Some Possible Hash Functions

For each of these hash functions, think about
- what Strings will cause collisions
- how long it will take to evaluate

Keys: strings of form $s_0 s_1 \ldots s_{k-1}$ (s_i are `chars` in range [0,256])

$h(K) = s_0$

$h(K) = \sum_{i=0}^{k-1} s_i$

6

Linear Probing

First idea: linear probing

h(key) % TableSize full?

Try (h(key) + 1) % TableSize.

Also full? (h(key) + 2) % TableSize.

Also full? (h(key) + 3) % TableSize.

Also full? (h(key) + 4) % TableSize.

…

16

Double Hashing

Instead of probing by a fixed value every time, probe by some new hash function!

h(key) % TableSize full?

Try (h(key) + g(key)) % TableSize.

Also full? (h(key) + 2*g(key)) % TableSize.

Also full? (h(key) + 3*g(key)) % TableSize.

Also full? (h(key) + 4*g(key)) % TableSize.

…

30

Summary

Separate Chaining
- Easy to implement
- Running times $O(1 + \lambda)$

Open Addressing
- Uses less memory.
- Various schemes:
- Linear Probing – easiest, but need to resize most frequently
- Quadratic Probing – middle ground
- Double Hashing – need a whole new hash function, but low chance of clustering.

Which you use depends on your application and what you're worried about.

33