

词典

排解冲突：封闭散列

旅客要在每个生人门口敲叩
才能敲到自己的家门
人要在外面到处漂流
最后才能走到最深的内殿

在我们出生之前，一切都在没有我们的宇宙里开着
在我们活着的时候，一切都在我们身体里闭着
当我们死去，一切重又打开
打开、关闭、打开，我们就是这样

邓俊辉

deng@tsinghua.edu.cn

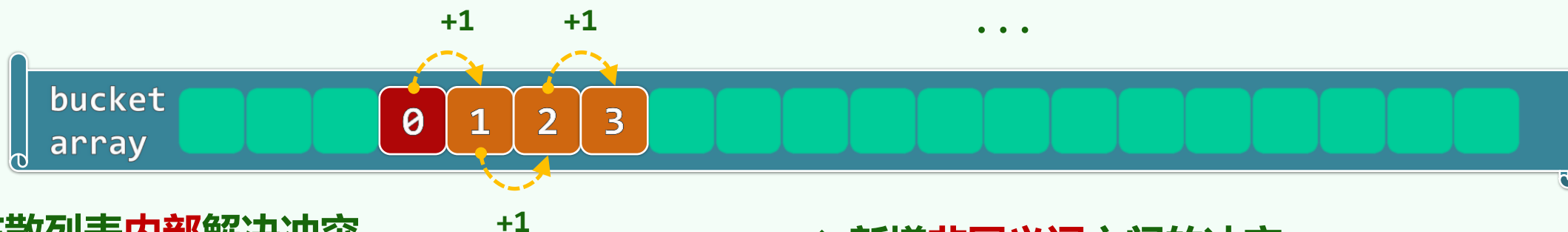
开放定址

- ❖ Closed Hashing, 必然对应于Open Addressing
 - 只要有必要, 任何散列桶都可以接纳**任何**词条
- ❖ Probe Sequence/Chain
 - 为每个词条, 都需**事先约定若干备用桶**, 优先级逐次下降
- ❖ 查找算法: 沿**试探链**, 逐个转向**下一桶**单元, 直到
 - 命中**成功**, 或者
 - 抵达一个**空桶** (存在则必能找到?) 而**失败**
- ❖ 相应地, 试探链又应如何**约定**?

线性试探

❖ Linear Probing

- 一旦冲突，则试探后一**紧邻的桶**
- 直到命中（成功），或抵达空桶（失败）



❖ 在散列表**内部**解决冲突

无需附加的指针、链表或溢出区等
整体结构保持**简洁**

❖ 只要还有空桶，迟早会找到

$$\diamond [\text{hash}(\text{key}) + 1] \% M$$

$$[\text{hash}(\text{key}) + 2] \% M$$

$$[\text{hash}(\text{key}) + 3] \% M$$

$$[\text{hash}(\text{key}) + 4] \% M$$

...

❖ 新增**非同义词**之间的冲突

❖ 数据堆积（clustering）现象严重

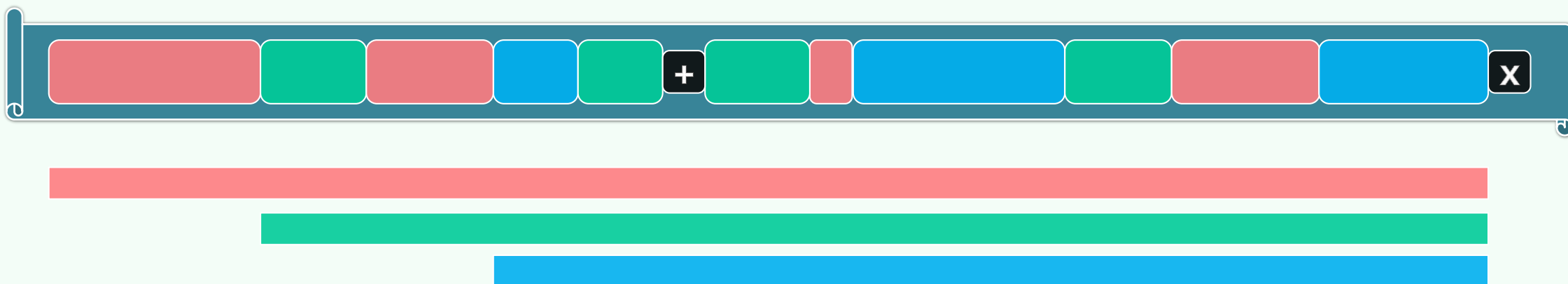
❖ 好在，试探链连续，数据**局部性**良好

❖ 通过装填因子，冲突与堆积都可有效控制

插入 + 删除

❖ 插入：新词条若尚不存在，则存入试探**终止**处的空桶

❖ 试探链：可能因而彼此**串接**、**重叠**！



❖ 删除：简单地清除命中的桶？

经过它的试探链都将因此**断裂**，导致后续词条**丢失**——明明存在，却访问不到

❖ 那么，如何才能**简明**、**高效**地完成删除呢？