Hash tables

Π "compartments"

N elements

ex 1: keys are integers

$$h(k) = k \,\%\, \Pi$$

ex 2: reals in $[0, 1)$

$$h(k) = \text{int}(\Pi k)$$

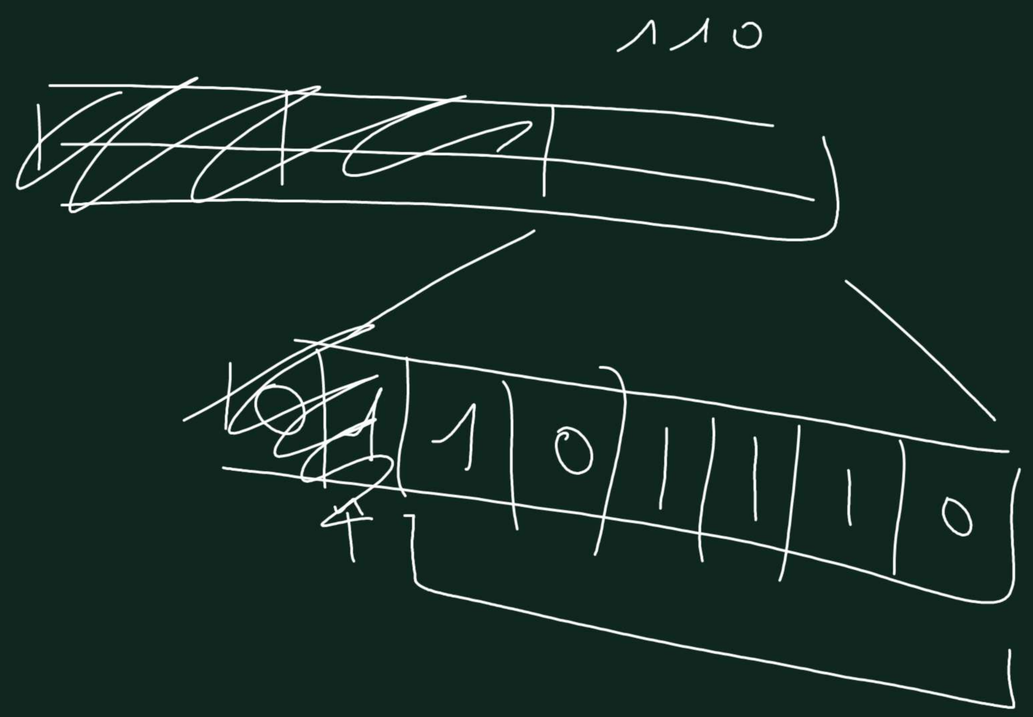
$$h : \mathcal{K}(\text{set of keys}) \longrightarrow [1 \,..\, \Pi] \quad (0 \,..\, \Pi - 1)$$

- $h \to$ in $0 \,..\, \Pi - 1$
- equally distributed over $0 \,..\, \Pi - 1$
- easy to compute

counter-example:

keys: 3 char-words (24 bits) ~~$n = 64$~~ n should be prime

$h(k) = k \% n$

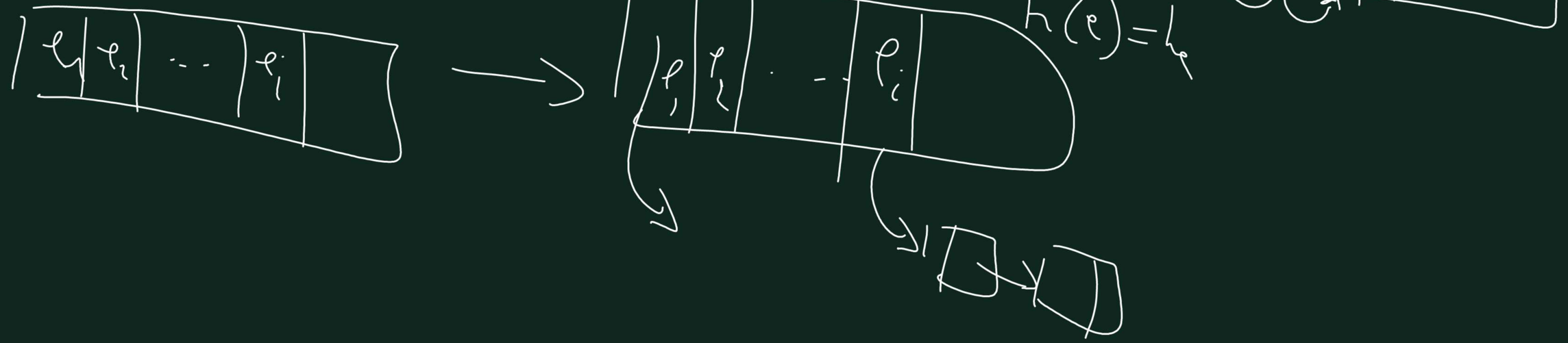


$card(K) >> n$ sometimes

→ collision: 2 different keys have the same h-value

2 approaches to cope with this issue:

- separate chaining
- linear probing



def: load factor of a hash-table: $\frac{N}{M}$

separate chaining
$N < M$
$N > M$ ok

linear probing
$N < M$
$N \ll M$

insert "EASYQUTION" in an initially empty table

ex: separate chaining

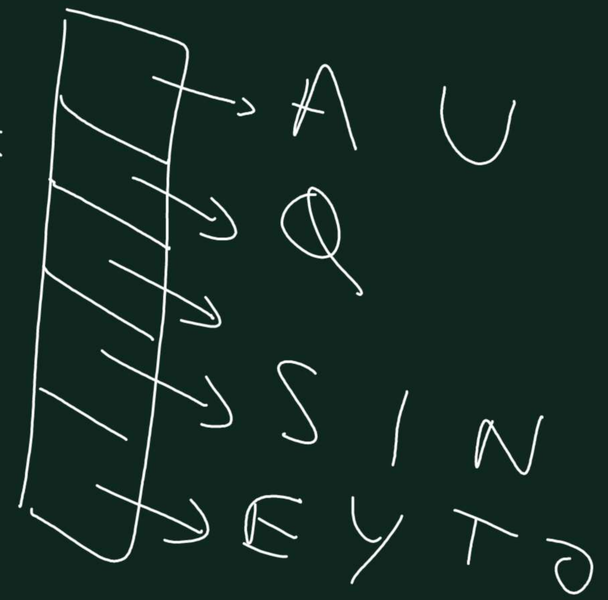
keys: letters ($0 \to 25$)

$M = 5$

$h(k) = 11k \% 5$

k	E	A	S	Y	Q	U	T	I	O	N
h(k)	4	0	3	4	1	0	4	3	4	3

HT:



load factor: $\frac{10}{5} = 2$

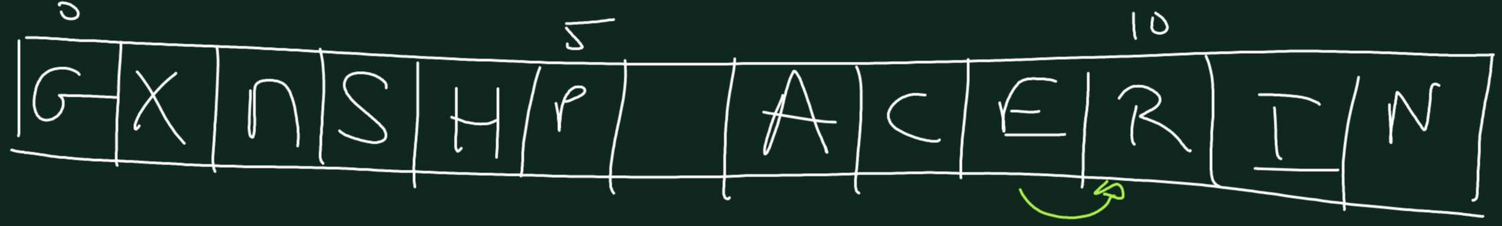
length of lists

search is $\Theta(1)$

linear probing:

	A	S	E	R	C	H	I	N	G	X	M	P
h	7	3	9	9	8	4	11	7	10	12	0	8

M : 13



→ we should always have a load factor <= 1

load factor	1/2	2/3	3/4	9/10
successful search	1.5	2	3	5.5
unsuccessful	2.5	5	8.5	55.5

exercice

linear probing, $\Pi = 2N$

1) what is the best (worst) placement for elements in the table?

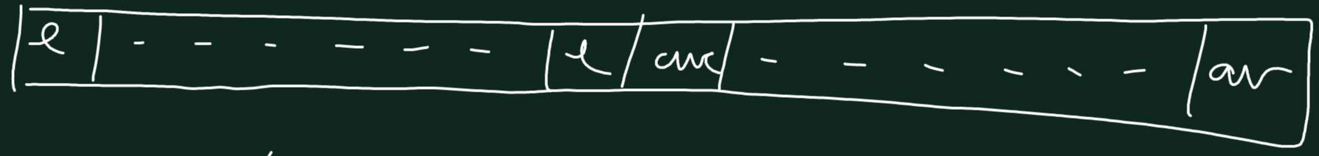
2) ——— average length of clusters in both cases?

3) average number of probes (unsuccessful search), in the best-case, is independent from N and Π

4) average number in worst case is $\simeq N/4$

best case 

$\bar{l} = 1/2$

worst case 

$\bar{l} = N/2$

avg number $\frac{1}{2^N}\left(N + N-1 + N-2 + \dots 1\right) = \frac{N(N+1)}{2 \times 2N} \sim \frac{N}{4}$

---

best: $\frac{1}{2^N}\left(2+1+2+1+\dots+2+1\right) = \frac{3N}{2N} = \frac{3}{2}$

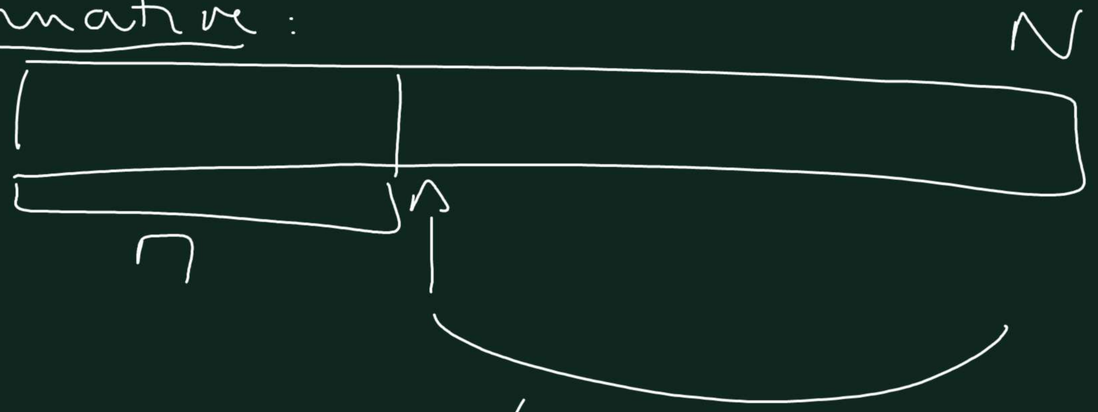
Comparison AVL / H_table for search

HT: $\Pi = 20000$, separate chaining

N	t(AVL) (ms)	t(HT)
5000	22.88	5.25
10000	32.88	5.5
20000	52.88	6
40000	52.88	7
80000	62.88	9
160000	72.88	13
320000	82.88	21
640000	92.88	37
1280000	102.88	69
2560000	112.88	133
5120000	122.88	261

Pb: find highest Π values in an array of size N
(is it worth sorting the array? Why?) ④ $(N \log N)$

alternative:



$$(N-\Pi) \times \Pi \sim N\Pi = \textcircled{4}(N)$$

extension to ADT List

with orderedInsert operation:

$L : (A \quad D \quad G \quad K)$

$\text{orderedInsert}(L, F) \rightarrow (A \quad D \quad F \quad G \quad K)$

ADT extension List

Use Element ($\leq$)

Operations

$\text{orderedInsert} : \text{List} \times \text{Elment} \rightarrow \text{List}$

Axioms

A_1 $\text{orderedInsert}(new, e) \equiv cons(e, new)$

A_2 $\text{orderedInsert}(l, first(l)) \equiv l$ (if duplicates forbidden)

A_3 $e \leq first(l) \Rightarrow \text{orderedInsert}(l, e) \equiv cons(e, l)$

A_4 $\text{orderedInsert}(l, e) \equiv cons(first(l), \text{orderedInsert}(rest(l), e))$

1.8€? no

1.9€? y

from (0)

canPay? wallet and price → y / y N

1) choose more params if needed
2) write code of function
3) evaluate its complexity in the worst case

```
canPay(w, price, from) {
    if (price == 0) return true
    if (from == w.length - 1) return (price == w[from])
    if (canPay(w, price - w[from], from + 1)) return true
    return canPay(w, price, from + 1)
}
```

initial call: from = 0 (first elt)

param: N, number of coins

init op: —

$C(1) = 1$

$C(n) = 2(C(n-1) + 1)$

$$C(1) = 1$$

$$C(n) = 2\,C(n-1) + 1$$

$$C(n) + 1 = 2\left(C(n-1) + 1\right)$$

$$D(n) = C(n) + 1$$

$$\begin{cases} D(1) = 2 \\ D(n) = 2\,D(n-1) \end{cases}$$

$$= 2^2 D(n-2)$$

$$\vdots$$

$$= 2^i D(n-i)$$

$$i = n - 1$$

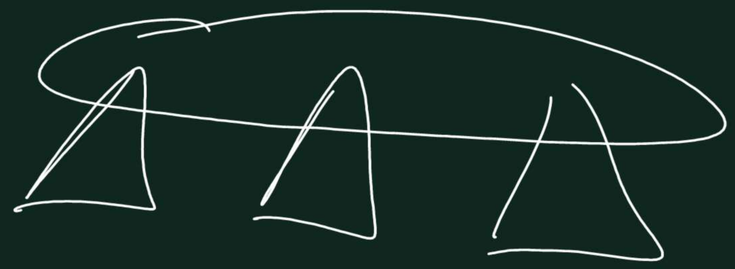
$$D(n) = 2^{n-1} D(1) \times 2$$

$$= 2^n$$

$$\Rightarrow C(n) = 2^n - 1$$

$$= \mathcal{O}(2^n)$$

define an extension to ADT Tree (& Forest)
with the max Children operation:
(ex. maxChildren(t) = 5)



extension ADT Tree, Forest

Operations

maxChildren : Tree → Integer
maxChildrenF : Forest → Integer

Axioms

A_1 maxChildren(new(e, f)) ≡ max(nTrees(f), maxChildrenF(f))
A_2 maxChildrenF(newF) ≡ 0
A_3 maxChildrenF(addTree(f, t, n)) ≡ max(maxChildren(t), maxChildrenF(f))

extension to ADT List with a subList(l, from, to) operation (total)

(hyp: indexes out of list bounds are ignored)

ADT extension List

Operations

subList : List × Integer × Integer → List

subList(A B C D, 1, 2)

Axioms

A_1 subList(nil, from, to) ≡ nil

A_2 to < from ⇒ subList(l, from, to) ≡ nil

A_3 subList(l, 1, to) ≡ Cons(first(l), subList(rest(l), 1, to - 1))

A_4 subList(l, from, to) ≡ subList(rest(l), from - 1, to - 1)

A_3 Cons(A, subList((B C D), 1, 1))

A_3 Cons(A, Cons(B), subList((C D), 2, 2))

$l = (A \quad B \quad C \quad D)$

sublist$(1, 2, 3)$

A_4 sublist$((B \quad C \quad D), 1, 2)$

A_3 cons$(B,$ sublist$((C \quad D), 1, 1))$

A_3 cons$(B,$ cons$(C,$ sublist$((D), 1, 0)))$

A_2 cons$(B,$ cons$(C,$ nil$)) \equiv (B \quad C)$

each step, score is incremented by the product of sizes of the subPiles just created

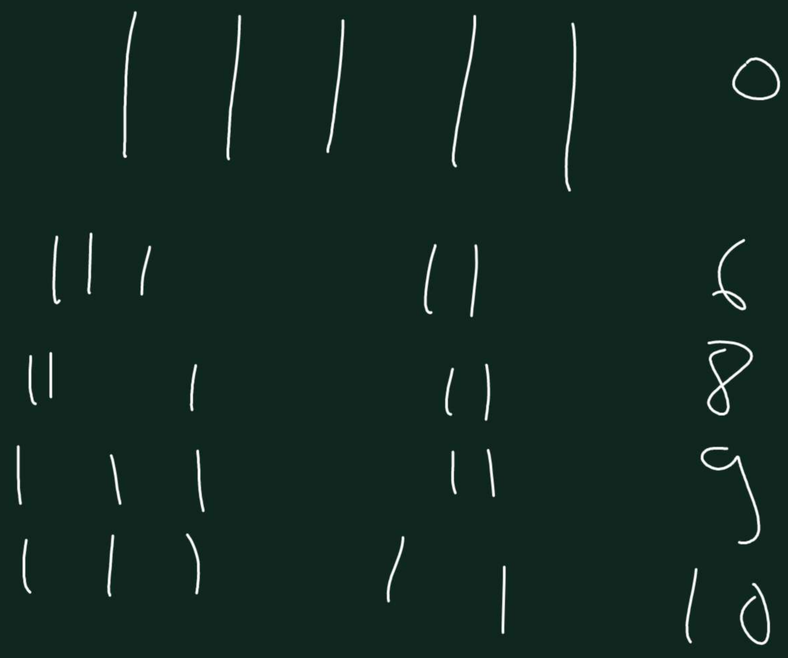
Q: best possible score? $S(n) = ?$

guess: $S(n) = \frac{n(n-1)}{2}$

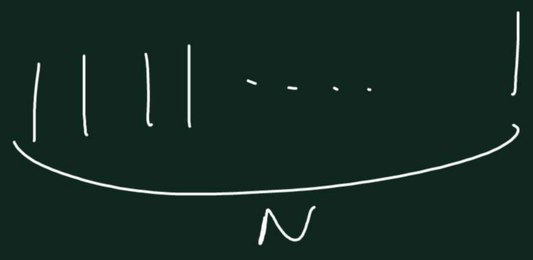
$P(n)$: $S(n) = \frac{n(n-1)}{2}$

$P(1), P(2) \ldots P(6)$ all $\top$

n	S(n)
1	0
2	1
3	3
4	6
5	10
6	15



hyp : $\forall i < n,\ p(i) = T \iff \forall i < N,\ S(i) = \frac{i(i-1)}{2}$



$$S(i) + S(N-i) + i(N-i)$$

Soit :

$$= \frac{i(i-1)}{2} + \frac{(N-i)(N-i-1)}{2} + i(N-i)$$

$$= \frac{i^2}{2} - \frac{i}{2} + \frac{N^2 - 2iN + i^2 - N + i}{2} + i(N-i)$$

$$= \frac{N^2 - N}{2} = \frac{N(N-1)}{2}$$

qcd