Overview:

Today: Hashing
1. Dictionary Data Structures
2. Hash Families
3. k-wise independence

1 Dictionary Data Structures

Definition
Suppose that the universe of all words is U and the set of all valid words is W, s.t.

\[ |U| = N, \text{ N is very large} \]
\[ W \subseteq U, |W| = n, \text{ n is much smaller} \]

Additional data is associated to each element of W.

1.1 Lookup

Given \( x \in U \), we would like to find the data associated with \( x \).
We follow the two steps below to implement this functionality:
1. Determine if \( x \in W \)
2. If \( x \in W \), then look up associated data
Space complexity \( \approx O(n) \)
Time complexity \( \approx O(1) \) (ideally)

1.2 Two Dictionary Structures

Static Dictionary Structure: W is given and fixed, does not support insert or delete.
1. Build a dictionary
2. Only do lookup operations
Dynamic Dictionary Structure:
1. Build a dictionary
2. Lookup
3. Insert
4. Delete
The last two operations add or remove elements from W correspondingly.

1.3 Lookup with Static Dictionary

The solution to implement look up operation for a static dictionary is relatively easy. Simply sort W when building the dictionary. Look up corresponding data by binary search.
Time complexity: $O(\log n)$

1.4 Lookup with Dynamic Dictionary

How to solve the same problem with dynamic dictionary?
Idea: Indexing into an array
Suppose we have an array of length $m$. Each entry stores an element of W. Given an $x \in U$. Where do we store and find the data? We need to use a hash function $h: U \to [m]$. Compute $h(x)$ and store word $x$ in location $h(x)$

What do we need from $h$?
- Easy to compute
  Problem: Collisions
  What happens if $h(x) = h(y)$ for some $x \neq y, x, y \in W$?
- Few collisions on $W$
  How do we deal with collisions?
  Note if $m < N = |U|$, by pigeonhole theorem, there will be elements of $U$ that collide. Hence deterministic $h$ does not work.
  **Pigeonhole Theorem**
  Pigeonhole Theorem states that if $n$ items are put into $m$ containers, with $n > m$, then at least one container must contain more than one item.
  Fix: Random $h$
  Let $h$ be a uniform random function $h: U \to [m]$

$$E_h[\text{Number of } x, y \in W : h(x) = h(y), x \neq y] = \sum_{x \neq y \in W} I_{h(x) = h(y)} = \frac{n \choose 2}{m} \approx \frac{n^2}{m}$$

**Birthday Paradox**
Birthday paradox concerns the probability that, in a set of $n$ randomly chosen people, some pair of them
will have the same birthday. When \( n \approx \sqrt{m} \), there is about 50\% chance of matching.

If \( n \approx m \), then there will be \( O(n) \) collisions \( \Rightarrow \) too much to ask for no collisions

Store uniform random function. - Can record which \( h \) used with few bits.

## 2 Hash Family

**Definition**
Hash family is a family of (probability distribution over) function \( h : U \rightarrow [m] \)

**Requirements**
We would like the hash family to meet the following requirements:
1. easy to store an element of the family
2. evaluate function given identifier and input quickly
3. some sort of guarantee

**Ex: guarantee**
For every \( x, y \in U \) s.t. \( x \neq y \),

\[
Pr_h(h(x) = h(y)) = \frac{1}{m}
\]

## 3 k-wise Independence

Stronger Guarantee: k-wise independence

**Definition**
For distinct \( x_1, x_2, ..., x_k \in U, x_1, ..., x_k \in [m] \)

\[
Pr_h(h(x_1) = y_1 \, \&\, h(x_2) = y_2 \, \&\, ..., \, h(x_k) = y_k) = \frac{1}{m^k}
\]

**Standard k-wise independence family**
If \( U = F_q \, \&\, [m] = F_q \), \( h(x) = \) random degree \( k - 1 \) polynomial \( (x) \)

**Standard poly interpolation**
Given \( x_1, x_2, ..., x_k, y_1, ..., y_k, \exists! \) degree \( k - 1 \) polynomial s.t. \( p(x_1) = y_1, ..., p(x_k) = y_k \)

\( U \subset F_{2^t}, h : U \rightarrow F_{2^t} \rightarrow [2^s] \) where \( s \leq t \)

Number of bits to specify \( h \): \( klg(N \, \ast \, m) \)

Submitted by Yang, Zhiyue on October 20, 2017.