09-B1

词典

散列函数：基本

此刻他就在占卜，方法是要从办公室到法庭扶手椅座位的步数可以被三除尽，那么新的疗法肯定能治好他的胃炎；要是除不尽，那就治不好。走下来是二十六步，但他把最后一步缩小，这样就正好走了二十七步

邓俊辉
deng@tsinghua.edu.cn

评价标准 + 设计原则

- 确定（determinism）

  同一关键码总是被映射至同一地址

- 快速（efficiency）

  expected-$O(1)$

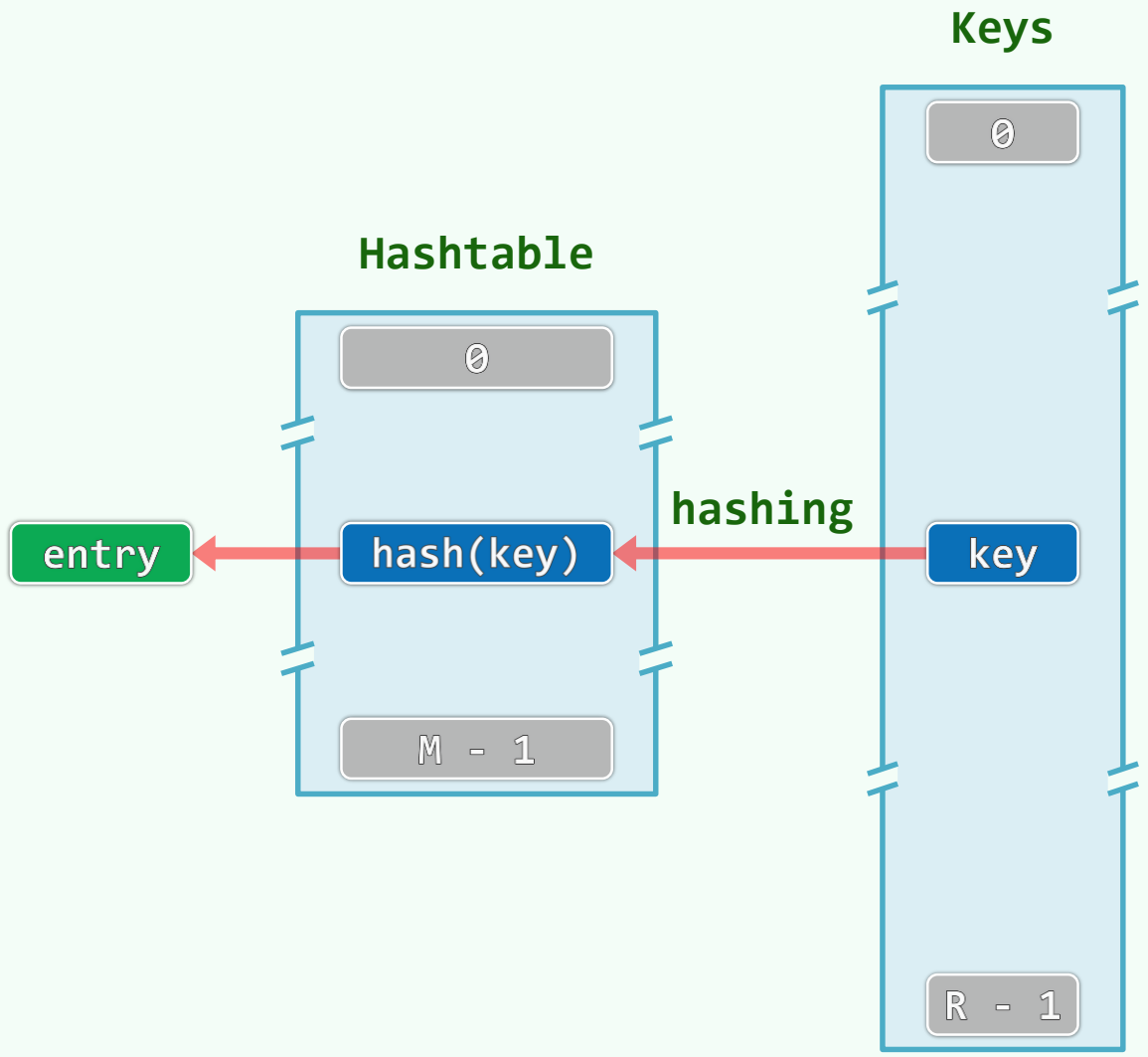
- 满射（surjection）

  尽可能充分地利用整个散列空间

- 均匀（uniformity）

  关键码映射到散列表各位置的概率尽量接近

  有效避免聚集（clustering）现象



除余法

- $hash(key) = key \,\%\, \mathcal{M}$ //前例中，为何选M = 90001?

- 据说：M为素数时，数据对散列表的覆盖最充分，分布最均匀

  其实：对于理想随机的序列，表长是否素数，无关紧要！

- 序列的Kolmogorov复杂度：生成该序列的算法，最短可用多少行代码实现？
  - 算术级数： 7 12 17 22 27 32 37 42 47 ... //单调性 / 线性：从7开始，步长为5
  - 循环级数： 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 ... //周期性：12345不断循环
  - 英文：data structures and algorithms ... //局部性：频率、关联、词根、...

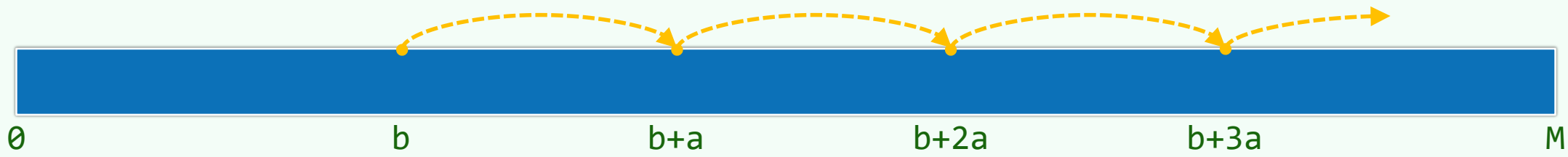
- 实际应用中的数据序列远非理想随机，上述规律性普遍存在：Next-Token Prediction

- 蝉的哲学：面对往往具有周期的天敌/词条，将生命期/散列表长取作素数，可将聚集之概率降至最低

MAD法

❖ 除余法的缺陷

- 不动点：无论表长M取值如何，总有：$hash(0) \equiv 0$
- 相关性：[0,R)的关键码尽管系平均分配至M个桶；但相邻关键码的散列地址也必相邻



❖ Multiply-Add-Divide

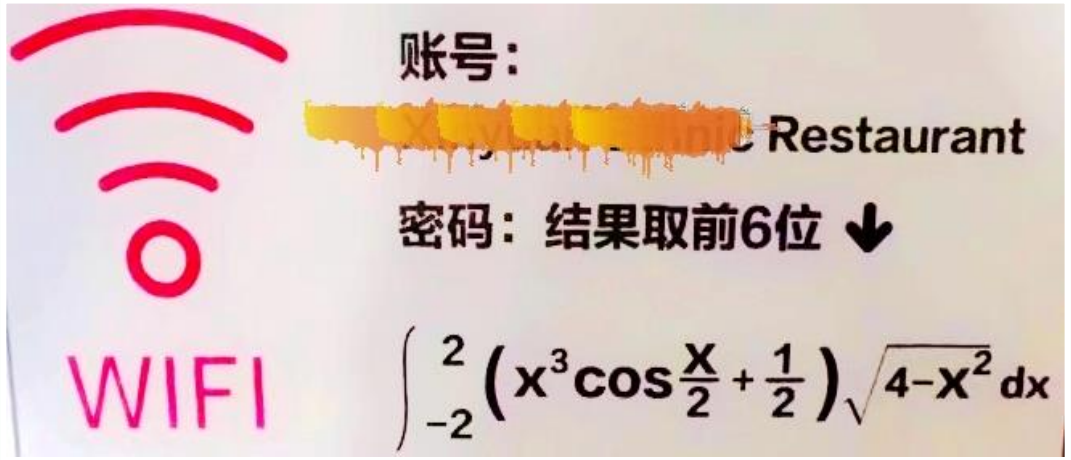
$$hash(key) = (a \times key + b) \,\%\, \mathcal{M} \;,\; \mathcal{M} \text{ prime},\; a > 1,\; b > 0,\; \text{and } \mathcal{M} \nmid a$$

更多散列函数

❖ 数字分析/selecting digits

抽取key中的某几位，构成地址

- 比如，取十进制表示的奇数位
- $hash(\begin{smallmatrix}3&4&5&2&5&5\\1&1&9&6&3&9\end{smallmatrix}) = 345255$



❖ 平方取中/mid-square

取key²的中间若干位，构成地址

- $hash(123) = middle(123 \times 123) = 1\boxed{512}9 = 512$
- $hash(1234567) = 15241\boxed{556}77489 = 556$

更多散列函数

❖ 折叠法/folding：将key分割成等宽的若干段，取其总和作为地址

- hash($_{123}456^{789}$) = 123 + 456 + 789 = 1368 //自左向右
- hash($_{123}456^{789}$) = 123 + 654 + 789 = 1566 //往复折返

❖ 位异或法/XOR：将key分割成等宽的二进制段，经异或运算得到地址

- hash($_{110}011^{011}{}_{b}$) = 110 ^ 011 ^ 011 = 110_b //自左向右
- hash($_{110}011^{011}{}_{b}$) = 110 ^ 110 ^ 011 = 011_b //往复折返

❖ ......

总之，越是随机，越是没有规律，越好