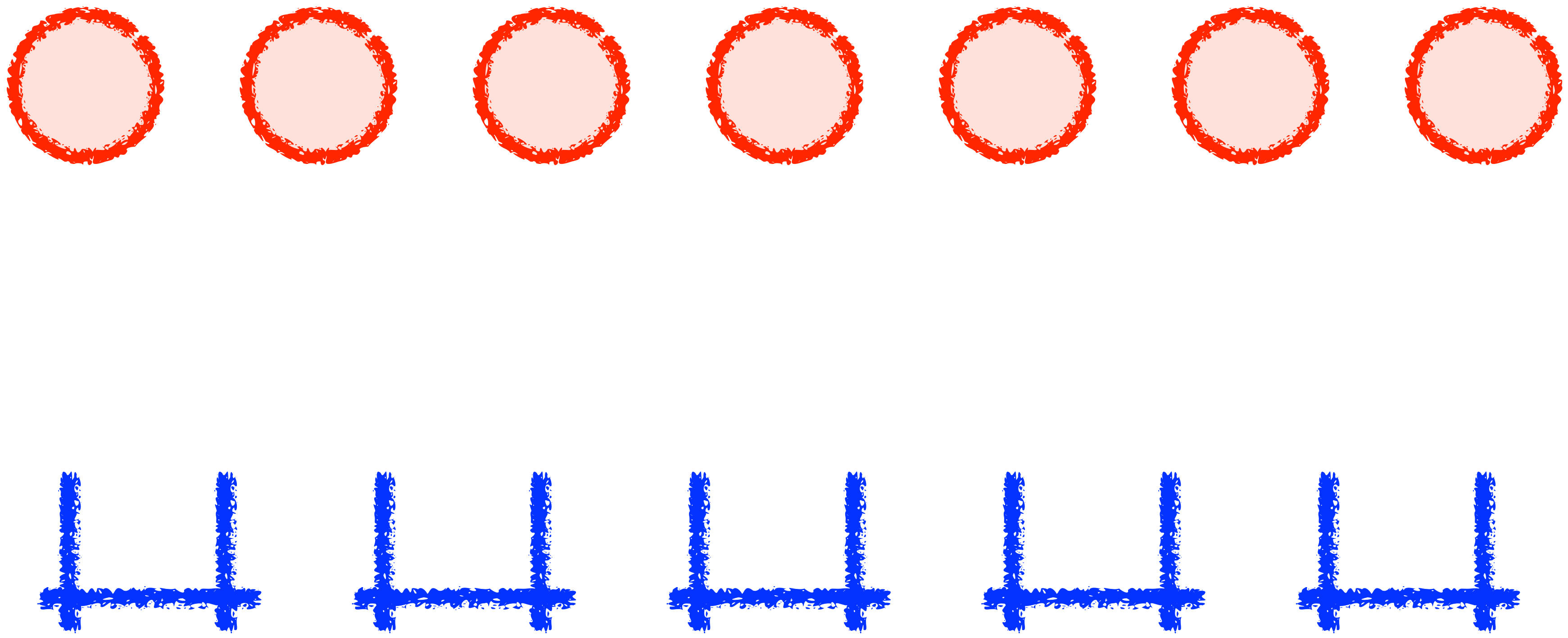
Advanced Algorithms

Hashing and Sketching

尹一通 Nanjing University, 2023 Fall

Balls into Bins



Balls into Bins (Random Function)

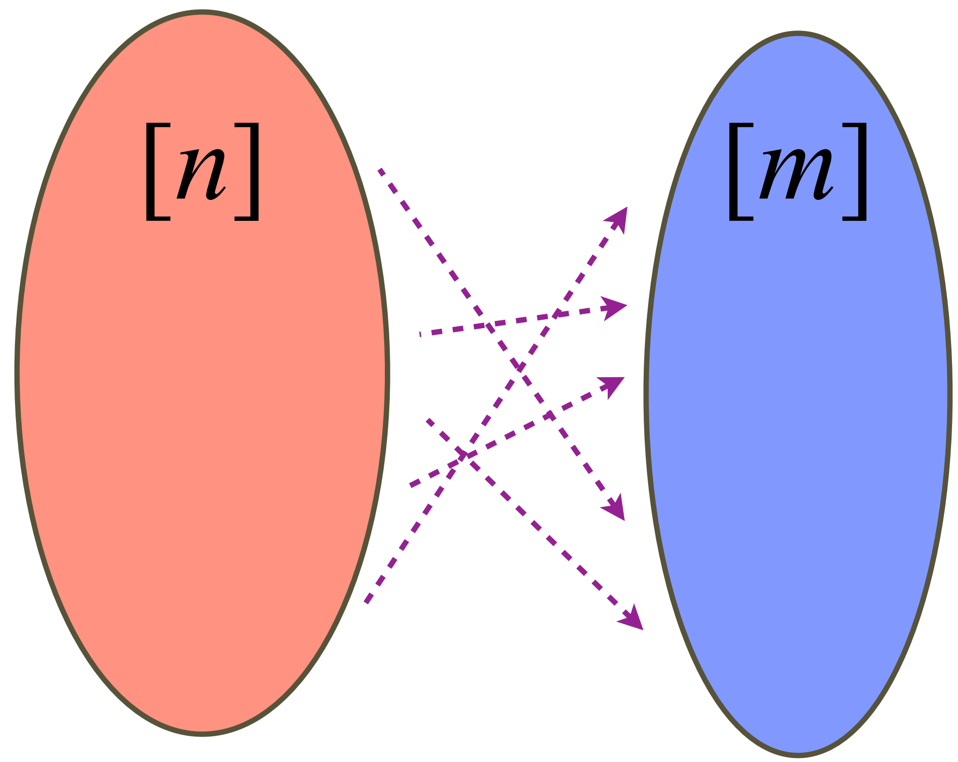
- n balls into m bins:

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

- uniform random function:

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

1-1	birthday
on-to	coupon collector
pre-image size	occupancy



uniform random function

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

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 $\mathscr{E}$: each bin receives ≤ 1 balls

Birthday Paradox

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

$$\Pr[\mathscr{E}] = \frac{\left| [n] \xrightarrow{1-1} [m] \right|}{\left| [n] \to [m] \right|} = \frac{m(m-1)\cdots(m-n+1)}{m^n}$$

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

Birthday Paradox

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

Suppose that balls are thrown one-by-one:

$\Pr[\mathscr{E}] = \Pr[$all n balls are thrown into ditinct bins$]$

chain rule $\displaystyle = \prod_{i=1}^{n} \Pr[$the ith ball is thrown into an empty bin | first $i-1$ balls are thrown into ditinct 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 $\mathscr{E}$: each bin receives ≤ 1 balls

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

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

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

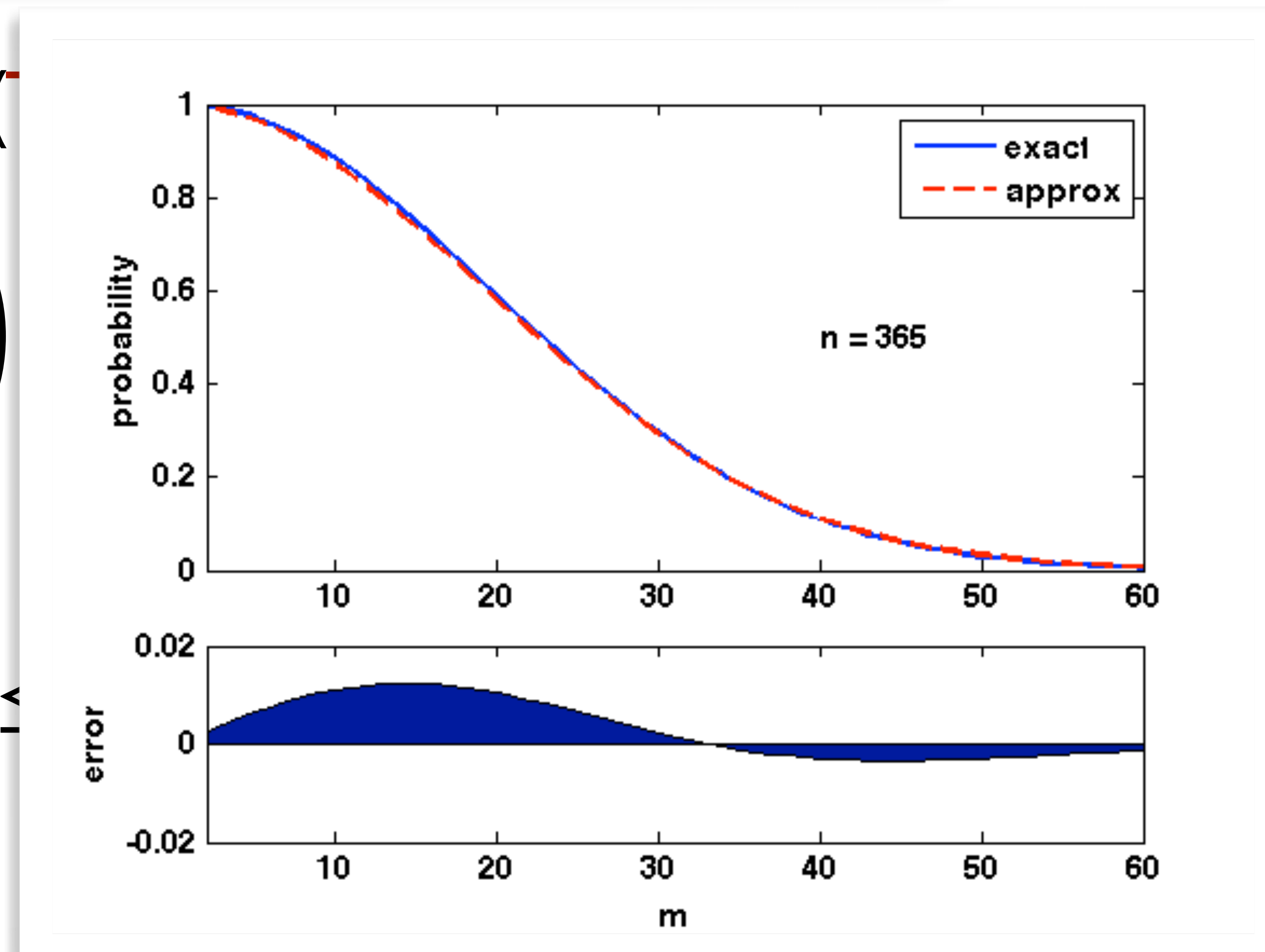
(assuming $n \ll m$)

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

Birthday Paradox

> n balls are thrown into m bins:
> event $\mathscr{E}$: each bin receives ≤ 1 balls

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



Formally: $\mathrm{e}^{-(1+o(1))n^2/2m} \leq$

(assuming $n \ll m$)

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

Hash Tables & Filters

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: $\log \binom{N}{n} = O(n \log N)$ 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

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

uniform random $\boxed{h}$ $[N] \to [m]$

no collision

$\Pr[\text{perfect}] \approx e^{-n^2/2m} > 1/2$

Table T:

e	b		d		f		c	a	

$m = n^2$

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 $\mathscr{H}$ of hash functions in $U \to [m]$ is **k-universal** if for any distinct $x_1, \ldots, x_k \in U$,

$$\Pr_{h \in \mathscr{H}} \left[h(x_1) = \cdots = h(x_k) \right] \le \frac{1}{m^{k-1}}.$$

Moreover, $\mathscr{H}$ is **strongly k-universal** (k-wise independent) if for any distinct $x_1, \ldots, x_k \in U$ and any $y_1, \ldots, y_k \in [m]$,

$$\Pr_{h \in \mathscr{H}} \left[\bigwedge_{i=1}^{k} h(x_i) = y_i \right] = \frac{1}{m^k}.$$

k-Universal Hash Family

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

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

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

- Degree-k polynomial in finite field with random coefficients
- Hashing between binary fields: $GF(2^w) \to GF(2^l)$

$$h_{a,b}(x) = \texttt{(a*x+b)>>(w-l)}$$

Birthday Paradox (pairwise independence)

n balls are thrown into m bins: by 2-universal hashing
event $\mathscr{E}$: each bin receives ≤ 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:**

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

2-universal

Markov's Inequality

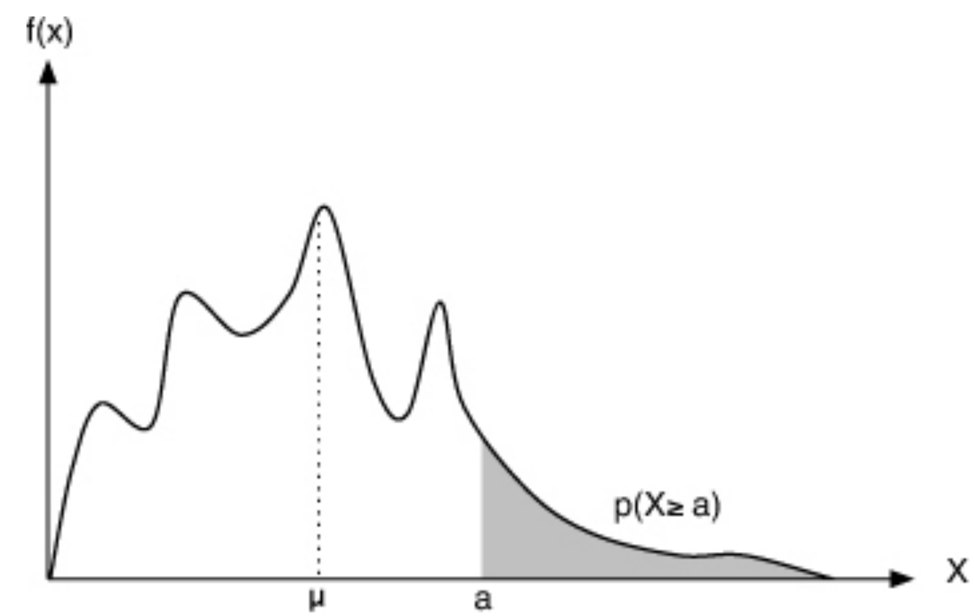
Markov's Inequality

For *nonnegative* random variable X, for any $t > 0$,

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

Let $Y = \begin{cases} 1 & X \geq t \\ 0 & \text{o.w.} \end{cases} \implies Y \leq \left\lfloor \frac{X}{t} \right\rfloor \leq \frac{X}{t}$

$$\Pr[X \geq t] = \mathbb{E}[Y] \leq \mathbb{E}\left[\frac{X}{t}\right] = \frac{\mathbb{E}[X]}{t}$$



Birthday Paradox (pairwise independence)

n balls are thrown into m bins: by 2-universal hashing
event $\mathscr{E}$: each bin receives ≤ 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:**

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

2-universal

when
$n \leq \sqrt{2m\epsilon}$

- **Markov's inequality:** $\Pr[\neg\mathscr{E}] = \Pr[Y \geq 1] \leq \mathbb{E}[Y] \leq \epsilon$

Perfect Hashing

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

2-universal $\boxed{h}$ $[N] \to [m]$ Pr[*imperfect*] $= \dfrac{n(n-1)}{2m}$

Table T: m

e	b		d		f		c	a	

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

Query(x):

retrieve hash function h;

check whether $T[h(x)] = x$;

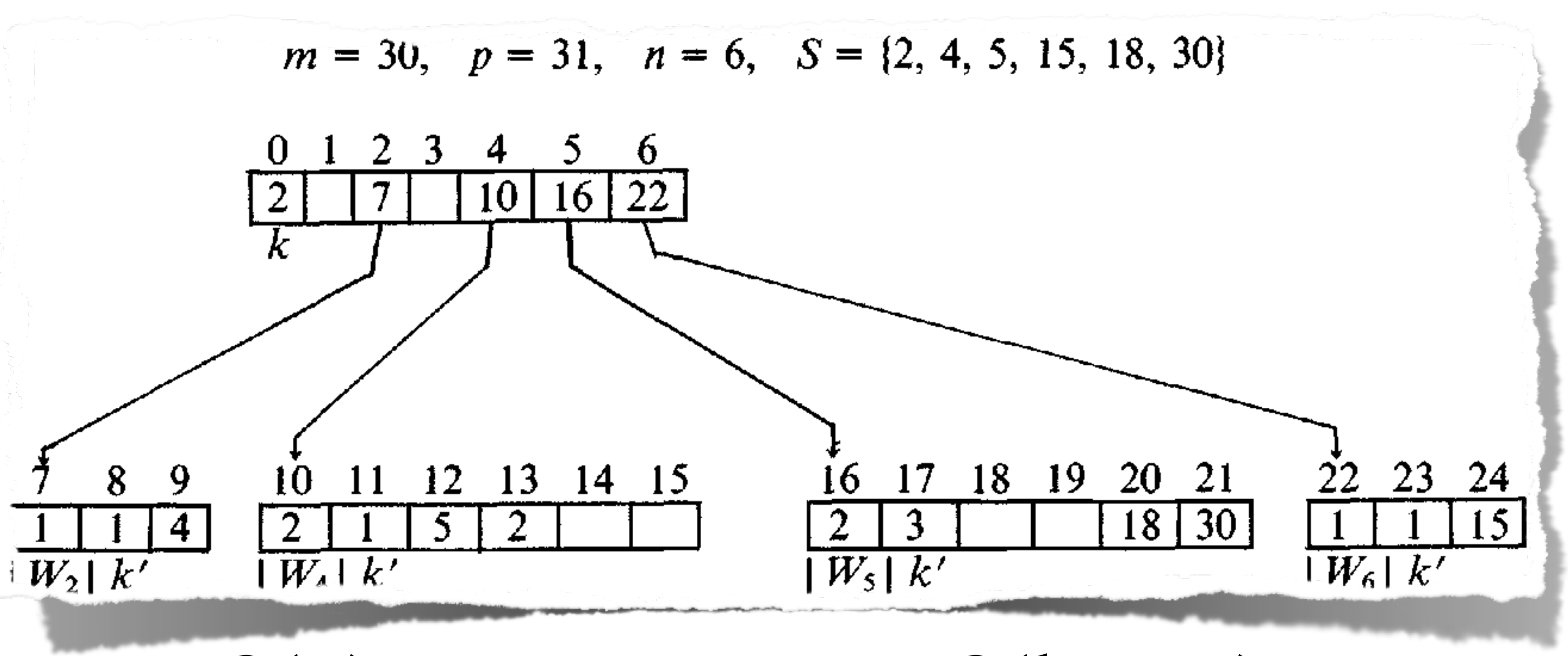
FKS Perfect Hashing

(Fredman, Komlós, Szemerédi, 1984)

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

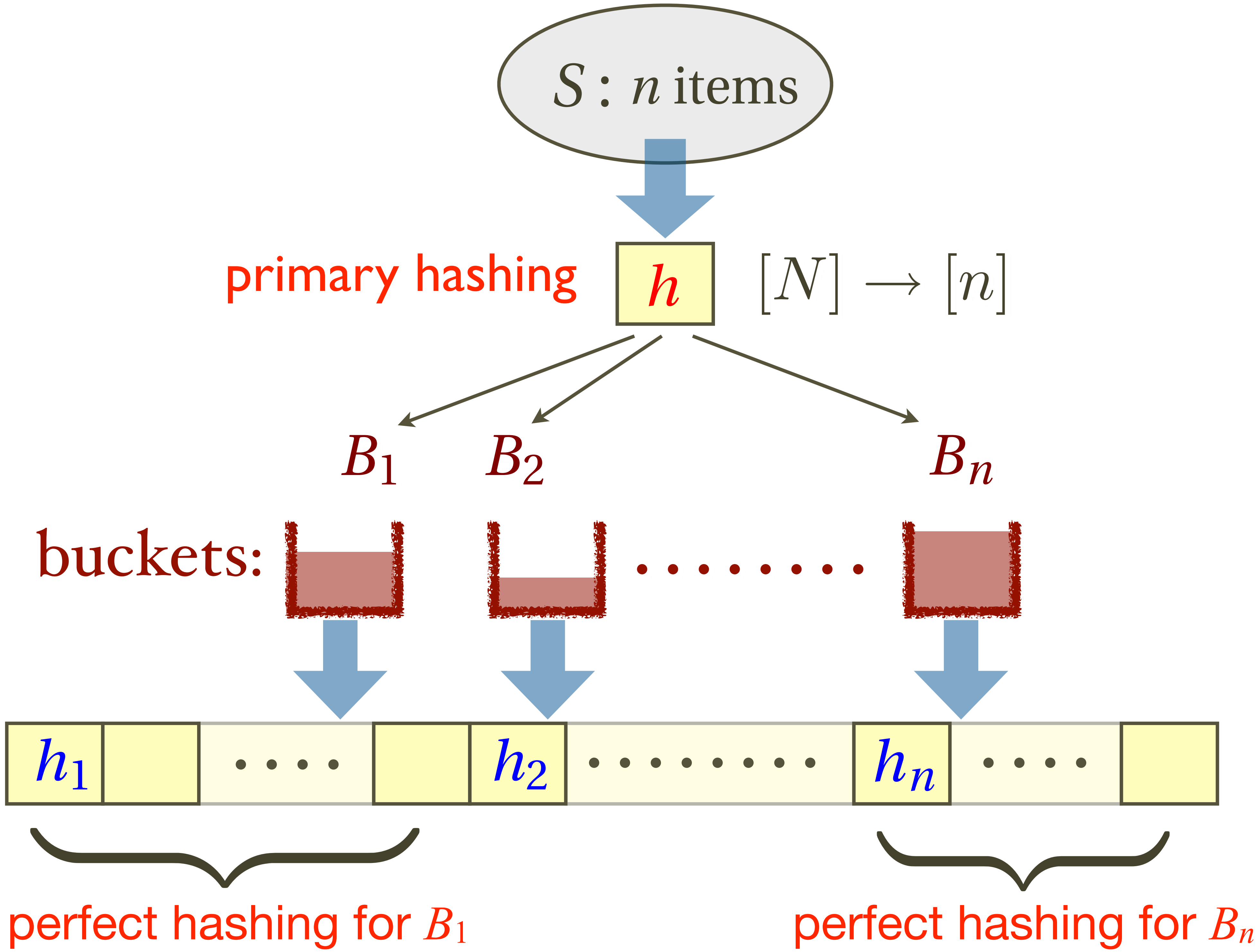
Query: an item $x \in U$

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

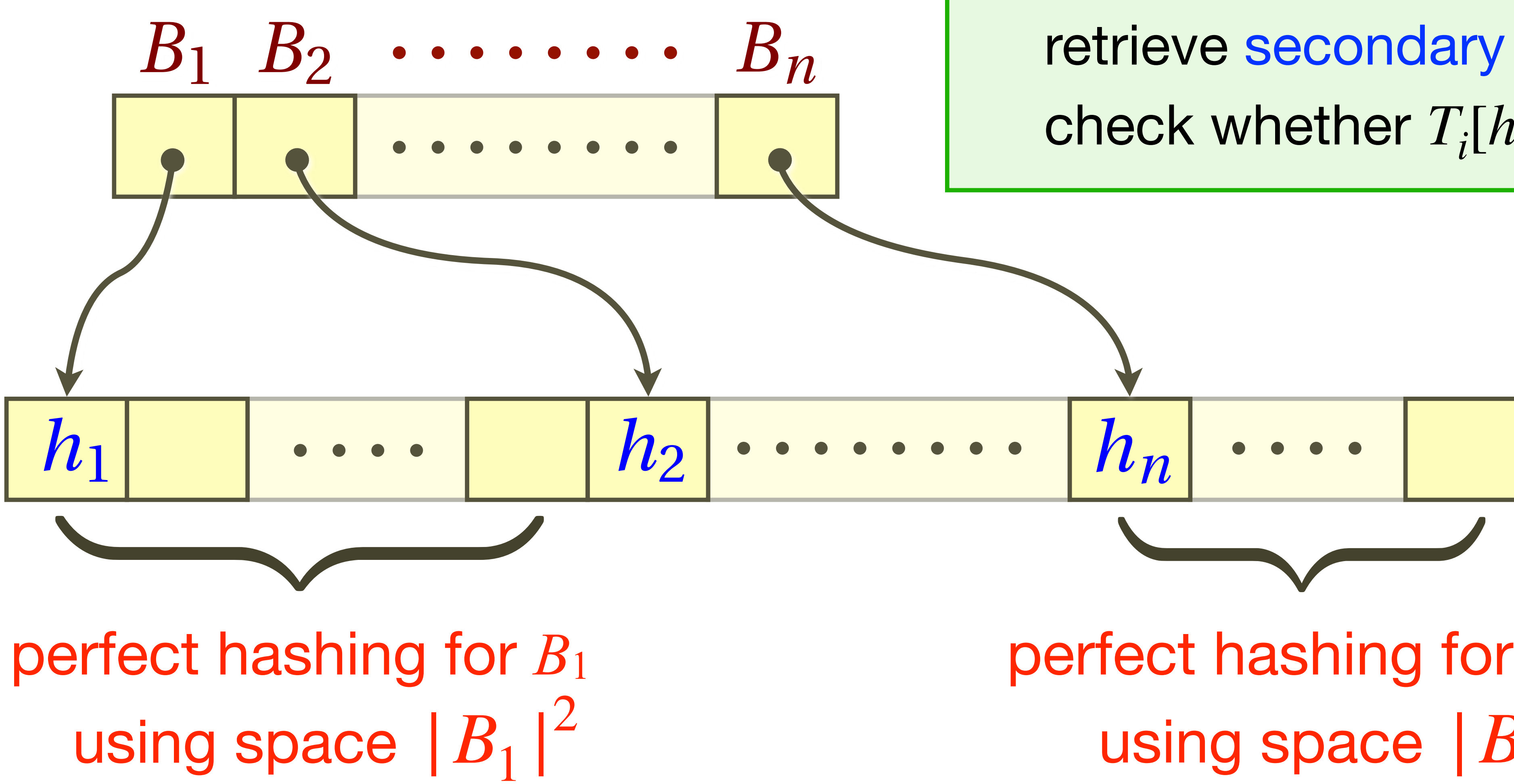
FKS Perfect Hashing



FKS Perfect Hashing

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

$h \quad [N] \longrightarrow [n]$



Query(x):

retrieve primary hash h;

goto bucket $i = h(x)$;

retrieve secondary hash h_i;

check whether $T_i[h_i(x)] = x$;

- $\exists\, h_1, \ldots, h_n$ 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:

$$\mathbb{E}[Y] = \sum_{i<j} \Pr[X_i = X_j] \le \binom{n}{2} \frac{1}{m}$$

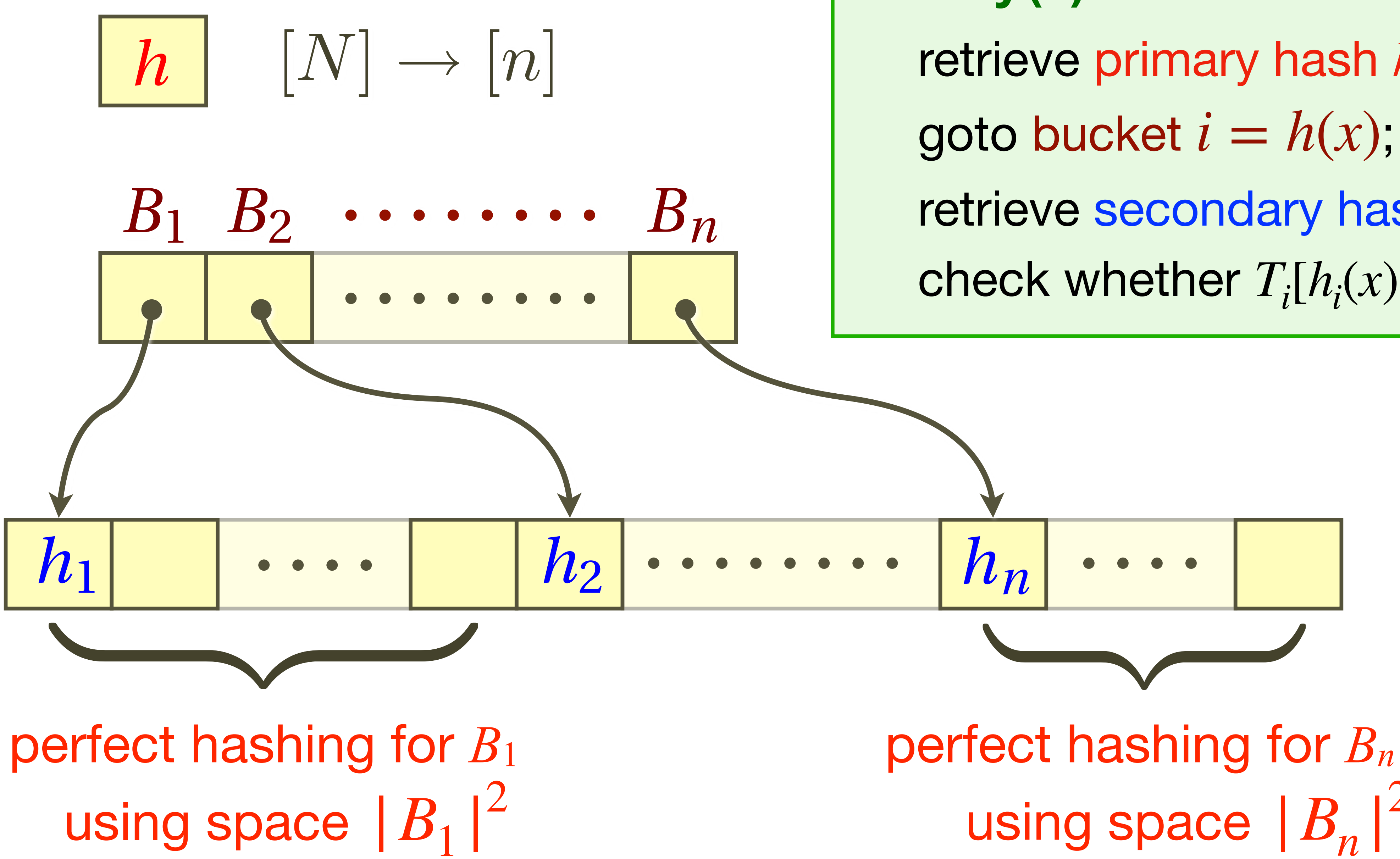
2-universal

- Size of the i-th bin: $|B_i|$

$$Y = \sum_{i=1}^{n} \binom{|B_i|}{2} = \frac{1}{2} \sum_{i=1}^{n} |B_i|(|B_i| - 1) \implies \mathbb{E}\left[\sum_{i=1}^{n} |B_i|^2\right] = \frac{n(n-1)}{m} + n$$

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)

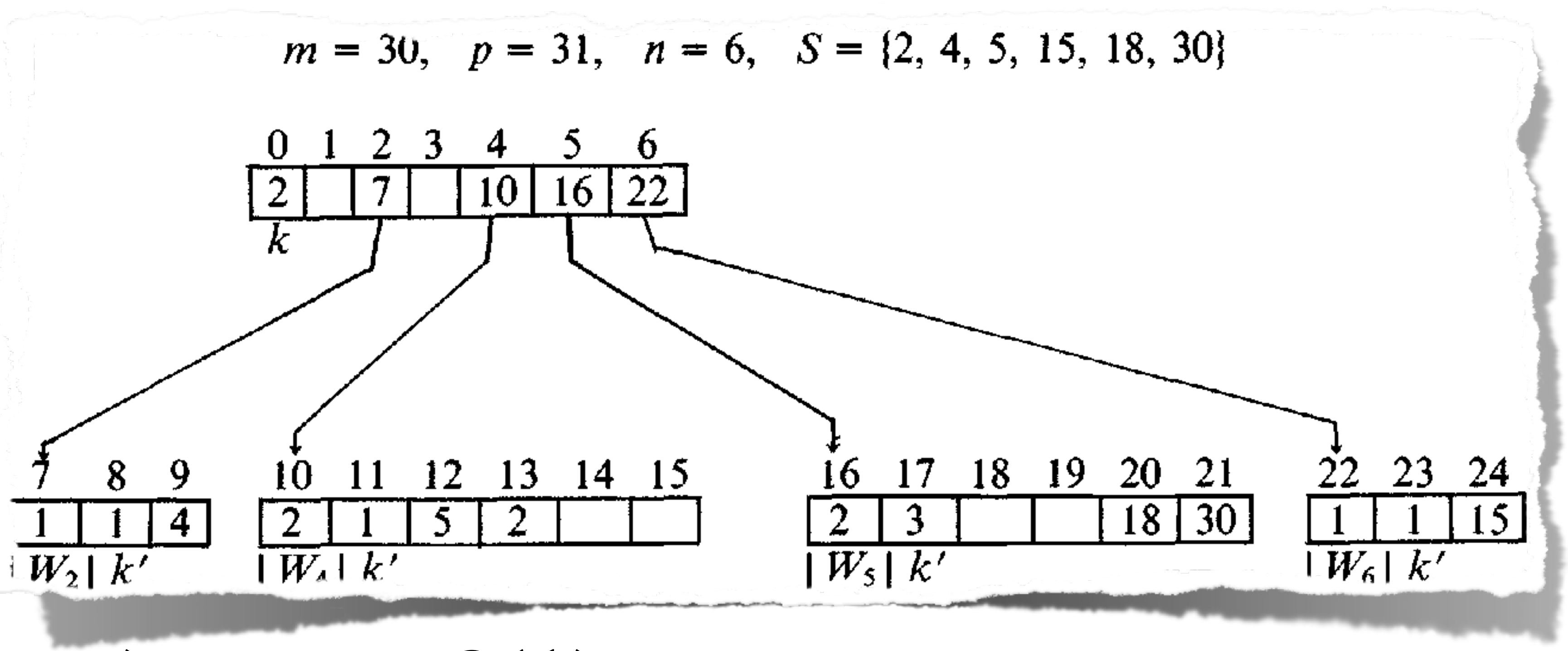
FKS Perfect Hashing

(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, Meyer auf der Heide, Rohnert, Tarjan, 1984]**

Optimal Dynamic Perfect Hashing

(Upper Bound, STOC 2022)

On the Optimal Time/Space Tradeoff for Hash Tables

Michael A. Bender
Stony Brook University

Martín Farach-Colton
Rutgers University

John Kuszmaul
Yale University

William Kuszmaul
MIT

Mingmou Liu
NTU

Abstract

For nearly six decades, the central open question in the study of hash tables has been to determine the optimal achievable tradeoff curve between time and space. State-of-the-art hash tables offer the following guarantee: If keys/values are $\Theta(\log n)$ bits each, then it is possible to achieve constant-time insertions/deletions/queries while wasting only $O(\log\log n)$ bits of space per key when compared to the information-theoretic optimum. Even prior to this bound being achieved, the target of $O(\log\log n)$ wasted bits per key was known to be a natural end goal, and was proven to be optimal for a number of closely related problems (e.g., stable hashing, dynamic retrieval, and dynamically-resized filters).

This paper shows that $O(\log\log n)$ wasted bits per key is not the end of the line for hashing. In fact, for any $k \in [\log^* n]$, it is possible to achieve $O(k)$-time insertions/deletions, $O(1)$-time queries, and

$$O(\log^{(k)} n) = O\left(\underbrace{\log\log\cdots\log}_{k} n\right)$$

wasted bits per key (all with high probability in n). This means that, each time we increase insertion/deletion time by an *additive constant*, we reduce the wasted bits per key *exponentially*. We further show that this tradeoff curve is the best achievable by any of a large class of hash tables, including any hash table designed using the current framework for making constant-time hash tables succinct.

Optimal Dynamic Perfect Hashing

(Lower Bound, FOCS 2023)

Tight Cell-Probe Lower Bounds for Dynamic Succinct Dictionaries

Tianxiao Li * Jingxun Liang † Huacheng Yu ‡ Renfei Zhou §

Abstract

A dictionary data structure maintains a set of at most n keys from the universe $[U]$ under key insertions and deletions, such that given a query $x \in [U]$, it returns if x is in the set. Some variants also store values associated to the keys such that given a query x, the value associated to x is returned when x is in the set.

This fundamental data structure problem has been studied for six decades since the introduction of hash tables in 1953. A hash table occupies $O(n \log U)$ bits of space with constant time per operation in expectation. There has been a vast literature on improving its time and space usage. The state-of-the-art dictionary by Bender, Farach-Colton, Kuszmaul, Kuszmaul and Liu [BFCK+22] has space consumption close to the *information-theoretic optimum*, using a total of

$$\log \binom{U}{n} + O(n \log^{(k)} n)$$

bits, while supporting all operations in $O(k)$ time, for any parameter $k \le \log^* n$. The term $O(\log^{(k)} n) = O(\underbrace{\log \cdots \log}_{k} n)$ is referred to as the *wasted bits per key*.

In this paper, we prove a matching *cell-probe* lower bound: For $U = n^{1+\Theta(1)}$, any dictionary with $O(\log^{(k)} n)$ wasted bits per key must have expected operational time $\Omega(k)$, in the cell-probe model with word-size $w = \Theta(\log U)$. Furthermore, if a dictionary stores values of $\Theta(\log U)$ bits, we show that *regardless of the query time*, it must have $\Omega(k)$ expected update time. It is worth noting that this is the first cell-probe lower bound on the trade-off between space and update time for general data structures.

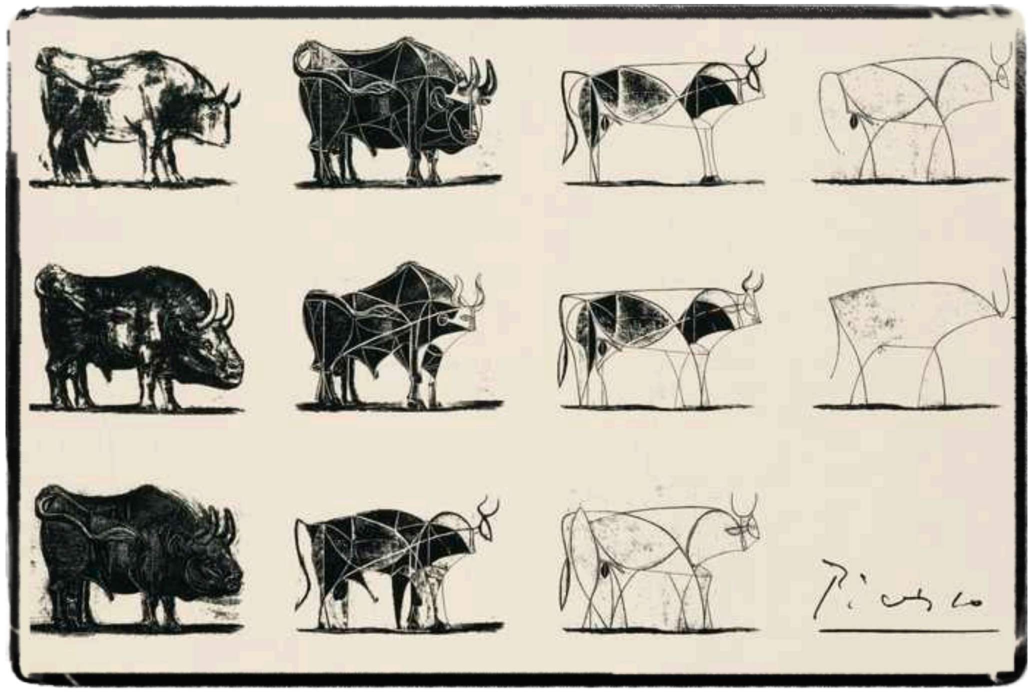
Data Structure for Set

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: $\log \binom{N}{n} = O(n \log N)$ bits (when $N \gg n$)
- **Sketch**: lossy representation of S using < entropy space



Approximate Set

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

Query: an item $x \in U$

Answer whether $x \in S$ with bounded error.

- uniform hash function $h : U \to [m]$ (m to be fixed)

Data Structure: bit array $A \in \{0,1\}^m$

A is initialized to all 0's;

for each $x_i \in S$: set $A[h(x_i)] = 1$;

Query x: answer "yes" iff $A[h(x)] = 1$

- $x \in S$: always correct
- $x \notin S$: false positive $\Pr\left[A[h(x)] = 1\right] = 1 - (1 - 1/m)^n \approx 1 - \mathrm{e}^{-n/m}$

Bloom Filters (Bloom 1970)

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

Query: an item $x \in U$

Answer whether $x \in S$ with bounded error.

- uniform & independent hash function $h_1, \ldots, h_k : U \to [m]$

(k and m to be fixed)

Data Structure: bit array $A \in \{0,1\}^m$

A is initialized to all 0's;

for each $x_i \in S$: set $A[h_j(x_i)] = 1$ for all $1 \leq j \leq k$;

Query x: "yes" iff $A[h_j(x)] = 1$ for all $1 \leq j \leq k$

Bloom Filters (Bloom 1970)

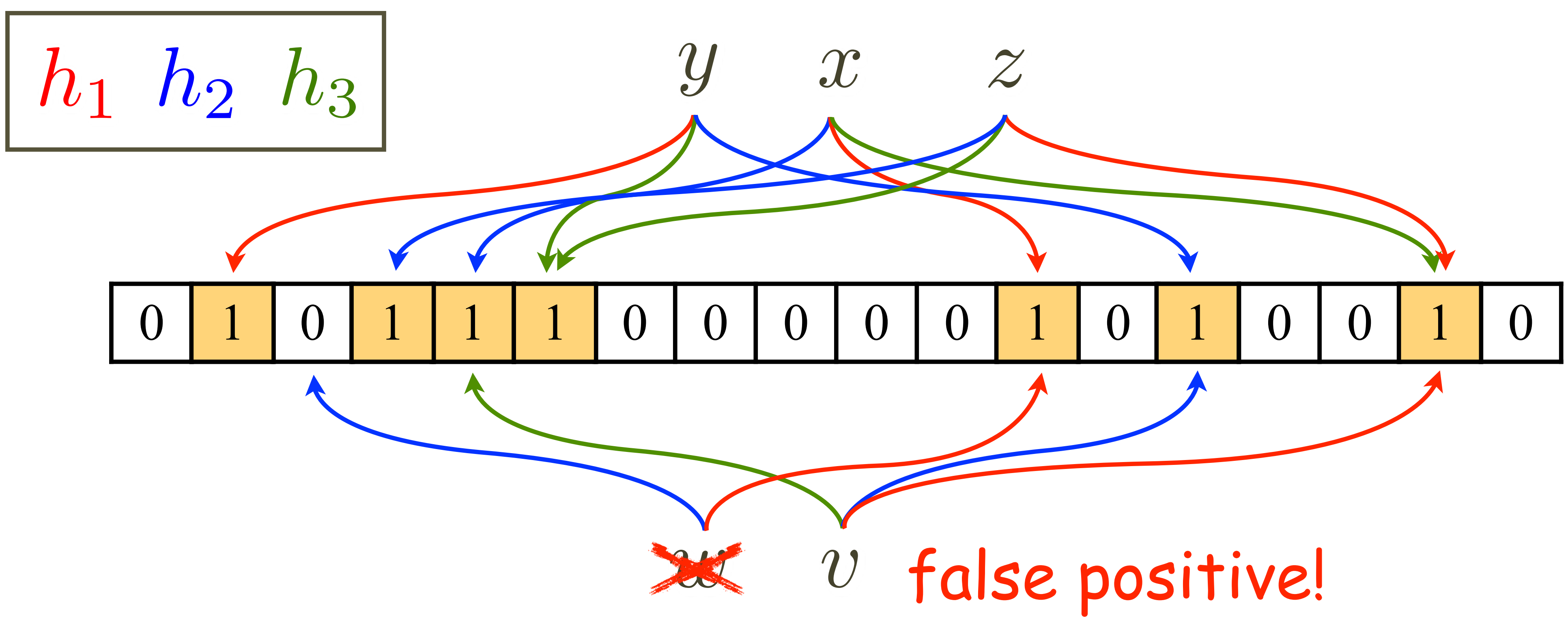
- uniform & independent hash function $h_1, \ldots, h_k : U \to [m]$

Data Structure: bit array $A \in \{0,1\}^m$

A is initialized to all 0's;

for each $x_i \in S$: set $A[h_j(x_i)] = 1$ for all $1 \leq j \leq k$;

Query x: "yes" iff $A[h_j(x)] = 1$ for all $1 \leq j \leq k$



Data: set $S \subseteq U$ of size n **Query**: $x \in U$

- uniform & independent hash function $h_1, \ldots, h_k : U \to [m]$

Data Structure: bit array $A \in \{0,1\}^m$

A is initialized to all 0's;

for each $x_i \in S$: set $A[h_j(x_i)] = 1$ for all $1 \le j \le k$;

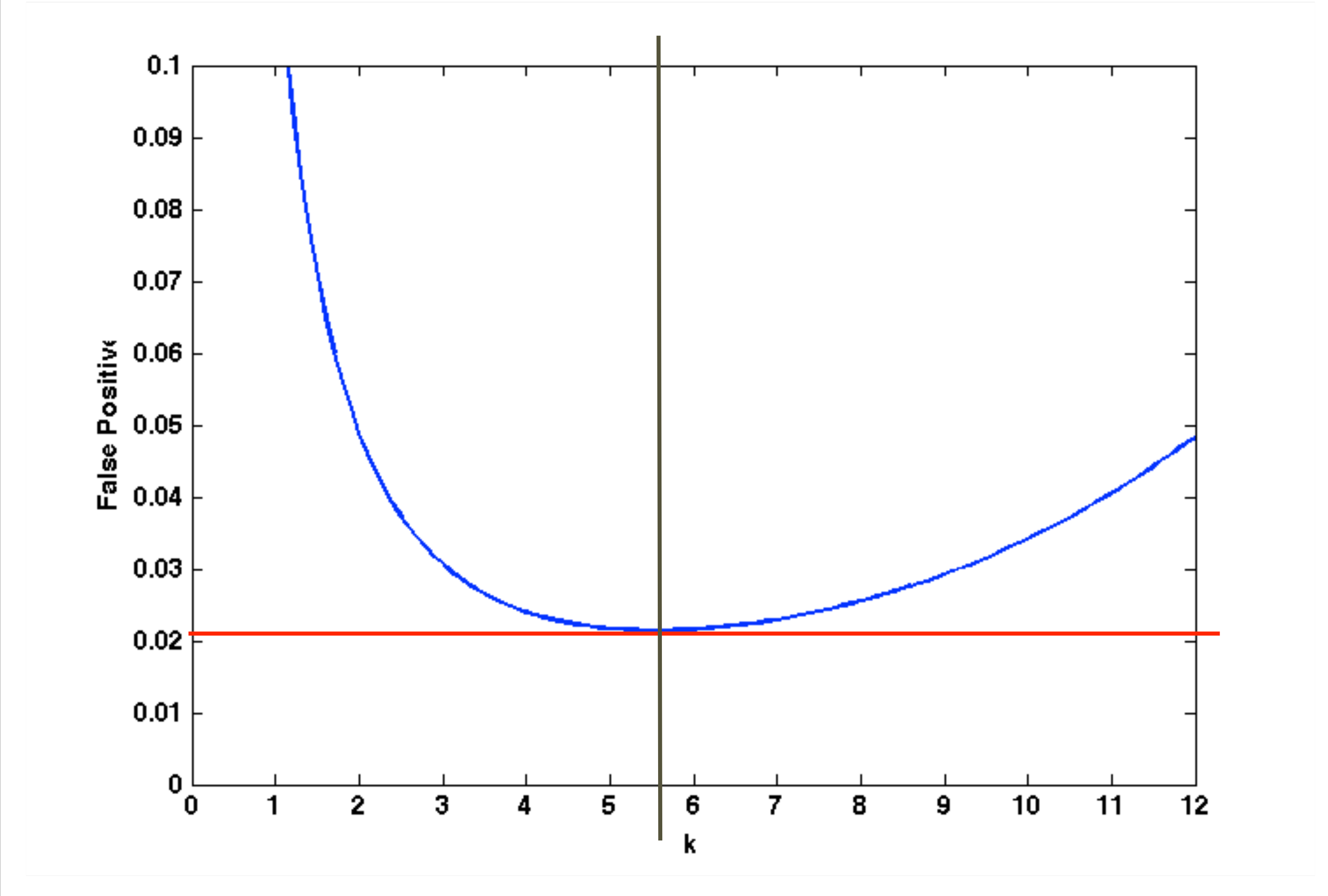
Query x: "yes" iff $A[h_j(x)] = 1$ for all $1 \le j \le k$

- $x \in S$: always correct
- $x \notin S$: false positive

$$\Pr\left[\forall 1 \le j \le k : \; A[h_j(x)] = 1\right]$$

$$= \left(\Pr\left[A[h_j(x)] = 1\right]\right)^k = \left(1 - \Pr\left[A[h_j(x)] = 0\right]\right)^k$$

$$\le (1 - (1 - 1/m)^{kn})^k \; \approx \left(1 - \mathrm{e}^{-kn/m}\right)^k$$



y: $x \in U$

$\ldots, h_k : U \to [m]$

all $1 \le j \le k$;

$\le j \le k$

- $x \notin S$: false positive

$$\Pr\left[\forall 1 \le j \le k: \ A[h_j(x)] = 1\right]$$

choose $k = c \ln 2$

$m = cn$

$$= \left(\Pr\left[A[h_j(x)] = 1\right]\right)^k = \left(1 - \Pr\left[A[h_j(x)] = 0\right]\right)^k$$

$$\le (1 - (1 - 1/m)^{kn})^k \approx \left(1 - \mathrm{e}^{-kn/m}\right)^k = 2^{-c \ln 2} \le (0.6185)^c$$

Bloom Filters (Bloom 1970)

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

Query: an item $x \in U$

Answer whether $x \in S$ with bounded error.

- uniform & independent hash function $h_1, \ldots, h_k : U \to [m]$

Data Structure: bit array $A \in \{0,1\}^m$

A is initialized to all 0's;

for each $x_i \in S$: set $A[h(x_i)] = 1$;

Query x: answer "yes" iff $A[h(x)] = 1$

- choose $k = c \ln 2$ and $m = cn$
  - **space cost:** $m = cn$ bits, **time cost:** $k = c \ln 2$
  - **false positive** $\leq (0.6185)^c$

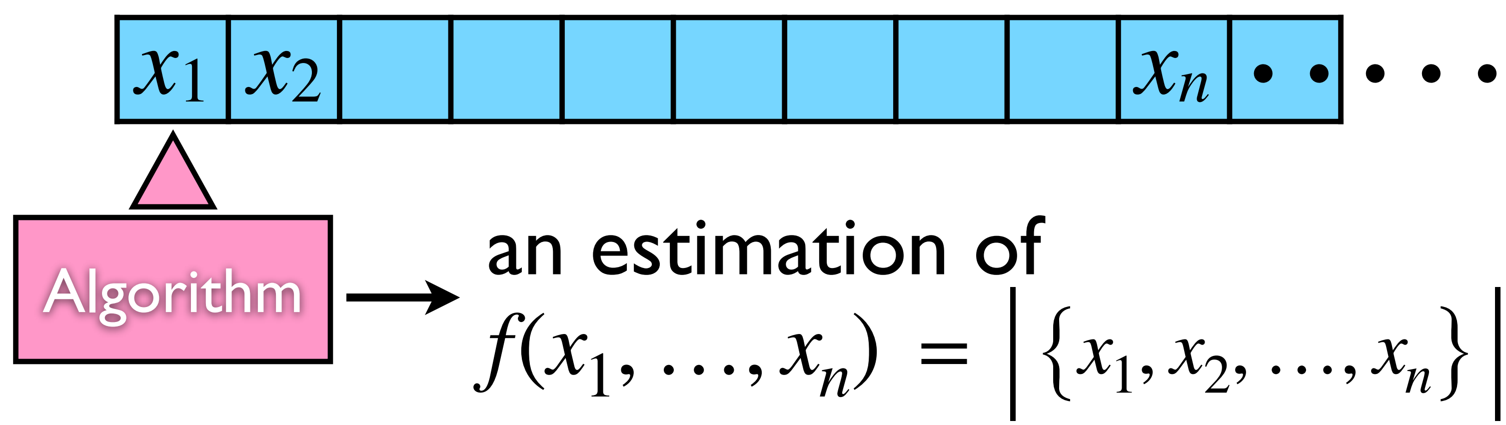
Distinct Elements
(*Frequency Moments*)

Count Distinct Elements

Input: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- **Data stream model**: input data item comes one at a time



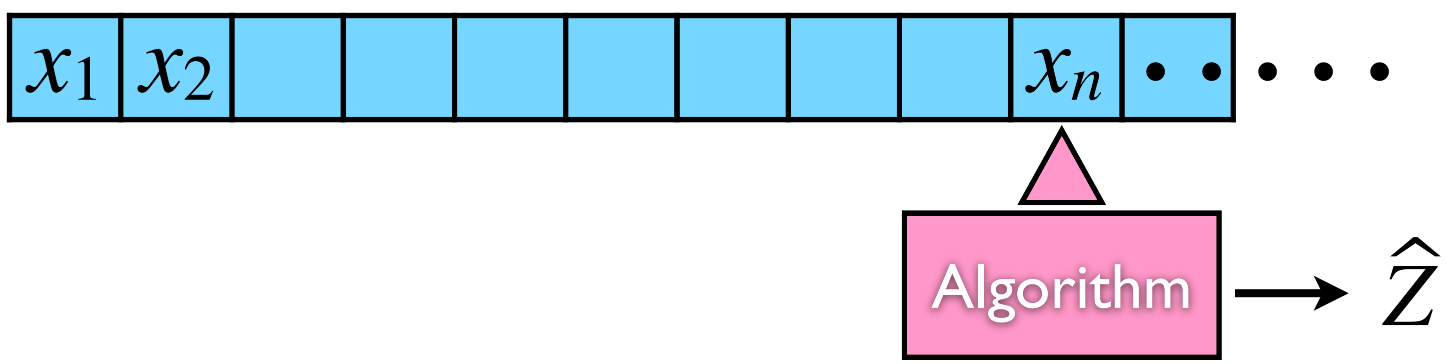
- Naïve algorithm: store all distinct data items using $\Omega(z \log N)$ bits
- **Sketch**: (lossy) representation of data using space $\ll z$
- **Lower bound** (Alon-Matias-Szegedy): any deterministic (exact or approx.) algorithm must use $\Omega(N)$ bits of space in the worst case

Count Distinct Elements

Input: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- **Data stream model**: input data item comes one at a time



- **(ϵ, δ)-estimator**: randomized variable $\widehat{Z}$

$$\Pr\left[(1-\epsilon)z \le \widehat{Z} \le (1+\epsilon)z \right] \ge 1-\delta$$

Using only memory equivalent to 5 lines of printed text, you can estimate with a typical accuracy of 5% and in a single pass the total vocabulary of Shakespeare.

——Durand and Flajolet 2003

Input: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Output: an estimation of $z = \left| \left\{ x_1, x_2, \ldots, x_n \right\} \right|$

Simple Uniform Hash Assumption (SUHA):

A uniform function is available, whose preprocessing, representation and evaluation are considered to be easy.

- (*idealized*) uniform hash function $h : U \to [0,1]$
  - $x_i = x_j \longrightarrow$ the same hash value $h(x_i) = h(x_j) \in_r [0,1]$
- $\left\{ h(x_1), \ldots, h(x_n) \right\}$: $z \times$ uniform and independent values in $[0,1]$
  - partition $[0,1]$ into $z + 1$ subintervals (with *identically distributed* lengths)

$$\mathbb{E}\left[\min_{1 \le i \le n} h(x_i) \right] = \mathbb{E}[\text{length of a subinterval}] = \frac{1}{z+1} \quad \text{(by symmetry)}$$

- estimator: $\widehat{Z} = \dfrac{1}{\min_i h(x_i)} - 1$?   Variance is too large!

Markov's Inequality

Markov's Inequality

For *nonnegative* random variable X, for any $t > 0$,

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

Corollary

For random variable X and *nonnegative-valued* function f, for any $t > 0$,

$$\Pr[f(X) \geq t] \leq \frac{\mathbb{E}[f(X)]}{t}$$

Chebyshev's Inequality

Chebyshev's Inequality

For random variable X, for any $t > 0$,

$$\Pr\Big[\, |X - \mathbb{E}[X]| \geq t \,\Big] \leq \frac{\mathbf{Var}[X]}{t^2}$$

- Variance:

$$\mathbf{Var}[X] = \mathbb{E}[(X - \mathbb{E}[X])^2] = \mathbb{E}[X^2] - (\mathbb{E}[X])^2$$

Apply Markov's inequality to $Y = (X - \mathbb{E}[X])^2$:

$$\Pr\Big[\, |X - \mathbb{E}[X]| \geq t \Big] = \Pr[Y \geq t^2] \leq \frac{\mathbb{E}\left[Y\right]}{t^2} \leq \frac{\mathbf{Var}[X]}{t^2}$$

Input: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- (*idealized*) uniform hash function $h : U \to [0,1]$

Min Sketch:

let $Y = \min_{1 \le i \le n} h(x_i)$;

return $\widehat{Z} = \frac{1}{Y} - 1$;

- By symmetry:

$$\mathbb{E}[Y] = \frac{1}{n+1}$$

- Goal:

$$\Pr\left[\boxed{\widehat{Z} < (1-\epsilon)z \text{ or } \widehat{Z} > (1+\epsilon)z}\right] \le \delta$$

assuming $\epsilon \le 1/2$

$$\left| Y - \mathbb{E}[Y] \right| > \frac{\epsilon/2}{z+1} \iff \left| Y - \frac{1}{z+1} \right| > \frac{\epsilon/2}{z+1}$$

Input: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- (*idealized*) uniform hash function $h : U \to [0,1]$

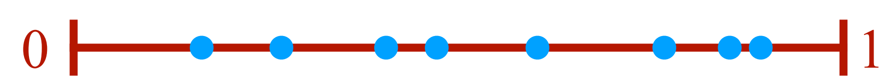
Min Sketch:

let $Y = \min_{1 \le i \le n} h(x_i)$;

return $\widehat{Z} = \frac{1}{Y} - 1$;

- Uniform independent hash values:

$$H_1, \ldots, H_z \in [0,1]$$



- $Y = \min_{1 \le i \le z} H_i$

geometry probability: $\Pr[Y > y] = (1-y)^z$ ⟹ pdf: $p(y) = z(1-y)^{z-1}$

$$\mathbb{E}[Y^2] = \int_0^1 y^2 p(y)\,\mathrm{d}y = \int_0^1 y^2 z(1-y)^{z-1}\,\mathrm{d}y = \frac{2}{(z+1)(z+2)}$$

$$\mathbf{Var}[Y] = \mathbb{E}[Y^2] - \mathbb{E}[Y]^2 = \frac{z}{(z+1)^2(z+2)} \le \frac{1}{(z+1)^2}$$

Input: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- (*idealized*) uniform hash function $h : U \to [0,1]$

Min Sketch:

let $Y = \min_{1 \le i \le n} h(x_i)$;

return $\widehat{Z} = \frac{1}{Y} - 1$;

- By symmetry:

$$\mathbb{E}[Y] = \frac{1}{z+1}$$

- Goal:

$$\Pr\left[\boxed{\widehat{Z} < (1-\epsilon)z \text{ or } \widehat{Z} > (1+\epsilon)z}\right] \le \delta$$

assuming $\epsilon \le 1/2$

$$\mathbf{Var}[Y] \le \frac{1}{(z+1)^2} \xrightarrow{\text{(Chebyshev)}} \Pr\left[\left|Y - \mathbb{E}[Y]\right| > \frac{\epsilon/2}{z+1}\right] \le \frac{4}{\epsilon^2}$$

The Mean Trick (for Variance Reduction)

- **Variance** and **covariance**:

$$\mathbf{Var}[X] = \mathbb{E}[(X - \mathbb{E}[X])^2] = \mathbb{E}[X^2] - (\mathbb{E}[X])^2$$

$$\mathbf{Cov}(X, Y) = \mathbb{E}\left[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])\right]$$

- Useful properties:

$$\mathbf{Var}[X + a] = \mathbf{Var}[X]$$

$$\mathbf{Var}[aX] = a^2\mathbf{Var}[X]$$

$$\mathbf{Var}\left[\sum_i X_i\right] = \sum_i \mathbf{Var}[X_i] + \sum_{i \neq j} \mathbf{Cov}(X_i, X_j)$$

- For pairwise independent identically distributed X_i's:

$$\mathbf{Var}\left[\frac{1}{k}\sum_{i=1}^{k} X_i\right] = \frac{1}{k^2}\sum_{i=1}^{k} \mathbf{Var}[X_i] = \frac{1}{k}\mathbf{Var}[X_1]$$

Input: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- uniform & independent hash functions $h_1, \ldots, h_k : U \to [0,1]$

Min Sketch:

for each $1 \le j \le k$, let $Y_j = \min_{1 \le i \le n} h_j(x_i)$;

return $\widehat{Z} = \frac{1}{\overline{Y}} - 1$ where $\overline{Y} = \frac{1}{k} \sum_{j=1}^{k} Y_j$;

- For every $1 \le j \le k$:

$$\mathbb{E}\left[Y_j\right] = \frac{1}{z+1} \xrightarrow{\text{linearity of expectation}} \mathbb{E}\left[\overline{Y}\right] = \frac{1}{z+1}$$

$$\mathbf{Var}[Y_j] \le \frac{1}{(z+1)^2} \xrightarrow{\text{independence}} \mathbf{Var}\left[\overline{Y}\right] \le \frac{1}{k(z+1)^2}$$

Input: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- uniform & independent hash functions $h_1, \ldots, h_k : U \to [0,1]$

Min Sketch:

for each $1 \le j \le k$, let $Y_j = \min_{1 \le i \le n} h_j(x_i)$;

return $\widehat{Z} = \frac{1}{\overline{Y}} - 1$ where $\overline{Y} = \frac{1}{k} \sum_{j=1}^{k} Y_j$;

$$\mathbb{E}\left[\overline{Y}\right] = \frac{1}{z+1}$$

$$\mathbf{Var}\left[\overline{Y}\right] \le \frac{1}{k(z+1)^2}$$

- Goal: $\Pr\left[\widehat{Z} < (1-\epsilon)z \text{ or } \widehat{Z} > (1+\epsilon)z\right] \le \delta$

assuming $\epsilon \le 1/2$

$$\Pr\left[\left|\overline{Y} - \mathbb{E}\left[\overline{Y}\right]\right| > \frac{\epsilon/2}{z+1}\right] \le \frac{4}{k\epsilon^2} \le \delta$$

(Chebyshev)

Set $k = \left\lceil \frac{4}{\epsilon^2 \delta} \right\rceil$

Input: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- uniform & independent hash functions $h_1, \ldots, h_k : U \to [0,1]$

Min Sketch: set $k = \lceil 4/(\epsilon^2 \delta) \rceil$

for each $1 \le j \le k$, let $Y_j = \min_{1 \le i \le n} h_j(x_i)$;

return $\widehat{Z} = \frac{1}{\overline{Y}} - 1$ where $\overline{Y} = \frac{1}{k} \sum_{j=1}^{k} Y_j$;

$$\Pr\left[(1-\epsilon)z \le \widehat{Z} \le (1+\epsilon)z \right] \ge 1 - \delta$$

- **Space cost**: $k = O\left(\frac{1}{\epsilon^2 \delta}\right)$ ***real numbers*** in $[0,1]$
- Storing k ***idealized*** hash functions.

Universal Hashing

Universal Hash Family (Carter and Wegman 1979):

A family $\mathscr{H}$ of hash functions in $U \to [m]$ is **k-universal** if for any distinct $x_1, \ldots, x_k \in U$,

$$\Pr_{h \in \mathscr{H}} \left[h(x_1) = \cdots = h(x_k) \right] \leq \frac{1}{m^{k-1}}.$$

Moreover, $\mathscr{H}$ is **strongly k-universal** (k-wise independent) if for any distinct $x_1, \ldots, x_k \in U$ and any $y_1, \ldots, y_k \in [m]$,

$$\Pr_{h \in \mathscr{H}} \left[\bigwedge_{i=1}^{k} h(x_i) = y_i \right] = \frac{1}{m^k}.$$

k-Universal Hash Family

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

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

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

- Degree-k polynomial in finite field with random coefficients
- Hashing between binary fields: $GF(2^w) \to GF(2^l)$

$$h_{a,b}(x) = \texttt{(a*x+b)>>(w-l)}$$

Flajolet-Martin Algorithm

Input: a sequence $x_1, x_2, \ldots, x_n \in [N] \subseteq [2^w]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- 2-wise independent hash function $h : [2^w] \to [2^w]$
- For $y \in [2^w]$, let $\mathrm{zeros}(y) = \max\{i : 2^i \mid y\}$ denote # of trailing 0's

Flajolet-Martin Algorithm:

let $R = \max_{1 \le i \le n} \mathrm{zeros}(h(x_i))$;

return $\widehat{Z} = 2^R$;

$$\Pr\left[\widehat{Z} < \frac{z}{C} \text{ or } \widehat{Z} > C \cdot z\right] \le \frac{3}{C}$$

Input: a sequence $x_1, x_2, \ldots, x_n \in [N] \subseteq [2^w]$

Output: an estimation of $z = \left| \left\{ x_1, x_2, \ldots, x_n \right\} \right|$

- 2-wise independent hash function $h : [2^w] \to [2^w]$
- For $y \in [2^w]$, let $\text{zeros}(y) = \max\{i : 2^i \mid y\}$ denote # of trailing 0's

Flajolet-Martin Algorithm:

let $R = \max_{1 \le i \le n} \text{zeros}(h(x_i))$;

return $\widehat{Z} = 2^R$;

Let

$$Y_r = \sum_{x \in \{x_1, \ldots, x_n\}} I\left[\text{zeros}\left(h(x)\right) \ge r\right]$$

(linearity of expectation)

$$\mathbb{E}[Y_r] = \sum_{x \in \{x_1, \ldots, x_n\}} \Pr\left[\text{zeros}\left(h(x)\right) \ge r\right] = z2^{-r}$$

(pairwise independence)

$$\mathbf{Var}[Y_r] = \sum_{x \in \{x_1, \ldots, x_n\}} \mathbf{Var}\left[I[\text{zeros}\left(h(x)\right) \ge r]\right] = z2^{-r}(1 - 2^{-r}) \le z2^{-r}$$

Pairwise Independent Trials

Proposition:

If X is a sum of pairwise independent random variables taking values in $\{0,1\}$, then $\mathbf{Var}[X] \leq \mathbb{E}[X]$.

$$\mathbf{Var}[X] = \sum_i \mathbf{Var}[X_i] = \sum_i (\mathbb{E}[X_i^2] - \mathbb{E}[X_i]^2) = \sum_i (\mathbb{E}[X_i] - \mathbb{E}[X_i]^2)$$

$$= \mathbb{E}[X] - \sum_i \mathbb{E}[X_i]^2 \leq \mathbb{E}[X]$$

Corollary (Chebyshev's Inequality):

If X is a sum of pairwise independent random variables taking values in $\{0,1\}$, for any $t > 0$,

$$\Pr\left[\,|X - \mathbb{E}[X]| \geq t\,\right] \leq \frac{\mathbb{E}[X]}{t^2}$$

Input: a sequence $x_1, x_2, \ldots, x_n \in [N] \subseteq [2^w]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- 2-wise independent hash function $h : [2^w] \to [2^w]$
- For $y \in [2^w]$, let $\mathrm{zeros}(y) = \max\{i : 2^i \,|\, y\}$ denote # of trailing 0's

Flajolet-Martin Algorithm:

let $R = \max_{1 \le i \le n} \mathrm{zeros}(h(x_i))$;

return $\widehat{Z} = 2^R$;

Let

$$Y_r = \sum_{x \in \{x_1, \ldots, x_n\}} I\left[\mathrm{zeros}\left(h(x)\right) \ge r\right]$$

(linearity of expectation)

$$\mathbb{E}[Y_r] = \sum_{x \in \{x_1, \ldots, x_n\}} \Pr\left[\mathrm{zeros}\left(h(x)\right) \ge r\right] = z2^{-r}$$

(pairwise independence) $\quad \mathbf{Var}[Y_r] \le \mathbb{E}[Y_r] \le z2^{-r}$

Input: a sequence $x_1, x_2, \ldots, x_n \in [N] \subseteq [2^w]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- 2-wise independent hash function $h : [2^w] \to [2^w]$
- For $y \in [2^w]$, let $\mathrm{zeros}(y) = \max\{i : 2^i \mid y\}$ denote # of trailing 0's

Flajolet-Martin Algorithm:

let $R = \max_{1 \le i \le n} \mathrm{zeros}(h(x_i))$;

return $\widehat{Z} = 2^R$;

Let

$$Y_r = \sum_{x \in \{x_1, \ldots, x_n\}} I\left[\mathrm{zeros}\left(h(x)\right) \ge r\right]$$

$$\mathbb{E}[Y_r] = z2^{-r} \qquad \mathbf{Var}[Y_r] \le z2^{-r}$$

(denote $r^* = \lceil \log_2 Cz \rceil$) $\quad \Pr\left[\widehat{Z} > Cz\right] \le \Pr[R \ge r^*]$

(observe $R = \max\{r : Y_r > 0\}$) $\quad \le \Pr[Y_{r^*} > 0] = \Pr[Y_{r^*} \ge 1]$

(Markov's inequality) $\quad \le \mathbb{E}[Y_{r^*}] = z/2^{r^*} \le 1/C$

Input: a sequence $x_1, x_2, \ldots, x_n \in [N] \subseteq [2^w]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- 2-wise independent hash function $h : [2^w] \to [2^w]$
- For $y \in [2^w]$, let $\mathrm{zeros}(y) = \max\{i : 2^i \mid y\}$ denote # of trailing 0's

Flajolet-Martin Algorithm:

let $R = \max_{1 \le i \le n} \mathrm{zeros}(h(x_i))$;

return $\widehat{Z} = 2^R$;

Let

$$Y_r = \sum_{x \in \{x_1, \ldots, x_n\}} I\left[\mathrm{zeros}\left(h(x)\right) \ge r\right]$$

$$\mathbb{E}[Y_r] = z2^{-r} \qquad \mathbf{Var}[Y_r] \le z2^{-r}$$

(denote $r^{**} = \lceil \log_2(z/C) \rceil$) $\quad \Pr\left[\widehat{Z} < z/C\right] \le \Pr[R < r^{**}]$

(observe $R = \max\{r : Y_r > 0\}$) $\quad \le \Pr[Y_{r^{**}} = 0]$

(Chebyshev's inequality) $\quad \le \mathbf{Var}[Y_{r^{**}}]/\mathbb{E}[Y_{r^{**}}]^2 \le 2^{r^{**}}/z$

$\le 2/C$

Input: a sequence $x_1, x_2, \ldots, x_n \in [N] \subseteq [2^w]$

Output: an estimation of $z = \left| \{x_1, x_2, \ldots, x_n\} \right|$

- 2-wise independent hash function $h : [2^w] \to [2^w]$
- For $y \in [2^w]$, let $\mathrm{zeros}(y) = \max\{i : 2^i \mid y\}$ denote # of trailing 0's

Flajolet-Martin Algorithm:

let $R = \max_{1 \le i \le n} \mathrm{zeros}(h(x_i))$;

return $\widehat{Z} = 2^R$;

$$\Pr\left[\widehat{Z} < \frac{z}{C} \text{ or } \widehat{Z} > C \cdot z\right] \le \frac{3}{C}$$

- **Space cost:** $O(\log \log N)$ bits for maintaining R
- $O(\log N)$ bits for storing 2-wise independent hash function

BJKST Algorithm

Input: a sequence $x_1, x_2, \ldots, x_n \in [N]$

Output: an estimation of $z = \left| \left\{ x_1, x_2, \ldots, x_n \right\} \right|$

- 2-wise independent hash function $h : [N] \to [M] = \{1, \ldots, M\}$

BJKST Algorithm:

let $Y_1, \ldots, Y_k$ be the k smallest hash values among

$$\left\{ h(x_1), h(x_2) \ldots, h(x_n) \right\};$$

return $\widehat{Z} = \dfrac{kM}{Y_k}$;

(Bar-Yossef, Jayram, Kumar, Sivakumar and Trevisan, 2002)

Input: a sequence $x_1, x_2, \ldots, x_n \in [N]$

Output: an estimation of $z = \left|\left\{x_1, x_2, \ldots, x_n\right\}\right|$

- 2-wise independent hash function $h : [N] \to [M] = \{1, \ldots, M\}$

BJKST Algorithm:

let $Y_1, \ldots, Y_k$ be the k smallest hash values among
$\left\{ h(x_1), h(x_2) \ldots, h(x_n) \right\}$;

return $\widehat{Z} = \dfrac{kM}{Y_k}$;

- Goal: $\Pr\left[\widehat{Z} < (1-\epsilon)z \text{ or } \widehat{Z} > (1+\epsilon)z\right] \leq \delta$

assuming $\epsilon \leq 1$

$$\left| Y_k - \frac{kM}{z} \right| > \frac{\epsilon}{2} \cdot \frac{kM}{z}$$

- uniform and 2-wise independent $X_1, \ldots, X_n \in [N^3]$
- let $Y_1, \ldots, Y_z$ be these elements in non-decreasing order

Let $V = \sum_{i=1}^{z} I\left[X_i \le \left(1 - \frac{\epsilon}{2}\right)\frac{kM}{z}\right]$ $\quad W = \sum_{i=1}^{z} I\left[X_i \le \left(1 + \frac{\epsilon}{2}\right)\frac{kM}{z}\right]$

$$\mathbb{E}[V] = \left(1 - \frac{\epsilon}{2} \pm o(1)\right)k \qquad \mathbb{E}[W] = \left(1 + \frac{\epsilon}{2} \pm o(1)\right)k$$

$$Y_k < \left(1 - \frac{\epsilon}{2}\right)\frac{k(M+1)}{z} \implies V \ge k \qquad Y_k > \left(1 + \frac{\epsilon}{2}\right)\frac{k(M+1)}{z} \implies W \le k$$

(Chebyshev's inequality for sum of pairwise independent trials)

$$\Pr[V \ge k] \le \frac{8}{k\epsilon^2} \qquad \Pr[W \le k] \le \frac{8}{k\epsilon^2}$$

- Goal: $\Pr\left[\left|Y_k - \frac{kM}{z}\right| > \frac{\epsilon}{2} \cdot \frac{kM}{z}\right] \le \delta$ $\quad$ Set $k = \left\lceil \frac{16}{\epsilon^2\delta} \right\rceil$

Input: a sequence $x_1, x_2, \ldots, x_n \in [N]$

Output: an estimation of $z = \left| \left\{ x_1, x_2, \ldots, x_n \right\} \right|$

- 2-wise independent hash function $h : [N] \to [N^3]$

BJKST Algorithm: Set $k = \lceil 16/(\epsilon^2 \delta) \rceil$

let $Y_1, \ldots, Y_k$ be the k smallest hash values among

$$\left\{ h(x_1), h(x_2) \ldots, h(x_n) \right\};$$

return $\widehat{Z} = \dfrac{kM}{Y_k}$;

$$\Pr\left[(1-\epsilon)z \le \widehat{Z} \le (1+\epsilon)z \right] \ge 1 - \delta$$

- **Space cost:** $O(k \log N) = O(\epsilon^{-2} \log N)$ bits when $\delta = \Omega(1)$

Frequency Moments

- **Data stream:** $x_1, x_2, \ldots, x_n \in U$
- for each $x \in U$, define *frequency* of x as $f_x = |\{i : x_i = x\}|$

k-th *frequency moments*: $F_k = \sum_{x \in U} f_x^k$

- **Space complexity** for (ϵ, δ)-estimation: constant ϵ, δ
  - for $k \leq 2$: $\mathrm{polylog}(N)$ **[Alon-Matias-Szegedy '96]**
  - for $k > 2$: $\tilde{\Theta}(N^{1-2/k})$ **[Indyk-Woodruff '05]**
- **Count distinct elements:** F_0
  - optimal algorithm **[Kane-Nelson-Woodruff '10]**: $O(\epsilon^{-2} + \log N)$ bits

Frequency Estimation

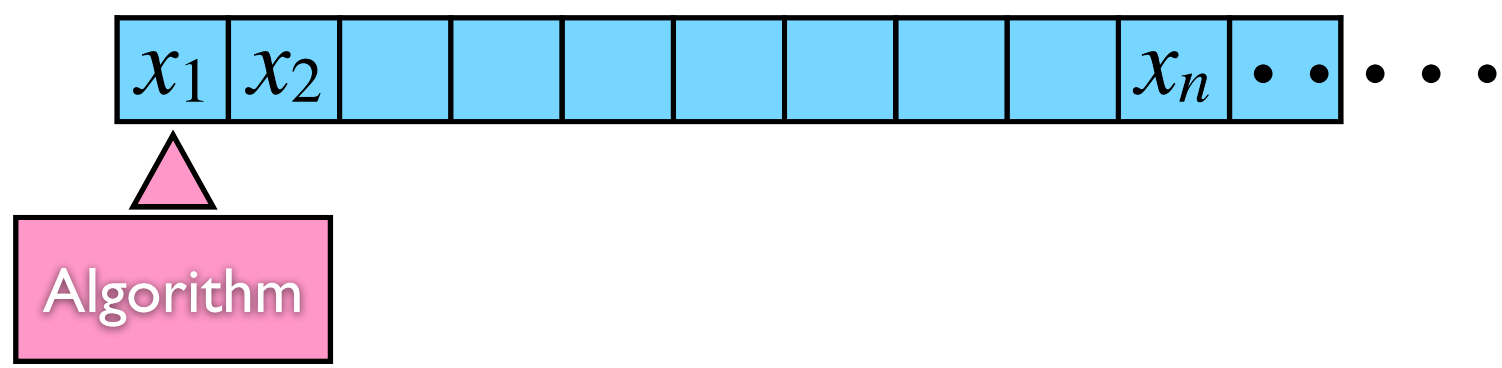
Frequency Estimation

Data: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Query: an item $x \in U$

Estimate the ***frequency*** $f_x = |\{i : x_i = x\}|$ of x.

- **Data stream model**: input data item comes one at a time



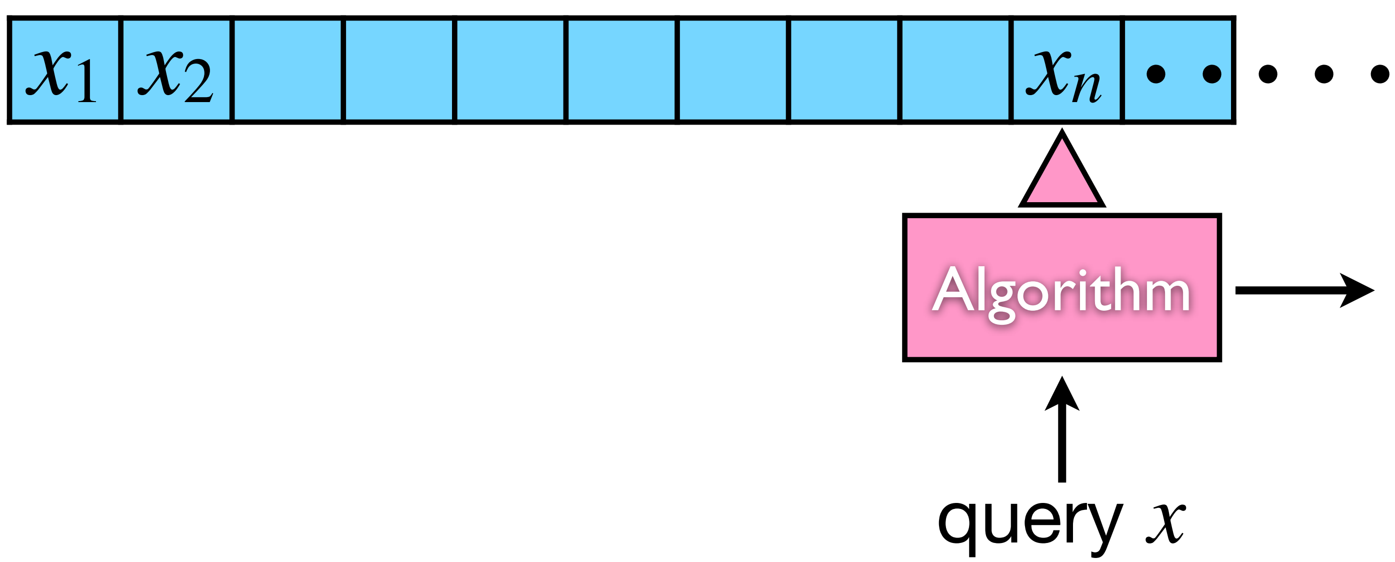
Frequency Estimation

Data: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Query: an item $x \in U$

Estimate the ***frequency*** $f_x = |\{i : x_i = x\}|$ of x.

- **Data stream model**: input data item comes one at a time



$\hat{f}_x$: estimation of f_x

within additive error

$$\Pr\left[\,|\hat{f}_x - f_x| \geq \epsilon n\,\right] \leq \delta$$

- Heavy hitters: items that appears $> \epsilon n$ times

Count-Min Sketch

Data: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Query: an item $x \in U$

Estimate the ***frequency*** $f_x = |\{i : x_i = x\}|$ of x.

- k independent 2-universal hash functions $h_1, \ldots, h_k : [N] \to [m]$

Count-Min Sketch: $\mathrm{CMS}[k][m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{CMS}[j][h_j(x_i)] + +$ for all $1 \leq j \leq k$;

Query x: return $\hat{f}_x = \min\limits_{1 \leq j \leq k} \mathrm{CMS}[j][h_j(x)]$

Observation: $\mathrm{CMS}[j][h_j(x)] \geq f_x$ for all $1 \leq j \leq k$

$$f_x \leq \hat{f}_x \leq \, ?$$

Data: sequence $x_1, \ldots, x_n \in [N]$ **Query**: $x \in [N]$

frequency $f_x = |\{i : x_i = x\}|$ of x

- k independent 2-universal hash functions $h_1, \ldots, h_k : [N] \to [m]$

Count-Min Sketch: $\mathrm{CMS}[k][m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{CMS}[j][h_j(x_i)]++$ for all $1 \le j \le k$;

Query x: return $\hat{f}_x = \min_{1 \le j \le k} \mathrm{CMS}[j][h_j(x)]$

- for any $x \in [N]$ and any $1 \le j \le k$:

$$\mathrm{CMS}[j][h_j(x)] = f_x + \sum_{\substack{y \in \{x_1, \ldots, x_n\} \setminus \{x\} \\ h_j(y) = h_j(x)}} f_y$$

$$\mathbb{E}\Big[\mathrm{CMS}[j][h_j(x)]\Big] = f_x + \sum_{y \in \{x_1, \ldots, x_n\} \setminus \{x\}} f_y \Pr[h_j(y) = h_j(x)]$$

Data: sequence $x_1, \ldots, x_n \in [N]$ **Query**: $x \in [N]$

frequency $f_x = |\{i : x_i = x\}|$ of x

- k independent 2-universal hash functions $h_1, \ldots, h_k : [N] \to [m]$

Count-Min Sketch: $\mathrm{CMS}[k][m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{CMS}[j][h_j(x_i)] + +$ for all $1 \le j \le k$;

Query x: return $\hat{f}_x = \min_{1 \le j \le k} \mathrm{CMS}[j][h_j(x)]$

- for any $x \in [N]$ and any $1 \le j \le k$:

$$\mathbb{E}\left[\mathrm{CMS}[j][h_j(x)]\right] = f_x + \sum_{y \in \{x_1, \ldots, x_n\} \setminus \{x\}} f_y \Pr[h_j(y) = h_j(x)]$$

$$\le f_x + \frac{1}{m} \sum_{y \in \{x_1, \ldots, x_n\} \setminus \{x\}} f_y \le f_x + \frac{1}{m} \sum_{y \in \{x_1, \ldots, x_n\}} f_y = f_x + \frac{n}{m}$$

Data: sequence $x_1, \ldots, x_n \in [N]$ **Query**: $x \in [N]$

frequency $f_x = |\{i : x_i = x\}|$ of x

- k independent 2-universal hash functions $h_1, \ldots, h_k : [N] \to [m]$

Count-Min Sketch: $\mathrm{CMS}[k][m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{CMS}[j][h_j(x_i)] + +$ for all $1 \le j \le k$;

Query x: return $\hat{f}_x = \min_{1 \le j \le k} \mathrm{CMS}[j][h_j(x)]$

$\forall x, \forall j$: $\mathrm{CMS}[j][h_j(x)] \ge f_x$

$$\mathbb{E}\left[\mathrm{CMS}[j][h_j(x)]\right] \le f_x + \frac{n}{m}$$

(Markov's inequality) $\Pr\left[\mathrm{CMS}[j][h_j(x)] - f_x \ge \epsilon n\right] \le \frac{1}{\epsilon m}$

$$\Pr\left[|\hat{f}_x - f_x| \ge \epsilon n\right] = \Pr\left[\forall 1 \le j \le k : \mathrm{CMS}[j][h_j(x)] - f_x \ge \epsilon n\right] \le \left(\frac{1}{\epsilon m}\right)^k$$

Data: a sequence $x_1, x_2, \ldots, x_n \in U = [N]$

Query: an item $x \in U$

Estimate the ***frequency*** $f_x = |\{i : x_i = x\}|$ of x.

- k independent 2-universal hash functions $h_1, \ldots, h_k : [N] \to [m]$

Count-Min Sketch: $\mathrm{CMS}[k][m]$ (initialized to all 0's)

Upon each x_i: $\mathrm{CMS}[j][h_j(x_i)]++$ for all $1 \le j \le k$;

Query x: return $\hat{f}_x = \min_{1 \le j \le k} \mathrm{CMS}[j][h_j(x)]$

$$\Pr\left[|\hat{f}_x - f_x| \ge \epsilon n\right] \le \left(\frac{1}{\epsilon m}\right)^k \le \delta$$

- choose $m = \lceil \mathrm{e}/\epsilon \rceil$ and $k = \lceil \ln(1/\delta) \rceil$
  - **space cost:** $O\left(\frac{1}{\epsilon} \log(1/\delta) \log n\right)$ bits
  - $O\left(\log(1/\delta) \log N\right)$ bits for hash functions
  - **time cost for query:** $O\left(\log(1/\delta)\right)$