**Admin:** PS #2 out

6.006
Rivest
L6.1
9/22/08

**Reading:** CLRS 11.1-3, 17

**Outline:**

- ☐ Computing a hash function
- ☐ Resizing a hash table
- ☐ Rabin-Karp string-matching & "rolling hashes"

# How to compute $h(x)$?

Lots of ways: here's one that's good
assume $x$ is an integer
let $m$ be hash table size
let $p$ be prime, $p \geq m$ (ok if $p=m$ if prime)
pick $a$: $0<a<p$
pick $b$: $0 \leq b<p$
$\Big\}$ can choose randomly
let

$$h(x) = ((ax+b) \bmod p) \underbrace{\bmod m}_{\text{not needed if } p=m}$$

**example:**

$m = 1{,}000{,}000$
$p = 1{,}000{,}003$
$a = 314159$
$b = 271828$

$x \leftarrow x \bmod p$
$y \leftarrow (a \cdot x + b) \bmod p$
output $y \bmod m$

If $x =$ "ATTGCATA" treat as base-4 integer
If $x =$ "weather" treat as base-26 integer
Note: can compute $x \bmod p$ as first step: $h(x) = h(x \bmod p)$
Note: if $p$ reasonably large, can use same $a, b, p$
with tables of different size $m$

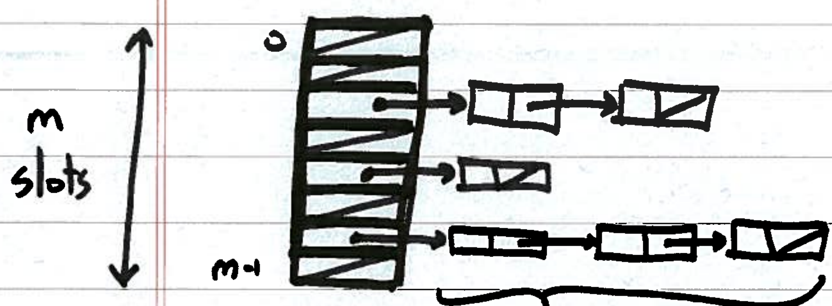
See text for other methods (division method, multiplication method).

# Resizing a hash table (Ref. Chapter 17)

6.006
Rivest
L6.3
9/22/08

(also applies to resizing arrays in general...)

Imagine we are using chaining for collision resolution




n keys arranged in m lists (some are empty)

average list length = load factor = $\alpha = n/m$

$\Rightarrow$ Want $m = \Theta(n)$ at all times

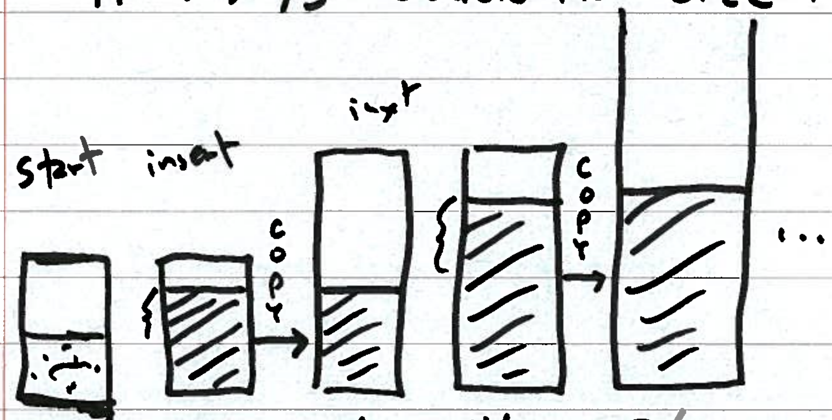
$\Rightarrow$ Don't know how large n might get to... what m to use?

... m too large : costly to create, wasteful

... m too small : slow to search, as lists get long

$\therefore$ want to dynamically adjust m as appropriate

if $\alpha \geq 4/5$ : double table size : so $\alpha = 2/5$ afterwards




$\alpha$ ? $4/5$ $2/5$ $4/5$ $2/5$

so $2/5 \leq \alpha < 4/5$ always (assuming we are always inserting, never deleting)

## Analysis

6.006
Rivest
L6.4
9/22/08

"Amortized analysis"

- cost of one insert can be large, since we might have to copy entire table!

- So, let's look at cost $T(n)$ for a sequence of $n$ inserts; then $T(n)/n$ is "average" (or amortized) cost per insert

Suppose $m=5$ initially

$T(1) = 1$

$T(2) = T(1)+1 = 2$

$T(3) = 3$

$T(4) = 3+1+4 = 8$ (copy 4)

$T(5) = 9$

$T(6) = 10$

$T(7) = 11$

$T(8) = 11+1+8 = 20$ (copy 8)

$$\text{if } n=2^k:\ T(n) = \underbrace{n}_{\text{inserts}} + \underbrace{(4+8+16+\cdots+n)}_{\text{copying}}$$
$$\le n + 2n = 3n$$

so average cost per insert is $\le 3$ per insert (worst-case in an amortized sense: $n$ inserts never take more time than $3n$)

**Or:** When we insert an element, we pay 1 now, and set aside 2 "units of work" to do later. (savings) When savings account is big enough to copy entire table over, do it! (Illustrate)

# What about deletions?

If $\alpha \le 1/5$, halve table size
(so $\alpha$ becomes $2/5$).

Can do both insertions & deletions;
amortized cost per operation is still $\le 3$.

(Example: if we decrease $n$ from $m \cdot 2/5$ to $m \cdot 1/5$,
we have done $m/5$ deletions. We pay $m/5$ for those
deletions, and put $2 \cdot (m/5)$ in the bank. That more then
pays for putting those remaining $m/5$ elts in a new, smaller, table.)

**Exercise:** why don't we halve table size when $\alpha < 2/5$?
(instead of $1/5$)?

# String-matching (Rabin-Karp) & "rolling hash"

6.006
Rivest
L6.6
9/22/08

Given: pattern $P[1..m]$ ⎫
text $T[1..n]$ ⎭ of chars

does $P$ occur in $T$?

e.g. find "ATG" (corresponds to "start codon") in "AATCGC... "

**Idea:**

1. compute hash($P$)
2. for each length-$m$ window of $T$ (i.e. for each $T[i..i+m-1]$)

A A T C G C

compute hash$(T[i..i+m-1])$ & compare to hash($P$)
if $=$ : check to see if they really match
if $\neq$ : move on to next $i$

Want hash function s.t. we can go from one window to next easily! Want to be able to compute

$T[i+1..i+m]$

easily, given

$T[i..i+m-1]$

"rolling hash"

How?

(continuing DNA example) [A=0, C=1, G=2, T=3]

pick prime $p = 1009$; hash = string mod $p$

suppose $m = 9$

$T[i \ldots i+8] =$ C T A T T A C G T

$= 130330123_4$ (base 4)

$= 184091_{10}$ (base 10)

$= 453$ mod $p$

what is effect of dropping high-order C?

C in high order has value $1 \cdot 4^8$

$= 1 \cdot \underline{960}$ mod $p$

precompute!

so hash(TATTACGT) $= 453 - 960$

$= 502$ mod $p$

What is effect of shifting left & appending "G" on right?

multiply by 4 & add 2

hash(TATTACGTG) $= 4 \cdot 502 + 2$

$= 1001$ (mod $p$)

$\left.\begin{array}{l}\text{one subtract}\\ \text{one multiply}\\ \text{one add}\end{array}\right\}$ to move over to next window