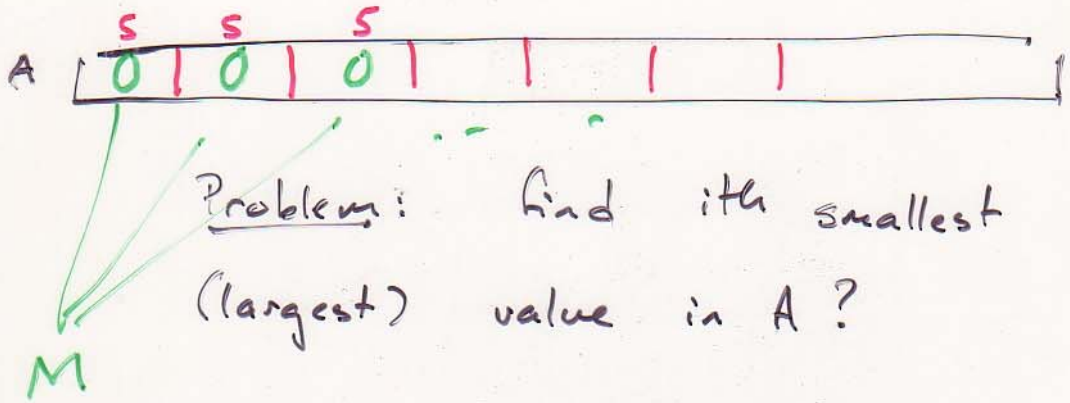
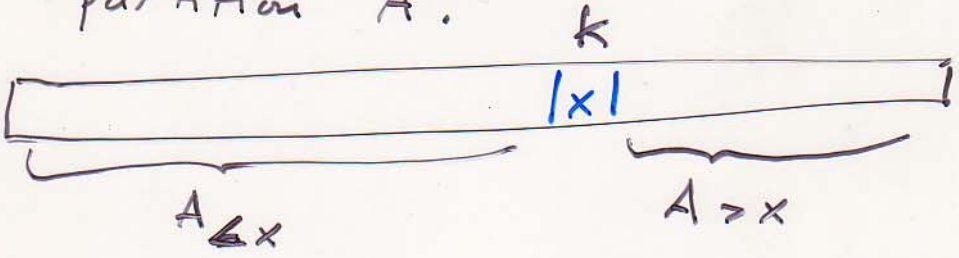
CS 223 Lec 6

Linear Time Selection



Problem: find ith smallest (largest) value in A?

- select median x of M ⇐ rec. call #1
- use x as a pivot to partition A.



- depending on $i < k$ or $i > k$ we recurse on left hand or right hand side... ⇐ rec. call #2.

Running Time Analysis

$T(n)$ = time used to select any position on an input array of size n.

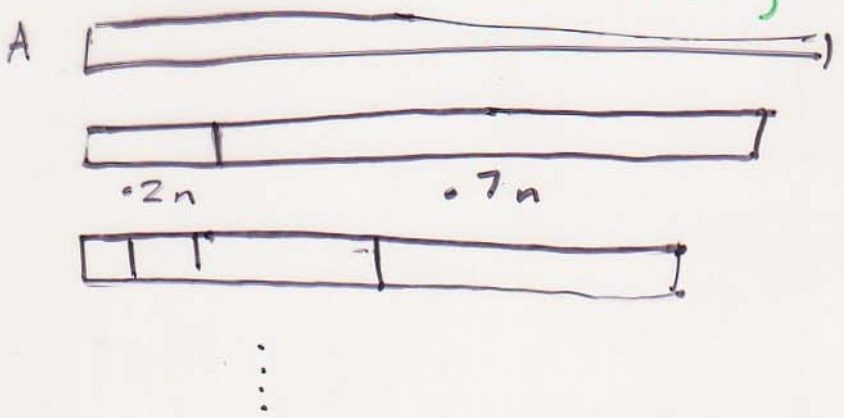
Claim: rec. call #1 rec. call #2

$$T(n) \leq T(\lceil n/5 \rceil) + T\left(\frac{7n}{10} + 6\right) + O(n)$$

$\lceil n/5 \rceil = |M|$ (finding median of M)

partition around x

call history



$O(n)$

$O(.9n)$

$O(.9^2 n)$

$\vdots$

$$O(n) + O(.9n) + O(.9^2 n) + \ldots$$

$$= O(n)\left(1 + .9 + (.9)^2 + (.9)^3 + \ldots\right)$$

geometric series

$$1 + x + x^2 + x^3 + \ldots$$

$$= O(n)\left(\frac{1}{1-(.9)}\right)$$

$$= 10 \cdot O(n)$$

$$= O(n)$$

So

$$T(n) = O(n).$$

Hash Tables

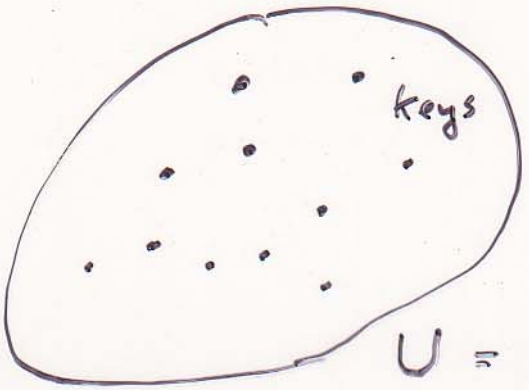
- for supporting dynamic sets of elements
- each element has a key value (not necessarily unique)

OPERATIONS

INSERT
DELETE
SEARCH

Direct-access table

use a simple array that is indexed by set of key values



U = universe of (potential) key values

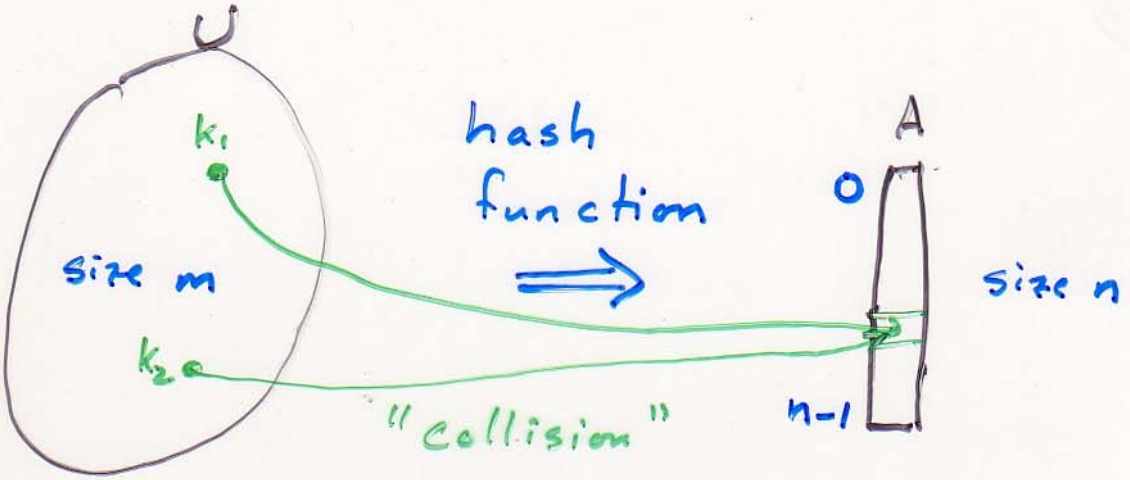
example

U = positive integers $\leq 10^9$

K = the actual keys used in an application

typically $|K| \ll |U|$

Hash Tables



Example

$$h(x) = x \bmod n$$

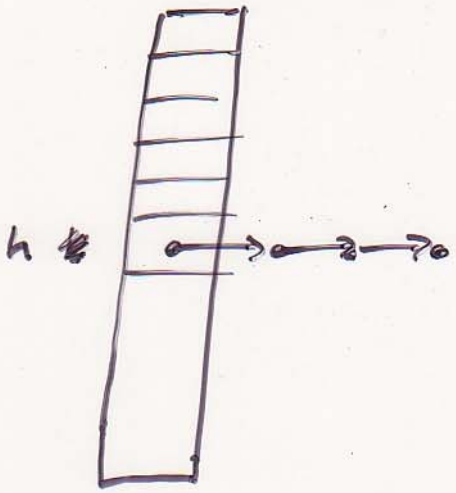
↑

assuming x is a positive integer.

How to deal with collisions?

"Chaining"

- use a linked list at each position in the hash table to store multiple elements ...



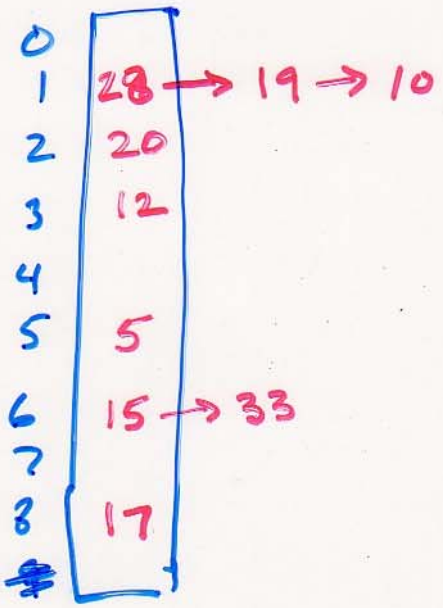
Worst-case behavior?

is bad, it occurs when all keys have the same hash value..

Example

Insert keys 5, 28, 19, 15, 20, 33, 12, 17, 10 into a hash table with 9 slots

$h(k) = k \bmod 9$. (use chaining)



Analysis of Hash Tables

∉ suppose

1. hash table has size m
2. there are n elements to store.

Defn the **load factor** is $\frac{n}{m}$

Assumption

simple uniform hashing

- $\Pr\left[\begin{array}{l}\text{a given key}\\ \text{hashs to}\\ \text{location } i\end{array}\right] = \frac{1}{m}$

- hashing is independent.