Fast Pseudo-Random Fingerprints

Yoram Bachrach, Microsoft Research Cambridge

Ely Porat – Bar Ilan-University

Agenda

- •Massive Data Sets
- •Fingerprinting / Sketching
- •Previous techniques
- •Contribution
  - –Computation time
  - –Fingerprint size
- •Pseudo-Random fingerprints
- •Key components of the analysis
- •Conclusions

Massive Data Sets

- Huge increase in volumes of data
  - Numbers of users
  - Data users produce
- Burst of research of techniques that deal with massive data sets
  - Impossible to store all data
  - Cannot examine the data more than once
    - Or more than few times

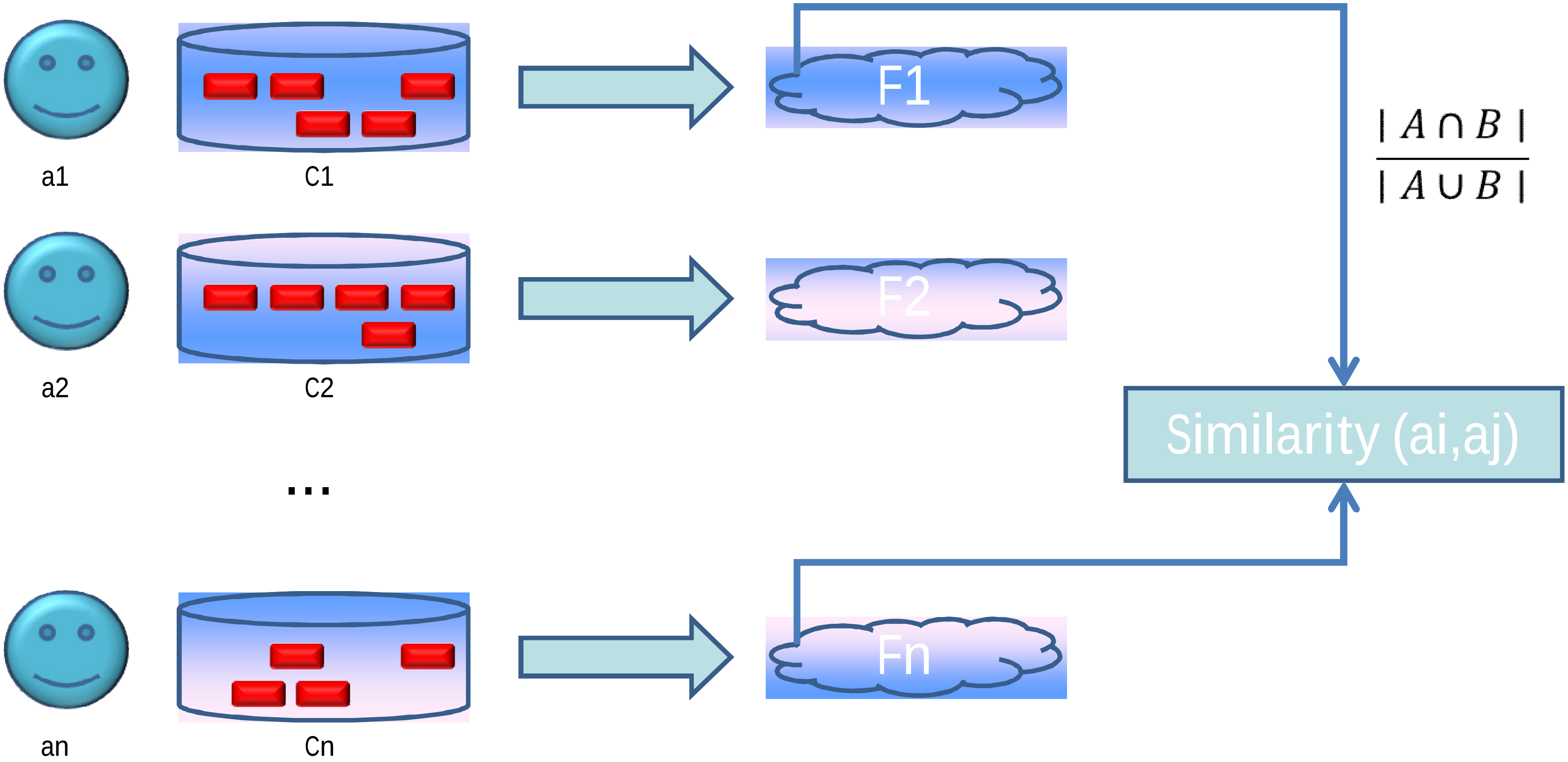
Example: Recommender Systems

- Recommend items to users
  - Content like books, music, videos and web pages
- **Content based** approach
  - Examine content consumed by target user in the past
  - Measure **content similarity** to available items
  - Recommend item with high similarity to past content
- **Collaborative filtering** approach
  - **Many** users **rank** items they consume
  - Find users who have similar tastes to target user
  - Recommend items similar users liked
    - And that the target user never examined
  - E.g. the Netflix challenge

Massive Recommender Systems

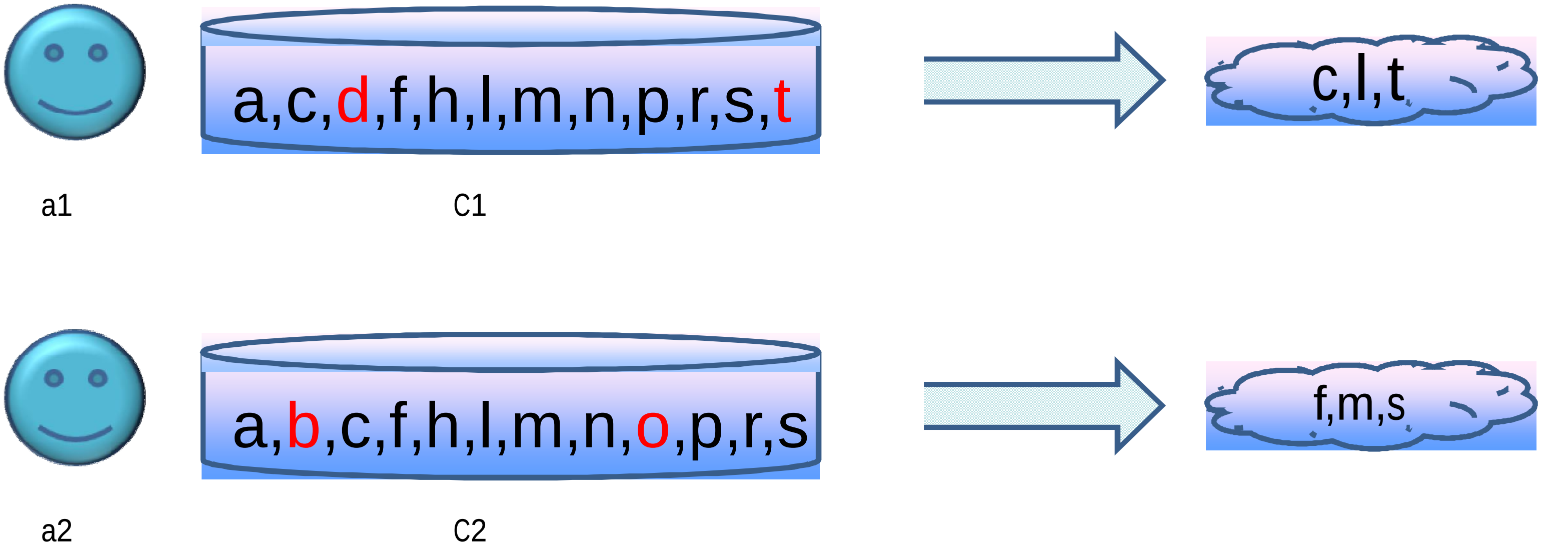
- Consider designing recommender system for web pages
  - Time a user examines a page is an implicit rating
  - Millions of users
  - Each user examines thousands of pages throughout the year
  - Hard to store and process the information
- Solution approach: fingerprints
  - Do not store the full data for each user
  - Keep a fingerprint of the user tastes
    - Allow finding out user similarity

Fingerprint Based Approach



$$\frac{|A \cap B|}{|A \cup B|}$$

Fingerprint Based Approach



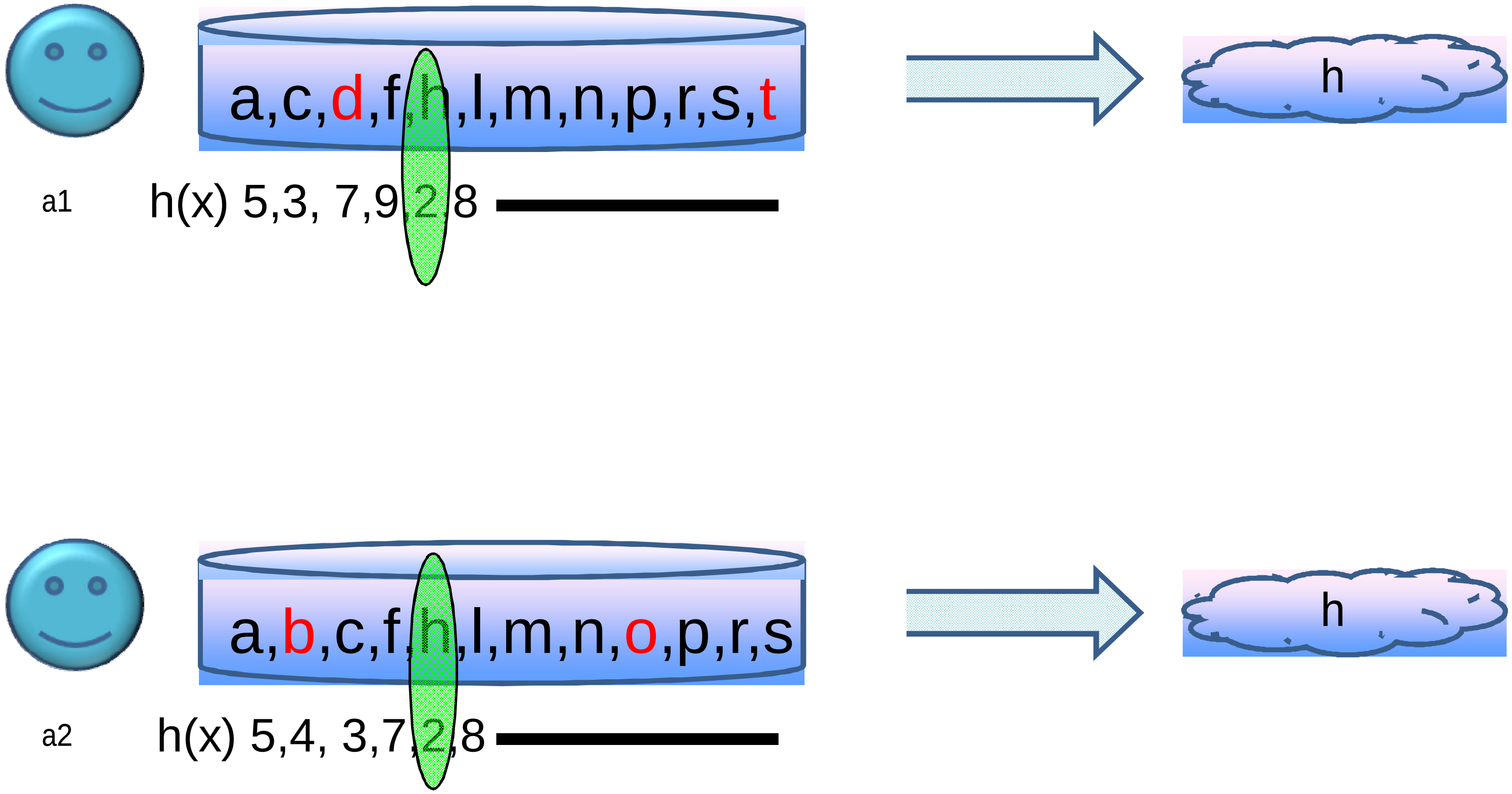
a,c,d,f,h,l,m,n,p,r,s,t → c,l,t

a1 C1

a,b,c,f,h,l,m,n,o,p,r,s → f,m,s

a2 C2

Fingerprint Based Approach



a,c,d,f,h,l,m,n,p,r,s,t

a1

h(x) 5,3, 7,9,2,8

h

a,b,c,f,h,l,m,n,o,p,r,s

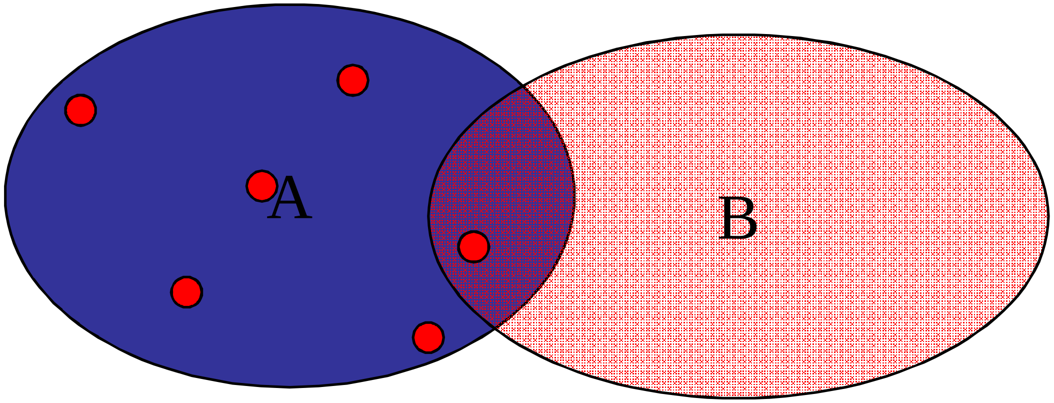
a2

h(x) 5,4, 3,7,2,8

h

Min wise hash function

$$\forall Y \subset U, c \in Y \; Pr_h[argmin_{x \in Y} h(x) = c] = \frac{1}{|Y|}$$



$$\text{argmin}_{x \in A} h(x) = argmin_{x \in B} h(x)$$

$$= argmin_{x \in A \cap B} h(x)$$

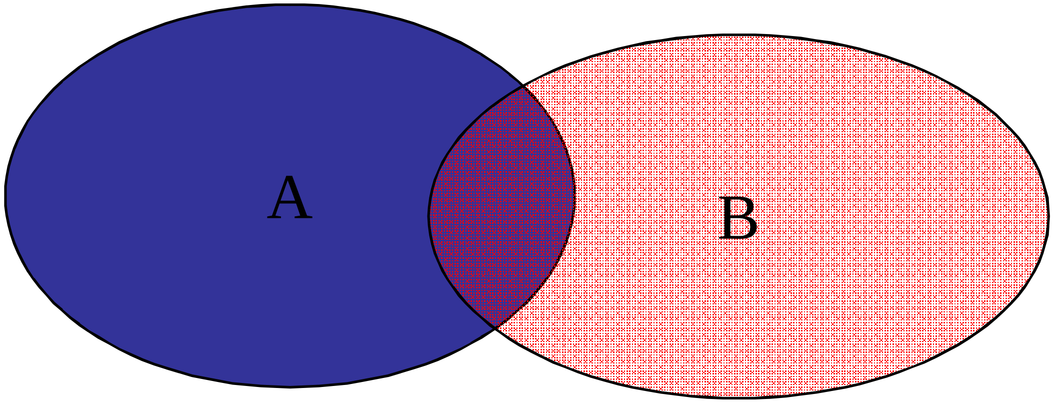
$$= argmin_{x \in A \cup B} h(x)$$

$$\Updownarrow$$

$$\text{argmin}_{x \in A \cup B} h(x) \in A \cap B$$

Min wise hash function

$$\forall Y \subset U, c \in Y \; Pr_h[argmin_{x \in Y} h(x) = c] = \frac{1}{|\,Y\,|}$$

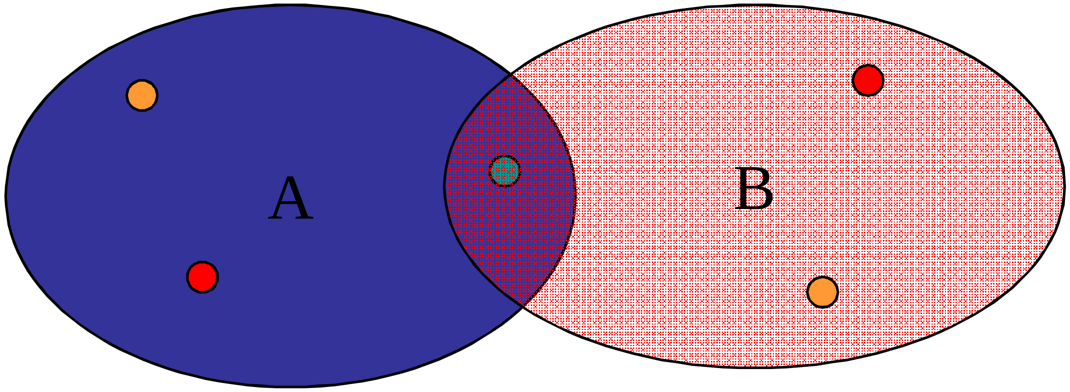


$$\mathrm{Pr}_h[argmin_{x \in A} h(x) = argmin_{x \in B} h(x)] =$$

$$Pr_h[argmin_{x \in A \cup B} h(x) \in A \cap B] = \frac{|\,A \cap B\,|}{|\,A \cup B\,|}$$

Similarity

h_1 h_2 .... $h_{O(\frac{1}{\varepsilon^2} log\frac{1}{\delta})}$ Min wise independent



We get ±ϵ approximation with probability 1-δ

First problem

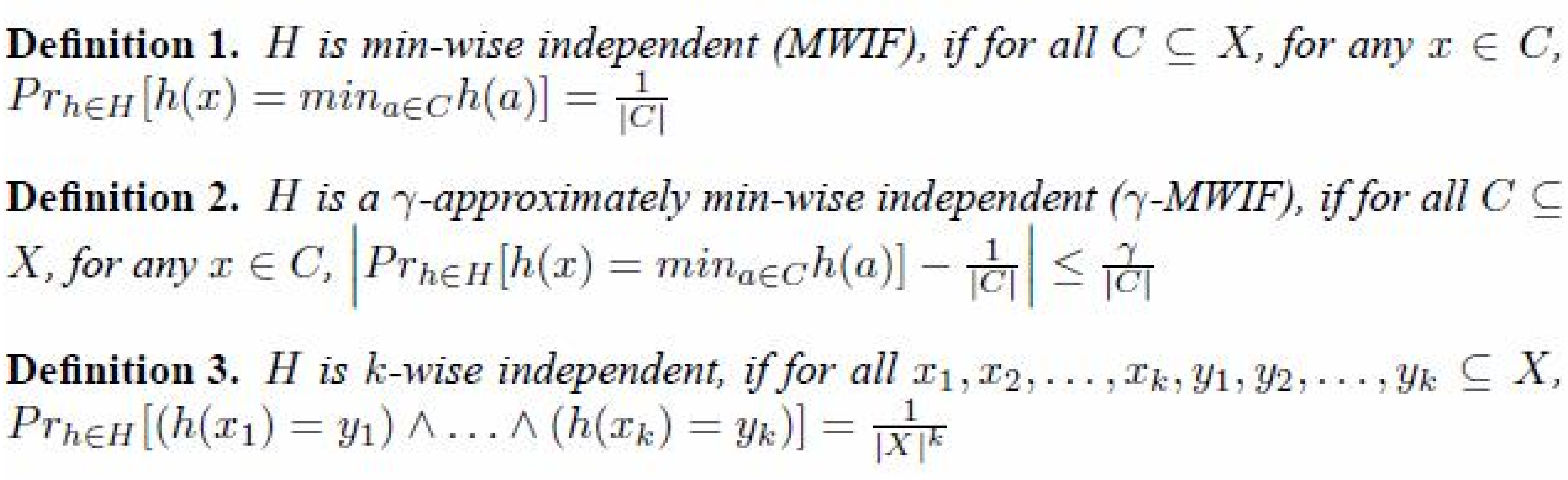
- Min-wise independent require Ω(U) space

- Use almost min wise independent [Indyk99]

- Require O(log1/є) independent function

- O(log1/є log U) bits space
- O(log1/є) time for evaluation.

Hash independence



Definition 1. *H is min-wise independent (MWIF), if for all $C \subseteq X$, for any $x \in C$,* $Pr_{h \in H}[h(x) = min_{a \in C} h(a)] = \frac{1}{|C|}$

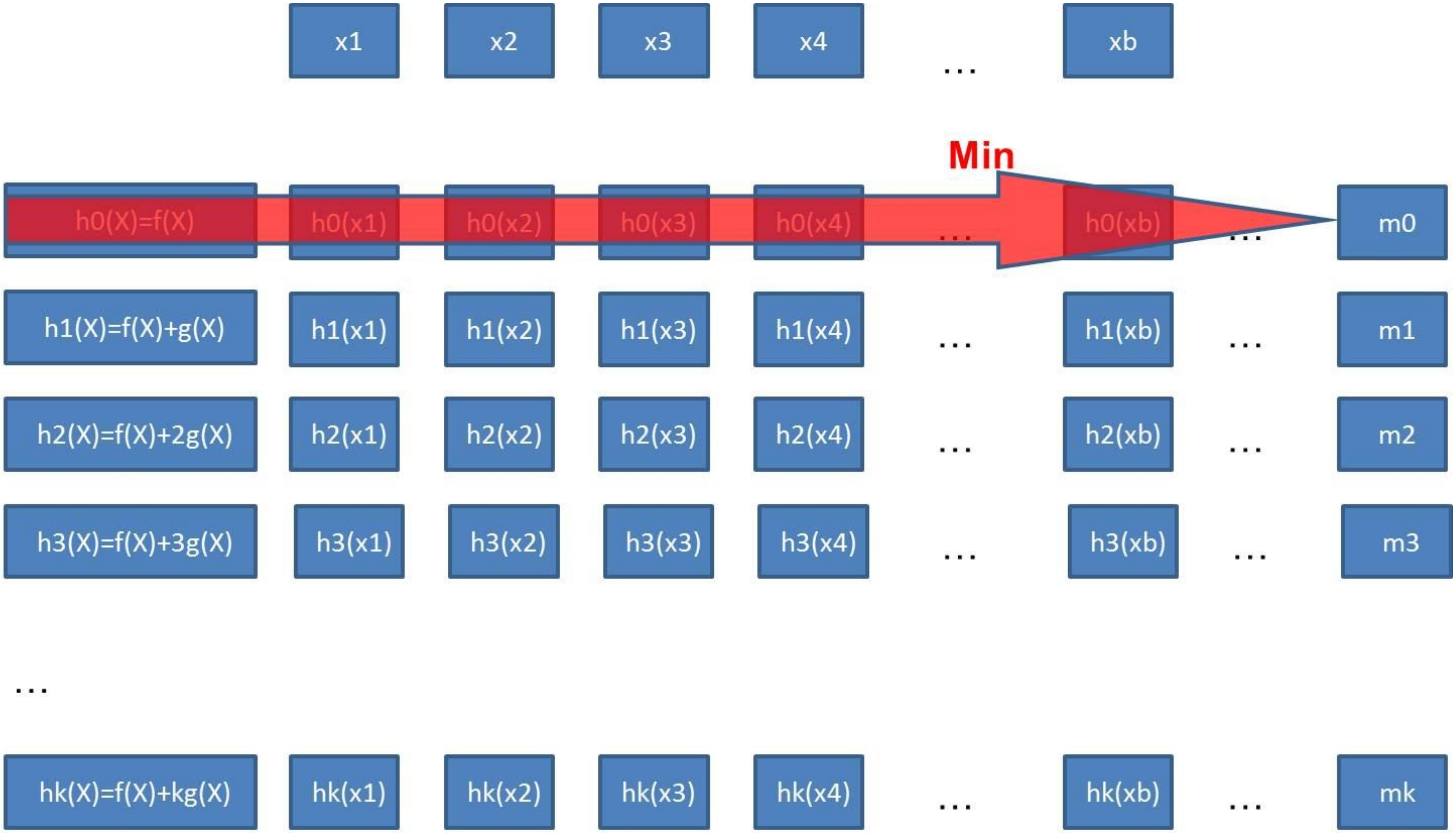
Definition 2. *H is a γ-approximately min-wise independent (γ-MWIF), if for all $C \subseteq X$, for any $x \in C$,* $\left| Pr_{h \in H}[h(x) = min_{a \in C} h(a)] - \frac{1}{|C|} \right| \leq \frac{\gamma}{|C|}$

Definition 3. *H is k-wise independent, if for all $x_1, x_2, \ldots, x_k, y_1, y_2, \ldots, y_k \subseteq X$,* $Pr_{h \in H}[(h(x_1) = y_1) \wedge \ldots \wedge (h(x_k) = y_k)] = \frac{1}{|X|^k}$

Block Structure

	x1	x2	x3	x4	…	xb		
h0(X)=f(X)	h0(x1)	h0(x2)	h0(x3)	h0(x4)	…	h0(xb)	…	m0
h1(X)=f(X)+g(X)	h1(x1)	h1(x2)	h1(x3)	h1(x4)	…	h1(xb)	…	m1
h2(X)=f(X)+2g(X)	h2(x1)	h2(x2)	h2(x3)	h2(x4)	…	h2(xb)	…	m2
h3(X)=f(X)+3g(X)	h3(x1)	h3(x2)	h3(x3)	h3(x4)	…	h3(xb)	…	m3
…								
hk(X)=f(X)+kg(X)	hk(x1)	hk(x2)	hk(x3)	hk(x4)	…	hk(xb)	…	mk

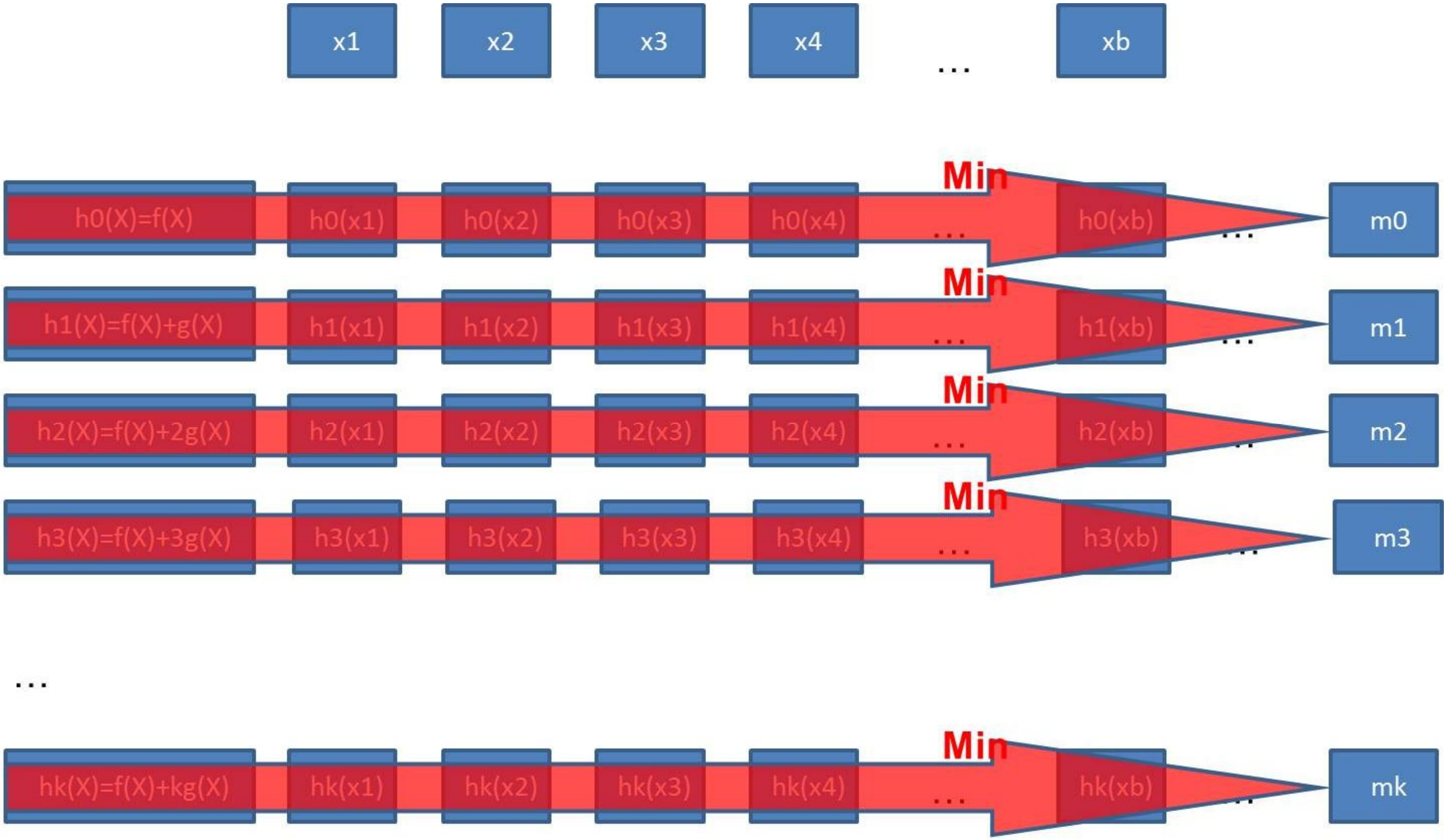
Minimal elements under block hash



Min

	x1	x2	x3	x4	…	xb		
h0(X)=f(X)	h0(x1)	h0(x2)	h0(x3)	h0(x4)	…	h0(xb)	…	m0
h1(X)=f(X)+g(X)	h1(x1)	h1(x2)	h1(x3)	h1(x4)	…	h1(xb)	…	m1
h2(X)=f(X)+2g(X)	h2(x1)	h2(x2)	h2(x3)	h2(x4)	…	h2(xb)	…	m2
h3(X)=f(X)+3g(X)	h3(x1)	h3(x2)	h3(x3)	h3(x4)	…	h3(xb)	…	m3
…								
hk(X)=f(X)+kg(X)	hk(x1)	hk(x2)	hk(x3)	hk(x4)	…	hk(xb)	…	mk

Required Computation



x1 x2 x3 x4 … xb

h0(X)=f(X) h0(x1) h0(x2) h0(x3) h0(x4) … Min h0(xb) … m0

h1(X)=f(X)+g(X) h1(x1) h1(x2) h1(x3) h1(x4) … Min h1(xb) … m1

h2(X)=f(X)+2g(X) h2(x1) h2(x2) h2(x3) h2(x4) … Min h2(xb) … m2

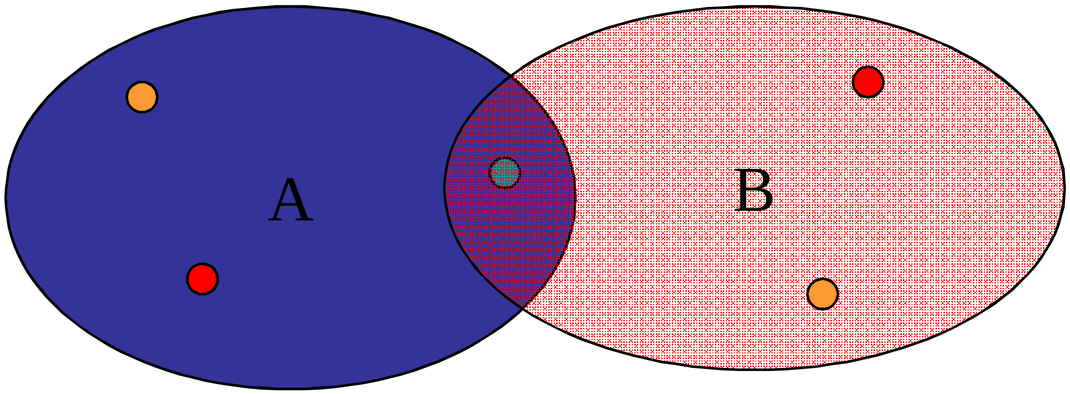
h3(X)=f(X)+3g(X) h3(x1) h3(x2) h3(x3) h3(x4) … Min h3(xb) … m3

…

hk(X)=f(X)+kg(X) hk(x1) hk(x2) hk(x3) hk(x4) … Min hk(xb) … mk

Space analysis

h_1 h_2 .... $h_{O(\frac{1}{\varepsilon^2}log\frac{1}{\delta})}$ Min wise independent functions



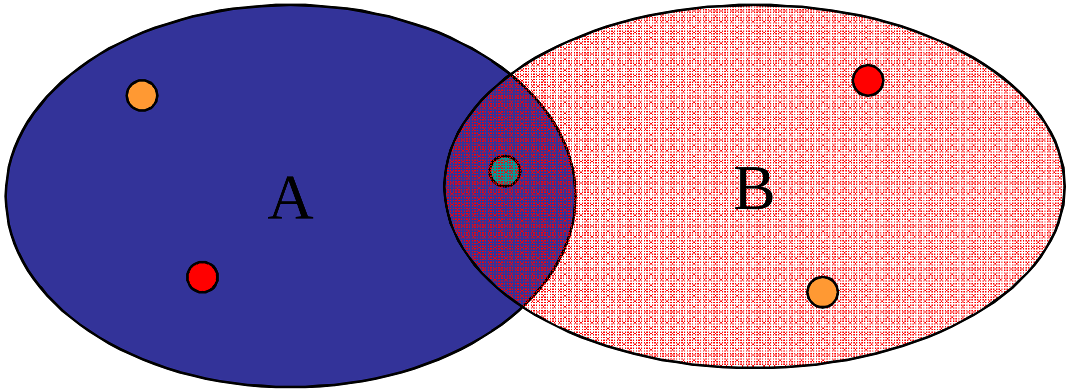
We are going to:

Sketch size: $O\,(\frac{1}{\varepsilon^2}\,log\frac{1}{\delta})\cdot\cancel{logU}$

Hashes required: $O\,(\frac{1}{\varepsilon^2}\,log\frac{1}{\delta})\cdot O\,(log\frac{1}{\varepsilon}\,logU) = O\,(\cancel{\frac{1}{\varepsilon^2}}\,log\frac{1}{\delta}\,log\frac{1}{\varepsilon}\,logU)$

Time analysis

$h_1 \quad h_2 \quad \ldots\ldots\ldots \quad h_{O(\frac{1}{\varepsilon^2} log\frac{1}{\delta})}$ Min wise independent



Time: $O\,(\frac{1}{\varepsilon^2}\,log\frac{1}{\delta})\cdot O\,(log\frac{1}{\varepsilon}) = O\,(\frac{1}{\varepsilon^2}\,log\frac{1}{\delta}\,log\frac{1}{\varepsilon})$

We are going to reduce it to: $O\,(log\frac{1}{\varepsilon}\,log\frac{1}{\delta})$

Contribution

•Fingerprint computation time

–Time depends on accuracy and confidence

–Previous methods

•Each item requires time of k

–k is quadratic in accuracy, logarithmic in confidence

–Current approach

•Each item requires time of log(k)

•Exponential speedup over previous approaches

•Fingerprint size (per universe size u)

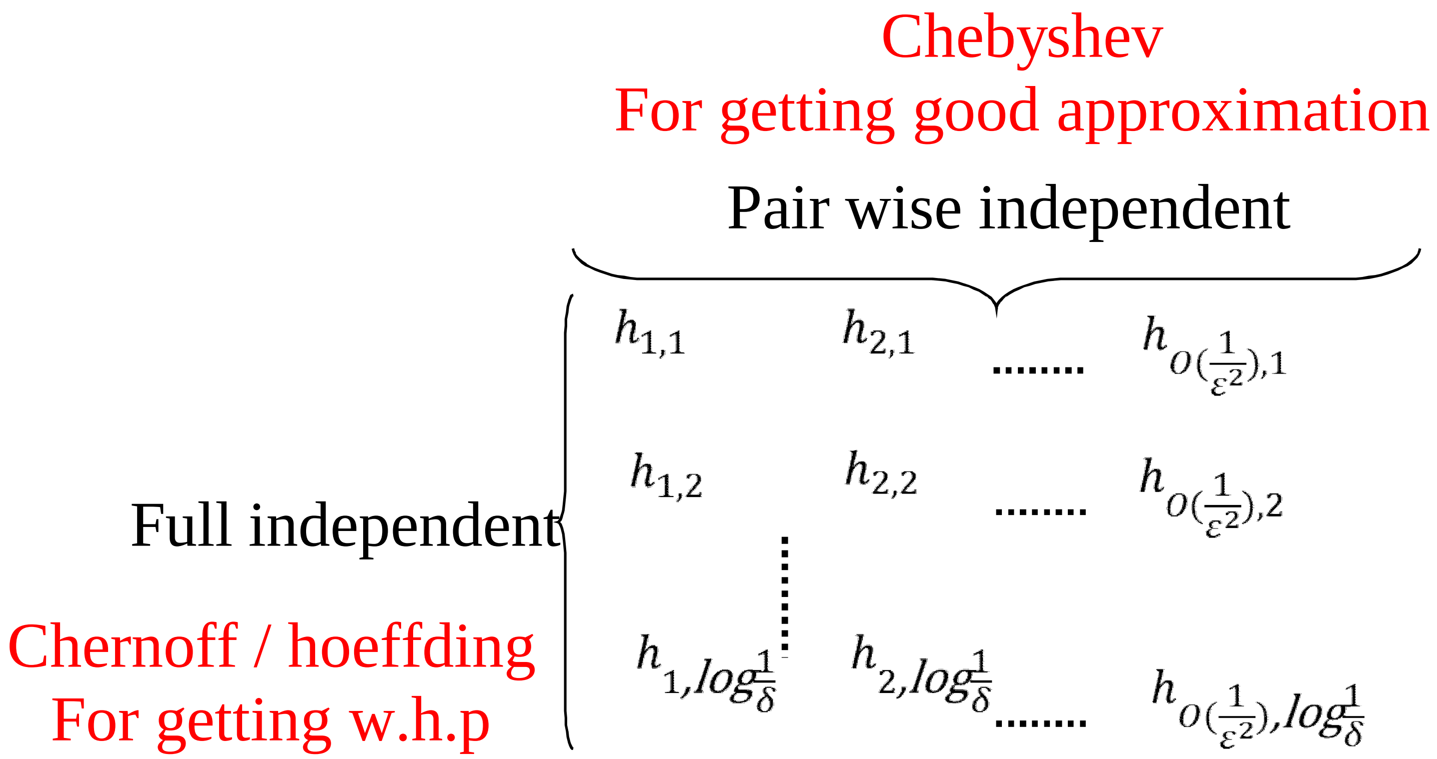
–Previous approaches require log(u) bits per item

–Each items requires a single bit per item

•General/generic method

–Applicable to many previous fingerprints

Pair wise independent



Chebyshev
For getting good approximation

Pair wise independent

Full independent

Chernoff / hoeffding
For getting w.h.p

$$\begin{array}{cccc} h_{1,1} & h_{2,1} & \ldots\ldots & h_{O(\frac{1}{\varepsilon^2}),1} \\ h_{1,2} & h_{2,2} & \ldots\ldots & h_{O(\frac{1}{\varepsilon^2}),2} \\ & \vdots & & \\ h_{1,log\frac{1}{\delta}} & h_{2,log\frac{1}{\delta}} & \ldots\ldots & h_{O(\frac{1}{\varepsilon^2}),log\frac{1}{\delta}} \end{array}$$

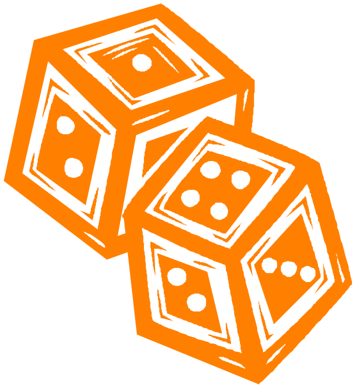
Min-wise or Pair-wise?

- In our case, every function by itself is (almost) min-wise independent

- Can construct simpler hashes
  - Almost min-wise independent
  - Only pair wise independent between themselves

Pseudo-Random Fingerprints

- Specific family of *pseudo-random* hashes
  - Shown to be *approximately min-wise independent*
  - Can quickly locate hashes resulting in *small values*
- Members are only *pair-wise* independent



Min-wise or Pair wise

- We choose f,g randomly from a family of O(log1/є) independent functions.

- Define $h_i(x)=f(x)+i*g(x)$

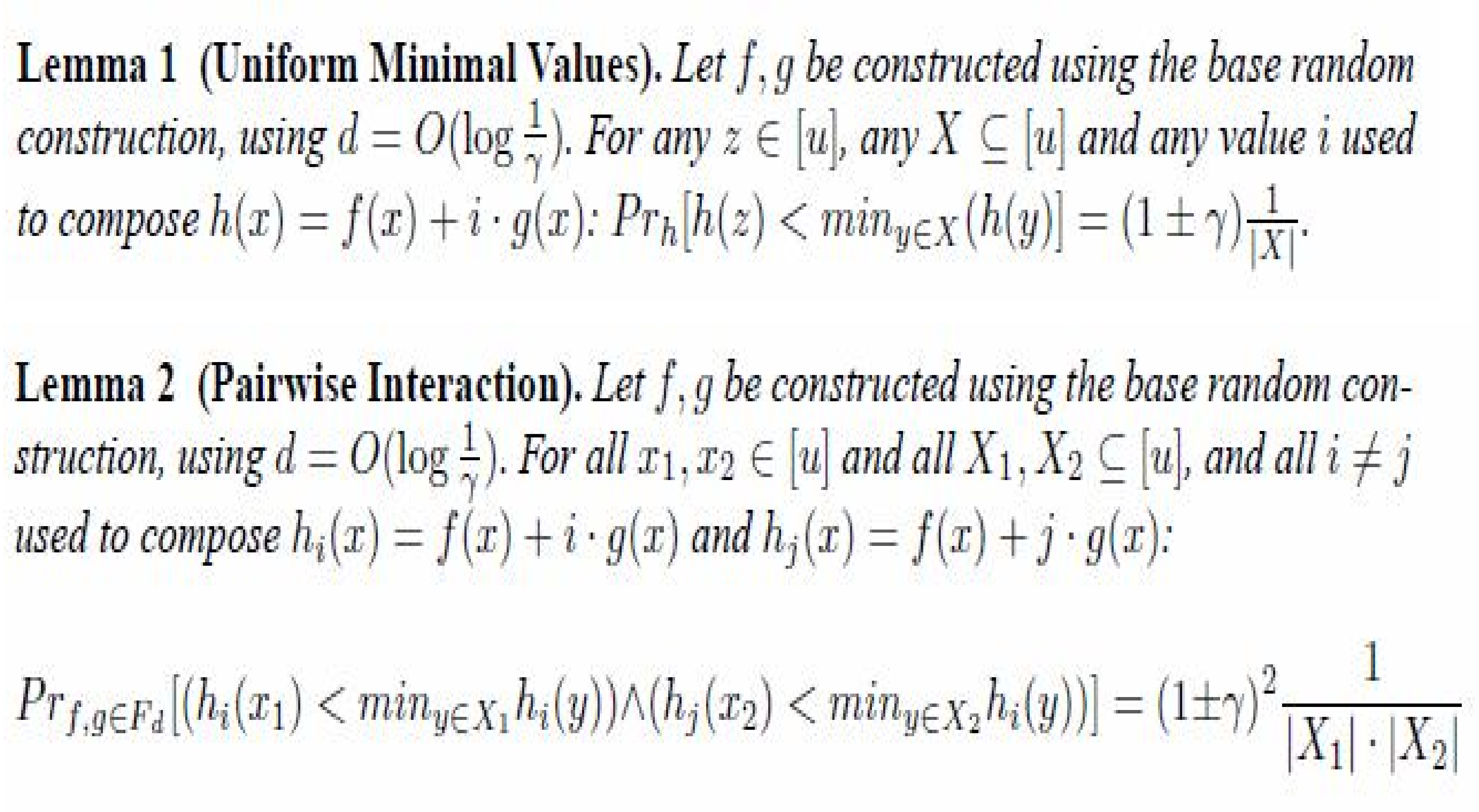
- Hash h_i behaves as a hash chosen randomly from a family of O(log1/є) independent functions
  - Therefore almost min wise independent.

Min-wise or Pair wise

- Define $h_i(x)=f(x)+i*g(x)$

- For any i and j h_i is independent of h_j

Properties



Lemma 1 (Uniform Minimal Values). *Let f, g be constructed using the base random construction, using $d = O(\log \frac{1}{\gamma})$. For any $z \in [u]$, any $X \subseteq [u]$ and any value i used to compose $h(x) = f(x) + i \cdot g(x)$: $Pr_h[h(z) < min_{y \in X}(h(y)] = (1 \pm \gamma)\frac{1}{|X|}$.*

Lemma 2 (Pairwise Interaction). *Let f, g be constructed using the base random construction, using $d = O(\log \frac{1}{\gamma})$. For all $x_1, x_2 \in [u]$ and all $X_1, X_2 \subseteq [u]$, and all $i \neq j$ used to compose $h_i(x) = f(x) + i \cdot g(x)$ and $h_j(x) = f(x) + j \cdot g(x)$:*

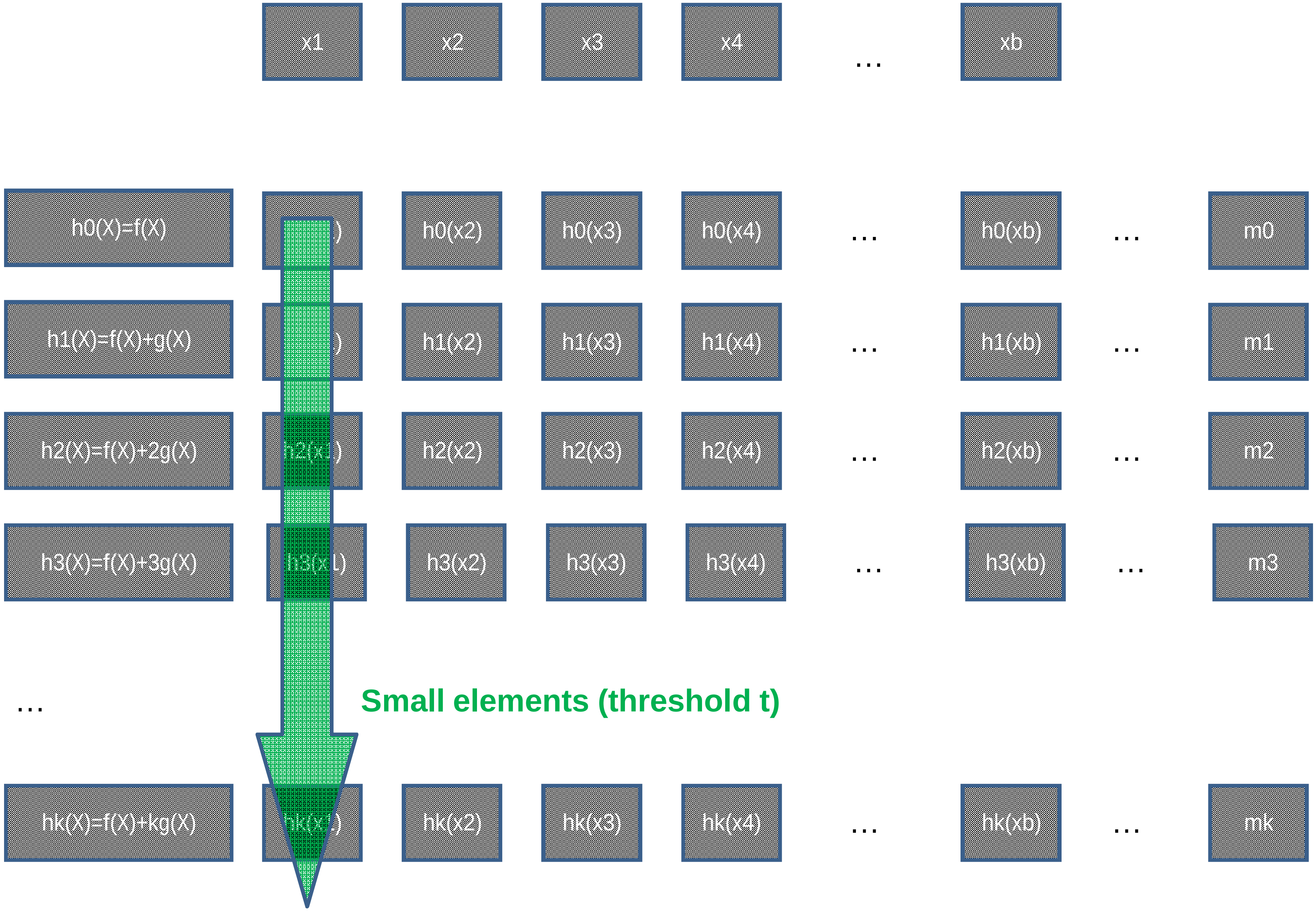
$$Pr_{f,g \in F_d}[(h_i(x_1) < min_{y \in X_1} h_i(y)) \wedge (h_j(x_2) < min_{y \in X_2} h_i(y))] = (1 \pm \gamma)^2 \frac{1}{|X_1| \cdot |X_2|}$$

What we got so far

$$
\text{Full independent}\left\{\begin{array}{lllll}
 & \multicolumn{4}{c}{\overbrace{\hphantom{h_{1,1} \quad h_{2,1} \quad — \quad h_{O(\frac{1}{\varepsilon^2}),1}}}^{\text{Pair wise independent}}} \\
f_1, g_1 & h_{1,1} & h_{2,1} & — & h_{O(\frac{1}{\varepsilon^2}),1} \\
f_2, g_2 & h_{1,2} & h_{2,2} & — & h_{O(\frac{1}{\varepsilon^2}),2} \\
 & \vert & & & \\
f_{log\frac{1}{\delta}}, g_{log\frac{1}{\delta}} & h_{1,log\frac{1}{\delta}} & h_{2,log\frac{1}{\delta}} & — & h_{O(\frac{1}{\varepsilon^2}),log\frac{1}{\delta}}
\end{array}\right.
$$

	Time	Space
Was	$O\,(\frac{1}{\varepsilon^2}\,log\frac{1}{\delta}\,log\frac{1}{\varepsilon})$	$O\,(\frac{1}{\varepsilon^2}\,log\frac{1}{\delta}\,log\frac{1}{\varepsilon}\,logU)$
Now	$O\,(log\frac{1}{\delta}\,log\frac{1}{\varepsilon} + \frac{1}{\varepsilon^2}\,log\frac{1}{\delta})$	$O\,(log\frac{1}{\delta}\,log\frac{1}{\varepsilon}\,logU)$

Fast computational element



	x1	x2	x3	x4	…	xb		
h0(X)=f(X)	)	h0(x2)	h0(x3)	h0(x4)	…	h0(xb)	…	m0
h1(X)=f(X)+g(X)	)	h1(x2)	h1(x3)	h1(x4)	…	h1(xb)	…	m1
h2(X)=f(X)+2g(X)	h2(x1)	h2(x2)	h2(x3)	h2(x4)	…	h2(xb)	…	m2
h3(X)=f(X)+3g(X)	h3(x1)	h3(x2)	h3(x3)	h3(x4)	…	h3(xb)	…	m3
…								
hk(X)=f(X)+kg(X)	hk(x1)	hk(x2)	hk(x3)	hk(x4)	…	hk(xb)	…	mk

Small elements (threshold t)

Finding Small Elements

- Can find all elements are smaller then a threshold in time: $O\ (log\frac{1}{\varepsilon} + Occ)$
    - Similar to an idea used by Pavan and Tirthapura

The idea

f(x)=18 g(x)=21 p=53 i=0,1,...,14

18,39,7,28,49,17,38,6,27,48,16,37,5,26,47

18,39,7,28,49,17,38,6,27,48,16,37,5,26,47

f(x)=7 g(x)=10 p=21
i=0,1,2,3,4

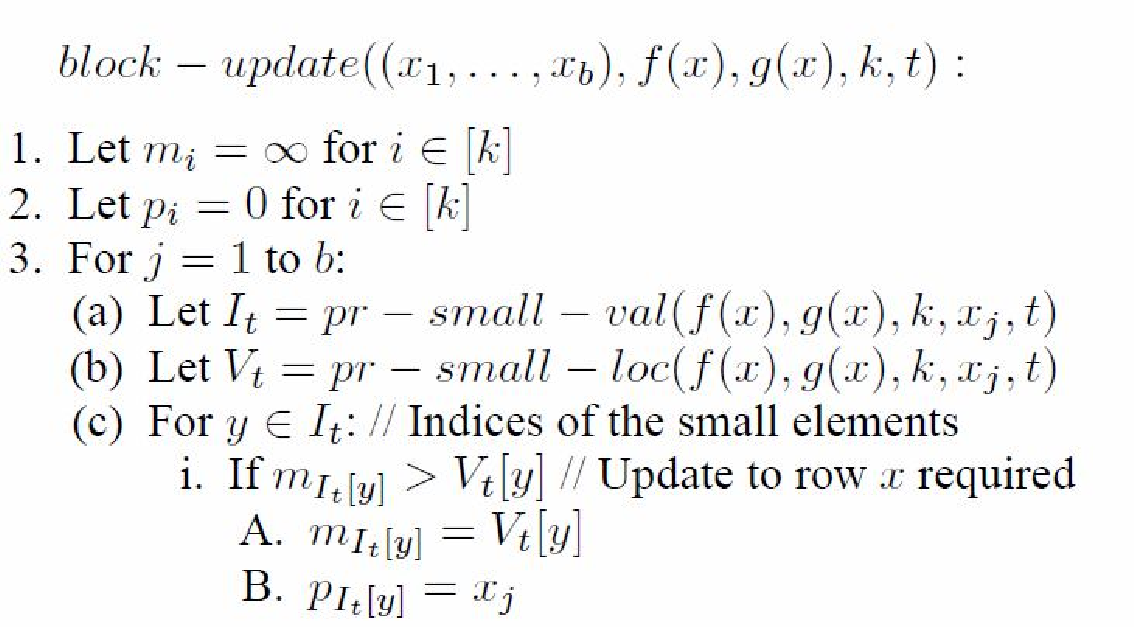
Algorithm

Maintain a bound on minimal row element

Update by iterating the columns

Find small elements (may trigger “missing” updates)

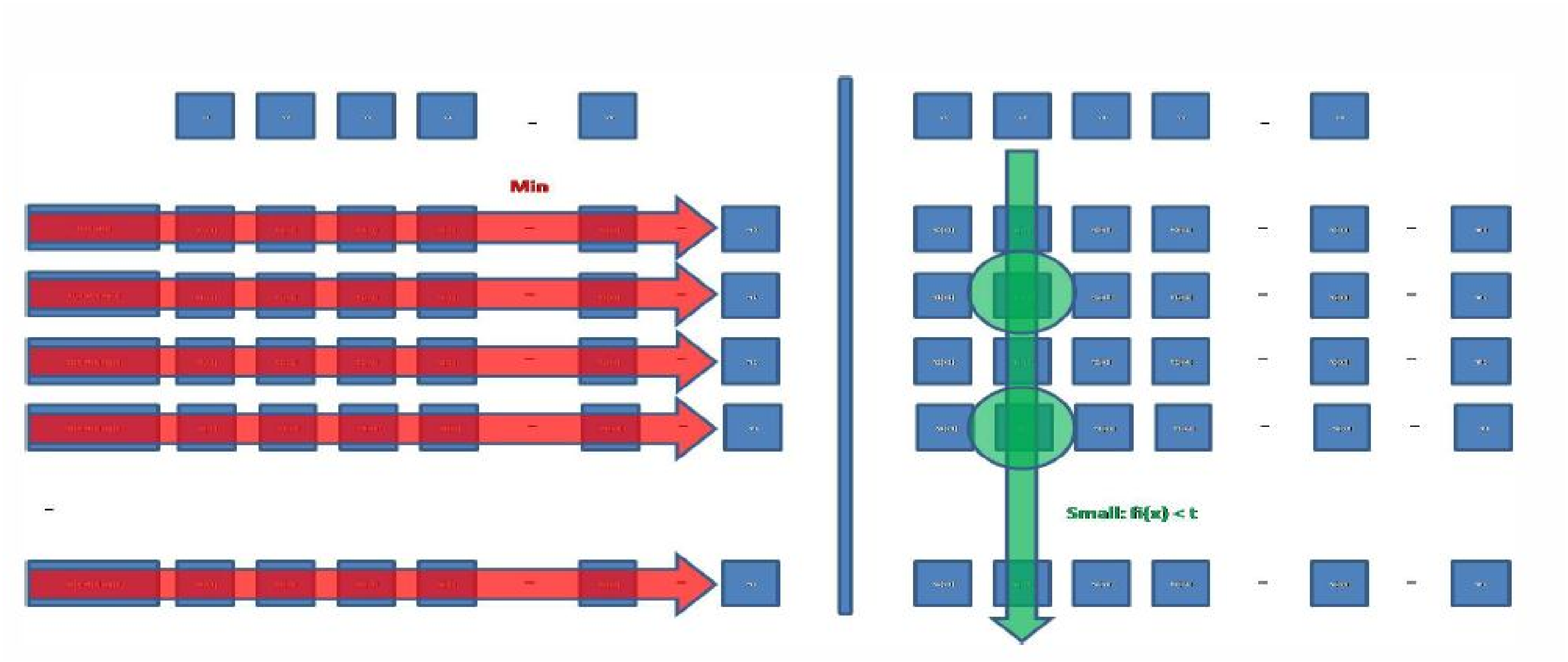
Update the rows where the y occur



```
$block - update((x_1, \ldots, x_b), f(x), g(x), k, t)$ :

1. Let $m_i = \infty$ for $i \in [k]$
2. Let $p_i = 0$ for $i \in [k]$
3. For $j = 1$ to $b$:
   (a) Let $I_t = pr - small - val(f(x), g(x), k, x_j, t)$
   (b) Let $V_t = pr - small - loc(f(x), g(x), k, x_j, t)$
   (c) For $y \in I_t$: // Indices of the small elements
       i. If $m_{I_t[y]} > V_t[y]$ // Update to row $x$ required
          A. $m_{I_t[y]} = V_t[y]$
          B. $p_{I_t[y]} = x_j$
```

Heart of the technique

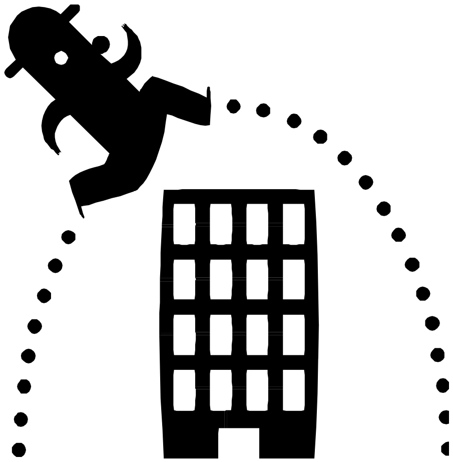


Required threshold and runtime

Column procedure time $O\ (log\frac{1}{\varepsilon} + Occ)$

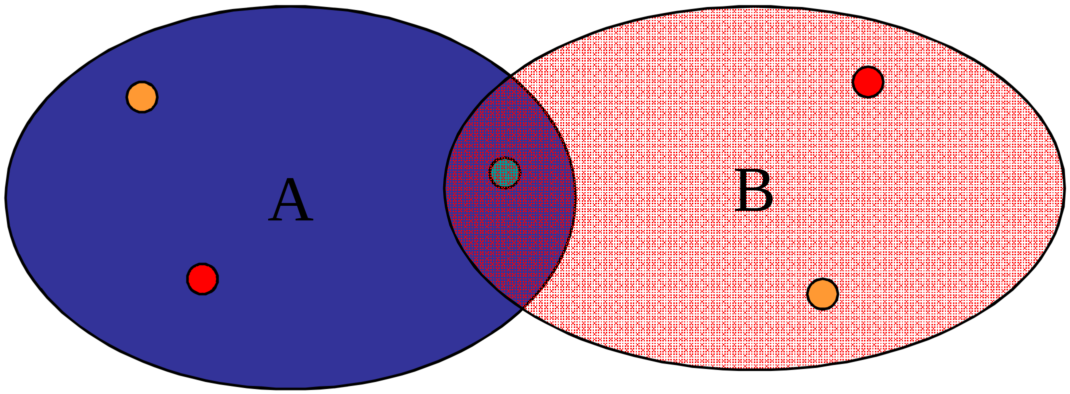
Threshold choice affects the runtime

But also the probability an error (missing updates)



Space analysis

$h_1 \quad h_2 \quad \ldots\ldots\ldots \quad h_{O(\frac{1}{\varepsilon^2}\log\frac{1}{\delta})}$ Min wise independent



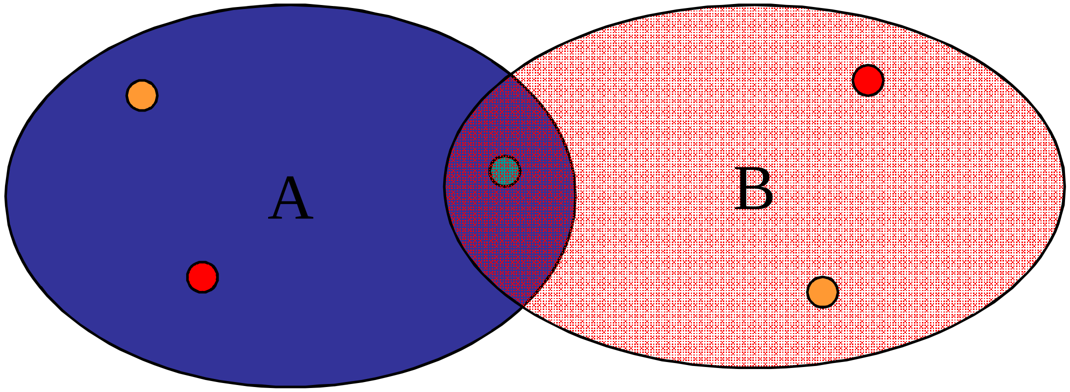
We are going to:

Sketch size: $O\left(\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right)\cdot\cancel{\log U}$

Hashes required: $O\left(\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right)\cdot O\left(\log\frac{1}{\varepsilon}\log U\right) = O\left(\cancel{\frac{1}{\varepsilon^2}}\log\frac{1}{\delta}\log\frac{1}{\varepsilon}\log U\right)$

Reducing Sketching Space

h_1 h_2 …….. $h_{O(\frac{1}{\varepsilon^2}log\frac{1}{\delta})}$ Min wise independent



We are going to:

Sketch size: $O\,(\frac{1}{\varepsilon^2}\,log\frac{1}{\delta})\cdot\cancel{logU}$

We hash each point to one bit

Reducing sketching space

Instead of

$$\mathrm{Pr}_{h_i}[min_{x\in A} h_i(x) = min_{x\in B} h_i(x)] = \frac{|A \cap B|}{|A \cup B|}$$

Additional pairwise independent hash

$$\mathrm{Pr}_{h,h_i}[h(min_{x\in A} h_i(x)) = h(min_{x\in B} h_i(x))] =$$

$$\frac{|A \cap B|}{|A \cup B|} + (1 - \frac{|A \cap B|}{|A \cup B|}) \cdot \frac{1}{2} =$$

$$\frac{1}{2} + \frac{1}{2}\frac{|A \cap B|}{|A \cup B|} = p$$

Reducing sketching space

$$p = \frac{1}{2} + \frac{1}{2}\frac{|A \cap B|}{|A \cup B|}$$

Our algorithm estimates

$$p' = p \pm \varepsilon$$

$$2p' - 1 = \frac{|A \cap B|}{|A \cup B|} \pm 2\varepsilon$$

Conclusion

Fast fingerprinting for massive datasets

General technique applicable to many fingerprints

Using pseudo-random hashes

Exponential speedup of computation

Future research

Speeding up computation even further

Similar techniques to fingerprints not based on minimal elements under the hash