Map as Hash Table

20144987 4 4 4 CLT OAK X 17 1.00 X
20144756 2 4 4 CLT BOS X US 1.00 X
201441020 2 4 4 CLT MCO X US 1.00 X
201442094 3 4 4 CHS CLT - US 1.00 X
20144214 3 4 4 IAH CLT - US 1.00 X
201441020 1 4 4 AVL CLT - 16 1.00 X
201441110 3 4 4 PHX OAK X US 1.00 X
20144794 3 4 4 CLE CLT - 16 1.00 X
201441020 5 4 4 BNA PHL - YX 1.00 X
201442030 3 4 4 DCA BHM X US 1.00 X
201441020 3 4 4 MCO CLT - US 1.00 X

contains key: compute hash(key)/% slots
set: compute hash(key)/% slots
put: compute hash(key)/% slots
get: compute hash(key)/% slots

collision by hash(key)/% slots
collision by hash(key)/% slots
collision by hash(key)/% slots

hash table w/ chaining

contains key: compute hash(key)/% size
if NULL at that location, return NO
if something there, do comparison
resolve collisions on put

hash table

hash fan should disperse keys evenly

contains key (OAK)

load factor

total keys in map

# of slots

78009 % 16 = 9

(expected) running time = O(load factor)
expected running time = $O(\text{load factor})$
if $m \geq n$

$= O(1)$

worst case $O(n)$
(all keys collide in same slot)

hash table - open addressing
linear probing

1. Compute hash
2. Compute $k_i$
3. Search through slots $h_i\ldots h_i+k_i-1$
   until key found or empty
   or wrapped all the way around

if $d \leq \frac{1}{2}\quad \Rightarrow \quad O(1)$ expected

if $d \leq \frac{1}{2}\quad \Rightarrow \quad O(1)$ expected

O(1) expected

Oct 15, 2020 Page 2
long fib_rec(int n) {
    if (n < 2) {
        return n;
    } else {
        return fib_rec(n - 1) + fib_rec(n - 2);
    }
}

outcome_class(p) {
    if p is end of game
        return value according to rules
    else
        S ← positions reachable in 1 move from p
        if S contains a losing position
            return win, and correspondingly max
        else
            return lose
}

Chomp

Play on m x n grid. Take turns selecting remaining cell, remove all above and to right
Last move loses.

5
4
3
2
1
A
B
C
D
E
F

pos in finite impartial game

long fib_rec(int n) {
    if (n < 2) {
        return n;
    } else {
        return fib_rec(n - 1) + fib_rec(n - 2);
    }
}

Chomp

outcome_class(p) {
    if p is end of game
        return value according to rules
    else
        S ← positions reachable in 1 move from p
        if S contains a losing position
            return win, and correspondingly max
        else
            return lose
}

Chomp

Play on m x n grid. Take turns selecting remaining cell, remove all above and to right
Last move loses.

5
4
3
2
1
A
B
C
D
E
F

pos in finite impartial game

outcome_class(p) {
    if p is end of game
        return value according to rules
    else
        S ← positions reachable in 1 move from p
        if S contains a losing position
            return win, and correspondingly max
        else
            return lose
}

Chomp

Play on m x n grid. Take turns selecting remaining cell, remove all above and to right
Last move loses.