A weakness of hashing

**Problem:** For any hash function $h$, a set of keys exists that can cause the average access time of a hash table to skyrocket.

- An adversary can pick all keys from $\{k \in U : h(k) = i\}$ for some slot $i$.

**Idea:** Choose the hash function at random, independently of the keys.

- Even if an adversary can see your code, he or she cannot find a bad set of keys, since he or she doesn’t know exactly which hash function will be chosen.
Universal hashing

**Definition.** Let $U$ be a universe of keys, and let $\mathcal{H}$ be a finite collection of hash functions, each mapping $U$ to \{0, 1, ..., $m-1$\}. We say $\mathcal{H}$ is *universal* if for all $x, y \in U$, where $x \neq y$, we have $|\{h \in \mathcal{H} : h(x) = h(y)\}| = |\mathcal{H}|/m$.

That is, the chance of a collision between $x$ and $y$ is $1/m$ if we choose $h$ randomly from $\mathcal{H}$.
Universality is good

**Theorem.** Let $h$ be a hash function chosen (uniformly) at random from a universal set $\mathcal{H}$ of hash functions. Suppose $h$ is used to hash $n$ arbitrary keys into the $m$ slots of a table $T$. Then, for a given key $x$, we have

$$E[\text{#collisions with } x] < \frac{n}{m}.$$
Proof of theorem

Proof. Let $C_x$ be the random variable denoting the total number of collisions of keys in $T$ with $x$, and let

$$c_{xy} = \begin{cases} 1 & \text{if } h(x) = h(y), \\ 0 & \text{otherwise}. \end{cases}$$

Note: $E[c_{xy}] = 1/m$ and $C_x = \sum_{y \in T - \{x\}} c_{xy}$. 
Proof (continued)

\[ E[C_x] = E \left[ \sum_{y \in T - \{x\}} c_{xy} \right] \]

• Take expectation of both sides.
Proof (continued)

\[
E[C_x] = E \left[ \sum_{y \in T - \{x\}} c_{xy} \right] = \sum_{y \in T - \{x\}} E[c_{xy}]
\]

- Take expectation of both sides.
- Linearity of expectation.
Proof (continued)

\[
E[C_x] = E\left[ \sum_{y \in T - \{x\}} c_{xy} \right]
\]

\[
= \sum_{y \in T - \{x\}} E[c_{xy}]
\]

\[
= \sum_{y \in T - \{x\}} 1 / m
\]

- Take expectation of both sides.
- Linearity of expectation.
- \( E[c_{xy}] = 1 / m \).
Proof (continued)

\[ E[C_x] = E \left[ \sum_{y \in T - \{x\}} c_{xy} \right] \]

\[ = \sum_{y \in T - \{x\}} E[c_{xy}] \]

\[ = \sum_{y \in T - \{x\}} 1/m \]

\[ = \frac{n - 1}{m}. \]

- Take expectation of both sides.
- Linearity of expectation.
- \( E[c_{xy}] = 1/m \).
- Algebra.
Constructing a set of universal hash functions

Let \( m \) be prime. Decompose key \( k \) into \( r + 1 \) digits, each with value in the set \( \{0, 1, \ldots, m-1\} \). That is, let \( k = \langle k_0, k_1, \ldots, k_r \rangle \), where \( 0 \leq k_i < m \).

**Randomized strategy:**

Pick \( a = \langle a_0, a_1, \ldots, a_r \rangle \) where each \( a_i \) is chosen randomly from \( \{0, 1, \ldots, m-1\} \).

Define \( h_a(k) = \sum_{i=0}^{r} a_i k_i \mod m \).

How big is \( \mathcal{H} = \{h_a\} \)? \( |\mathcal{H}| = m^{r+1} \).
Universality of dot-product hash functions

Theorem. The set $\mathcal{H} = \{h_a\}$ is universal.

Proof. Suppose that $x = \langle x_0, x_1, \ldots, x_r \rangle$ and $y = \langle y_0, y_1, \ldots, y_r \rangle$ be distinct keys. Thus, they differ in at least one digit position, wlog position 0. For how many $h_a \in \mathcal{H}$ do $x$ and $y$ collide?

We must have $h_a(x) = h_a(y)$, which implies that

$$\sum_{i=0}^{r} a_i x_i \equiv \sum_{i=0}^{r} a_i y_i \pmod{m}.$$
Proof (continued)

Equivalently, we have

$$\sum_{i=0}^{r} a_i (x_i - y_i) \equiv 0 \pmod{m}$$

or

$$a_0 (x_0 - y_0) + \sum_{i=1}^{r} a_i (x_i - y_i) \equiv 0 \pmod{m},$$

which implies that

$$a_0 (x_0 - y_0) \equiv -\sum_{i=1}^{r} a_i (x_i - y_i) \pmod{m}.$$
Fact from number theory

**Theorem.** Let $m$ be prime. For any $z \in \mathbb{Z}_m$ such that $z \neq 0$, there exists a unique $z^{-1} \in \mathbb{Z}_m$ such that

$$z \cdot z^{-1} \equiv 1 \pmod{m}.$$

**Example:** $m = 7$.

<table>
<thead>
<tr>
<th>$z$</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$z^{-1}$</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>6</td>
</tr>
</tbody>
</table>
Back to the proof

We have

\[ a_0(x_0 - y_0) \equiv -\sum_{i=1}^{r} a_i(x_i - y_i) \pmod{m}, \]

and since \( x_0 \neq y_0 \), an inverse \( (x_0 - y_0)^{-1} \) must exist, which implies that

\[ a_0 \equiv \left( -\sum_{i=1}^{r} a_i(x_i - y_i) \right) \cdot (x_0 - y_0)^{-1} \pmod{m}. \]

Thus, for any choices of \( a_1, a_2, \ldots, a_r \), exactly one choice of \( a_0 \) causes \( x \) and \( y \) to collide.
Proof (completed)

**Q.** How many $h_a$’s cause $x$ and $y$ to collide?

**A.** There are $m$ choices for each of $a_1, a_2, \ldots, a_r$, but once these are chosen, exactly one choice for $a_0$ causes $x$ and $y$ to collide, namely

$$a_0 = \left( \left( - \sum_{i=1}^{r} a_i(x_i - y_i) \right) \cdot (x_0 - y_0)^{-1} \right) \mod m.$$ 

Thus, the number of $h_a$’s that cause $x$ and $y$ to collide is $m^r \cdot 1 = m^a = |\mathcal{H}|/m$. 

© 2001 by Charles E. Leiserson
Perfect hashing

Given a set of $n$ keys, construct a static hash table of size $m = O(n)$ such that \texttt{SEARCH} takes $\Theta(1)$ time in the \textit{worst case}.

\textbf{IDEA:} Two-level scheme with universal hashing at both levels.

\textit{No collisions at level 2!}
Collisions at level 2

**Theorem.** Let \( \mathcal{H} \) be a class of universal hash functions for a table of size \( m = n^2 \). Then, if we use a random \( h \in \mathcal{H} \) to hash \( n \) keys into the table, the expected number of collisions is at most \( 1/2 \).

**Proof.** By the definition of universality, the probability that 2 given keys in the table collide under \( h \) is \( 1/m = 1/n^2 \). Since there are \( \binom{n}{2} \) pairs of keys that can possibly collide, the expected number of collisions is

\[
\binom{n}{2} \cdot \frac{1}{n^2} = \frac{n(n-1)}{2} \cdot \frac{1}{n^2} < \frac{1}{2}.
\]
No collisions at level 2

Corollary. The probability of no collisions is at least 1/2.

Proof. Markov’s inequality says that for any nonnegative random variable \( X \), we have

\[
\Pr\{X \geq t\} \leq \frac{E[X]}{t}.
\]

Applying this inequality with \( t = 1 \), we find that the probability of 1 or more collisions is at most 1/2.

Thus, just by testing random hash functions in \( \mathcal{H} \), we’ll quickly find one that works.
Analysis of storage

For the level-1 hash table $T$, choose $m = n$, and let $n_i$ be random variable for the number of keys that hash to slot $i$ in $T$. By using $n_i^2$ slots for the level-2 hash table $S_i$, the expected total storage required for the two-level scheme is therefore

$$E\left[\sum_{i=0}^{m-1} \Theta(n_i^2)\right] = \Theta(n),$$

since the analysis is identical to the analysis from recitation of the expected running time of bucket sort. (For a probability bound, apply Markov.)