Separate chaining

Have a list referenced by each slot of hash table that holds all elements that hash to that slot (one hash function)

e.hashCode()

insert(e) add it to list table[hash(x)]

locate(e) search within list table[hash(x)]

remove(e) remove e from list table[hash(x)]

Resizing hash table

No absolute limit on n/m (could go arbitrarily high) but cost is too high

Open addressing $\alpha < 1$

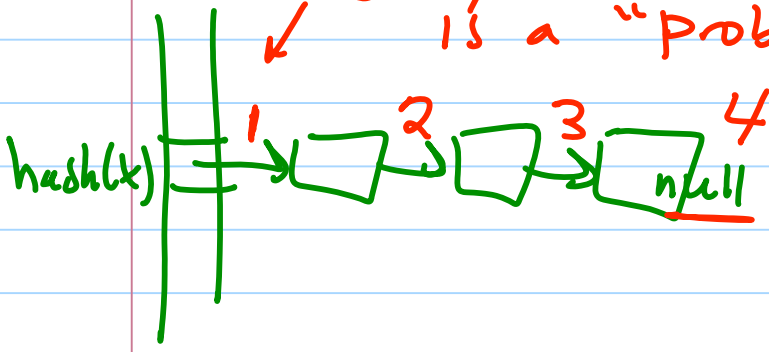
resize upward when α reaches $2\alpha_*$

resize downward when α reaches $\alpha_*/2$

Analysis

Expected cost for unsuccessful search

every reference we follow is a "probe"



expected list length

$$\frac{n}{m} = \alpha$$

$$E\left[\begin{array}{c}\#\text{ probes in an} \\ \text{unsuccessful search}\end{array}\right] = 1+\alpha$$

open addressing

$$\frac{1}{1-\alpha} = 1+\alpha+\alpha^2+\ldots$$

$$E\left[\begin{array}{c}\#\text{ probes in an} \\ \text{successful search}\end{array}\right] = 1+\frac{\alpha}{2}-\frac{\alpha}{2n}$$