Dynamic Sets with Dictionary Operations

- Insert
- Delete
- Search

Can we do better ?

Hash Tables

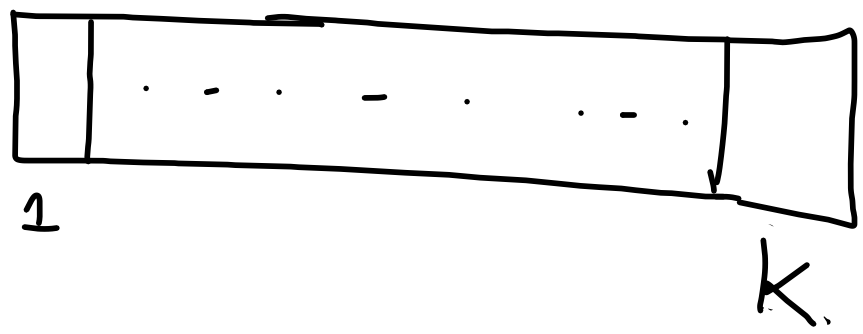
Possible Data Structures

	Insert at end	Search	Delete
Array	$O(1)$	$O(n)$	$O(n)$
Sorted Array	$O(n)$	$O(n)$	$O(n)$
Linked List	$O(1)$	$O(n)$	$O(n)$
DLL	-	-	/
BST	$O(\log n)$	$O(\log n)$	$O(n)$
(balanced)			$O(\log n)$

Direct Address Tables

(universe of keys)

Keys: $1, \ldots, k$



Universe of keys is very large

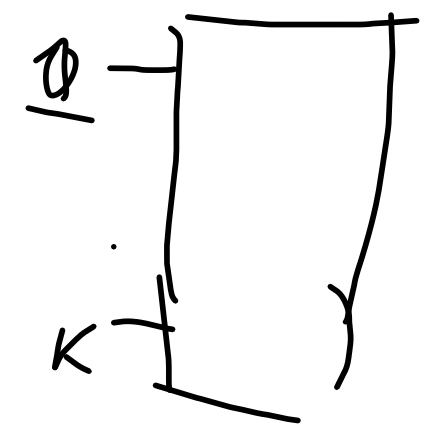
- Initially all 0
- Insert (i) ⇒ A[i] = 1
- Delete (i) ⇒ A[i] = 0
- Search(i) ⇒ if A[i] == 1 Yes

} O(1)

Universe of Keys very Large (K)

- Dictionary size very less (M)

$K \gg M$



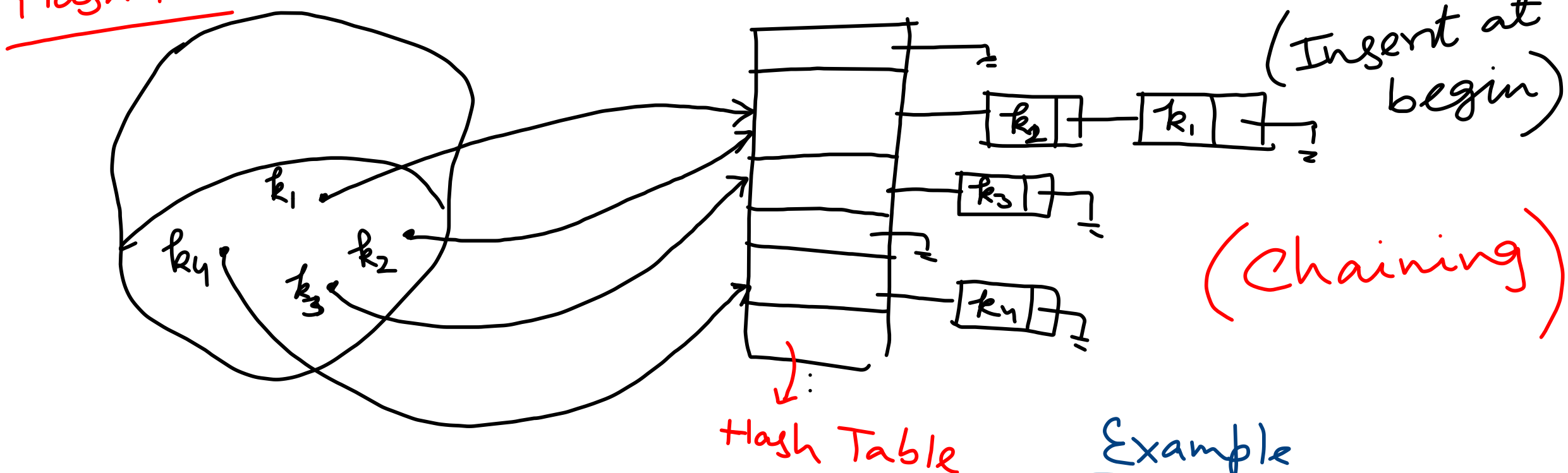
Use Hash

$H: \{1, \ldots, K\} \rightarrow \{1, \ldots, M\}$ { Dictionary size ≤ 10

- Insert (i) : A[H(i)] = 1
- Delete (i) : A[H(i)] = 0
- Search (i) : — —

} Collision

Hash Table



(Insert at begin)

(chaining)

Insertion → $O(1)$

Deletion → Order of the list

Search → Order of the list

Example

$H(i) = i \bmod M$

Key: $i, i+M, i+2M, \ldots$

Delete / Search → $O(n)$

Universal Hashing

Choose hash function $H: K \to \{0, \ldots, M\}$ s.t.

$\forall\, k_1, k_2 \in K$,

$\#\, h \in H$ for which $h(k_1) = h(k_2)$ is at most $\frac{|H|}{M}$.

$p \to$ prime

$a \in \mathbb{Z}_p^*$

$b \in \mathbb{Z}_p$

$$h_{a,b}(k) = ((ak + b) \bmod p) \bmod M$$

→ Universal Hash

Universal Hashing & we use that in Hash Table

⇓

Search Complexity $\Theta(1+\alpha)$

↳ $\alpha = \frac{N}{M}$ → # elements in the dictionary

M → Size of Hash Table

{ N may be greater than M

Proof HomeWork

Hash Table

Resolve Collision

- Chaining
- Open Addressing

} Expected → Constant time
} Worst → Linear

Open Addressing

- Store all the key values in the hash Table itself
- $N \leq M$
- Examine (Probe) the hash table until it finds an empty location.

$h: K \times \{0, \ldots, M\} \rightarrow \{1, \ldots, M\}$

Probe Sequence: $\langle h(k,0), h(k,1), \ldots, h(k,m-1) \rangle$

$\forall k \in K, \quad h(k,i) \neq h(k,j)$

Insert (k_1)

- Check if $h(k_1, 1)$ is empty or not
  - If empty, insert at that place.
  - o/w, check if $h(k_1, 2)$ is empty or not

How to Define h?

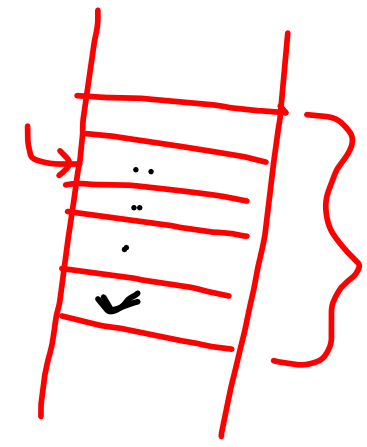
- Linear Probing
- Quadratic Probing
- Double Hashing

Linear Probing

$$h(k,i) = (h'(k) + i) \bmod M + 1$$

h' → any auxiliary hash function

- Easy to implement
- Primary Clustering



Quadratic Probing

$$h(k,i) = (h'(k) + c_1 \cdot i + c_2 \cdot i^2) \bmod M + 1$$

What if $h(k_1,0) = h(k_2,0) \Rightarrow$ Secondary Clustering

-

Double Hashing

$$h(k,i) = \left(h_1(k) + i \cdot h_2(k)\right) \bmod M + 1$$

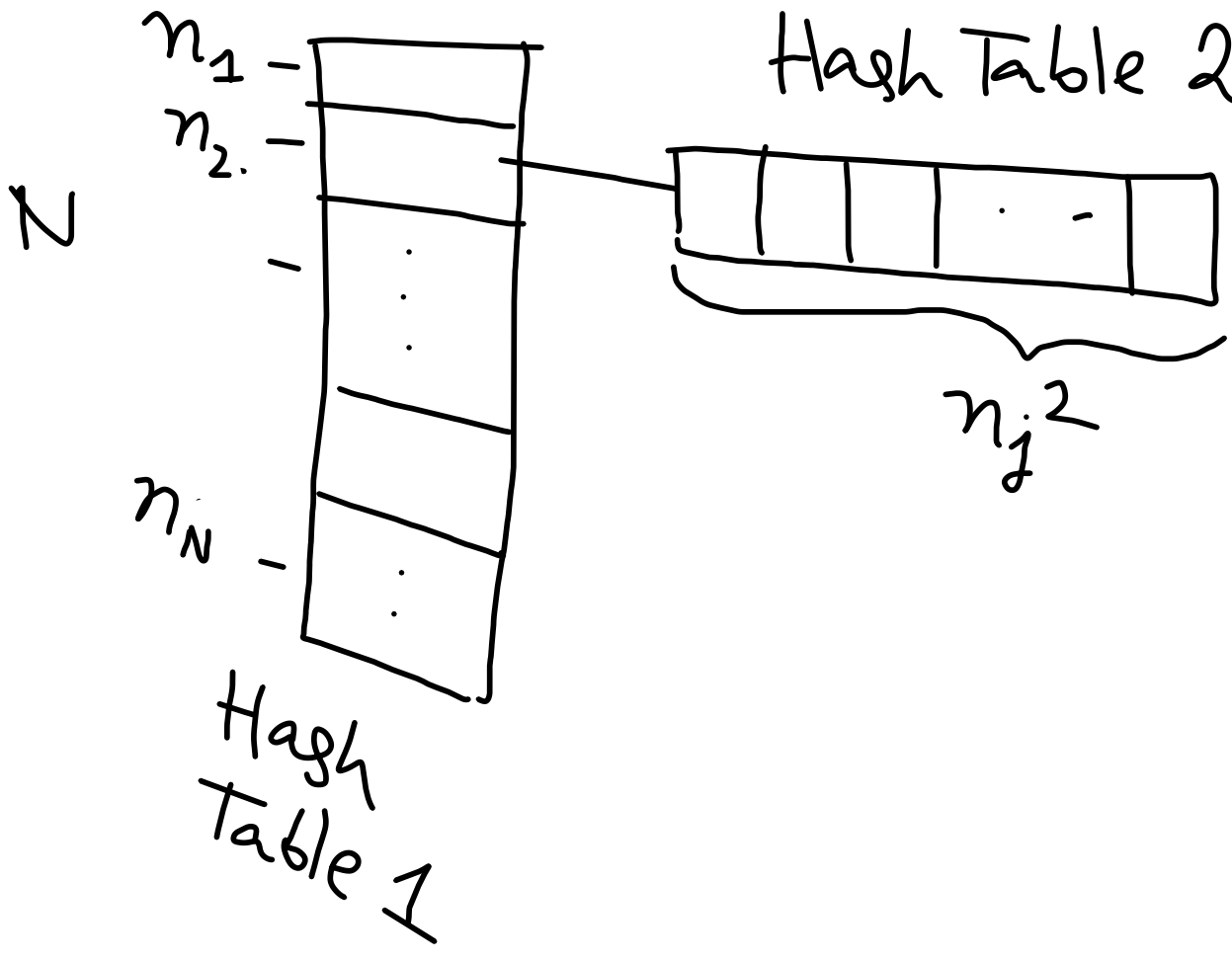
If h is universal hashing

* Insertion → $\Theta\left(\frac{1}{1-\alpha}\right)$ $\qquad \alpha = \frac{N}{M}$

  Search → $\Theta\left(\frac{1}{\alpha} \ln \frac{1}{1-\alpha}\right)$

Perfect Hashing

- Fixed keys
- Worst Case Constant time



Space

$E[\sum n_i^2] < 2n$

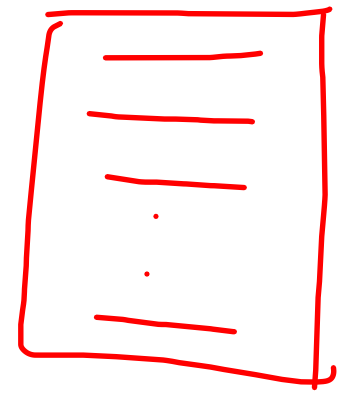
Project

Wordle

- Implementation
- Strategy

  - Agnita, Sangita
  - Puja, Shreya
  - Sougata, Saikat
  - Tamojit, Rajdeep

File / Dictionary



- First digit N
- No E
- No I
- One digit S