Lecture 9: Pairwise-Independent Hashing

Stefano Tessaro
tessaro@cs.washington.edu
This week – Applications + Random Variables

• **Today:** Data structures!
  – The power of pairwise-independence

• **Wednesday:** (Simple) Machine Learning
  – Naïve Bayes Learning
  – (Optional) Project

• **Friday:** Random Variables
Definition. The events $\mathcal{A}_1, \ldots, \mathcal{A}_n$ are **independent** if for every $k \leq n$ and $1 \leq j_1 < j_2 < \cdots < j_k \leq n$,

$$
\mathbb{P}(\mathcal{A}_{j_1} \cap \mathcal{A}_{j_2} \cap \cdots \cap \mathcal{A}_{j_k}) = \mathbb{P}(\mathcal{A}_{j_1}) \cdot \mathbb{P}(\mathcal{A}_{j_2}) \cdots \mathbb{P}(\mathcal{A}_{j_k}).
$$
Definition. The events $\mathcal{A}_1, \ldots, \mathcal{A}_n$ are **independent** if for every $k \leq n$ and $1 \leq j_1 < j_2 < \cdots < j_k \leq n$,

$$\mathbb{P}(\mathcal{A}_{j_1} \cap \mathcal{A}_{j_2} \cap \cdots \cap \mathcal{A}_{j_k}) = \mathbb{P}(\mathcal{A}_{j_1}) \cdot \mathbb{P}(\mathcal{A}_{j_2}) \cdots \mathbb{P}(\mathcal{A}_{j_k}).$$

Definition. The events $\mathcal{A}_1, \ldots, \mathcal{A}_n$ are **pairwise-independent** if for all distinct $i, j \in [n]$,

$$\mathbb{P}(\mathcal{A}_i \cap \mathcal{A}_j) = \mathbb{P}(\mathcal{A}_i) \cdot \mathbb{P}(\mathcal{A}_j).$$

Today: Application to CS of pairwise-independence!
Basic Problem

Problem: Store a subset $S$ of a large set $X$.

Example. $X =$ set of all US ZIP codes
$S =$ set of ZIP codes of CSE 312 students

$|X| \approx 42000$
$|S| \approx 50$

Two goals:
1. **Constant-time** answering of queries “Is $x \in S$?”
2. **Minimize storage** requirements.

Imagine for simplicity $X = \{1, \ldots, K\} = [K]$
Naïve Solution – Constant Time

**Idea:** Represent $S$ as an array $a$ with $K$ entries.

$$S = \{1, 3, ..., K − 1\}$$

**Membership test:** To check $i \in S$ just check whether $a[i] = 1$.

→ constant time! 😊😊

**Storage:** Require storing $K$ bits, even for small $S$. 😞😔
Naïve Solution – Small Storage

**Idea:** Represent $S$ as a list with $|S|$ entries.

$$S = \{1, 3, \ldots, K - 1\}$$

**Storage:** Grows with $|S|$ only

**Membership test:** Check $i \in S$ requires time linear in $|S|$.

(Can be made logarithmic by using a tree)
Today – Hash Table

Idea: Represent $S$ as an array $a$ with $M \ll K$ entries.

$S = \{1, 3, \ldots, K - 1\}$

Membership test: To check $i \in S$ just check whether $a[h(i)] = i$.

Storage: $M$ elements from $\{0\} \cup [K]$

$$a[h(i)] = \begin{cases} i & \text{if } i \in S \\ 0 & \text{if } i \notin S \end{cases}$$
Our Solution

**Challenge 1:** Ensure $h(i) \neq h(j)$ for all $i, j \in S$

**Membership test:** To check $i \in S$ just check whether $a[h(i)] = i$.

**Storage:** $M$ elements from $\{0\} \cup [K]$

**Challenge 2:** Ensure $M \approx |S|^2$

We will show today $M \approx |S|^2$
Our Solution

**Challenge 1:** Ensure $h(i) \neq h(j)$ for all $i, j \in S$

**Membership test:** To check $i \in S$ just check whether $a[h(i)] = i$.

**Impossible!** Because $M < K$, for every $h$, we can always come up with a set $S$ where this is not true!

(By the pigeonhole principle)

**Solution:** We will pick $h$ randomly and show it is good for $S$ with good probability (e.g., $\geq 1/2$)

hash function $h: [K] \to [M]$
How to choose $h$?

Fix set $S \subseteq [K]$ with $n$ elements. Wlog $S = \{1, \ldots, n\}$

**First idea:** Pick $h: [K] \to [M]$ randomly from the set of all functions.

**Theorem.** $\mathbb{P}(\exists i \neq j: h(i) = h(j)) \leq \frac{n(n-1)}{2M}$

Set $M = n^2 = |S|^2$ for probability $< \frac{1}{2}$

**Note:** This will not be a good idea in the end. Why? We need to store entire description of $h$! Let’s stick with it for now.
Proof – Random Hash

\[ \Omega = \{ \mathbf{h} \mid \mathbf{h}: [K] \rightarrow [M] \} \]

\[ \mathbb{P}(\mathbf{h}) = \frac{1}{M^K} \]

\[ \mathcal{C} = \{ \mathbf{h} \mid \exists i \neq j: \mathbf{h}(i) = \mathbf{h}(j) \} \]

For every \( i < j \):

\[ \mathcal{C}_{i,j} = \{ \mathbf{h} \mid \mathbf{h}(i) = \mathbf{h}(j) \} \]

**Claim.** \( \mathcal{C} = \mathcal{C}_{1,2} \cup \mathcal{C}_{1,3} \cup \cdots \mathcal{C}_{n-1,n} = \bigcup_{i<j} \mathcal{C}_{i,j} \)

“Proof”: \( \mathcal{C} \) happens if and only if (\( \mathbf{h}(1) = \mathbf{h}(2) \) or \( \mathbf{h}(1) = \mathbf{h}(3) \) or \( \mathbf{h}(1) = \mathbf{h}(4) \) or … or \( \mathbf{h}(n - 1) = \mathbf{h}(n) \))
Proof – Random Hash

\[ \Omega = \{ h \mid h : [K] \rightarrow [M] \} \]
\[ \mathbb{P}(h) = \frac{1}{M^K} \]

For every \( i < j \): \( C_{i,j} = \{ h \mid h(i) = h(j) \} \)

Claim. For all \( i < j \), \( \mathbb{P}(C_{i,j}) = \frac{1}{M} \)

Proof: Let \( \mathcal{A}_i(y) = \{ h \mid h(i) = y \} \) [i.e., we pick a function that maps \( i \) to \( y \).]

\[ \mathbb{P}(C_{i,j}) = \sum_y \mathbb{P}(\mathcal{A}_i(y) \cap \mathcal{A}_j(y)) \]

Note that \( \mathbb{P}(\mathcal{A}_i(y)) = \mathbb{P}(\mathcal{A}_j(y)) = \frac{M^{K-1}}{M^K} = \frac{1}{M} \)
\[ \mathbb{P}(\mathcal{A}_i(y) \cap \mathcal{A}_j(y)) = \frac{M^{K-2}}{M^K} = \frac{1}{M^2} = \frac{1}{M} \cdot \frac{1}{M} \]

Independent!
Proof – Random Hash

\[ \Omega = \{ h \mid h : [K] \to [M] \} \]

\[ \mathbb{P}(h) = \frac{1}{M^K} \]

For every \( i < j \): \( C_{i,j} = \{ h \mid h(i) = h(j) \} \)

**Claim.** For all \( i < j \), \( \mathbb{P}(C_{i,j}) = \frac{1}{M} \)

**Proof:** Let \( \mathcal{A}_i(y) = \{ h \mid h(i) = y \} \) [i.e., we pick a function that \( i \) maps to \( y \).]

\[ \mathbb{P}(C_{i,j}) = \sum_y \mathbb{P}(\mathcal{A}_i(y) \cap \mathcal{A}_j(y)) = \sum_y \mathbb{P}(\mathcal{A}_i(y)) \cdot \mathbb{P}(\mathcal{A}_j(y)) \]

\[ = \sum_y \frac{1}{M^2} = M \times \frac{1}{M^2} = \frac{1}{M} \]
Proof – Random Hash

\[ C = \bigcup_{i<j} C_{i,j} \quad \mathbb{P}(C_{i,j}) = \frac{1}{M} \]

Claim. For all \( i < j \), \( \mathbb{P}(C_{i,j}) = 1/M \)

\[ \mathbb{P}(C) = \mathbb{P}(\bigcup_{i<j} C_{i,j}) \leq \sum_{i<j} \mathbb{P}(C_{i,j}) = \sum_{i<j} \frac{1}{M} = \binom{n}{2} \frac{1}{M} = \frac{n(n-1)}{2M} \]

Union bound: \( \mathbb{P}(A_1 \cup \cdots \cup A_n) \leq \mathbb{P}(A_1) + \cdots + \mathbb{P}(A_n) \)

Theorem. \( \mathbb{P}(\exists i \neq j : h(i) = h(j)) \leq \frac{n(n-1)}{2M} \)
Back to Data Structures

Problem: Description of $h: [K] \rightarrow [M]$ needs to be stored along with the set $S$.

Need to store $K$ elements from $[M]$. 😞
Our proof did not need $h$ to be picked at random from all functions ...

**Claim.** For all $i < j$, $\mathbb{P}(C_{i,j}) = 1/M$

\[
\mathbb{P}(C_{i,j}) = \sum_y \mathbb{P}(A_i(y) \cap A_j(y)) = \sum_y \mathbb{P}(A_i(y)) \mathbb{P}(A_j(y)) \\
= \sum_y \frac{1}{M^2} = M \times \frac{1}{M^2} = \frac{1}{M}
\]

This only requires pairwise independence of the $A_i(y)$'s
Definition. A set $H$ of functions $[K] \rightarrow [M]$ is pairwise independent if for all distinct $i \neq j$, and all $y, y' \in [M]$

$$|\{h \in H \mid h(i) = y \land h(j) = y'\}| = \frac{|H|}{M^2}$$


Theorem. $\mathbb{P}(\exists i \neq j: h(i) = h(j)) \leq \frac{n(n-1)}{2M}$

Proof as before: Only one step different (next slide)
Definition. A set $H$ of functions $[K] \to [M]$ is pairwise independent if for all distinct $i \neq j$, and all $y, y' \in [M]$

$$|\{h \in H | h(i) = y \land h(j) = y'\}| = \frac{|H|}{M^2}$$

Let $\mathcal{A}_i(y) = \{h \in H | h(i) = y\}$

$$\mathbb{P}(\mathcal{A}_i(y) \cap \mathcal{A}_j(y)) = \frac{|\{h \in H | h(i) = y \land h(j) = y'\}|}{|H|} = \frac{1}{M^2}$$

This is all we needed!
Pairwise-Independent Functions

**Fact:** The set of all functions $[K] \rightarrow [M]$ is pairwise independent

– Size $M^K$
Pairwise-Independent Functions

Fact (informal)*: There exists a pairwise-independent set \( H \) of functions \( [K] \to [M] \) with size \( |H| = K^2 \)

- Described by two elements of \([K]\).
- Idea*: \( x \to (ax + b \mod K) \mod M \) i.e., function described by \( a,b \) in \([K]\).
- Overall solution takes storing \( |S|^2 + 2 \) elements from \([K] \cup \{0\} \) (i.e., array + description of a chosen good function)

Several other applications: Data structures, algorithms, cryptography, ...

*Some cheating here, as usually one gets an approximation of a pairwise independent hash function, where \( \mathbb{P}(\mathcal{A}_i(y) \cap \mathcal{A}_j(y)) \approx \mathbb{P}(\mathcal{A}_i(y)) \cdot \mathbb{P}(\mathcal{A}_j(y)) \)