Hashing (2)
Recall: Universal Hashing

Let $\mathcal{H} = \{h_\lambda\}_{\lambda \in \Lambda}$ be a family of hash functions from $\mathcal{U}$ to $\mathcal{V}$.

$m := |\mathcal{V}|$

**Def’n:** $\mathcal{H}$ is called **universal** if for all $a, b \in \mathcal{U}$ with $a \neq b$,

$$|\{\lambda \in \Lambda : h_\lambda(a) = h_\lambda(b)\}| \leq \frac{|\Lambda|}{m}.$$

**Probabilistic interpretation:** if $R$ is a random variable, uniformly distributed over $\Lambda$, then

$$\Pr[h_R(a) = h_R(b)] \leq \frac{1}{m}.$$
A universal family

Let $m$ be a prime, and $t$ a positive integer

Define $\mathcal{U} := \mathbb{Z}_m^t$, $\Lambda := \mathbb{Z}_m^t$, $\nu := \mathbb{Z}_m$

For $\lambda = (\lambda_1, \ldots, \lambda_t) \in \Lambda$, $a = (a_1, \ldots, a_t) \in \mathcal{U}$, define

$$h_\lambda(a) := \sum_{i=1}^t a_i \lambda_i$$

Define

$$\mathcal{H} := \{ h_\lambda \}_{\lambda \in \Lambda}$$

**Theorem:** $\mathcal{H}$ is universal
Proof of theorem.

Suppose \((a_1, \ldots, a_t) \neq (b_1, \ldots, b_t)\)

We want to count the number \(N\) of solutions \((\lambda_1, \ldots, \lambda_t)\)

to the equation

\[
\sum_i a_i \lambda_i = \sum_i b_i \lambda_i.
\]

Re-write this as

\[
\sum_i c_i \lambda_i = 0
\]

where \(c_i := a_i - b_i\)

By assumption, not all \(c_i\)'s are zero

Want to show: \(N \leq |\Lambda|/m = m^{t-1}\)
Proof (cont’d).

Let \( N \) be the number of solutions \((\lambda_1, \ldots, \lambda_t)\) to \(\sum_i c_i \lambda_i = 0\), where some \(c_j \neq 0\)

Want to show: \( N \leq |\Lambda|/m = m^{t-1} \)

Without loss of generality, assume \(c_1 \neq 0\)

For every choice of \(\lambda_2, \ldots, \lambda_t\), there is a unique \(\lambda_1\) such that \(\sum_i c_i \lambda_i = 0\), namely, \(\lambda_1 = -c_1^{-1} \sum_{i=2}^t c_i \lambda_i\)

There are \(m^{t-1}\) ways of choosing \(\lambda_2, \ldots, \lambda_t\), and each yields one solution

So \( N = m^{t-1} \)

QED
Practical considerations

Key space:

- View keys as bit strings of some fixed length $l$
- Break up into “chunks” of length $w$, where $2^w \leq m < 2^{w+1}$:
  
  \[
  \begin{array}{cccc}
  a_1 & a_2 & \cdots & a_t \\
  \end{array}
  \]

- View each $a_i$ as a number between 0 and $2^w - 1$, and each such number as an element of $\mathbb{Z}_m$
- This map from $\{0, 1\}^l$ to $\mathbb{Z}_m^t$ is injective
- Variable length keys (padding)
Practical considerations (cont’d)

Slot space: choice of prime $m$

Bertrand’s Postulate: There is always a prime between $x$ and $2x$ for all integers $x \geq 1$

Chebyshev said it
So I’ll say it again
There’s always a prime between $N$ and $2N$

Dynamically growing the table: when $n/m$ gets too large, choose a new $m' \geq 2m$, and rehash everything
Another universal family

Let $p$ be a prime, and $m$ a positive integer

Define $\mathcal{U} := \{0, \ldots, p-1\}$,
$\Lambda := \{1, \ldots, p-1\} \times \{0, \ldots, p-1\}$,
$\mathcal{V} := \{0, \ldots, m-1\}$

For $\lambda = (\lambda_1, \lambda_2) \in \Lambda$, $a \in \mathcal{U}$, define

$$h_\lambda(a) := (\lambda_1 a + \lambda_2) \mod p \mod m$$

**Theorem:** $\mathcal{H} := \{h_\lambda\}_{\lambda \in \Lambda}$ is universal (see text)

Pros: free choice of $m$

Cons: multiplication of large numbers
General problem: large hash function index space — almost as large as the key space

Solution: weaker (but still useful) hashing requirements
\( \varepsilon \)-universal Hashing

Let \( \mathcal{H} = \{ h_\lambda \}_{\lambda \in \Lambda} \) be a family of hash functions from \( \mathcal{U} \) to \( \mathcal{V} \)

**Def’n:** Let \( 0 \leq \varepsilon \leq 1 \). \( \mathcal{H} \) is called \( \varepsilon \)-universal if for all \( a, b \in \mathcal{U} \) with \( a \neq b \),

\[
\left| \{ \lambda \in \Lambda : h_\lambda(a) = h_\lambda(b) \} \right| \leq \varepsilon \cdot |\Lambda|.
\]

**Probabilistic interpretation:** if \( R \) is a random variable, uniformly distributed over \( \Lambda \), then

\[
\Pr[h_R(a) = h_R(b)] \leq \varepsilon
\]

universal = \( (1/m) \)-universal, where \( m := |\mathcal{V}| \)
Using $\varepsilon$-universal hash families

As long as $\varepsilon$ is not too big, many of the results we proved have useful analogs.

E.g., in a table with at most $n$ keys

- expected cost of each dictionary operation in a table containing $n$ keys is $\leq 1 + \varepsilon n$.
- expected value of maximum load is $\leq \sqrt{\varepsilon n^2 + n}$
**An $\varepsilon$-universal family**

Let $m$ be a prime, and $t$ a positive integer

Define $\mathcal{U} := \mathbb{Z}_m^t$, $\Lambda := \mathbb{Z}_m$, $\mathcal{V} := \mathbb{Z}_m$

For $\lambda \in \Lambda$, $a = (a_0, a_1, \ldots, a_{t-1}) \in \mathcal{U}$, define

$$h_\lambda(a) := \sum_{i=0}^{t-1} a_i \lambda^i$$

Define

$$\mathcal{H} := \{ h_\lambda \}_{\lambda \in \Lambda}$$

**Theorem:** $\mathcal{H}$ is $(t-1)/m$-universal
Proof. Suppose \((a_0, \ldots, a_{t-1}) \neq (b_0, \ldots, b_{t-1})\)

We want to count the number \(N\) of solutions \(\lambda\) to the equation

\[
\sum_i a_i \lambda^i = \sum_i b_i \lambda^i.
\]

Re-write this as

\[
\sum_i c_i \lambda^i = 0
\]

where \(c_i := a_i - b_i\)

\(N = \#\) roots of \(\sum_i c_i x^i\), which is a non-zero polynomial of degree at most \(t-1\)

\[
\therefore N \leq t - 1 = (t - 1)/m \cdot m.\quad \text{QED}
\]
Another $\varepsilon$-universal hash

More flexible choice of key space and slot space

Let $m$ be a positive integer, $p \geq m$ be a prime, and $t$ a positive integer

Define $\mathcal{U} := [0..m) \leq t$, $\Lambda := [0..p)$, $\mathcal{V} := [0..m)$

For $\lambda \in \Lambda$, $a = (a_1, a_2, \ldots, a_l) \in \mathcal{U}$, where $0 < l \leq t$, define

$$h_\lambda(a) := (\lambda^l + a_1\lambda^{l-1} + \cdots + a_{l-1}\lambda) \mod p + a_l \mod m$$

(empty string maps to 0)

Fact: $\{h_\lambda\}_{\lambda \in \Lambda}$ is $4t/m$-universal (exercise)
Pairwise Independent Hashing: a stronger notion

Let $\mathcal{H} = \{h_\lambda\}_{\lambda \in \Lambda}$ be a family of hash functions from $\mathcal{U}$ to $\mathcal{V}$.

Let $m := |\mathcal{V}|$

We will assume that $|\mathcal{U}| > 1$

**Def’n:** $\mathcal{H}$ is called **pairwise independent** if for all $a, b \in \mathcal{U}$ with $a \neq b$, and for all $r, s \in \mathcal{V}$, we have

$$|\{\lambda \in \Lambda : h_\lambda(a) = r \text{ and } h_\lambda(b) = s\}| = \frac{|\Lambda|}{m^2}.$$
Probabilistic interpretation

Let $R$ be a random variable, uniformly dist’ed over $\Lambda$

For each $a \in \mathcal{U}$, define the random variable $V_a := h_R(a)$

**Fact:** The family of random variables $\{V_a\}_{a \in \mathcal{U}}$ is pairwise independent, with each $V_a$ uniformly distributed over $\mathcal{V}$
A pairwise independent family

Let $m$ be a prime, and $t$ a positive integer
Define $\mathcal{U} := \mathbb{Z}_m^t$, $\Lambda := \mathbb{Z}_m^{t+1}$, $\mathcal{V} := \mathbb{Z}_m$

For $\lambda = (\lambda_0, \lambda_1, \ldots, \lambda_t) \in \Lambda$, $a = (a_1, \ldots, a_t) \in \mathcal{U}$, define

$$h_\lambda(a) := \lambda_0 + \sum_{i=1}^{t} a_i \lambda_i$$

Define

$$\mathcal{H} := \{ h_\lambda \}_{\lambda \in \Lambda}$$

Fact: $\mathcal{H}$ is pairwise independent (Exercise)
**Application: message authentication**

Alice and Bob share a random hash index \( R \)

Later, Alice sends a message \( M \) to Bob, together with a hash code \( C := h_R(M) \)

An adversary can try to fool Bob, by replacing Alice’s message \( M \) with a message \( M' \neq M \), and replacing the hash code \( C' \) such that \( h_R(M') = C' \)

Here, \( M' \) and \( C' \) are functions of \( M \) and \( C \)

Pairwise independent hashing implies

\[
\Pr[M' \neq M \text{ and } h_R(M') = C'] \leq \frac{1}{m}
\]

*Intuition:* the hash code \( h_R(M) \) reveals nothing about the hash code \( h_R(M') \)