Hashing

- Keys come from some large universe.
- \( S \subset U \) we actually care about. \(|S| = N \), \( N \ll |U| \).

- A function \( h : U \rightarrow \{0, 1, 2, \ldots, M-1\} \) where the codomain is the hash table. Generally, \( M = O(N) \).

- A collision happens when \( h(x) = h(y) \) when \( x \neq y \). We will handle collisions with a linked list

- Table
  
  \[
  \begin{array}{c|c|c}
  0 & 1 & 2 \\
  1 & & \\
  2 & x & y \\
  M-1 & & \\
  \end{array}
  \]

  Collision keys are chained together.
  \( h(x) = h(y) = h(z) = 2 \).

- Find, insertion, deletion properties are easy.
- \( R(x) \) is easy to implement. \( R(x) = B \times x \mod M \) where \( B \) is a multiplier.

- Hashing is good if the keys of \( S \) are uniformly distributed.

- How to counter this? Good hash functions should be random.
Universal Hashing

Def: A hashing function is universal, if for all \( x \neq y \) in \( U \), we have

\[
\Pr( h(x) = h(y) ) \leq \frac{1}{M}
\]
Constructing a universal hash family

Matrix method \( h: U \to \{0, 1, \ldots, M-1\} \)
- keys are \( u \)-bits long
- \( M = 2^b \); index is \( b \)-bits long.

Consider \( h \) to be a random \( b \times u \) 0/1 matrix \( h(x) = hx \) where we do addition mod 2.

\[
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 \\
1 & 1 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4
\end{bmatrix}
= 
\begin{bmatrix}
h(x)
\end{bmatrix}
\]

Claim: For \( x \neq y \), \( \Pr_h(h(x) = h(y)) = \frac{1}{M} = \frac{1}{2^b} \).

Proof: Multiply \( \Rightarrow \) add all columns whose corresponding entries in \( x \) is 1.

Consider an arbitrary pair \( y \) keys \( x \neq y \), \( x \neq y \).
Suppose \( x_i = 0 \) \& \( y_i = 1 \). Now consider all hashing matrix whose \( i \)-th column is 0. Over the remaining choices of \( i \)-th column, \( h(x) \) is the same. There are \( 2^b \) different ways to set \( i \)-th \( i \)-th column. Each such setting gives a different values \( y \neq h(y) \). Therefore, there are \( \frac{1}{2^b} \) chance of getting \( h(x) = h(y) \).
Creating a hash table for $S$.

1. Generate a random $b \times u$ matrix $h$.

2. For any key $x = \langle x_0, x_1, x_2, \ldots, x_{u-1} \rangle$, compute $h(x) \mod b$. Let $\langle z_0, z_1, z_2, \ldots, z_{b-1} \rangle$ be the index.

3. Start a in location Table $[z_0^2 + z_1^2 + \cdots + z_{b-1}^2]$.

# of expected collision: $\frac{N}{2^b} = \frac{N}{M}$.

If \( M = O(N) \), # of expected collisions is constant.

Do Insert, Find, Delete operations are $O(1)$. 