Advanced Algorithms
Balls into Bins
Balls into Bins

$n$ balls \[ \text{uniform & independent} \]

$m$ bins

random function $f : [n] \to [m]$

birthday, coupon collector, occupancy, ...
Random Function

- $n$ balls into $m$ bins:

$$\Pr[\text{assignment}] = \frac{1}{m} \cdots \frac{1}{m} = \frac{1}{m^n}$$

- uniform random function:

$$\Pr[f] = \frac{1}{| [n] \rightarrow [m] |} = \frac{1}{m^n}$$

<table>
<thead>
<tr>
<th>1-1</th>
<th>birthday</th>
</tr>
</thead>
<tbody>
<tr>
<td>on-to</td>
<td>coupon collector</td>
</tr>
<tr>
<td>pre-image size</td>
<td>occupancy</td>
</tr>
</tbody>
</table>
Birthday Paradox

Paradox:
(i) a statement that leads to a contradiction;
(ii) a situation which defies intuition.

In a class of $m > 57$ students, with $>99\%$ probability, there are two students with the same birthday.

Assumption: birthdays are uniformly & independently distributed.

$n$ balls are thrown into $m$ bins:

event $\mathcal{E}$: each bin receives $\leq 1$ balls
Birthday Paradox

$n$ balls are thrown into $m$ bins:
event $\mathcal{E}$: each bin receives $\leq 1$ balls

\[
\Pr[\mathcal{E}] = \frac{\left| [n] \rightarrow [m] \right|}{\left| [n] \rightarrow [m] \right|} = \frac{m(m-1)\ldots(m-n+1)}{m^n}
\]

\[
= \prod_{i=0}^{n-1} \left( 1 - \frac{i}{m} \right)
\]
Birthday Paradox

$n$ balls are thrown into $m$ bins:

event $\mathcal{E}$: each bin receives $\leq 1$ balls

Suppose that balls are thrown one-by-one:

$$\Pr[\mathcal{E}] = \Pr[\text{all } n \text{ balls are thrown into distinct bins}]$$

chain rule

$$\Pr[\text{the } i\text{th ball is thrown into an empty bin } \mid \text{first } i - 1 \text{ balls are thrown into distinct bins}]$$

$$= \prod_{i=1}^{n} \Pr[\text{the } i\text{th ball is thrown into an empty bin } \mid \text{first } i - 1 \text{ balls are thrown into distinct bins}]$$

$$= \prod_{i=1}^{n} \left(1 - \frac{i - 1}{m}\right) = \prod_{i=0}^{n-1} \left(1 - \frac{i}{m}\right)$$
Birthday Paradox

$n$ balls are thrown into $m$ bins:
event $\mathcal{E}$: each bin receives $\leq 1$ balls

(Taylor: $1 - x \approx e^{-x}$ for $x = o(1)$)

$$Pr[\mathcal{E}] = \prod_{i=0}^{n-1} \left(1 - \frac{i}{m}\right) \approx \prod_{i=0}^{n-1} e^{-\frac{i}{m}} \approx e^{-\frac{n^2}{2m}}$$

Formally:

$$e^{-(1+o(1))\frac{n^2}{2m}} \leq \prod_{i=0}^{n-1} \left(1 - \frac{i}{m}\right) \leq e^{-(1-o(1))\frac{n^2}{2m}}$$

(assuming $n \ll m$)

when $n = \sqrt{2m \ln \frac{1}{p}} \implies Pr[\mathcal{E}] = (1 \pm o(1))p$
Birthday Paradox

$n$ balls are thrown into $m$ bins:

event $\mathcal{E}$: each bin receives $\leq 1$ balls

$$Pr[\mathcal{E}] = \prod_{i=0}^{n-1} \left(1 - \frac{i}{m}\right)$$

Formally:

$$e^{-(1+o(1))n^2/2m}$$

(assuming $n \ll m$)

when $n = \sqrt{2m \ln \frac{1}{p}}$ $\implies$ $Pr[\mathcal{E}] = (1 \pm o(1))p$
Data Structure for Set

Data: a set $S$ of $n$ items $x_1, x_2, \ldots, x_n \in U = [N]$  
Query: an item $x \in U$  
Determine whether $x \in S$.

- **Space cost**: size of data structure (in bits)  
  - **entropy** of a set: $O(n \log N)$ bits (when $N \gg n$)  
- **Time cost**: time to answer a query (in memory accesses)  
- **Balanced tree**: $O(n \log N)$ space, $O(\log n)$ time  
- **Perfect hashing**: $O(n \log N)$ space, $O(1)$ time
Perfect Hashing

\[ S = \{a, b, c, d, e, f\} \subseteq [N] \text{ of size } n \]

uniform random \( h \) \([N] \rightarrow [m]\)  

no collision \[ \Pr[\text{perfect}] \approx e^{-n^2/2m} > 1/2 \]

Table \( T \):
\[
\begin{array}{cccccc}
  e & b & d & f & c & a \\
\end{array}
\]

\( m = n^2 \)  

Birthday

SUHA: Simple Uniform Hash Assumption

Query\( (x) \):

- retrieve hash function \( h \);
- check whether \( T[h(x)] = x \);
Universal Hashing

Universal Hash Family (Carter and Wegman 1979):
A family $\mathcal{H}$ of hash functions in $U \rightarrow [m]$ is $k$-universal if for any distinct $x_1, \ldots, x_k \in U$,
\[
\Pr_{h \in \mathcal{H}}[h(x_1) = \cdots = h(x_k)] \leq \frac{1}{m^{k-1}}.
\]
Moreover, $\mathcal{H}$ is strongly $k$-universal ($k$-wise independent) if for any distinct $x_1, \ldots, x_k \in U$ and any $y_1, \ldots, y_k \in [m]$,
\[
\Pr_{h \in \mathcal{H}}\left[\bigwedge_{i=1}^{k} h(x_i) = y_i\right] = \frac{1}{m^k}.
\]
k-Universal Hash Family

hash functions $h : U \rightarrow [m]$

- **Linear congruential hashing:**
  - Represent $U \subseteq \mathbb{Z}_p$ for sufficiently large prime $p$
  - $h_{a,b}(x) = ((ax + b) \mod p) \mod m$
  - $\mathcal{H} = \left\{ h_{a,b} \mid a \in \mathbb{Z}_p \setminus \{0\}, b \in \mathbb{Z}_p \right\}$

**Theorem:**
The linear congruential family $\mathcal{H}$ is 2-wise independent.

- **Degree-$k$ polynomial in finite field with random coefficients**
- **Hashing between binary fields:** $GF(2^w) \rightarrow GF(2^l)$
  $$h_{a,b}(x) = (a \cdot x + b) \gg (w-1)$$
Birthday Paradox (pairwise independence)

$n$ balls are thrown into $m$ bins: by 2-universal hashing

event $\mathcal{E}$: each bin receives $\leq 1$ balls

- Location of $n$ balls: $X_1, X_2, \ldots, X_n \in [m]$  
- Total # of collisions:
  \[ Y = \sum_{i<j} I[X_i = X_j] \]
- Linearity of expectation:
  \[ \mathbb{E}[Y] = \sum_{i<j} \Pr[X_i = X_j] \leq \binom{n}{2} \frac{1}{m} \]
  when $n \leq \sqrt{2m \epsilon}$
- Markov’s inequality:
  \[ \Pr[\neg \mathcal{E}] = \Pr[Y \geq 1] \leq \mathbb{E}[Y] \leq \epsilon \]
Perfect Hashing

\[ S = \{a, b, c, d, e, f\} \subseteq [N] \text{ of size } n \]

2-universal \[ h \colon [N] \to [m] \]

\[ \Pr[\text{imperfect}] = \frac{n(n-1)}{2m} \]

Table \( T \):

| a | b | c | d | e | f | m |

For 2-universal family \( \mathcal{H} \) from \([N]\) to \([m]\), if \( m > \binom{n}{2} \), for any \( S \subseteq [N] \) of size \( n \), there is an \( h \in \mathcal{H} \) that cause no collisions over \( S \).

Query\( (x) \):

- retrieve hash function \( h \);
- check whether \( T[h(x)] = x \);
FKS Perfect Hashing
(Fredman, Komlós, Szemerédi, 1984)

**Data:** a set $S$ of $n$ items $x_1, x_2, \ldots, x_n \in U = [N]$

**Query:** an item $x \in U$

Determine whether $x \in S$.

- **Space cost:** $O(n)$ words (each of $O(\log N)$ bits)
- **Time cost:** $O(1)$ for each query in the worst case
FKS Perfect Hashing

$S : n$ items

primary hashing $h$  

$[N] \rightarrow [n]$  

buckets:

perfect hashing for $B_1$

perfect hashing for $B_n$
FKS Perfect Hashing

Set $S \subseteq [N]$ of size $n$

- $h : [N] \rightarrow [n]$
- $B_1 B_2 \ldots B_n$

Query($x$):
- retrieve primary hash $h$;
- goto bucket $i = h(x)$;
- retrieve secondary hash $h_i$;
- check whether $T_i[h_i(x)] = x$;

Set of size $S$ using space $|B_1|^2$

- $h_1, \ldots, h_n$ from 2-universal family s.t. $h_i$ is perfect for $B_i$ for all $i$
Collision Number

$n$ balls are thrown into $m$ bins by 2-universal hashing

- Location of $n$ bins: $X_1, X_2, \ldots, X_n \in [m]$
  
  Collision #: $Y = \sum_{i<j} I[X_i = X_j]$

- Linearity of expectation:
  
  $$\mathbb{E}[Y] = \sum_{i<j} \Pr[X_i = X_j] \leq \binom{n}{2} \frac{1}{m}$$

- Size of the $i$-th bin: $|B_i|$
  
  $$Y = \sum_{i=1}^{n} \left( \frac{|B_i|}{2} \right) = \frac{1}{2} \sum_{i=1}^{n} |B_i| (|B_i| - 1) \implies \mathbb{E} \left[ \sum_{i=1}^{n} |B_i|^2 \right] = \frac{n(n-1)}{m} + n$$
FKS Perfect Hashing

Set $S \subseteq [N]$ of size $n$

Query($x$):
- retrieve primary hash $h$;
- goto bucket $i = h(x)$;
- retrieve secondary hash $h_i$;
- check whether $T_i[h_i(x)] = x$;

• $\exists h$ from a 2-universal family s.t. the total space cost is $O(n)$
FKS Perfect Hashing
(Fredman, Komlós, Szemerédi, 1984)

Data: a set $S$ of $n$ items $x_1, x_2, \ldots, x_n \in U = [N]$

Query: an item $x \in U$

Determine whether $x \in S$.

- $O(n \log N)$ space, $O(1)$ time in the worst case
- Dynamic version: [Dietzfelbinger, Karlin, Mehlhorn, Heide, Rohnert, Tarjan, 1984]
Balls into Bins
(Coupon Collector)

uniform & independent

surjection (cover all bins)
Coupon Collector

coupons in cookie box

each box comes with a uniformly random coupon

number of boxes bought to collect all $n$ coupons

number of balls thrown to cover all $n$ bins
**Coupon Collector**

- **$X$**: number of balls thrown to make all the $n$ bins nonempty

\[ X = \sum_{i=1}^{n} X_i \]

- $X_i$ is geometric!

- with $p_i = 1 - \frac{i - 1}{n}$

\[ \mathbb{E}[X_i] = \frac{1}{p_i} = \frac{n}{n - i + 1} \]
Coupon Collector

\[ X : \ \text{number of balls thrown to make all the } n \ \text{bins nonempty} \]

\[ X_i : \ \text{number of balls thrown while there are exactly } (i-1) \ \text{nonempty bins} \]

\[ \mathbb{E}[X] = \sum_{i=1}^{n} \mathbb{E}[X_i] \]

\[ = \sum_{i=1}^{n} \frac{n}{n - i + 1} \]

\[ = n \sum_{i=1}^{n} \frac{1}{i} \]

\[ = nH(n) \]

\[ \mathbb{E}[X_i] = \frac{1}{p_i} = \frac{n}{n - i + 1} \]

Expected \( n \ln n + O(n) \) balls

 línea de esperanzas

Harmonic number

linearity of expectations
Coupon Collector

\[ X : \text{ number of balls thrown to make all the } n \text{ bins nonempty } \]

**Theorem:** For \( c > 0 \),
\[ \Pr[ X \geq n \ln n + cn ] \leq e^{-c} \]

**Proof:** For one bin, it misses all balls with probability

\[
\left(1 - \frac{1}{n}\right)^{n \ln n + cn} = \left(1 - \frac{1}{n}\right)^{n(\ln n + c)} < e^{-(\ln n + c)} < \frac{1}{ne^c}
\]
**Coupon Collector**

**X:** number of balls thrown to make all the \( n \) bins nonempty

**Theorem:** For \( c > 0 \),
\[
\Pr[ X \geq n \ln n + cn ] \leq e^{-c}
\]

**Proof:** For one bin, it misses all balls with probability
\[
\frac{1}{ne^c}
\]
union bound!

\[
\Pr[ \exists \text{ a bin misses all balls } ] \leq n \Pr[ \text{ first bin misses all bins } ] < e^{-c}
\]
**Coupon Collector**

**Theorem**: For $c > 0$, 
\[
\Pr[ X \geq n \ln n + cn ] \leq e^{-c}
\]

**a sharp threshold**: 
\[
\lim_{n \to \infty} \Pr[X \geq n \ln n + cn] = 1 - e^{-e^{-c}}
\]
Stable Matching

- each man has a preference order of the \( n \) women;
- each woman has a preference order of the \( n \) men;
- solution: \( n \) couples
- Marriages are stable!
Stable Matching

$n$ men

$n$ women

unstable (blocking pair):

a man and a woman, who prefer each other to their current partners

stable: no blocking pairs

local optimum
fixed point
equilibrium
deadlock
Proposal Algorithm
(Gale-Shapley 1962)

Single man:
propose to the most preferable women who has not rejected him

Woman:
upon received a proposal: accept if she’s single or married to a less preferable man (divorce!)
Proposal Algorithm
(Gale-Shapley 1962)

- **woman**: once got married always married
  *(will only switch to better men!)*
- **man**: will only get worse ...
- once all women are married, the algorithm terminates, and the marriages are stable
- total number of proposals:
  $$\leq n^2$$

<table>
<thead>
<tr>
<th>Single man:</th>
</tr>
</thead>
<tbody>
<tr>
<td>propose to the most preferable women who has not rejected him</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Woman:</th>
</tr>
</thead>
<tbody>
<tr>
<td>upon received a proposal: accept if she’s single or married to a less preferable man <em>(divorce!)</em></td>
</tr>
</tbody>
</table>
### Average-Case Performance

*(Knuth 1976)*

- Every man/woman has a **uniform random permutation** as preference list.

- Expected total number of proposals?

<table>
<thead>
<tr>
<th>Single man:</th>
</tr>
</thead>
<tbody>
<tr>
<td>propose to the most preferable women who has not rejected him</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Woman:</th>
</tr>
</thead>
<tbody>
<tr>
<td>upon received a proposal: accept if she’s single or married to a less preferable man <em>(divorce!)</em></td>
</tr>
</tbody>
</table>
Principle of deferred decision

The decision of random choice in the random input is deferred to the running time of the algorithm.
Principle of Deferred Decisions

proposing in the order of a uniformly random permutation

at each time, proposing to a uniformly random woman who has not rejected him

decisions of the inputs are deferred to the time when Alg accesses them
Stochastic Domination

at each time, proposing to a uniformly & independently random woman

at each time, proposing to a uniformly random woman who has not rejected him

the man forgot who had rejected him (!)
Average-Case Performance

• uniformly and independently proposing to $n$ women

• Alg stops once all women got proposed.

• Coupon collector!

• Expected $n \ln n + O(n)$ proposals.