南京大学 NANJING UNIVERSITY

新型网络数据挖掘

Haipeng Dai (戴海鹏)

State Key Laboratory for Novel Software Technology,
Nanjing University, China.

2021.8.19

Outline

PART 01 **Background**

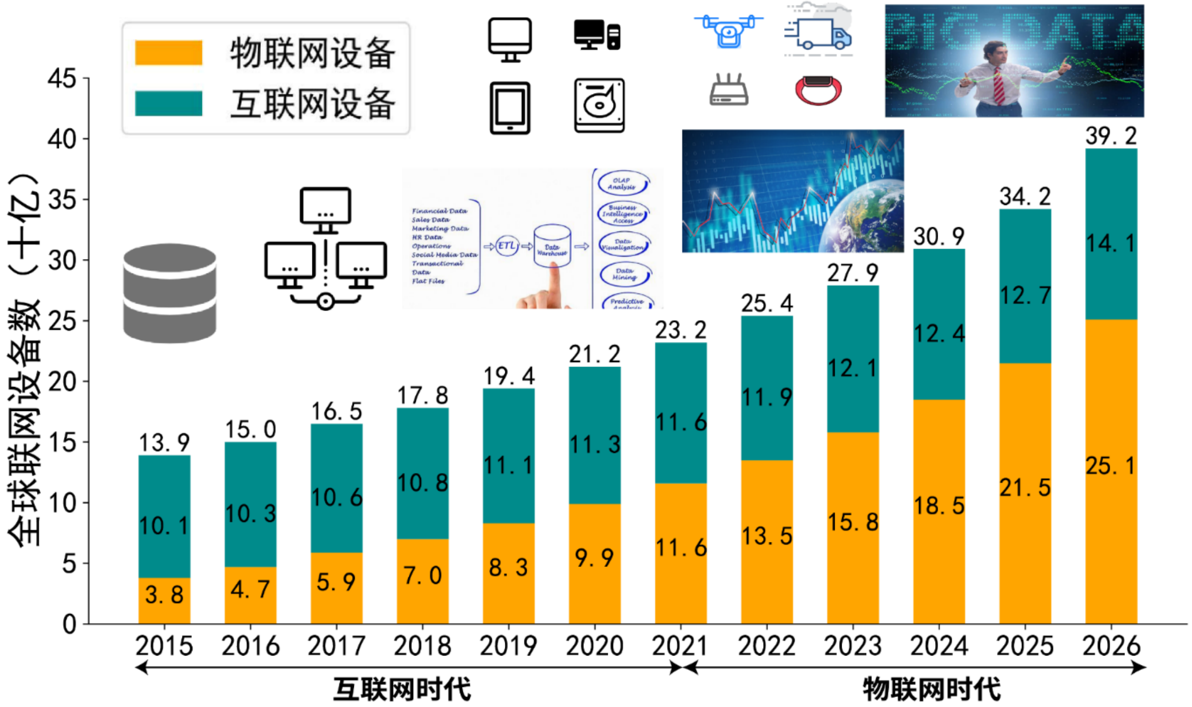
PART 02 Ming Items with Complex Pattern

- Persistent Items

PART 03 Utilizing Item Inherent Information

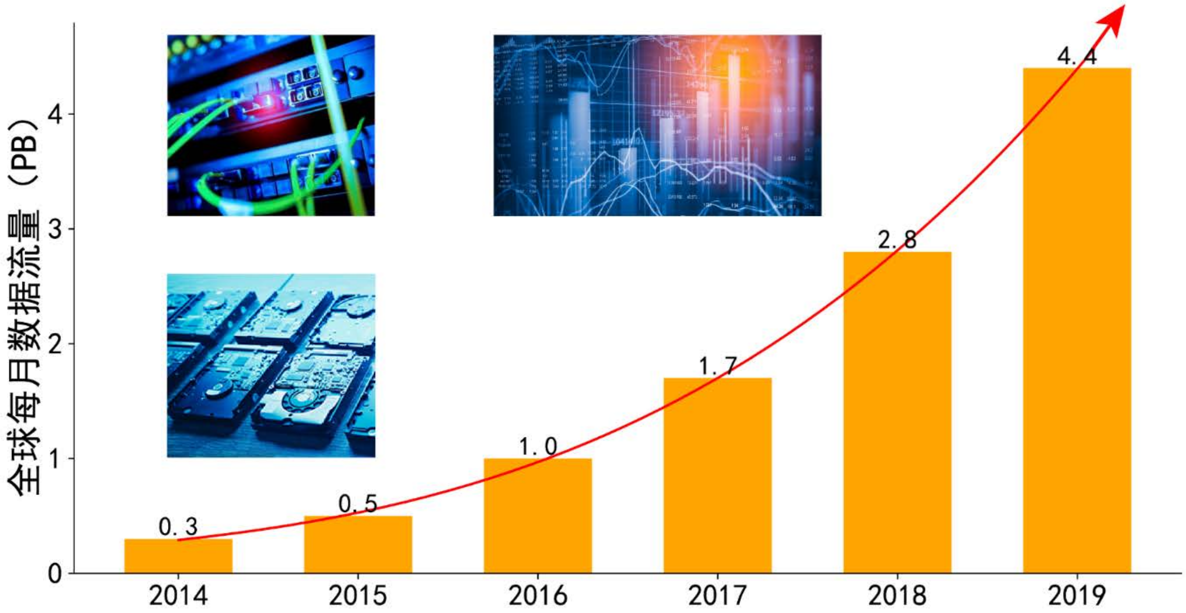
- Negative Items and Distribution

01/ Background



"Internet of Things (IoT) active device connections installed base worldwide,"
https://www.statista.com/statistics/1101442/iot-number-of-connected-devices-worldwide.

01/ Background



Massive data
vs.
Limited storage

“Global machine-to-machine (M2M) data traffic from 2014 to 2019," https://www.statista.com/statistics/267310/data-volume-of-non-internet-ip-traffic-by-category.

01/ Items Show Complex Pattern

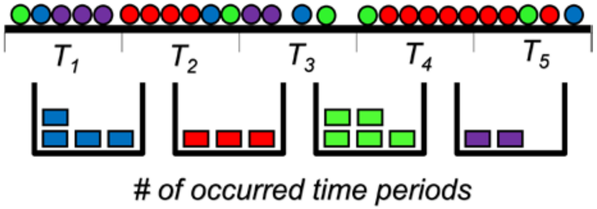
NANJING UNIVERSITY

- **Frequent Items**

  Charikar M, et al. Finding frequent items in data streams (ICALP 2002).

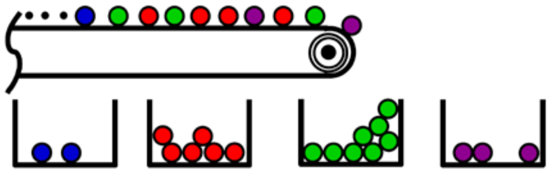
- **Items with Heavy Change**

  Monika R Henzinger. Algorithmic challenges in web search engines (Internet Mathematics 2004)



- **Persistent Items**

  Haipeng Dai, et al. Finding Persistent Items in Data Streams (VLDB 2017)



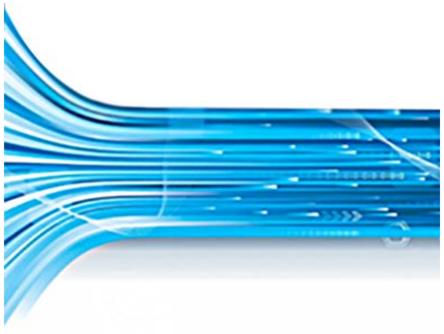
- **Significant Items**

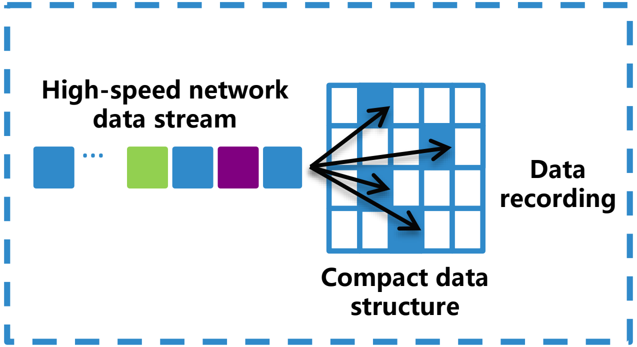
  Tong Yang, et al. Finding Significant Items in Data Streams (ICDE 2019)

- **… …**

01/ Challenge - I

NANJING UNIVERSITY





Challenge: How to design compact data structures with limited space to process, store and query the items with complex pattern in data stream efficiently?

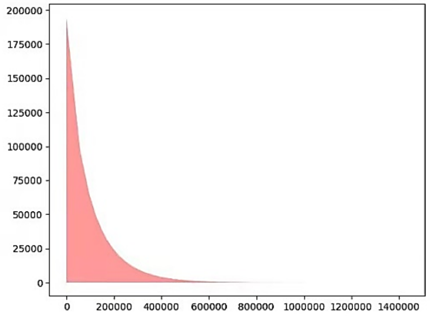
01/ Items Inherent Information

NANJING UNIVERSITY

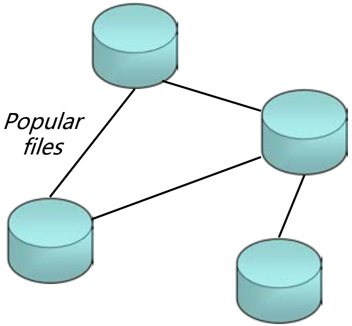
Availability of Negative (high-cost) Items

- **Publicly available**, like the online malicious IP address statistics for intrusion detection.
- **Obtained by cache**, most web servers can cache high-cost query records to improve the system performance.

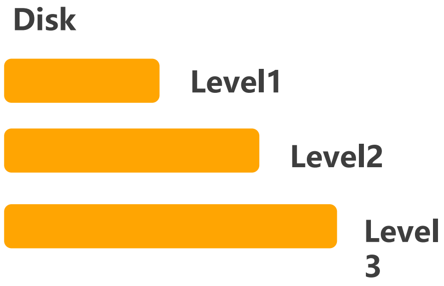
Items with Specific Distribution



Items are Skewed



Internet traffics



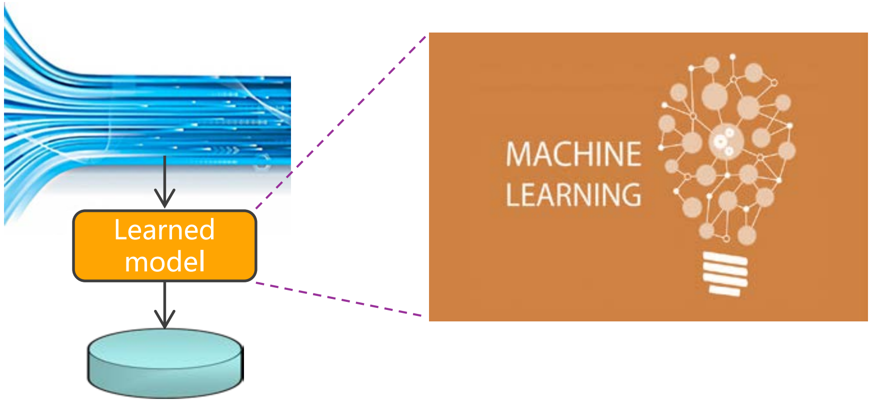
LSM-based k-v stores

01/ Challenge - II

南京大學
NANJING UNIVERSITY

Recently Proposed Learned Based Models is Impractical Now

- Prolonged training and query latency
- Relying on semantic knowledge of data



Challenge: How to design a smart data structure which can take full advantage of the information of workloads or dataset while achieving high performance.

Outline

NANJING UNIVERSITY

PART 01 Background

PART 02 Ming Items with Complex Pattern

- Persistent Items

PART 03 Utilizing Item Inherent Information

- Negative Items and Distribution

Finding Persistent Items in Data Streams (VLDB'17)

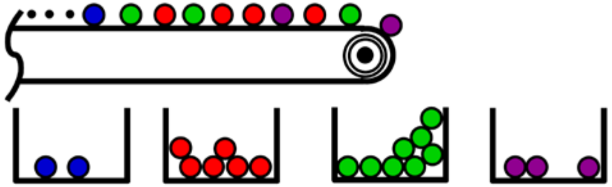
Haipeng Dai[1], Muhammad Shahzad[2], Alex X. Liu[1], and Yuankun Zhong[1].

[1] State Key Laboratory for Novel Software Technology,
Nanjing University, China.
[2] North Carolina State University, USA.

02/ Frequent Item vs. Persistent Item

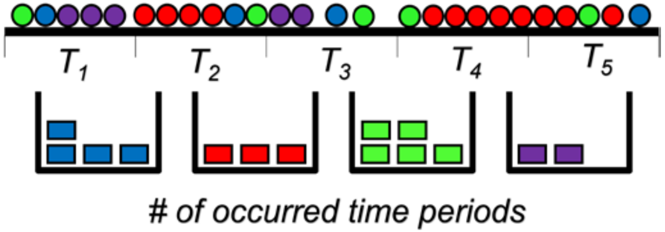
Frequent Item

- Given a stream of *N* items, find those that occur most frequently.



Persistent Item

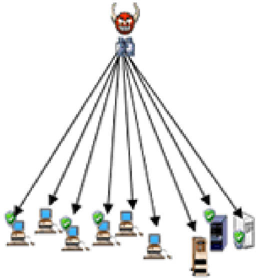
- Given a stream in *T* consecutive equally sized measurement periods, find those items that occur in most measurement periods (≥ a threshold T_{th}).

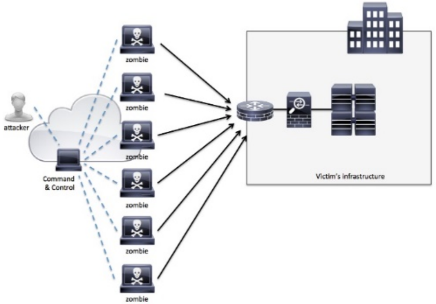


02/ Persistent Item

Potential Applications for Persistent Items Identification

- Stealthy DDoS attacks
- Stealthy port scanning attacks
- Communication between a bot and its C&C server
- Click-fraud detection

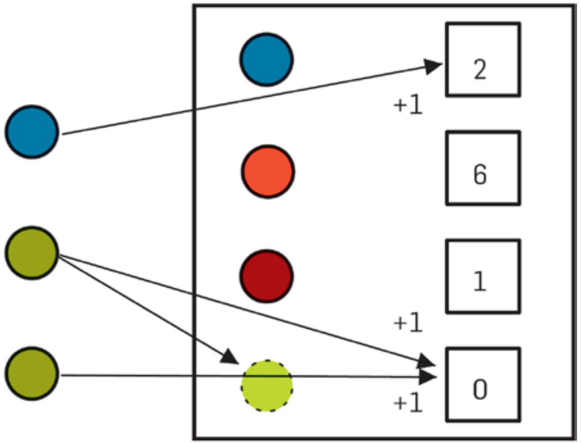




02/ Limitations of Prior Art

Prior Approach 1: Counter-based Algorithms

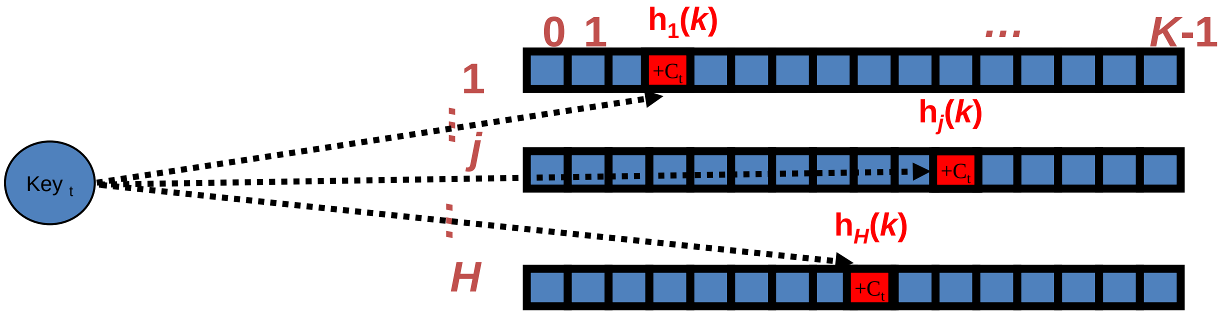
- Lossy Counting [Manku et al VLDB 2002]
- Space Saving [Metwally et al Database Theory-ICDT 2005]
- Limitations: low memory efficiency; cannot eliminate duplicate items within a measurement period



02/ Limitations of Prior Art

Prior Approach 2: Sketch-based Algorithms

- C-sketch [Charikar et al Automata, Languages and Programming 2002]
- CM-sketch [Cormode et al Journal of Algorithms 2005]
- Limitations: low memory efficiency; cannot eliminate duplicate items within a measurement period

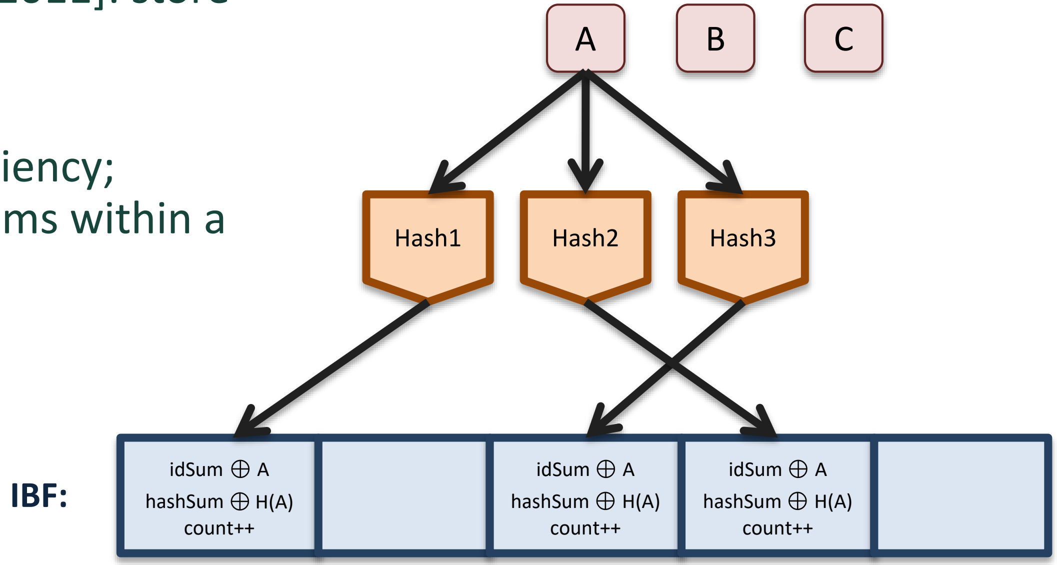


02/ Limitations of Prior Art

NANJING UNIVERSITY

Prior Approach 3: Invertible Bloom Filter

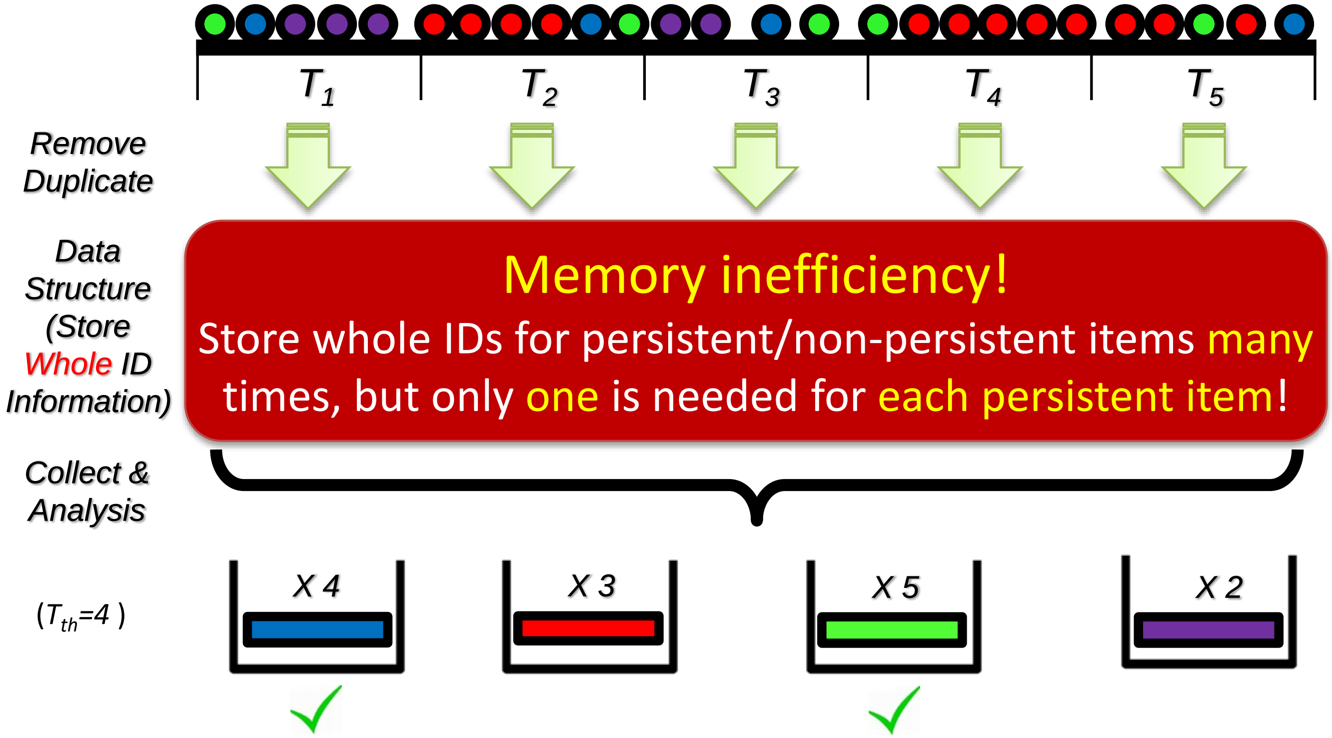
- IBF [Eppstein et al SIGCOMM 2011]: store set ID and auxiliary info.
- Limitations: low memory efficiency; cannot eliminate duplicate items within a measurement period



02/ Motivation

NANJING UNIVERSITY

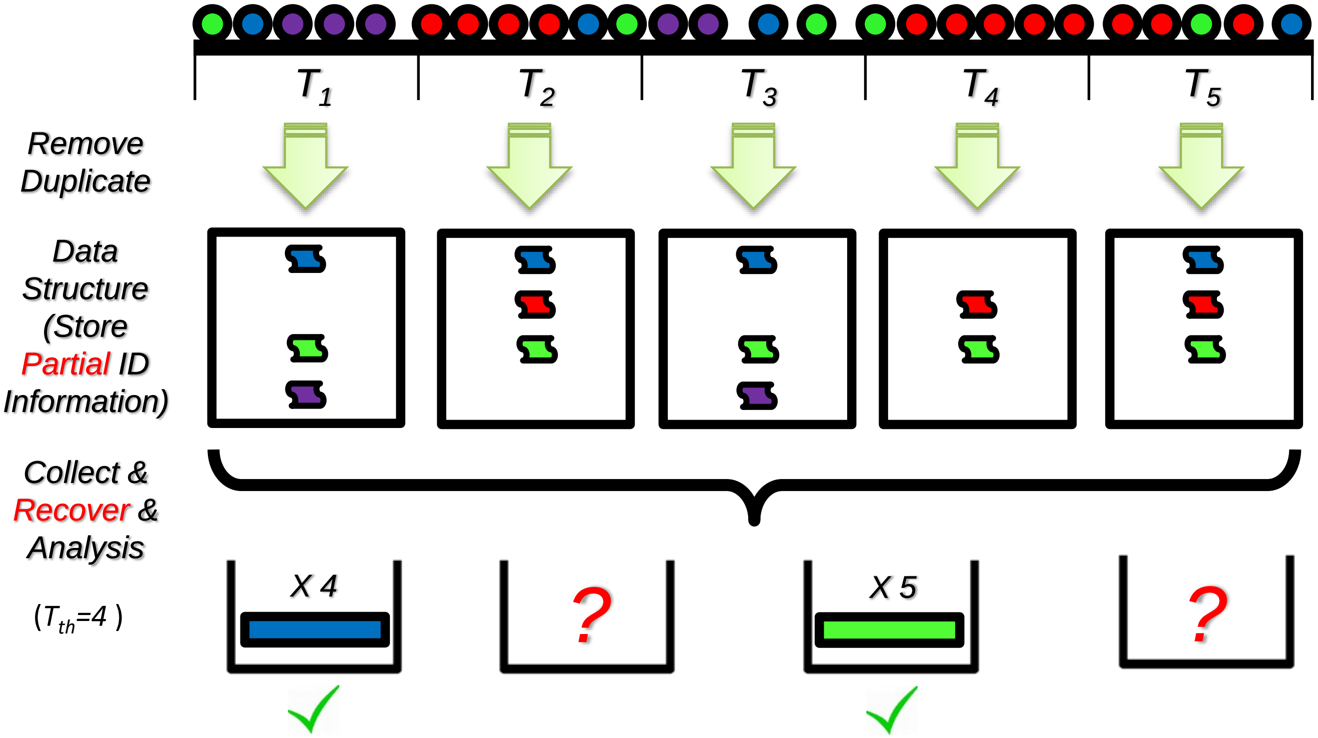
A naïve approach for persistent items identification



02/ Motivation

NANJING UNIVERSITY

Our solution: Persistent items Identification schemE (PIE)



02/ Key Idea of PIE

Our Solution: Persistent items Identification schemE (PIE)

Questions:

- How to store and recover ID information?

Encode + Decode

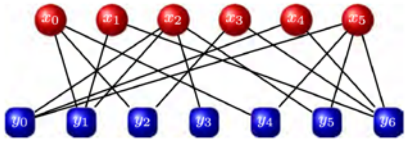
- How to pinpoint the stored information for the same item ID in the data structure without the knowledge of the item ID?

Position Info. + Hash-print Info.

02/ Key Idea of PIE

Introduction: encode *k* symbols into potentially limitless encoding symbols

$$
\begin{aligned}
y_0 &= x_2 + x_4 + x_5 \\
y_1 &= x_0 + x_1 + x_2 \\
y_2 &= x_0 + x_3 \\
y_3 &= x_2 \\
y_4 &= x_0 + x_5 \\
y_5 &= x_2 + x_5 \\
y_6 &= x_1 + x_3 + x_4 + x_5
\end{aligned}
$$



Advantages:

Linear time encoding and decoding speed

High decoding success probability (or low decoding failure probability)

…

Decoding failure probability

$$P_{df}(r;l) = \begin{cases} 1, & \text{if} \quad r < l \\ 0.85 \times 0.567^{r-l}, & \text{if} \quad r \geq l \end{cases}$$

02/ Space-Time Bloom Filter (STBF)

南京大學 NANJING UNIVERSITY

- Structure: an array C_i of cells with uniform length

- Structure of a cell: (Flag, Raptor codes, Hash-print)

Name (notation)	Length	Description
Flag: $C_{iF}[x]$	1	Indicating whether the cell is empty, singleton, or collided (combined with other bits)
Raptor Codes : $C_{iR}[x]$	*r*	Encoded codes for ID recovering, same for all mapped cells for an item in a period, but different for different periods
Hash-print: $C_{iP}[x]$	*p*	Fingerprint-like info. generated by hashing for an item, same for all mapped cells in all periods

- Cell status: empty, singleton, collided

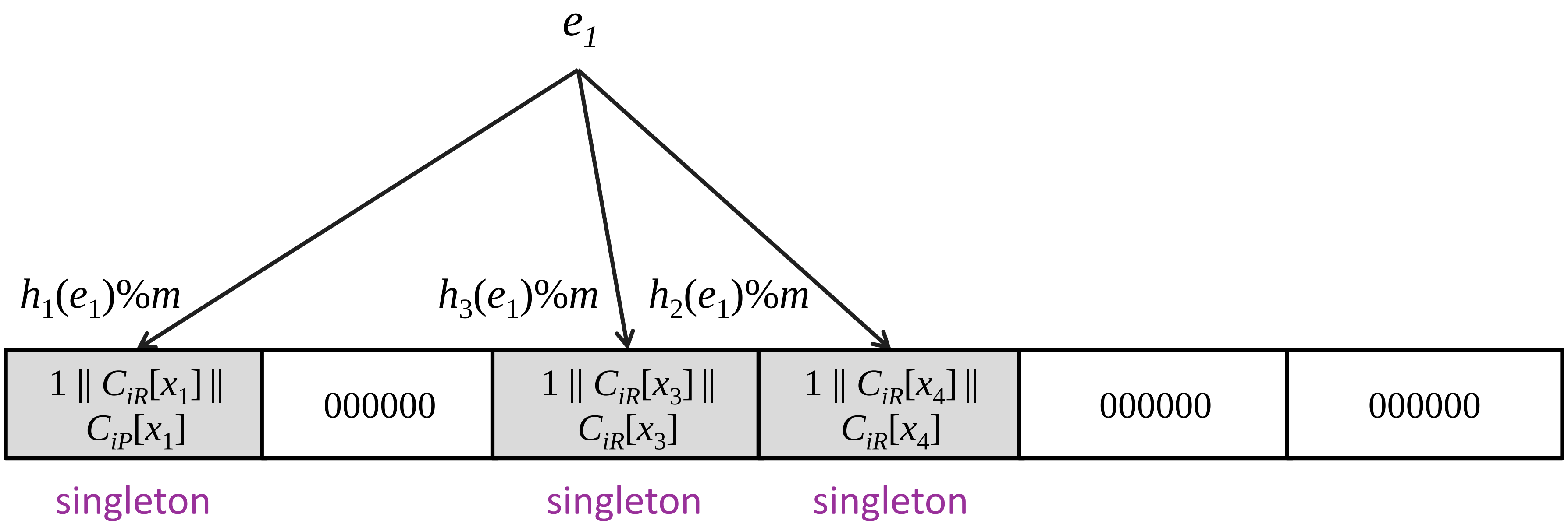
02/ Recording Phase of STBF

- E.g.:

000000	000000	000000	000000	000000	000000
empty	empty	empty	empty	empty	empty

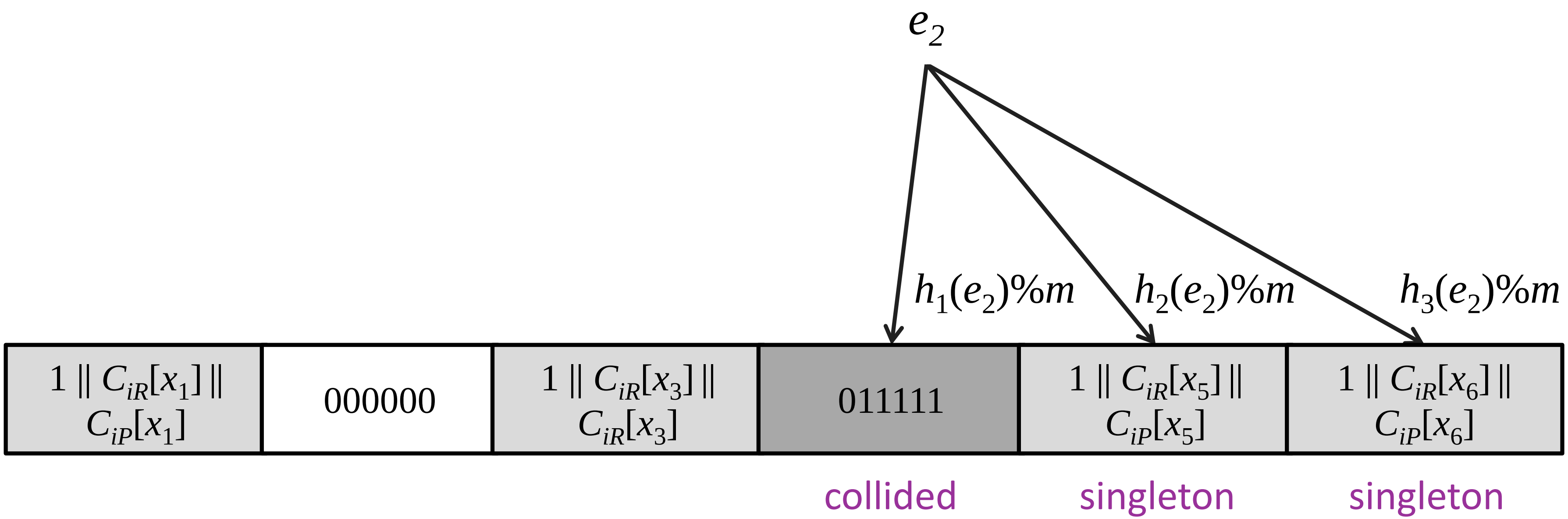
02/ Recording Phase of STBF

- E.g.:



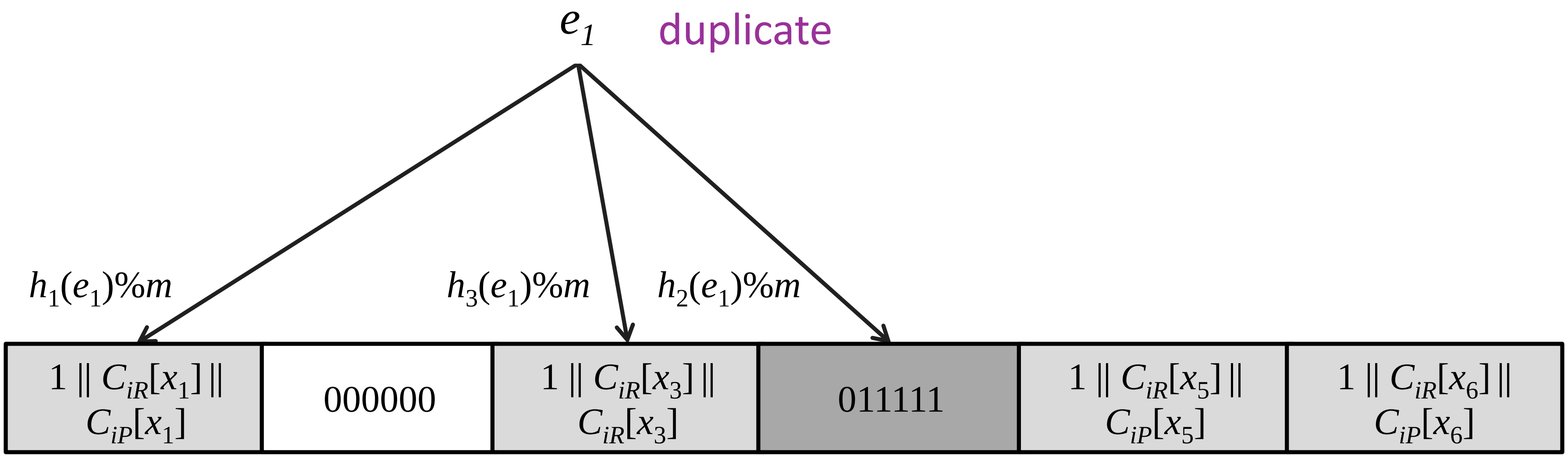
02/ Recording Phase of STBF

- E.g.:



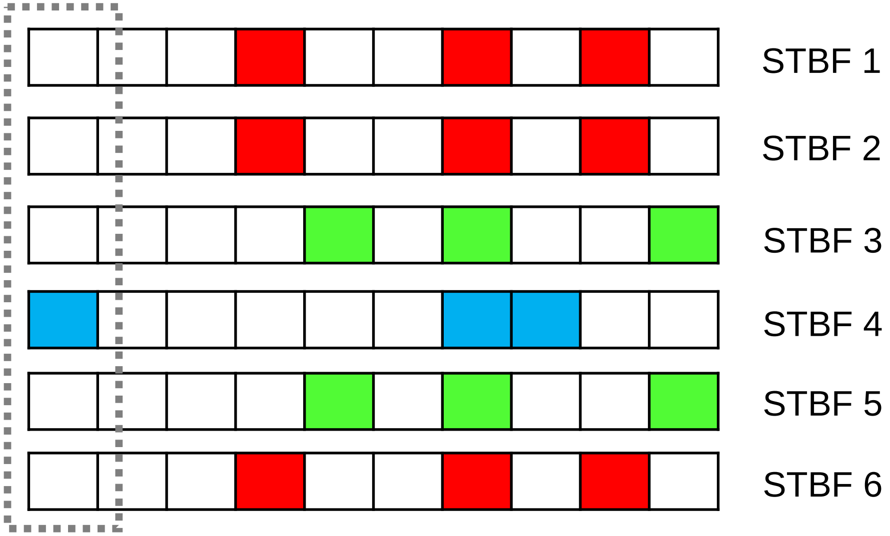
02/ Recording Phase of STBF

- E.g.:



02/ Identification Phase of STBF

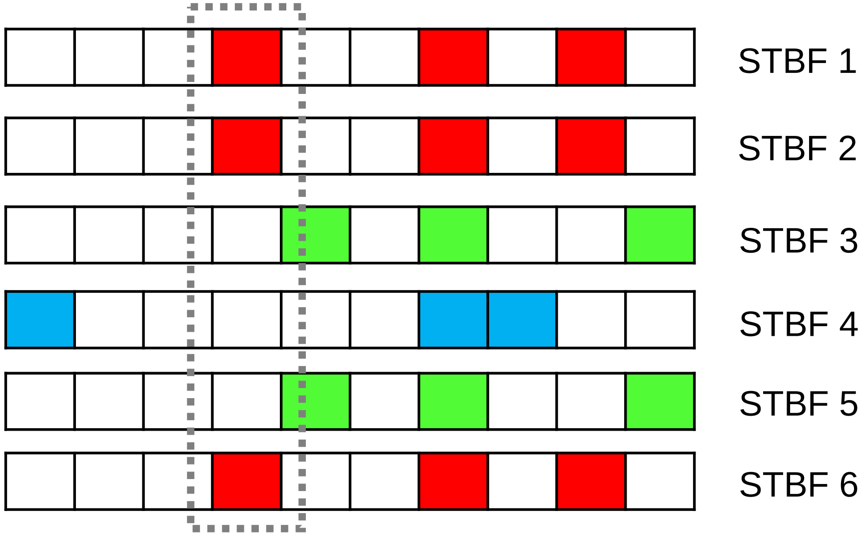
Process *cell lines* one by one, cluster stored Raptor codes into different groups in terms of cell lines and hash-prints, then decode codes in the same group



cell line 1 with 1 group

02/ Identification Phase of STBF

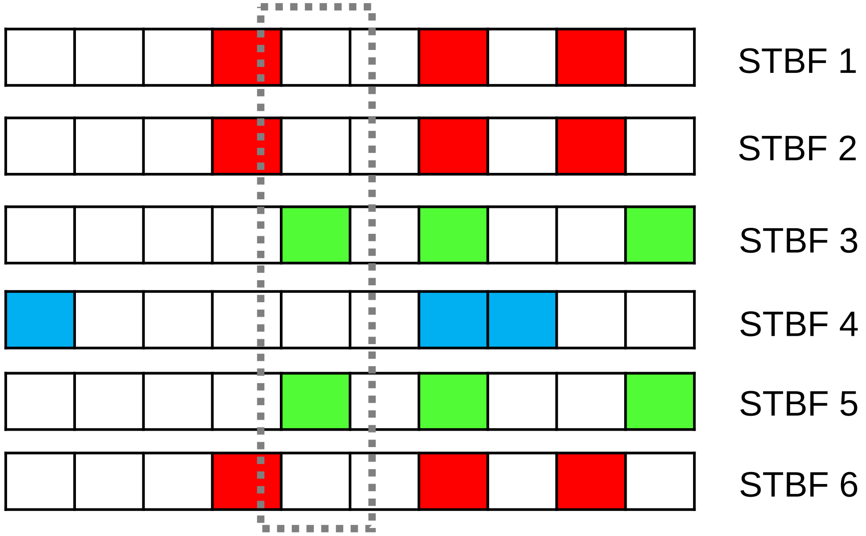
Process *cell lines* one by one, cluster stored Raptor codes into different groups in terms of cell lines and hash-prints, then decode codes in the same group



cell line 4 with 1 group

02/ Identification Phase of STBF

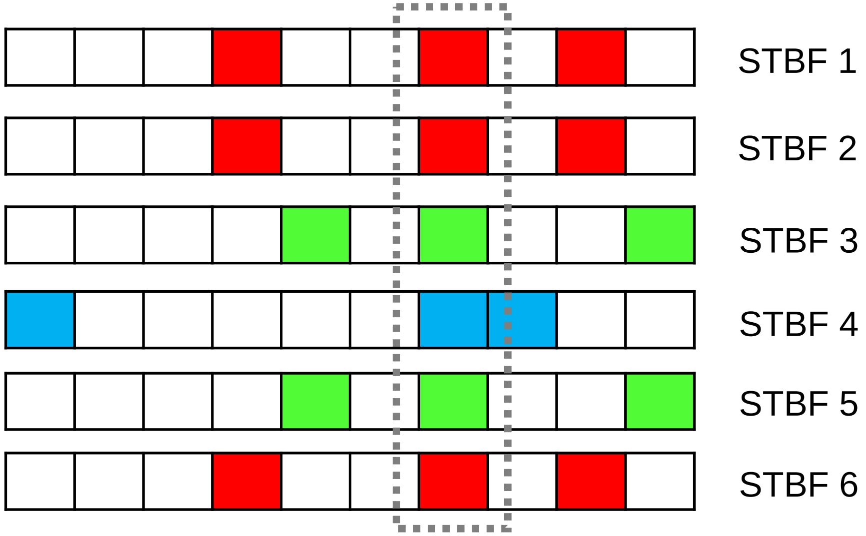
Process *cell lines* one by one, cluster stored Raptor codes into different groups in terms of cell lines and hash-prints, then decode codes in the same group



cell line 5 with 1 group

02/ Identification Phase of STBF

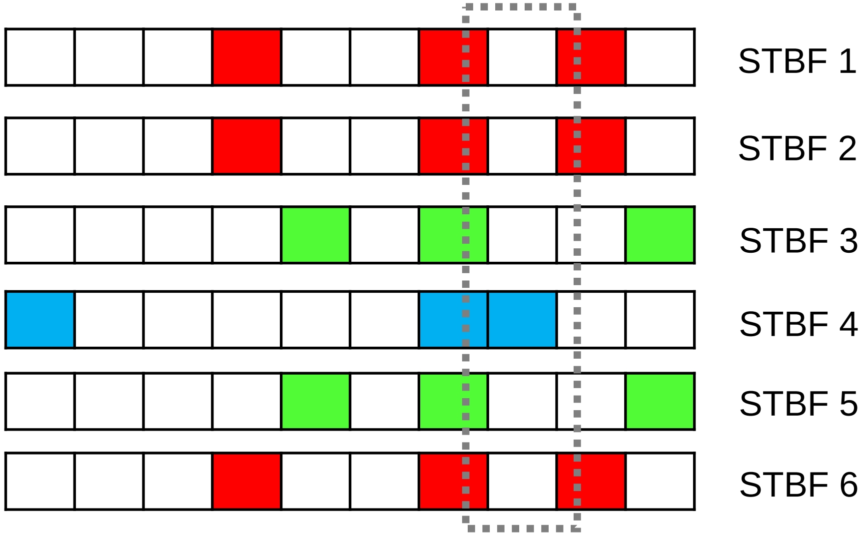
Process *cell lines* one by one, cluster stored Raptor codes into different groups in terms of cell lines and hash-prints, then decode codes in the same group



cell line 7 with 3 groups

02/ Identification Phase of STBF

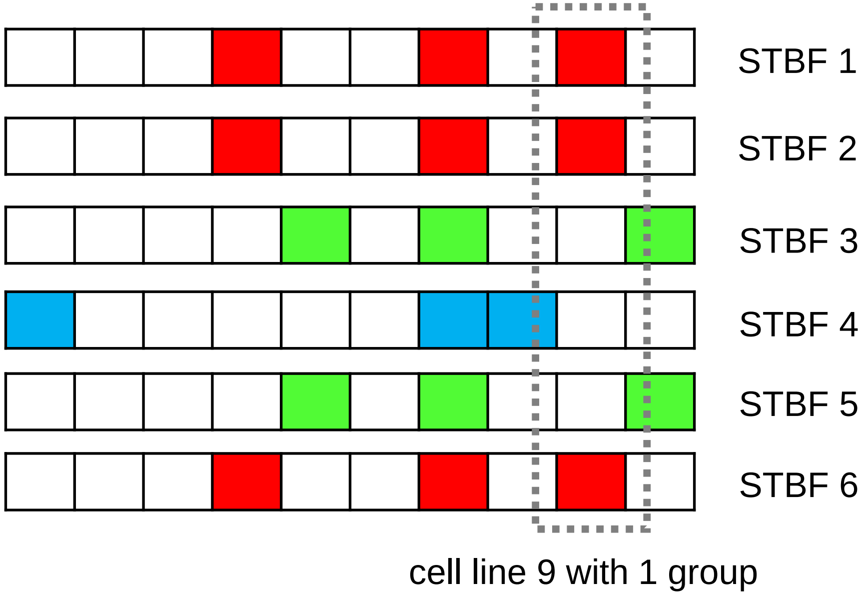
Process *cell lines* one by one, cluster stored Raptor codes into different groups in terms of cell lines and hash-prints, then decode codes in the same group



cell line 8 with 1 group

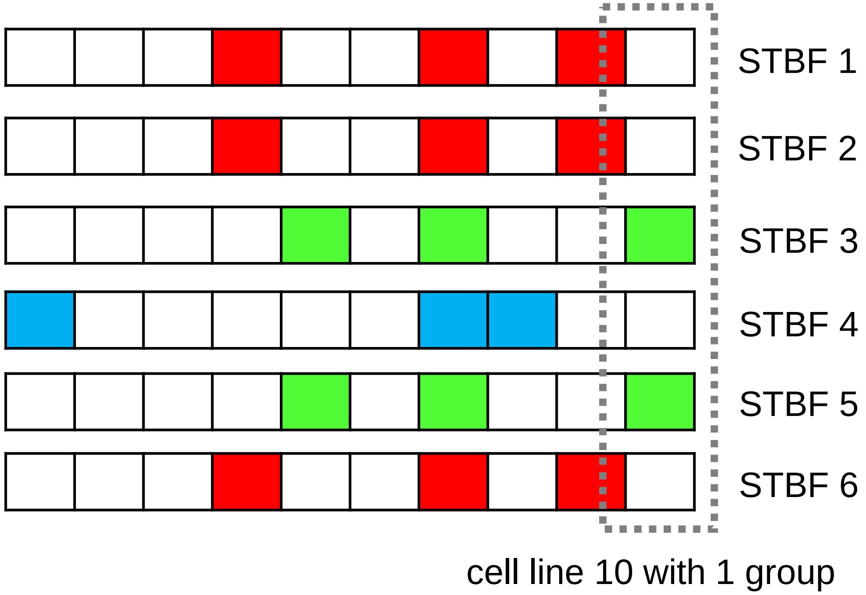
02/ Identification Phase of STBF

Process *cell lines* one by one, cluster stored Raptor codes into different groups in terms of cell lines and hash-prints, then decode codes in the same group



02/ Identification Phase of STBF

Process *cell lines* one by one, cluster stored Raptor codes into different groups in terms of cell lines and hash-prints, then decode codes in the same group



02/ Performance Analysis - FNR

- False Negative Rate (FNR): the rate of failing to recover the IDs of persistent items

- Two possible cases for FNR:
  - Hash-mapping Collision: one or more cells are collided and the Raptor codes are lost

    → easy to analyze

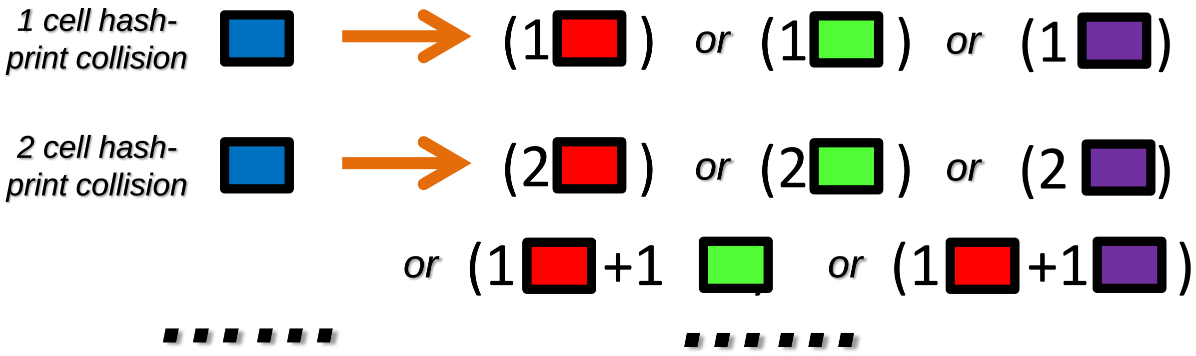
  - Hash-print Collision: some other items happen to have the **same** hash-prints with the considered persistent item, and their introduced Raptor codes make the recovering **fail**

    → hard to analyze

02/ Performance Analysis – FNR(cont’)

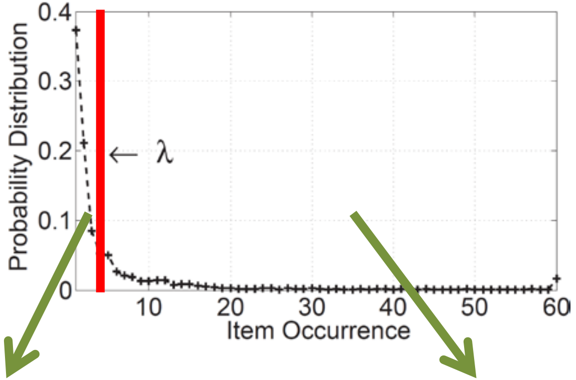
- Hash-print Collision analysis for FNR: need to enumerate all possibilities of collisions

- E.g.:



02/ Performance Analysis – FNR(cont')

- Our solution: two-partite approximation
  - Observation: distributions of item occurrences in practical applications typically follows Zipf or "Zipf-like" skewed distribution



Item occurrences ≈ 1

Analysis based on binomial distribution

Item occurrences ≈ sufficiently large to cause recovering failure

Analysis based on geometric distribution

02/ Performance Analysis – FPR

NANJING UNIVERSITY

- False Positive Rate (FPR): the rate of wrongly recovering the IDs of non-persistent items

- Two possible cases for FPR:
  - Phantom items: the recovered ID does not actually belong to any of the observed items during T measurement periods
  - Non-persistent items: the recovered ID is exactly the same as some other non-persistent item

- We derive an upper bound for FPR considering both cases

02/ Parameter Optimization for FNR

- Challenges:
  - Complicated mathematical expression of FNR
  - Optimization with three parameters (*r*, *p*, and *m*)

- Solutions:
  - Propose an approximation method to simplify the expression of FNR
  - Find the condition under which FNR is minimized if any parameter out of *r*, *p*, and *m* is fixed, and therefore, simplify the optimization

THEOREM 1. *The false negative rate is minimized if the inequality $r \times T_{th} \times P_{nc} \geq \kappa l$ takes the equal sign when $m(r+p+1) = M$, and the number of cells m is equal to $\lfloor m^* \rfloor$, where m^* satisfies the following equation*

$$\left(1-\left(1+\frac{kN}{T}\right)\frac{1}{m^*}\right)-\ln 2\cdot\frac{1}{m^*}\left(1-\frac{1}{m^*}\right)$$
$$\times\left(M-\frac{\kappa l}{T_{th}}\frac{kN}{T}\left(1-\frac{1}{m^*}\right)^{-\frac{kN}{T}-1}\right)=0 \quad (13)$$

and parameters r and p are calculated as below

$$r=\left\lceil\frac{\kappa l}{T_{th}}\left(1-\frac{1}{m}\right)^{-\frac{kN}{T}}\right\rceil \quad (14)$$

$$p=\left\lfloor\frac{M}{m}\right\rfloor-\left\lceil\frac{\kappa l}{T_{th}}\left(1-\frac{1}{m}\right)^{-kN/T}\right\rceil-1 \quad (15)$$

02/ Evaluation Setup

- Item Traces:
  - CHIC: a backbone header trace
  - ICSI: an enterprise network traffic trace
  - DC: a data center traffic trace collected at a university data center

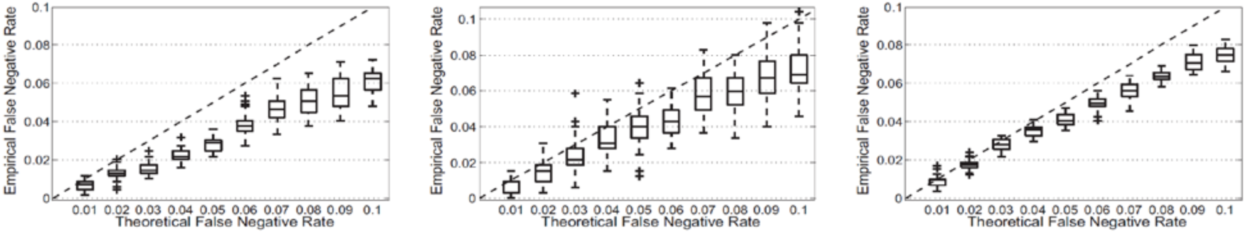
Trace	Duration	# pkts	# flows
CHIC	6 min	25.3 M	101,374
ICSI	1 hour	1.49 M	8,797
DC	1 hour	8.09 M	10,289

02/ Evaluation Setup

- Parameter Settings:
  - # of measurement periods T : 60
  - # of mapping hash functions k : 3
  - Mingling threshold g_T : 4
  - Memory for STBF M : 600Kb (CHIC), 100Kb (ICSI), 300 Kb (DC)
- Evaluation metrics:
  - False Negative Rate
  - False Positive Rate
- Side-by-side comparison:
  - CM sketch
  - IBF

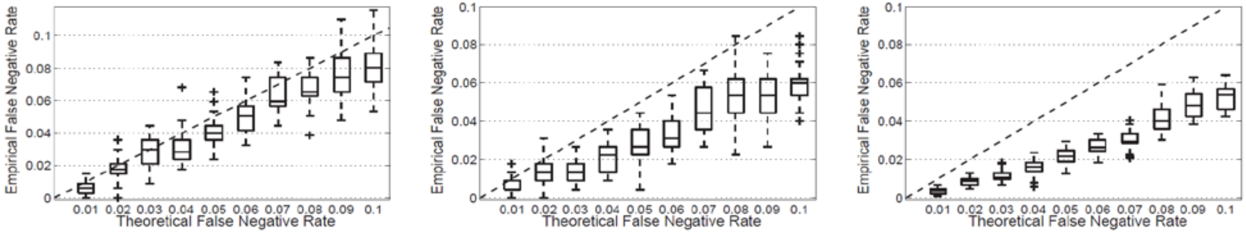
02/ False Negative Rate

- Our results show that the average FNR of PIE calculated from simulations is always less than the maximum desired FNR.



(a) CHIC (b) DC (c) ICSI

Empirical false negative rate vs. theoretical false negative rate when $T_{th} = 40$

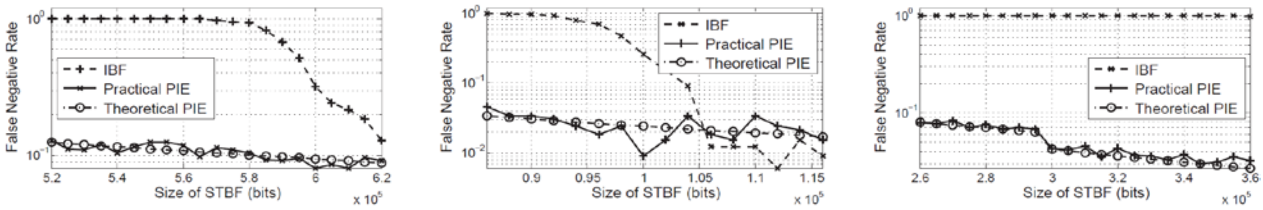


(a) CHIC (b) DC (c) ICSI

Empirical false negative rate vs. theoretical false negative rate when $T_{th} = 50$

02/ False Negative Rate

- Our results show that the average FNR of PIE is almost twice an order of magnitude smaller than the FNR of IBF.

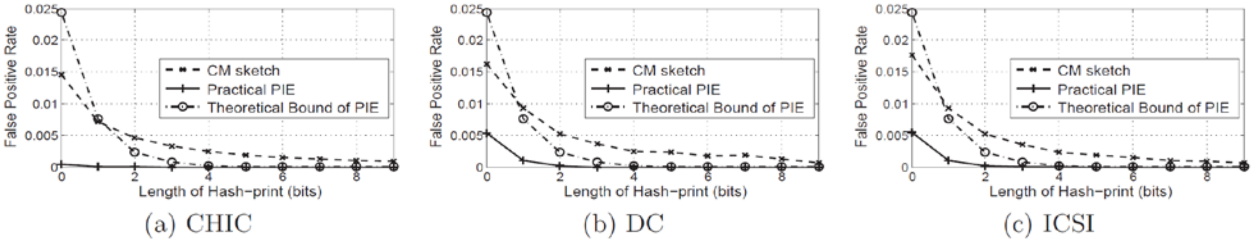


(a) CHIC (b) DC (c) ICSI

False negative rate when $T_{th} = 40$

02/ False Positive Rate

- Our results show that FPR of PIE is at least 426.1 times less than the FPR of CM sketch.



False positive rate when $T_{th} = 40$

02/ Conclusion

- Propose the notion of persistent item and define the problem of persistent items identification
- Propose the Space-Time Bloom Filter data structure for persistent items identification
- Analyze the False Negative Rate and False Positive Rate of PIE, and study parameter optimization
- Conduct numerical evaluations based on real traces to validate the performance of PIE

02/ Related Publications

- [VLDB'17] **Haipeng Dai**, Muhammad Shahzad, Alex X. Liu and Yuankun Zhong. "Finding Persistent Items in Data Streams". (**CCF A**)

- [TON'19] **Haipeng Dai**, Muhammad Shahzad, Alex X. Liu, Meng Li and Yuankun Zhong. "Identifying and Estimating Persistent Items in Data Streams". (**CCF A**)

02/ Citation

- Our VLDB paper “Finding Persistent Items in Data Streams” has been cited by 10 CCF A conferences and journals

Networking	SIGCOMM	CCF A
	TON	CCF A
	INFOCOM	CCF A
	TMC	CCF A
Parallel/distributed Computing	TPDS	CCF A
Database	SIGMOD	CCF A
	ICDE	CCF A
	VLDBJ	CCF A
Data Mining	SIGKDD	CCF A
	TKDE	CCF A

Finding Persistent Items in Distributed Datasets (INFOCOM'18)

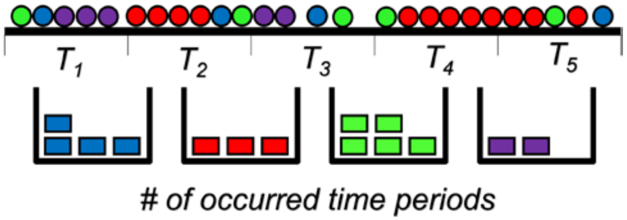
Haipeng Dai[1], Meng Li[1], and Alex X. Liu[1].

[1] State Key Laboratory for Novel Software Technology,
Nanjing University, China.

02/ Streams vs. Distributed Datasets

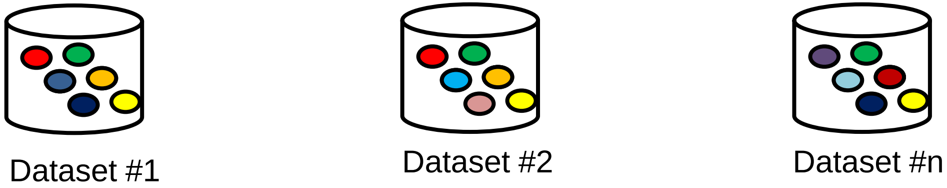
Persistent Item In Data Streams (Dynamic)

- A stream in T equally sized measurement periods, find those items in most measurement periods (≥ a threshold T_{th}).



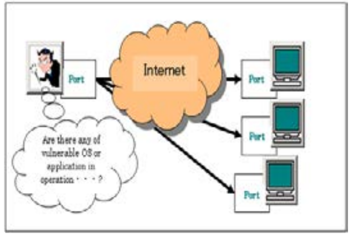
Persistent Item In Distributed Datasets (Static)

- Given T datasets, find those items in most datasets (≥ a threshold T_{th}).

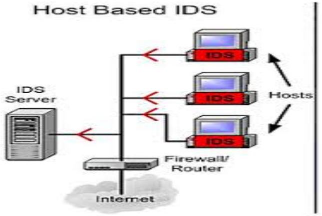


02/ Potential Applications

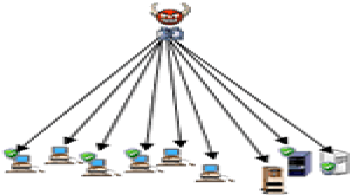
- Distributed port scanning attacks



- Distributed Intrusion Detection



- Distributed DDoS Attack Detection



02/ Related Works

NANJING UNIVERSITY

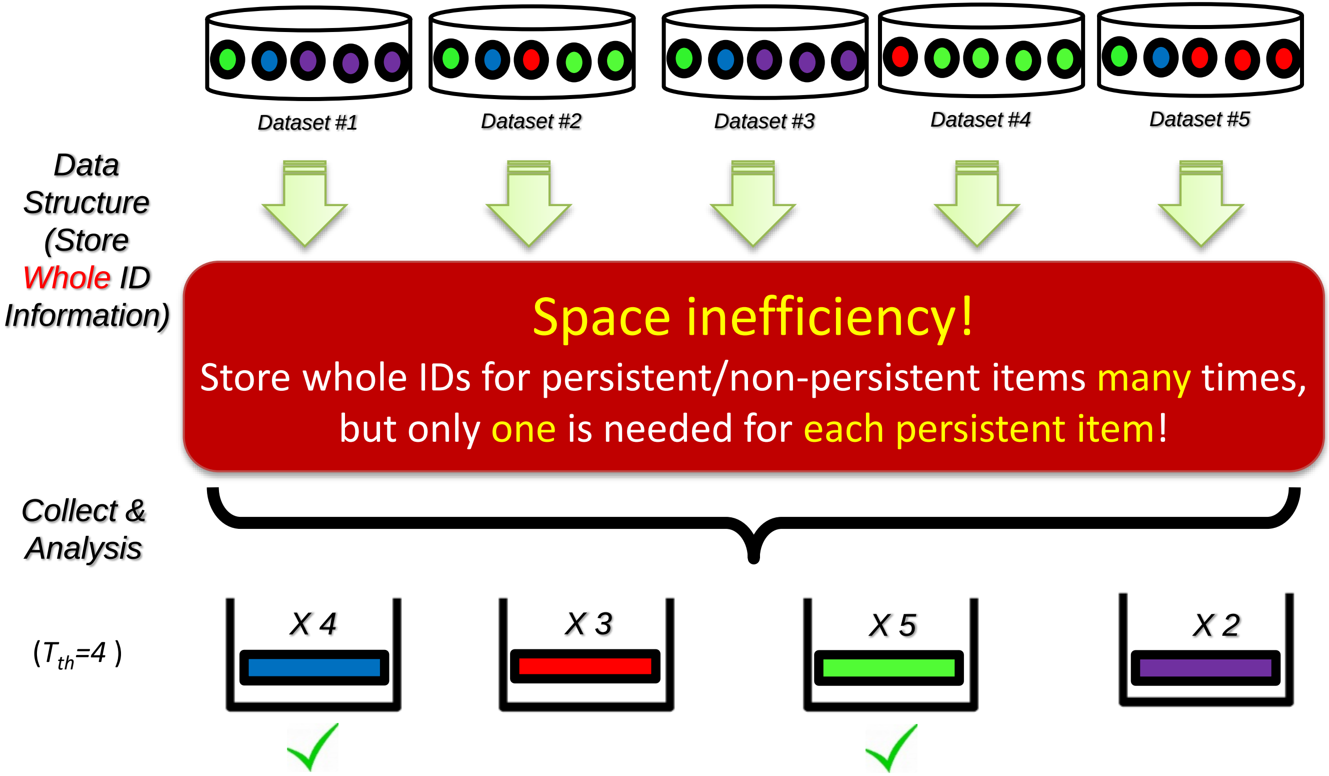
- Related Work #1: Frequent items identification in a distributed environment
  - Master- slave model: [B.Babcock et al Sigmod 2003] [Cao et al PODC 2004][Dai et al INFOCOM 2016]
  - Hierarchical model: [A. Manjhi et al ICDE 2005] [Li et al ICDCS 2008]
  - Decentralized model: [B. Lahiri et al JPDC 2010]
  - Limitation: poor performance in terms of communication cost

- Related Work #2: Cooperative monitoring systems
  - Specific aggregation functions: [G.Cormode et al TODS 2008] [S. Agrawal et al ICDE 2007] [C. Arackaparambil et al ICALP 2009] [L.Huang et al INFOCOM 2007] [M.Gabel et al IPDPS 2014]
  - Multi-set joining problem: [L. F. Mackert et al Sigmod 1986] [J.K. Mulin et al TSE 1990] [Z.Cai et al ICNP 2015]
  - Limitation: focus on specific aggregation or priori knowledge is needed

02/ Motivation

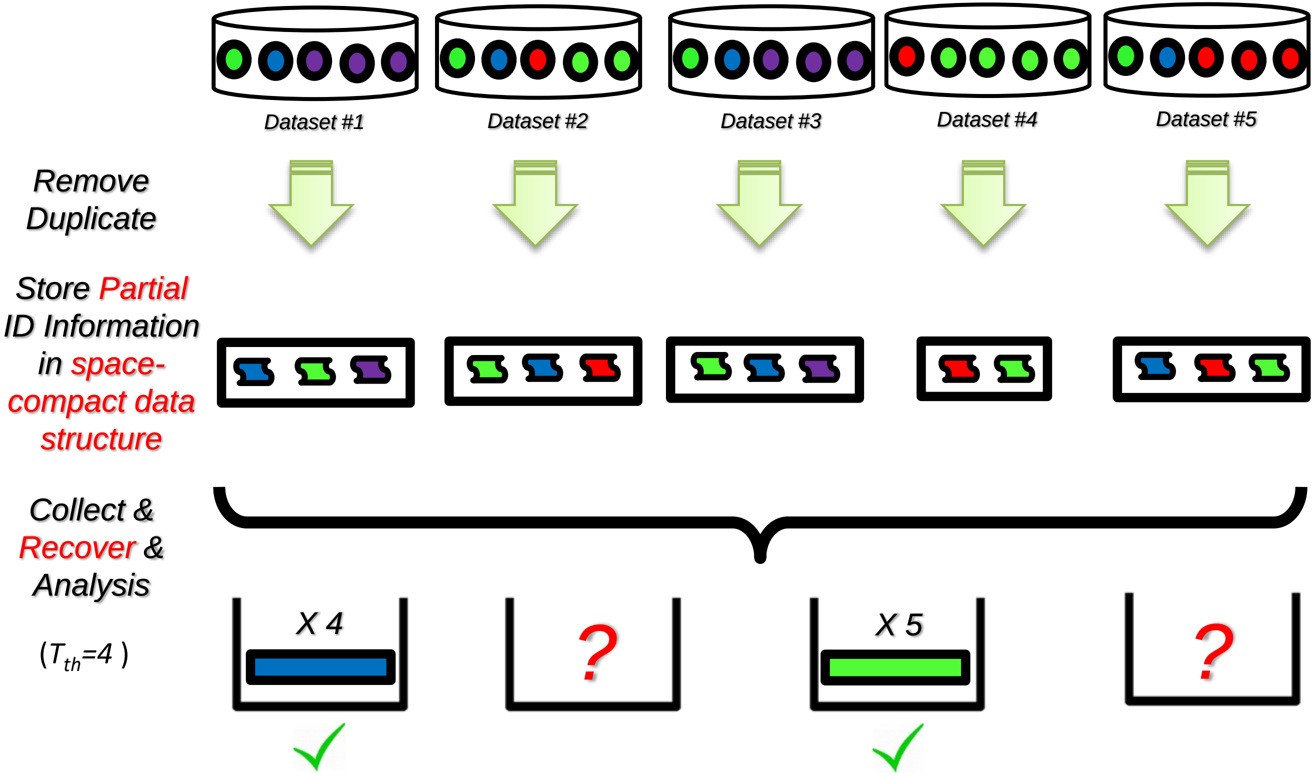
南京大学 NANJING UNIVERSITY

- **A naïve approach for persistent items identification**



02/ Key Idea of DISPERSE

- **Our Solution**: DIStributed PERSistent items SchemE (DISPERSE)



02/ Coding Cuckoo filter (CCF)

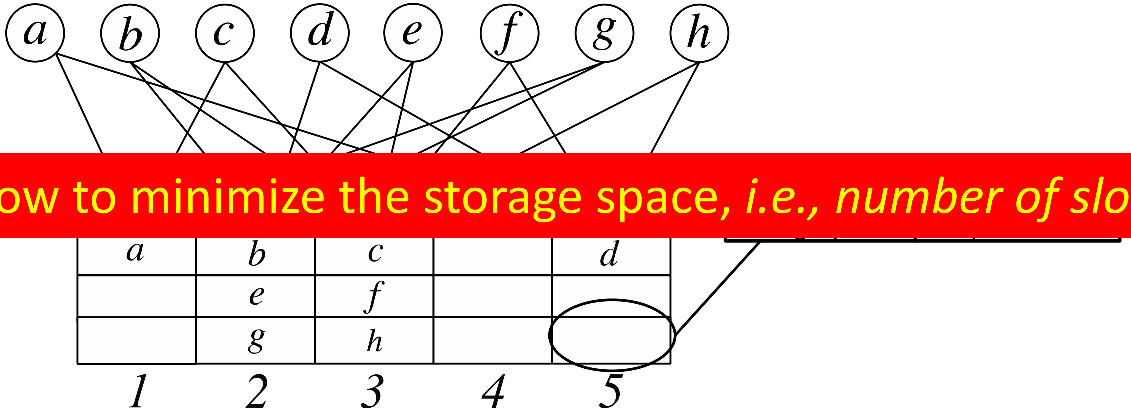
NANJING UNIVERSITY

- Structure: an array C_i of bucket, and each bucket consists of *w* slots.
- Space-compact data structure.
- Structure of a slot: (Hash-print, Raptor codes)

Name (notation)	Length	Description
Raptor Codes : $C_{iR}[x]$	*r*	Encoded codes for ID recovering, same for all mapped slots for an item, but different for different datasets.
Hash-print: $C_{iP}[x]$	*p*	Fingerprint-like info. generated by hashing for an item, same for mapped slots of an item.

02/ Recording Phase of CCF

- Inserting item into CCF
  - Each item has two associated candidate buckets.
  - Each item is to be placed in a slot.



Key Problem: How to minimize the storage space, *i.e., number of slots*?

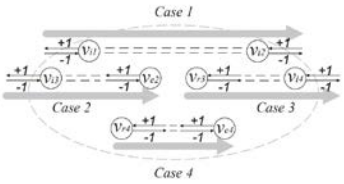
An example CCF with 5 buckets composed of 3 slots

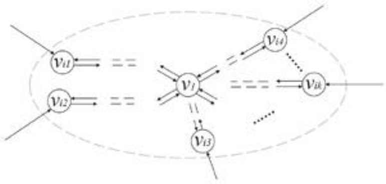
02/ Storage Space Optimization of CCF

- **Problem Formulation**: Min-max Cost Two-degree Matching (MCTM) Problem.

- **Solution**: Relationship Graph based Algo.
  - The proposed algorithm is proved to be optimal
  - O(|L| 2 + |L| · |R|) time complexity

Theorem 4.1: Algorithm 1 will terminate in finite steps, and its output is optimal.





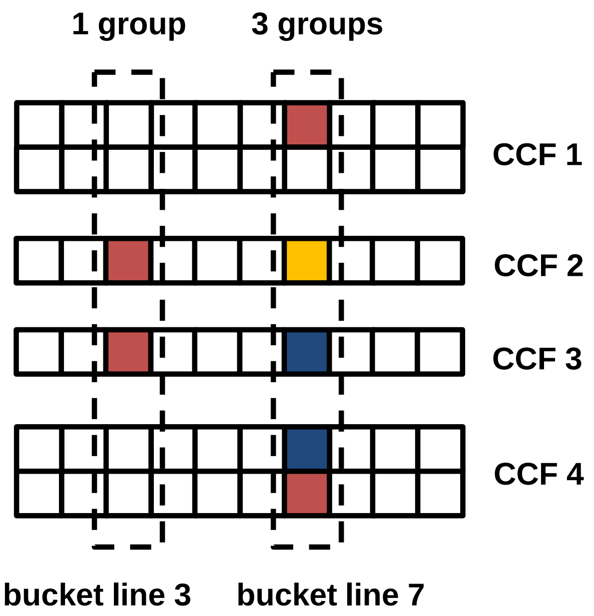
Algorithm 1: Relationship Graph based Algorithm

Input: A bipartite graph $G = (L \bigcup R, E)$ with any $v \in L$, $deg(v) = 2$.
Output: MCTM M on G with a min-max cost for vertices in R.

$M = \emptyset$.
while *there exists* $v_l \in L$ *not matched* **do**
  Randomly select $v_l \in L$ that is not matched and $v_r \in R$, where $(v_l, v_r) \in E$, add (v_l, v_r) into M.
Build a new graph $G' = (R, E')$, where E' is initialized to be $\emptyset$, based on the MCTM M as follows.
for *each* $v_l \in L$ *in* G *and* (v_l, v_{r_1}), $(v_l, v_{r_2}) \in E$ **do**
  if $(v_l, v_{r_1}) \in M$ **then**
    if $(v_{r_1}, v_{r_2}) \in E'$ **then**
      Increase the weight of (v_{r_1}, v_{r_2}) by 1.
    else
      Add a directed edge (v_{r_1}, v_{r_2}) into E' and set the edge weight to 1.
  else
    if $(v_{r_2}, v_{r_1}) \in E'$ **then**
      Increase the weight of (v_{r_2}, v_{r_1}) by 1.
    else
      Add a directed edge (v_{r_2}, v_{r_1}) into E' and set the edge weight to 1.
while *there exists* $v_r \in R$, *where* $deg^+(v_r) = \Delta^+(R)$ *and* $deg^+(v_r) > (\delta^+(\mathcal{R}(G', v_r)) + 1)$ **do**
  Use Breadth-First-Search algorithm to find a vertex v_{end} such that there exists a path p from v_r to v_{end} and $deg^+(v_{end}) = \delta^+(\mathcal{R}(G', v_r))$.
  for *each directed edge* $(v_i, v_{i+1}) \in p$ **do**
    Decrease the edge weight of (v_i, v_{i+1}) by 1 if it is larger than 1; otherwise remove it.
    Increase the edge weight of (v_{i+1}, v_i) by 1 if (v_{i+1}, v_i) exists; otherwise add edge (v_{i+1}, v_i) and set its weight to 1.
    Randomly select a vertex $v_l \in L$ in the original graph G such that $(v_l, v_i) \in M$, $(v_l, v_{i+1}) \notin M$, and (v_l, v_i), $(v_l, v_{i+1}) \in E$, remove the edge (v_l, v_i) from MCTM M and add (v_l, v_{i+1}) into M.
return *MCTM* M.

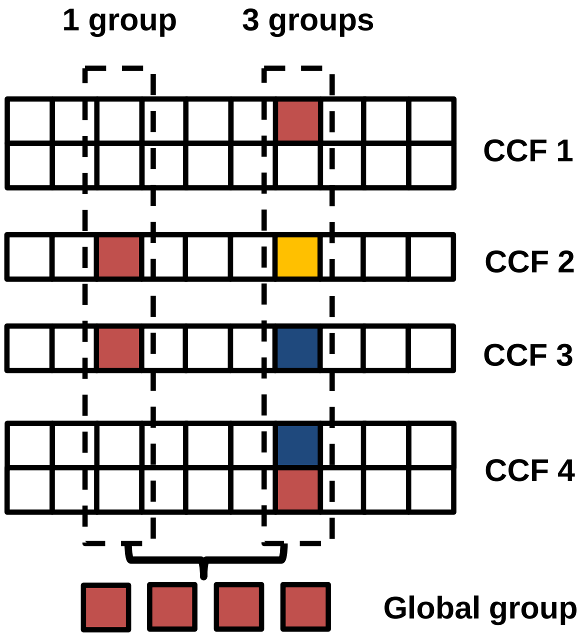
02/ Decoding Phase of CCF

- Isolate Raptor codes for different items
  - Align all CCFs and buckets of the same column are called bucket line
  - Divide the slots in each bucket line into groups according to fingerprints



02/ Decoding Phase of CCF

- Isolate Raptor codes for different items
  - Combine a group in a certain bucket line with another group having the same fingerprint at the associated bucket line
  - Decode Item IDs from the Raptor codes in global group



02/ Analysis – False Negative Rate

- Successfully recovering a persistent item
  - The probability of an item free from fingerprint mingling is given by P_{mf}
  - The probability of mingled item happens to have the same Raptor codes make items survive to be recovered is given by P_{ms}
  - The probability of successfully recovering an persistent item is P_{sr}, where w_t denote the percentage of items with t occurrences
  - The probability of failing to recover a persistent item is given by P_{FN}

$$\left.\begin{aligned}
P_{mf} &= \left[\left(1-\frac{2}{m}\right)+\frac{2}{m}\times\left(1-\frac{1}{2^p}\right)\right]^{\mathcal{N}-1} \approx \left(1-\frac{1}{m\times 2^{p-1}}\right)^{\mathcal{N}} \\
P_{ms} &= \left(\frac{2}{m}\times\frac{1}{2^p}\times\frac{1}{2^r}\right)^{\mathcal{N}-1} \approx \left(\frac{1}{m\times 2^{p+r-1}}\right)^{\mathcal{N}}. \\
P_{sr}(t) &= (P_{mf}+P_{ms})\times P_{ds}(r\cdot t;l) \\
P_{sr} &= \frac{\sum_{t=T_h}^{T} w_t\times P_{sr}(t)}{\sum_{t=T_h}^{T} w_t}.
\end{aligned}\right\} P_{FN} = 1 - P_{sr}$$

02/ Analysis – False Positive Rate

- Identifying a false persistent item
  - Due to mingling
  - The probability of recovering a false persistent item is given by P_{FP}

$$P_{FP} = \frac{1}{2^p} \times \frac{2}{m} = \frac{1}{m \times 2^{p-1}}$$

02/ Evaluation

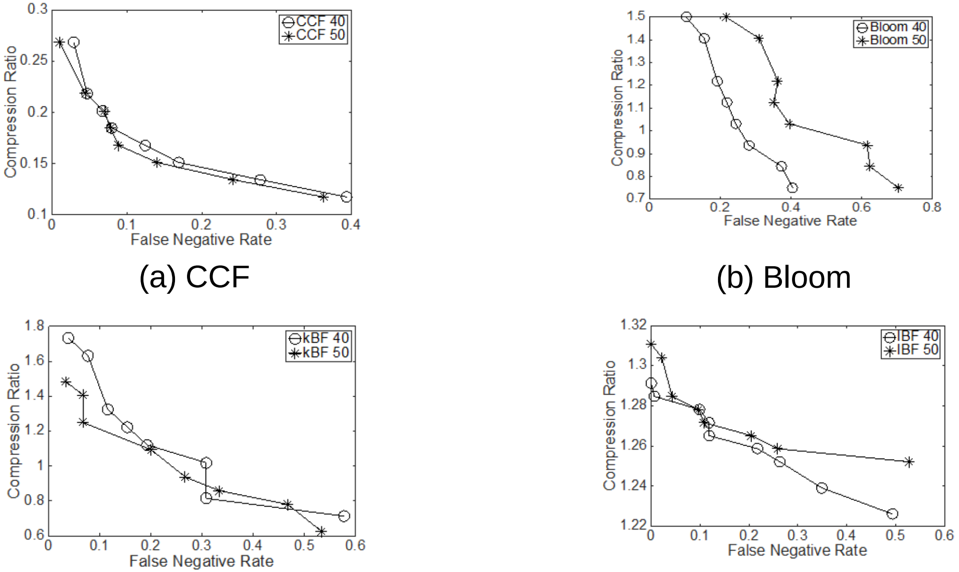
- Parameter Settings:
  - # of datasets T : 60
  - # Threshold T_{th}: 40 and 50
  - Data trace: CAIDA

- Evaluation metrics:
  - False Negative Rate (FNR)
  - False Positive Rate (FPR)
  - Compression ratio

- Side-by-side comparison:
  - Bloom
  - kBF
  - IBF

02/ Compression Ratio vs. FNR

- Our results show that our scheme can achieve 7.9, 5.7, and 6.6 times performance gains, respectively, in terms of compression ratio.



(a) CCF

(b) Bloom

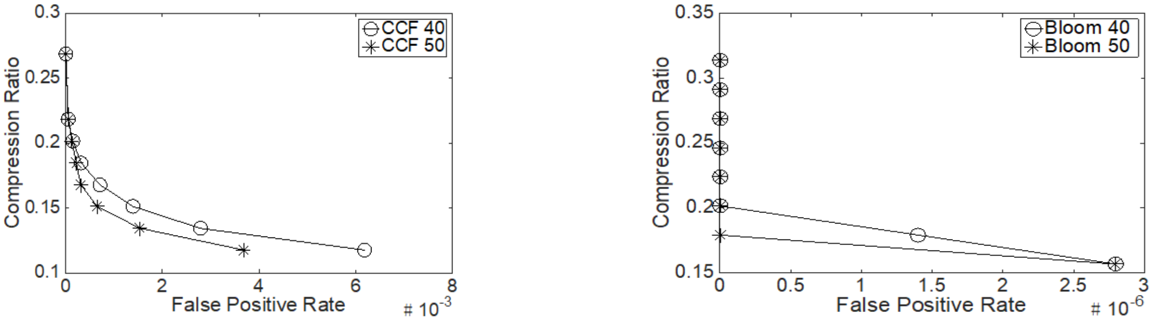
(c) kBF

(d) IBF

Space Compression Ratio vs. FNR

02/ Compression Ratio vs. FPR

- Our results show that only Bloom achieves a better FPR than CCF, which is, however, at the cost of hardly recovering any items.



(a) CCF (b) Bloom

Compression Ratio vs. FPR

02/ Conclusion

- Propose the first solution for finding persistent items among distributed datasets
- Propose to encode item ID to cut down the communication cost
- Propose an probabilistic data structure to store encoded items
- Conduct solid simulations for evaluation

02/ Related Publications

- [INFOCOM'18] **Haipeng Dai**, Meng Li and Alex X. Liu. "Finding Persistent Items in Distributed Datasets". (**CCF A**)

- [TON'20] **Haipeng Dai**, Meng Li, Alex X. Liu, Jiaqi Zheng and Guihai Chen. "Finding Persistent Items in Distributed Datasets". (**CCF A**)

Outline

PART 01 Background

PART 02 Ming Items with Complex Pattern

- Persistent Items

PART 03 Utilizing Item Inherent Information

- Negative Items and Distribution

NANJING UNIVERSITY

Hash Adaptive Bloom Filter (ICDE'21)

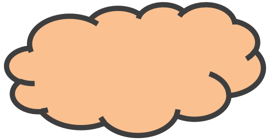
Rongbiao Xie[1], Meng Li[1], Zheyu Miao[2,3], Rong Gu[2]*, He Huang[3], Haipeng Dai[1]* and Guihai Chen[1].

[1] State Key Laboratory for Novel Software Technology, Nanjing University, China.
[2] Zhejiang University
[3] Alibaba Group
[4] Soochow University

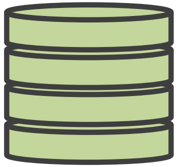
03/ Motivation

NANJING UNIVERSITY

Membership testing problem is a fundamental problem in numerous applications.



Big Data



Database

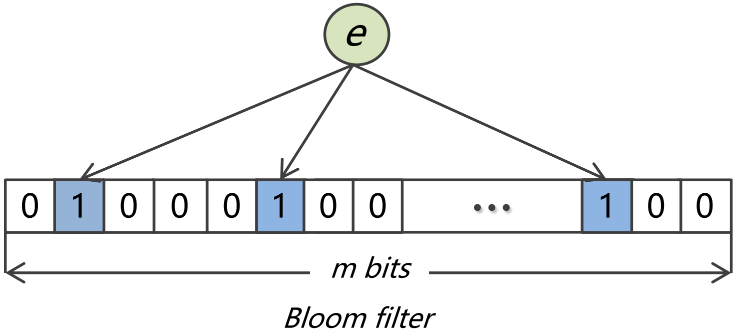
Network Security

Best solution: filters

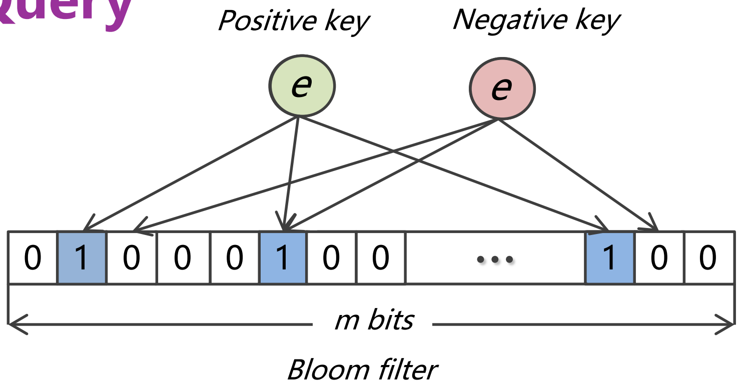
- High accuracy
- Memory efficient
- Fast construction and query

03/ Background – Bloom Filter

Insertion



Query



03/ Background

南京大學
NANJING UNIVERSITY

Availability of Negative Keys

- **Publicly available**, like the online malicious IP address statistics for intrusion detection.

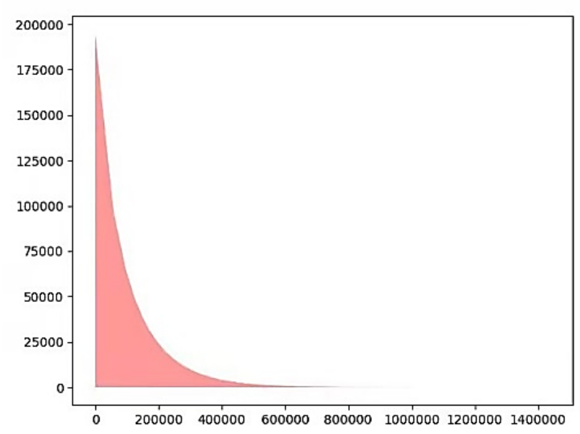
uribl

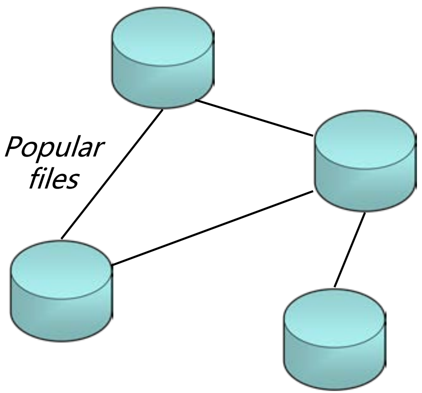
- **Obtained by cache**, for LSM-based k-v stores, frequently failed queries with heavy I/O overhead can be cached to reduce extra disk accesses.

RocksDB

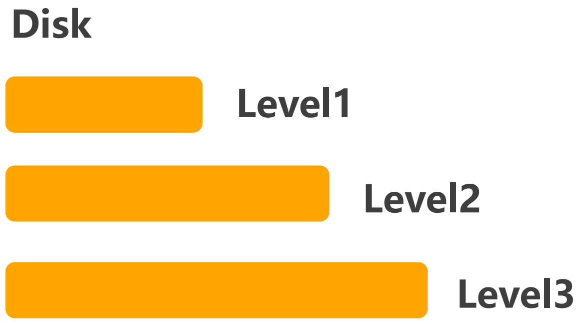
03/ Background

Further Scenario： Negative Keys Follow a Skewed Cost Distribution





Internet traffics



LSM-based k-v stores

03/ Prior filters' limitations

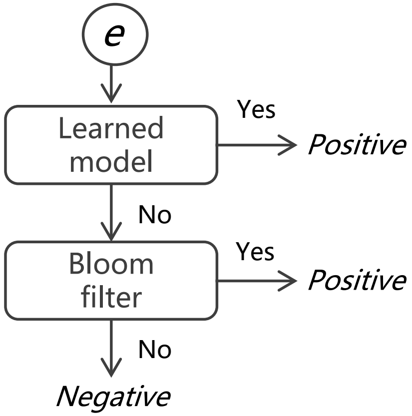
NANJING UNIVERSITY

For Bloom filter, Xor filter

- Insensitivity for negative keys and the cost distribution

For Learning-based Filters, like Learned Bloom filter

- The accuracy of learned model is hard to be guaranteed
- The training and query phase is much more expensive



For Cost-based Filters, like Weighted Bloom filter

- Elements need to carry their cost information all the time

03/ Design goal

Adaptability of Filters

- Sensitive to the negatives keys.
- Learning to be adaptive to the skewed cost distribution
- Applicable to various data like Bloom filter

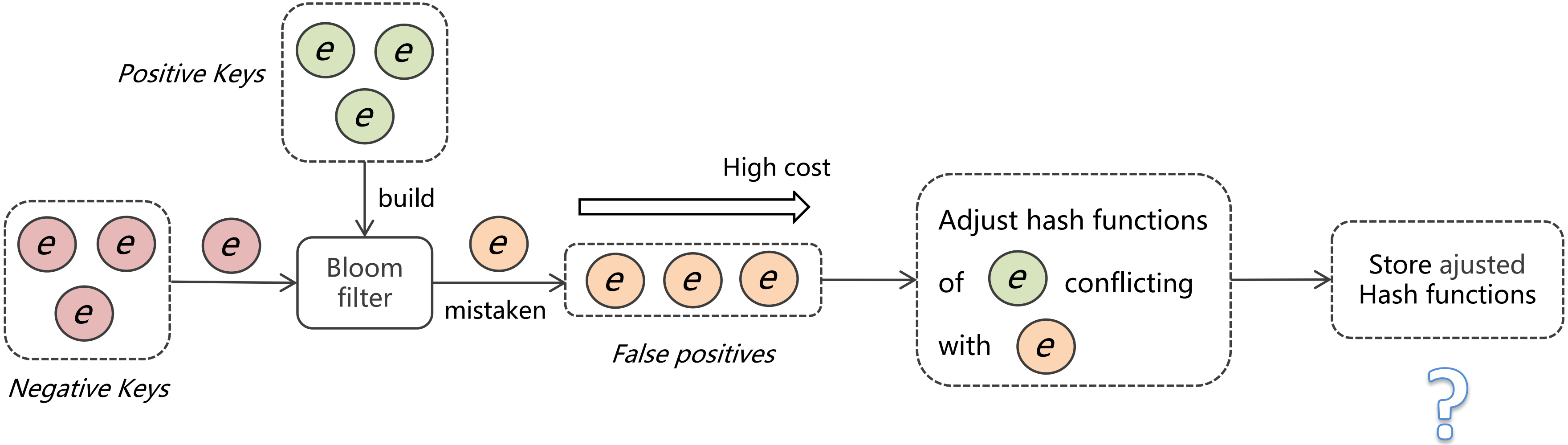
Construction and query speed

- Approaching the speed of Bloom filter

03/ Rationale

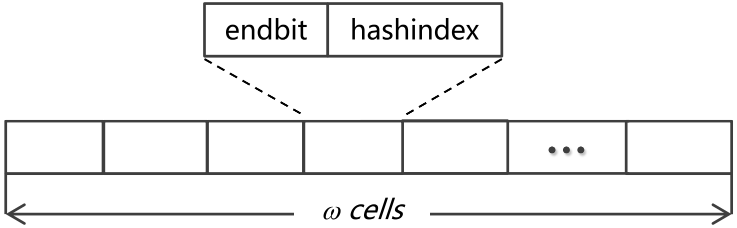
Key Idea: Customizing the hash functions for positive keys to avoid conflicting with high-cost negative keys.

Procedure:



03/ Architecture - HashExpressor

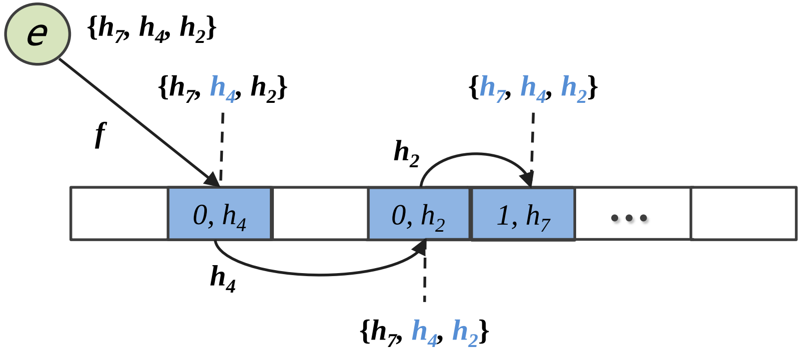
Data Structure



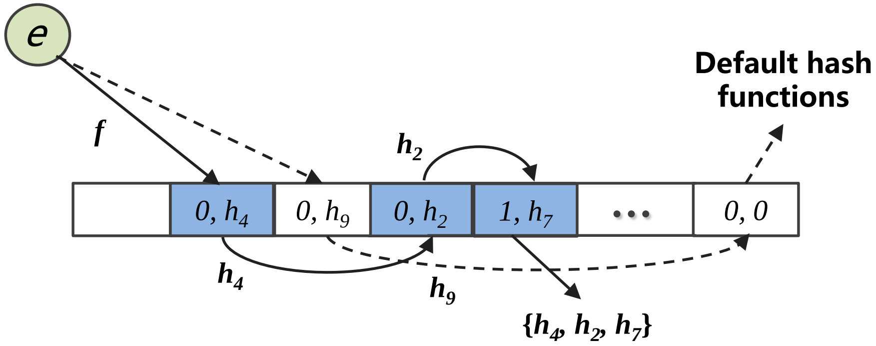
- **endbit:** whether the query comes from an adjusted positive key
- **hashindex**: The index of a hash function

Zero FNR
Small FPR

Insertion



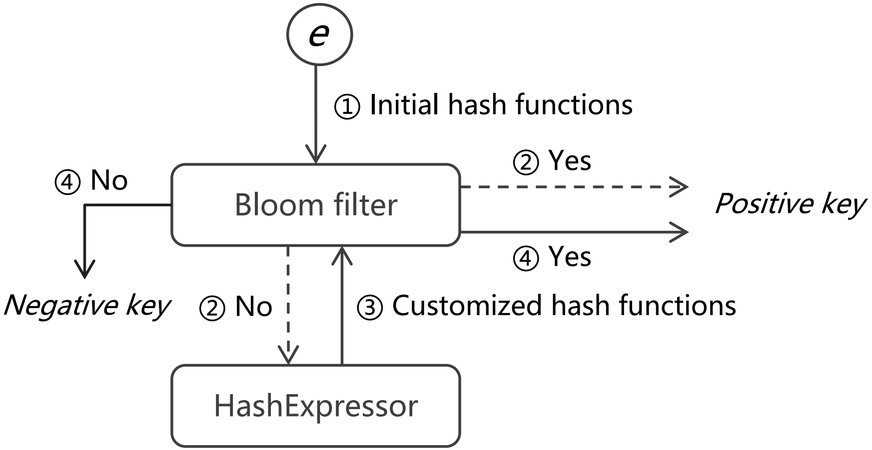
Query



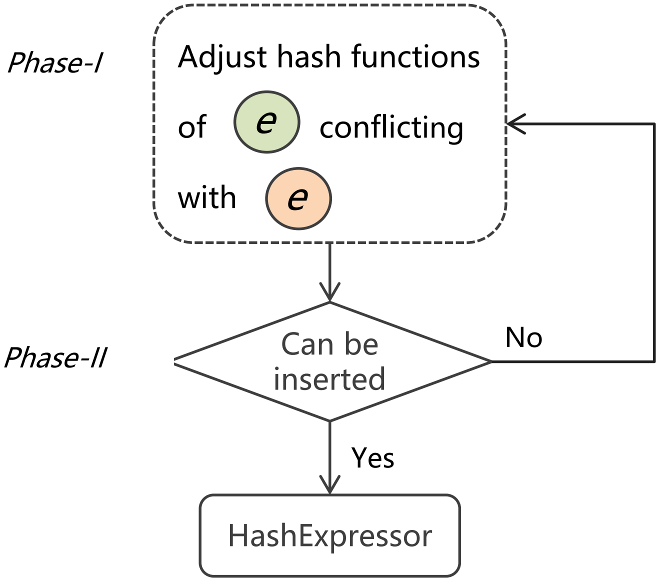
03/ Architecture - HABF

NANJING UNIVERSITY

Zero-FNR Query



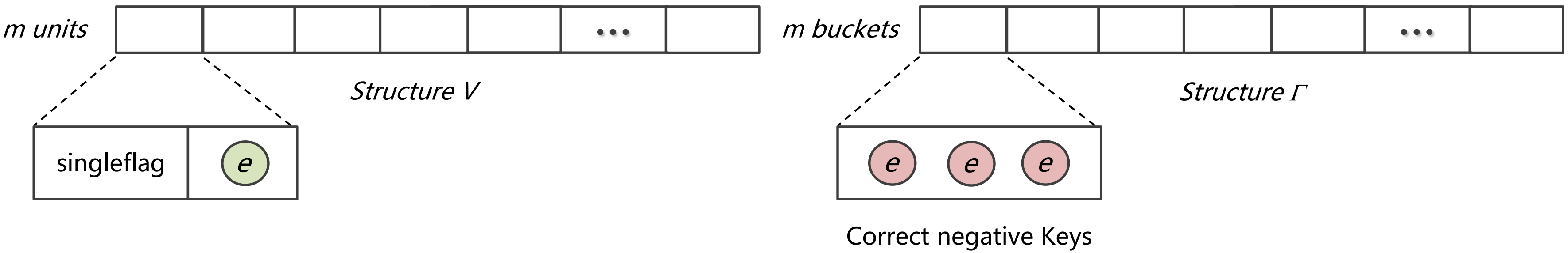
Two-phase Construction



03/ Architecture - HABF

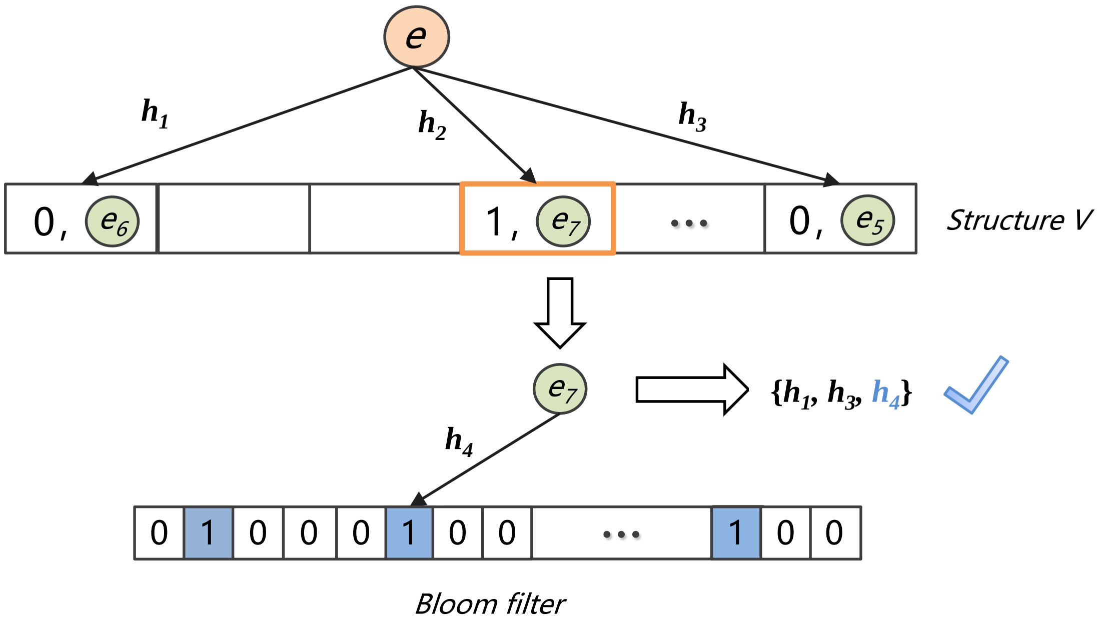
How to choose e to be adjusted, further which hash functions should we adjust, and adjust to which?

- **Two runtime auxiliary structures**



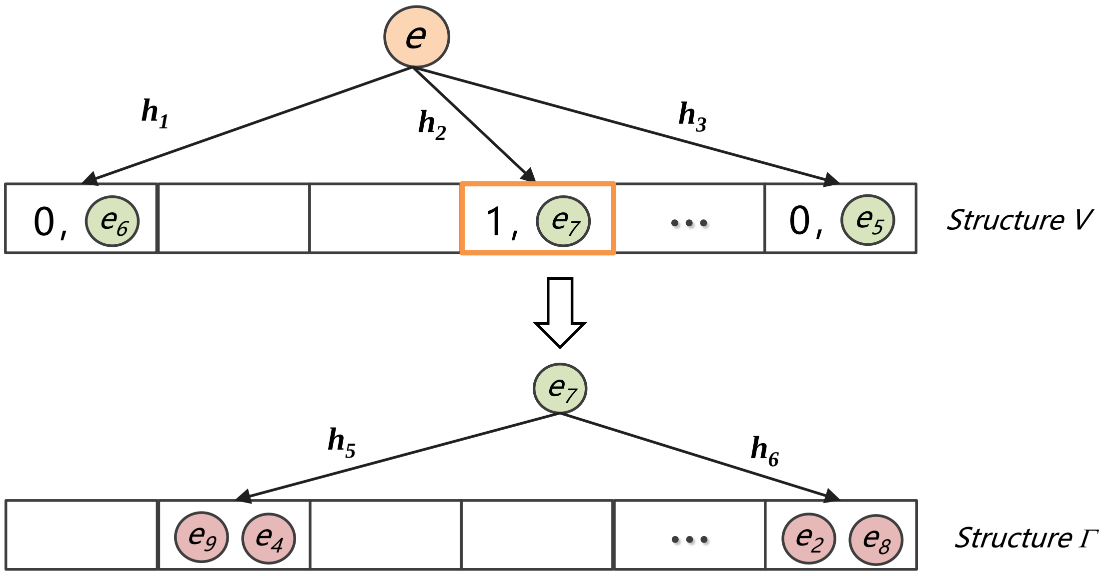
03/ Case Study

Default initial hash functions h_1, h_2, h_3 *Candidate hash functions* h_4, h_5, h_6



03/ Case Study

Default initial hash functions h_1, h_2, h_3 *Candidate hash functions* h_4, h_5, h_6



No confliction ⇒ $\{h_1, h_3, h_5\}$ ✓

e_2 is conflicted and cost(e_2) > cost(e) ✗

03/ Experiments - Setup

NANJING UNIVERSITY

Dataset:

- Shalla’s Blacklists: A URL dataset with evident characteristics
- YCSB: A benchmark for databases, and we modified its uniform generator to generate 24,074,812 keys

Dataset	Positive keys	Negative keys
Shalla	1,491,178	1,435,527
YCSB	12,500,611	11,574,201

03/ Experiments - Setup

NANJING UNIVERSITY

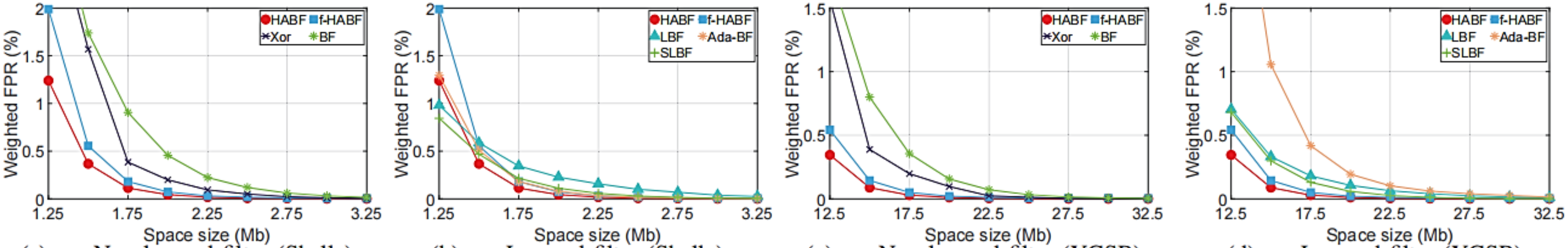
Metrics:

- Weighted FPR = $\dfrac{\text{Overall cost of false positives}}{\text{Overall cost of negative keys}}$

Comparison algorithms:

- **Non-learned Filters:** BF, Xor, WBF
- **Learned Filters:** LBF, Ada-BF, SLBF

03/ Under Uniform Distribution

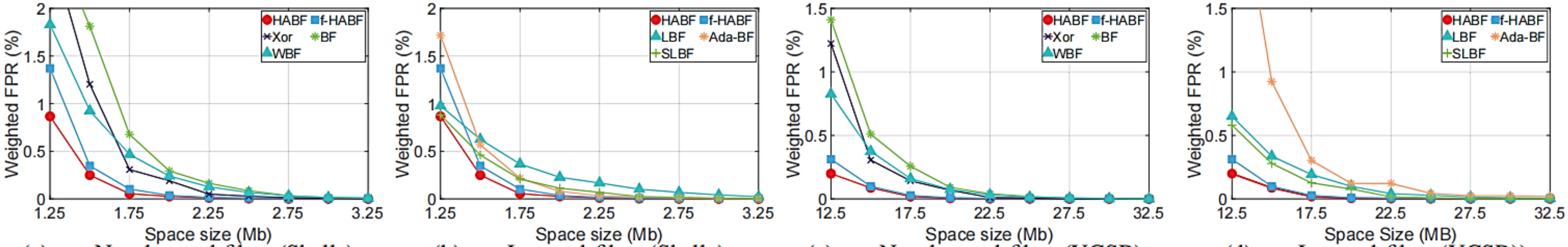


(a) vs. Non-learned filter (Shalla) (b) vs. Learned filter (Shalla) (c) vs. Non-learned filter (YCSB) (d) vs. Learned filter (YCSB)

- When keys have evident characteristics, HABF will use less space if a low weighted FPR is required.
- When the key schema is approximately random, HABF has the smallest weighted FPR for all our space settings.

03/ Under Skewed Distribution

南京大學
NANJING UNIVERSITY

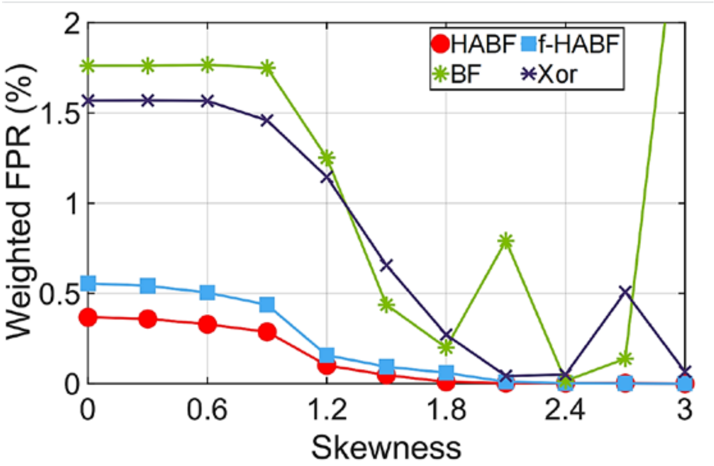


(a) vs. Non-learned filter (Shalla) (b) vs. Learned filter (Shalla) (c) vs. Non-learned filter (YCSB) (d) vs. Learned filter (YCSB))

- HABF always has the smallest weighted FPR under all the space settings

03/ Effect of Skewness

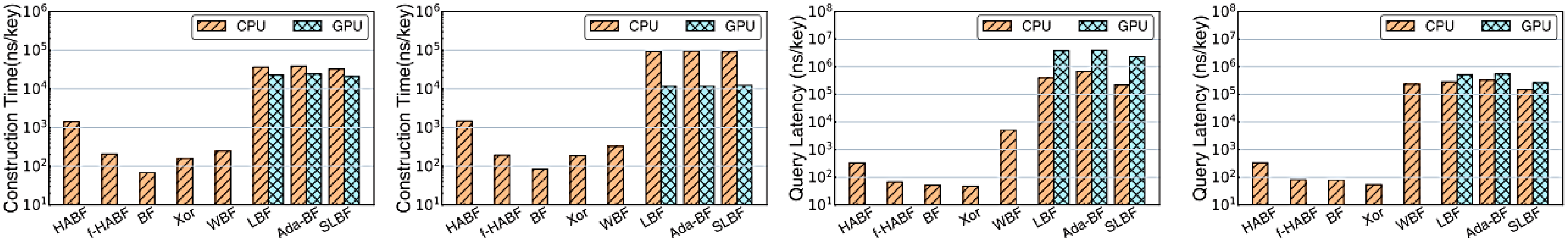
NANJING UNIVERSITY



- HABF and f-HABF continue to decrease steadily but for BF and Xor, the weighted FPRs show great fluctuations.

03/ Construction and Query Time

南京大學
NANJING UNIVERSITY



(a) Construction time (Shalla) (b) Construction time (YCSB) (c) Query time (Shalla) (d) Query time (YCSB)

- The construction time of HABF and f-HABF are around 19.0× and 2.7× larger than that of BF, respectively.

- The query time of HABF and f-HABF are around 5.4× and 1.2× than that of BF, respectively.

03/ Conclusion

- We study how to improve the performance when some negative keys and their costs are available
- We propose a novel framework named HABF to customize the hash functions for positive keys to avoid high-cost negative keys
- Experimental results: the performance of our algorithm is much better than related algorithms

03/ Related Publications

- [ICDE'21] Rongbiao Xie, Meng Li, Zheyu Miao, Rong Gu*, He Huang, **Haipeng Dai*** and Guihai Chen. "Hash Adaptive Bloom Filter". (**CCF A**)

- [VLDBJ] Meng Li, Rongbiao Xie, Deyi Chen, Rong Gu, He Huang, **Haipeng Dai***, Wanchun Dou and Guihai Chen*. "A Pareto Optimal Filter Family with Hash Adaptivity". (**CCF A, under review**)

Q & A

Haipeng Dai (戴海鹏)

Homepage: http://cs.nju.edu.cn/daihp/

Email: haipengdai@nju.edu.cn

Phone: 18951991961