1 Randomization – Does It Apply to Hashing?

- Randomization turned out to be a good way to ensure almost-certainly balanced binary trees.
- Now, we’ll look at another kind of dictionary structure – a hash table.
- If we can find a hash function that distributes inputs to hash slots uniformly, then we know that insertion, search, and deletion take only $O(1)$ time.
- If we assume that our inputs are integers drawn uniformly at random from some space $[0, N)$, then even a very basic hash function $h(x) = x \mod m$ will map each input to every slot with equal probability, resulting in no collisions with high probability for $O(\sqrt{m})$ elements.
- But of course, a nasty person who knows our hash function might try to break the table.
- In particular, there are definitely (by PHP) $N/m$ input values that all hash to the same slot, and inserting/searching for these values results in running times linear in the number of elements inserted.

Can randomization save us?

- Suppose we randomize the choice of our hash function $h$.
- $h$ needs to be chosen at the time we create the table, because it must remain fixed thereafter if we want to be able to find stuff that we inserted previously!
- We want to guarantee that, no matter what (small) subset of possible input values the table receives, it is very unlikely (over the choice of $h$) that many of these values will hash to the same slot.
- More specifically, suppose we receive two arbitrary inputs $x \neq y$. We want to ensure that we don’t always pick an $h$ that causes $x$ and $y$ to hash together.
- For a table of size $m$, the best we can hope for is that the chance that $x$ and $y$ hash together is $1/m$ – the same as if we assigned slots to $x$ and $y$ at random.
- We’re going to verify that this criterion guarantees good hashing behavior with high probability, then show how to choose hash functions to make the criterion true.
2 Universal Hash Functions

• Let $H$ be a family of functions, each mapping from $[0, N)$ to $[0, m)$.

• Any function $h \in H$ could be used as a hash function from integers $< N$ to slots in a table of size $m$.

• Defn: the family $H$ is universal if, for every pair $x \neq y$ in $[0, N)$,
  \[ \Pr_{h \in H}(h(x) = h(y)) = \frac{1}{m}. \]
  
• Hence, a randomly chosen $h$ from a universal family $H$ causes any two distinct keys to collide with probability $1/m$.

We now show that universal hash functions are grrrrrrreat!

• Thm: Let $H$ be a universal family of hash functions.

  • For any $0 < \delta < 1$, let $S \subseteq [0, N)$ be a set of at most $\sqrt{2m\delta}$ keys.
  
  • Then for a random $h \in H$, every pair $x, y \in S$ has $h(x) \neq h(y)$ with probability at least $1 - \delta$.

  • Pf: First, let’s compute the expected number of collisions in $S$ for a random $h$.

• Let $c_{xy}$ be a 0-1 indicator variable defined by
  \[ c_{xy} = \begin{cases} 
  1 & \text{if } h(x) = h(y) \\
  0 & \text{otherwise.} 
\end{cases} \]

• Taking the expectation w/r to the choice of $h$,
  \[ E[c_{xy}] = 1 \cdot \Pr(h(x) = h(y)) + 0 \cdot \Pr(h(x) \neq h(y)) \]
  \[ = \Pr(h(x) = h(y)) \]
  \[ = \frac{1}{m}. \]

• Let $C_S$ be the number of collisions in set $S$ under $h$. We have that
  \[ C_S = \sum_{x \neq y \in S} c_{xy}, \]
  and so
  \[ E[C_S] = E \left[ \sum_{x \neq y \in S} c_{xy} \right] \]
  \[ = \sum_{x \neq y \in S} E[c_{xy}] \]
  \[ = \frac{|S|(|S| - 1)}{2} \cdot \frac{1}{m} \]
  \[ < \frac{|S|^2}{2m} \]
  \[ = \frac{2m\delta}{2m} \]
  \[ = \delta. \]
where the second step follows by *linearity of expectation*—we can always distribute the expectation “operator” over a sum.

- OK, so the *expected* number of collisions in \( S \) is at most \( \delta \), but we want to show that the probability of even one collision occurring is at most \( \delta \).

- We can relate a distribution’s mean to the size of a random value drawn from it using *Markov’s inequality*.

- **Thm** (Markov): for any non-negative random variable \( x \),

\[
\Pr(x > kE[x]) < \frac{1}{k}
\]

for all \( k > 0 \).

- (Note that this inequality is mainly useful when \( k \) is big! It’s a very weak bound—much weaker than Chernoff bounds, which are specific to distributions arising from Bernoulli random processes.)

- We want to know the chance that \( C_S \), the number of collisions in \( S \), is \( > 0 \).

- Because \( \delta < 1 \),

\[
\Pr(C_S > 0) = \Pr(C_S > \frac{1}{\delta}E[C_S])
\]

since \( C_S \) is integer-valued, and we showed above that \( E[C_S] < \delta \).

- But by Markov’s inequality, the RHS above is \(< \delta \), as claimed. QED

We can further generalize this result to show that, even when we insert more than \( \sqrt{2m\delta} \) elements, the distribution of elements in the table remains close to uniform with high probability.

- **Corr**: Let \( H \) be a universal family of hash functions.

- For any \( 0 < \delta < 1 \), let \( S \subseteq [0, N] \) be a set of at most \( \sqrt{Bm\delta} \) keys.

- Then for a random \( h \in H \), \( h \) does not hash more than \( B + 1 \) values to any one slot with probability at least \( 1 - \delta \).

- **Pf**: fix any \( x \in S \).

- Let \( C_{x,S} \) be the number of *other* elements \( y \in S \) for which \( h(y) = h(x) \).

- We have that

\[
E[C_{x,S}] = \sum_{y \in S \setminus \{x\}} E[c_{xy}]
\]

\[
= (|S| - 1) \frac{1}{m}
\]

\[
< \frac{|S|}{m}
\]

\[
\leq \sqrt{\frac{B\delta}{m}}.
\]
Applying Markov’s inequality, we obtain

\[
\Pr(C_x, S > B) = \Pr\left( C_x, S > \sqrt{\frac{Bm}{\delta} \cdot \sqrt{\frac{B\delta}{m}}} \right) 
\leq \Pr\left( C_x, S > \sqrt{\frac{Bm}{\delta} \cdot E[C_x, S]} \right) 
< \sqrt{\frac{\delta}{Bm}}.
\]

Finally, applying the union bound over all elements \( x \in S \), the chance that some element hashes to the same slot as more than \( B \) other elements of \( S \) is less than

\[
|S| \cdot \sqrt{\frac{\delta}{Bm}} = \sqrt{Bm}\delta \cdot \sqrt{\frac{\delta}{Bm}} = \delta.
\]

QED

Conclude that, as long as we only insert \( O\left(\sqrt{m}\right) \) elements into our table, choosing a random hash function almost certainly gives us chains of of size \( O(1) \) and hence \( O(1) \) operation times.

### 3 An Aside: Proofs for the Skeptical

- **Thm** (Linearity of Expectation): let \( x \) and \( y \) be numerical random variables, not necessarily independent.

  Then \( E[x + y] = E[x] + E[y] \).

- **Pf:**

  \[
  E[x + y] = \sum_a \sum_b (a + b) \Pr(x = a \land y = b) 
  = \sum_a \sum_b a \Pr(x = a \land y = b) + \sum_a \sum_b b \Pr(x = a \land y = b) 
  = \sum_a a \sum_b \Pr(x = a \land y = b) + \sum_b \sum_a b \Pr(x = a \land y = b) 
  = \sum_a a \Pr(x = a) + \sum_b b \Pr(y = b) 
  = E[x] + E[y].
  \]

  QED

- **Thm** (Markov’s Inequality): for any non-negative random variable \( x \),

  \[
  \Pr(x > kE[x]) < \frac{1}{k}
  \]

  for all \( k > 0 \).
• Pf:
\[
E[x] = E[x \mid x > kE[x]] \cdot \Pr(x > kE[x]) + E[x \mid x \leq kE[x]] \cdot \Pr(x \leq kE[x])
\]
\[
> kE[x] \cdot \Pr(x > kE[x]) + 0.
\]
• Dividing both sides of the last inequality by \( kE[x] \) gives us the result. QED

4 Where Do Universal Hash Function Families Come From?

• There are a number of different constructions for universal hash families.
• Today, we’ll see a construction based on modular arithmetic.
• Fix the table size \( m \) to be a prime number.
• Suppose that \( N \) is at most \( m^\ell \) for some \( \ell > 0 \).
• For any \( x \in [0, N) \), we can express \( x \) “in base \( m \)” as an ordered list \( \vec{x} \) of \( \ell \) values, each in the range \([0, m - 1)\).
• Given any vector \( \vec{a} \in \{0, 1, \ldots, m - 1\}^\ell \), define
\[
h_{\vec{a}}(x) = \vec{a} \cdot \vec{x} \mod m
\]
\[
= \sum_{i=1}^{\ell} a_i x_i \mod m.
\]
• Our universal family \( H \) will be the family of functions \( h_{\vec{a}} \) for all \( m^\ell \) possible \( \vec{a} \).

We need to prove that this family \( H \) is actually universal.

• Our proof will rely on the uniqueness of inverses modulo a prime number.
• Given a prime \( m \) and \( 0 < a < m \), there exists a unique \( x < m \) s.t. \( ax = 1 \) (mod \( m \)).
• (We can find this inverse by the extended Euclid algorithm!)
• It follows that for prime \( m \) and any \( a, b < m \), there exists a unique \( x < m \) s.t. \( ax = b \) (mod \( m \)).
• Thm: for any \( x \neq y < N \) and a randomly chosen \( h_{\vec{a}} \in H \),
\[
\Pr(h_{\vec{a}}(x) = h_{\vec{a}}(y)) = 1/m.
\]
• Pf: let \( \vec{x} = [x_1, \ldots, x_\ell] \) and \( \vec{y} = [y_1, \ldots, y_\ell] \).
• Because \( x \neq y \), there exists some \( i \) s.t. \( x_i \neq y_i \).
• Now \( h_{\vec{a}}(x) = (a_i x_i + r) \mod m \) and \( h_{\vec{a}}(y) = (a_i y_i + s) \mod m \), for some \( r \) and \( s \) that depend on all the other coefficients of \( \vec{a} \) besides \( a_i \).
Clearly,
\[ h_{\vec{a}}(x) = h_{\vec{a}}(y) \] iff \[ a_i x_i + r = a_i y_i + s \pmod{m} \]
iff \[ a_i (x_i - y_i) = s - r \pmod{m} \].

We know, by uniqueness of inverses modulo a prime, that there is a unique value (modulo \( m \)) of the difference \( s - r \) for which the last equality holds.

Conclude that
\[
\Pr(h_{\vec{a}}(x) = h_{\vec{a}}(y)) = \sum_{r,s} \Pr(a_i (x_i - y_i) = s - r \pmod{m} | r, s) \Pr \left( \sum_{j \neq i} a_j x_j = r \land \sum_{j \neq i} a_j y_j = s \right)
\]
\[
= \frac{1}{m} \sum_{r,s} \Pr \left( \sum_{j \neq i} a_j x_j = r \land \sum_{j \neq i} a_j y_j = s \right)
\]
\[
= \frac{1}{m}.
\]

QED

We can tweak this family of hash functions to give an even stronger guarantee than universality.

Let \( b \) be a scalar value from \([0, m)\).

Define the family of hash functions
\[ h_{\vec{a},b} = \sum_{i=1}^{\ell} a_i x_i + b \pmod{m}. \]

for all possible choices of \( \vec{a} \) and \( b \).

The above proof of universality still holds for this family.

Let’s also consider this question: for fixed \( x < N \) and \( z < m \), what is \( \Pr(h(x) = z) \)? That is, does the family of hash functions \( h \) have a bias for which slot \( x \) winds up in?

For a randomly chosen \( h_{\vec{a}} \), \( \Pr(\sum_i a_i x_i = z \pmod{m}) \) depends strongly on \( x \!\!).

For example, if we consider \( x = 0 \), every hash function \( h_{\vec{a}} \) hashes \( x \) to 0.

But for a randomly chosen function \( h_{\vec{a},b} \),
\[
\Pr \left( \sum_i a_i x_i + b = z \pmod{m} \right) = \Pr \left( b = z - \sum_{i} a_i x_i \pmod{m} \right) = 1/m.
\]

because \( b \) is chosen uniformly at random.

Hence, the family of functions \( h_{\vec{a},b} \) is not only universal but unbiased.
Moreover, we can show that for a randomly chosen $h_{\vec{a},b}$ and $x \neq y$,

$$\Pr(h_{\vec{a},b}(x) = z \land h_{\vec{a},b}(y) = w) = \frac{1}{m^2},$$

that is, the marginal distribution on slots for pairs of elements is identical to that obtained by picking the slot for each element uniformly at random.

Indeed, by uniqueness of multiplicative inverses modulo a prime, we can show that the equations

$$a_i x_i + r + b = w \pmod{m}$$
$$a_i y_i + s + b = z \pmod{m}$$

are satisfied by unique values $a_i$ and $b$ for given $x_i, y_i, r, s, w,$ and $z$.

Hence, if we fix one $i$ such that $x_i \neq y_i$, we can argue by a similar proof to the previous one that, ignoring all the other choices of $a_j$, the chance that we pick the right $a_i$ and the right $b$ to make both equations true is $1/m \cdot 1/m$. 