Lecture 26

Wednesday, 13 October 2021 10:02 AM

Two Algorithms / Applications using Modular Arithmetics —— (mod p)

↑ prime

→ Universal Hashing

→ String / Pattern Matching (smaller space ?)

Let p be prime, take $r \in [1, p-1]$

$$F : \underbrace{z}_{[1,p-1]} \longrightarrow \underbrace{(z\, r)}_{[1,p-1]} \bmod p$$

$\left(\frac{3r}{p}\right) \neq \text{int}$

$\frac{4 \cdot 4}{8} = \text{int}$

$\frac{4 \cdot 4}{7} \neq \text{int}$

Properties

- 1-1 / Invertible
- If r is rndm $\Rightarrow$ For any z, $F(z)$ is rndm.

CLAIM 1 : For any $z \in [1, p-1]$, $z^{p-1} = 1 \pmod p$

Proof:

Take the set $S_1 = \{1, 2, \ldots, p-1\}$

Also, consider the set $S_2 = \{ \underset{(\text{mod } p)}{z}, \underset{(\text{mod } p)}{2z}, \ldots, \underset{(\text{mod } p)}{(p-1)z} \}$

$\subseteq [1, p-1]$

$\boxed{\text{Subclaim: } S_1 = S_2}$

Proof:

If $S_1 \neq S_2 \Rightarrow$ Two elements of S_2 are same

$\exists\, i \geq j$ such that $iz \,(\text{mod } p) = jz \,(\text{mod } p)$

$$\Rightarrow \frac{iz - jz}{p} = \text{integer} \Rightarrow \frac{(i-j)\,z}{p} = \text{integer} \Rightarrow i = j$$

$$(1)(2)(3)\ldots(p-1) = z^{p-1}(1)(2)\ldots(p-1) \quad (\text{mod } p)$$

$$\frac{\left(z^{p-1}-1\right)(1)(2)(3)\ldots(p-1)}{p} = \text{integer}$$

$$\Rightarrow z^{p-1} = 1 \;(\text{mod } p).$$

CLAIM 2 : $\boxed{F : z \to zr \pmod p}$ is invertible, and also 1-1.

Proof : $F(z) = \underline{z}\, r \pmod p$

Product by r^{p-2},

$$r^{p-2} \underbrace{F(z)}_{y} = z\, r^{p-1} \pmod p = \underline{z} \pmod p$$

Inverse Map is

$$F^{-1} : y \to (r^{p-2} y) \bmod p$$

CLAIM 3 : If $r \in [1, p-1]$ was random $\Rightarrow$ For a given z, $F(z)$ is any random value in $[1, p-1]$.

Proof :

Fix a z,

$$\text{Prob}\left(F(z) = i \right) = \text{Prob}\left((z r) \bmod p = i \right)$$

$$= \text{Prob}\left(r = \underline{z^{p-2}(i)} \bmod p \right)$$

$$= \frac{1}{p-1}$$

⊛ Universal Hashing:

Given: Universe $U = [1, M]$

Set $S = \{s_1, s_2 \dots s_n\} \subseteq [1, M]$ of size n.

Aim: Find a data-structure for S to answer search queries:

"Does $z \in S$?" where $1 \le z \le M$

Typically, $n <<< M$

Eg. $n = 10^3$ $M = 10^8$

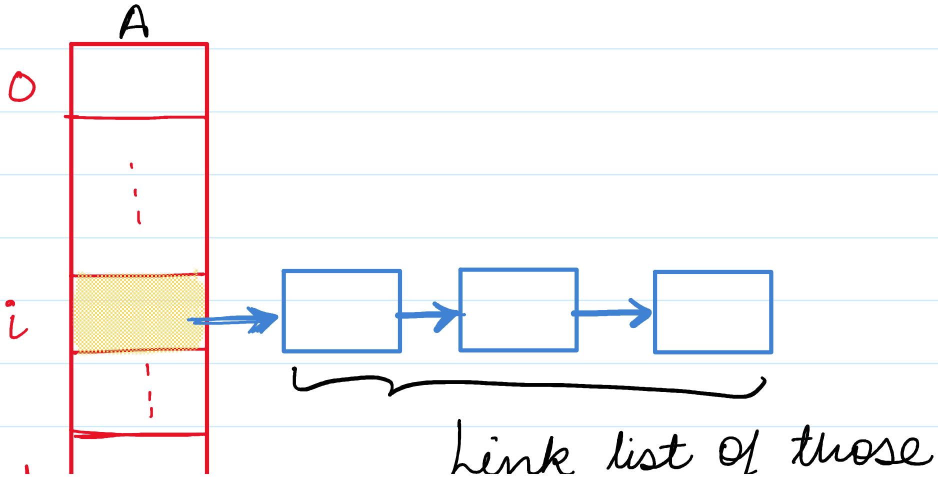
$\boxed{\text{Assumption} \quad \text{Word size} = O(\log M)}$

Some Solutions:

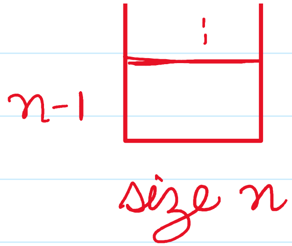
	Search time	Space
Array	$O(1)$	$O(M)$
Link-list	$O(n)$	$O(n)$
AVL trees	$O(\log n)$	$O(n)$

AIM → Hashing $O(1)$ $O(n)$

- Hashing involves a function $H : [1, M] \rightarrow [0, n-1]$
- Hash-Table
  - Array A of size n
  - A[i] stores



└ A[i] stores link-list

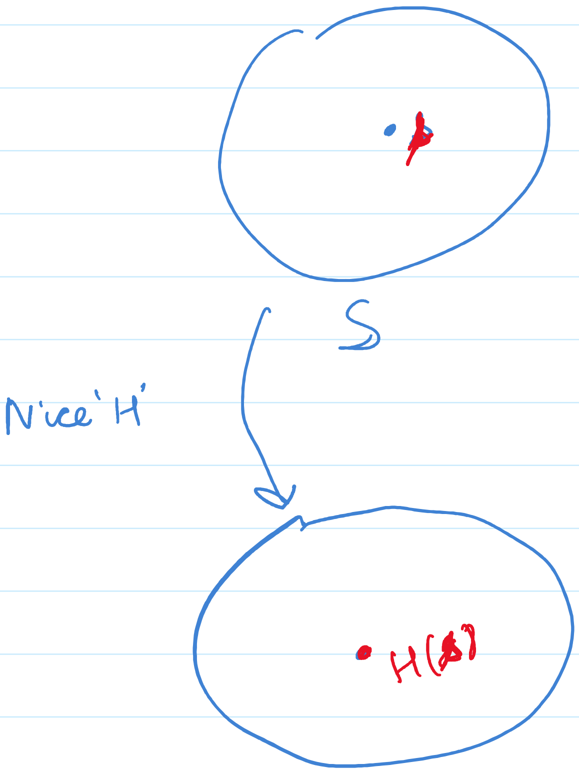


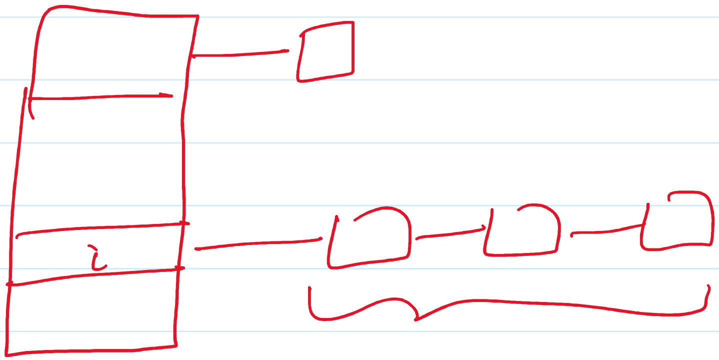
$\underbrace{\qquad\qquad}$ Link list of those elements $s \in S$ for which $H(s) = i$

Search-Algo(z)

1. Compute $i = H(z)$
2. Go to link-list at location i, and scan it.
3. If ($z \in$ Link-List-i): Return "FOUND".
   Else
   Return "Not-Found."

Total Time = $\underbrace{\text{Time to compute } H(z)}_{\text{Ideally should be } O(1)} + \underbrace{\max_{0 \le i \le n-1} \text{Size}(\text{Link-list-}i)}_{\text{Ideally should be } O(1)}$





All elements whose are mapped to i by H.

$\boxed{\text{CLAIM 1}}$: $H_1(z) = z \pmod{n}$ it will good iff S was random.

$\boxed{\text{CLAIM 2}}$ $H_0(z) = zr \pmod{p}$

$p \sim M$

$S \xrightarrow{H_0}$ make S look like random $\xrightarrow{\bmod n}$ Expected size of link list is $O(1)$.