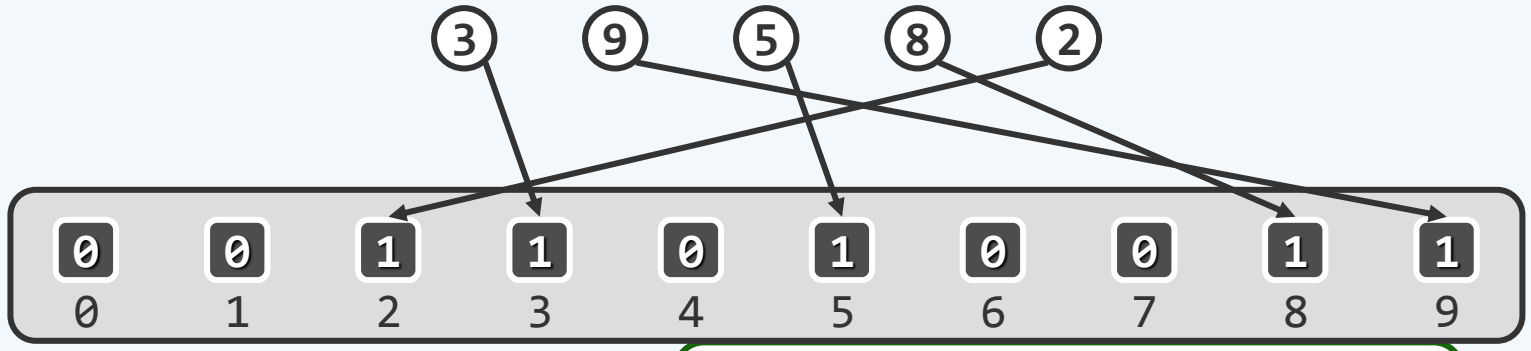
initialization　for i = 0 to m - 1, let E[i] = 0 //O(m)，可优化至O(1)

distribution　for each key in the input, let E[key] = 1 //O(n)

enumeration　for i = 0 to m - 1, output i if E[i] = 1 //O(m)

❖ 空间：O(m)

时间：O(n + m)



一般情况

- 进一步地，若允许关键码重复 //此时未必n ≤ m，甚至可能m << n
  - 清华大学2016级本科生按生日排序：

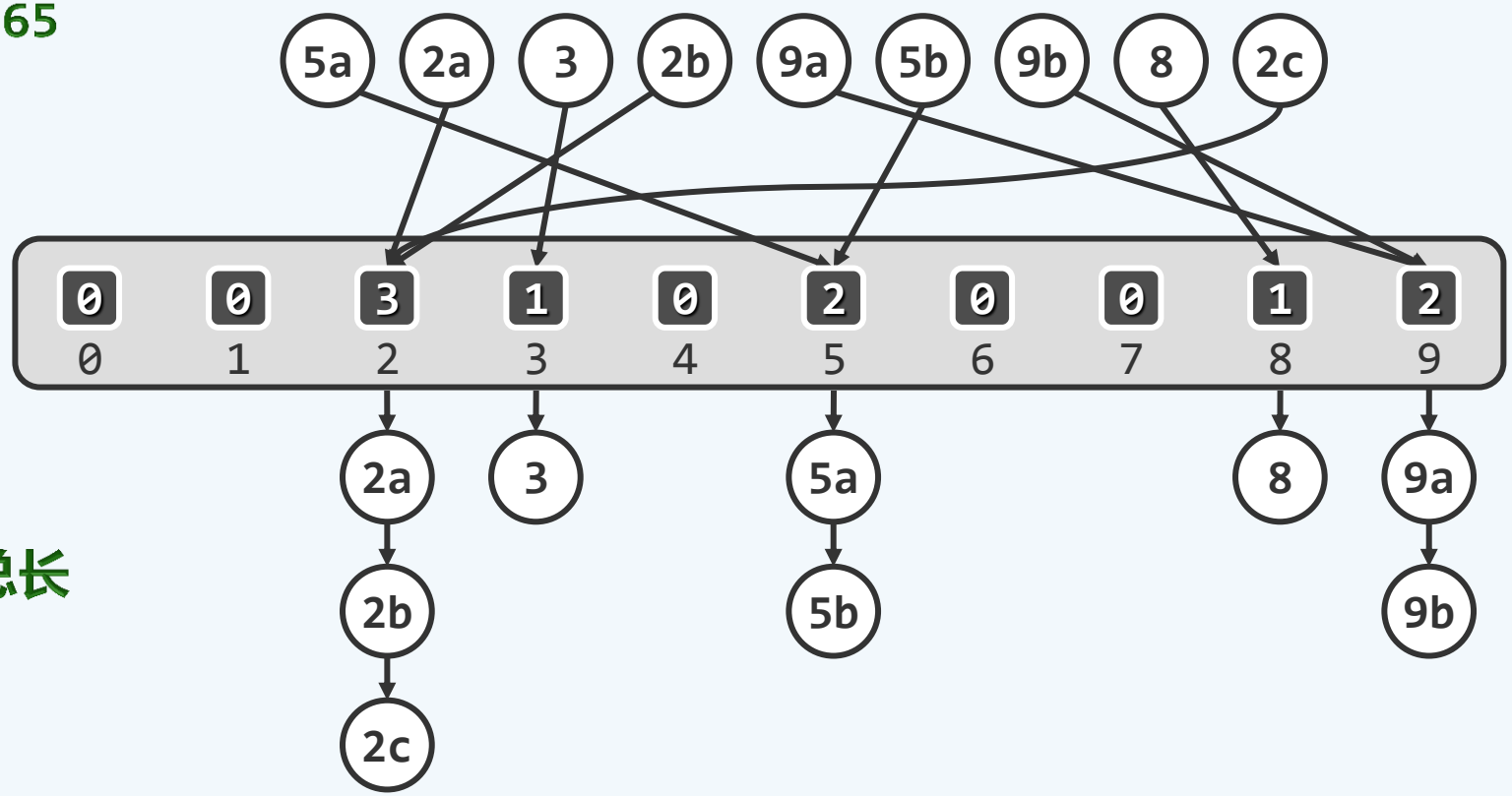
    n = 3300，m = 365

- 依然使用散列表
  - 每一组同义词
  - 各自组成一个链表
- 空间复杂度

  = 散列表长 + 所有链表总长

  = O(m + n)

  //改用向量呢？



一般情况

- `initialization` 初始化散列表（开辟空间、设置各桶的表头） //如有必要，可以优化
  - `distribution` 扫描各词条，**散列**并插至对应桶的链表 //插入位置有讲究
  - `collection` 扫描各桶，**串接**所有非空链表 //串接次序和方向也有讲究
- 只要实现得当，必能保证**稳定**性，即雷同词条的次序与输入相同 //其重要性，远超直觉
- 时间复杂度 = $O(m) + O(n) + O(m) = O(n + m)$
- 大量词条重复时`m << n`，性能接近于**线性**
- 关键码均匀分布时，亦是如此