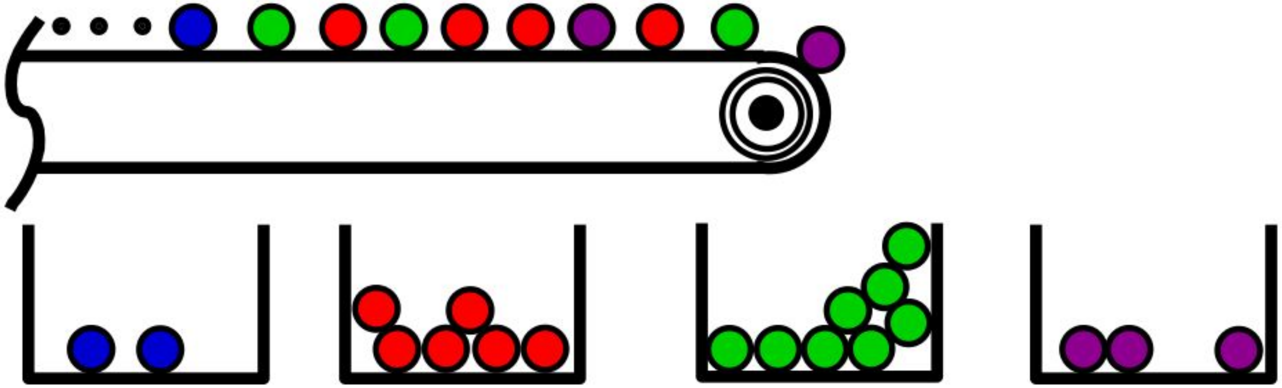
Buffered Count-Min Sketch on SSD: Theory and Experiments

Mayank Goswami, Dzejla Medjedovic, Emina Mekic, and **Prashant Pandey**

ESA 2018, Helsinki, Finland

The heavy hitters problem (HH(k))

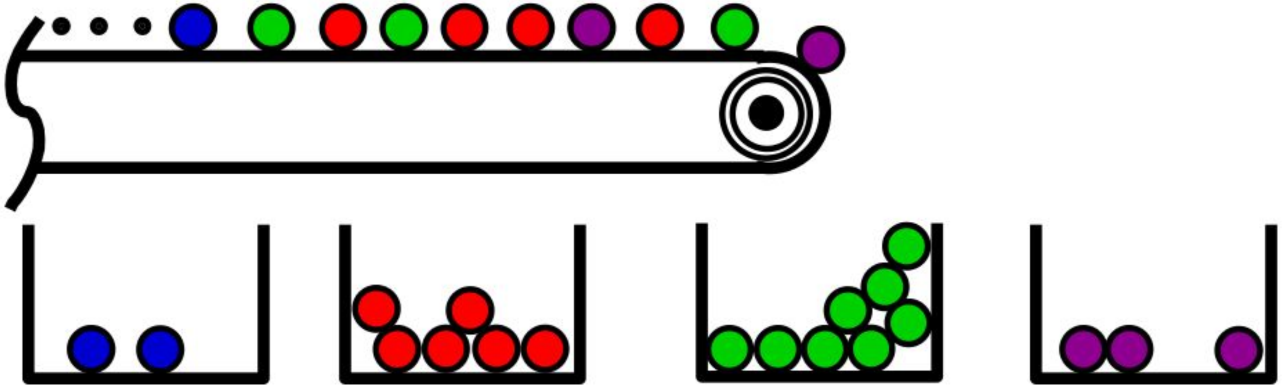
- Given stream of N items, report items whose frequency $\geq \varphi N$.
- General solution is “hard” in small space.
- Approximate solutions are employed.



Picture taken from: http://dmac.rutgers.edu/Workshops/WGUnifyingTheory/Slides/cormode.pdf.

The approximate heavy hitters problem (ε-HH(k))

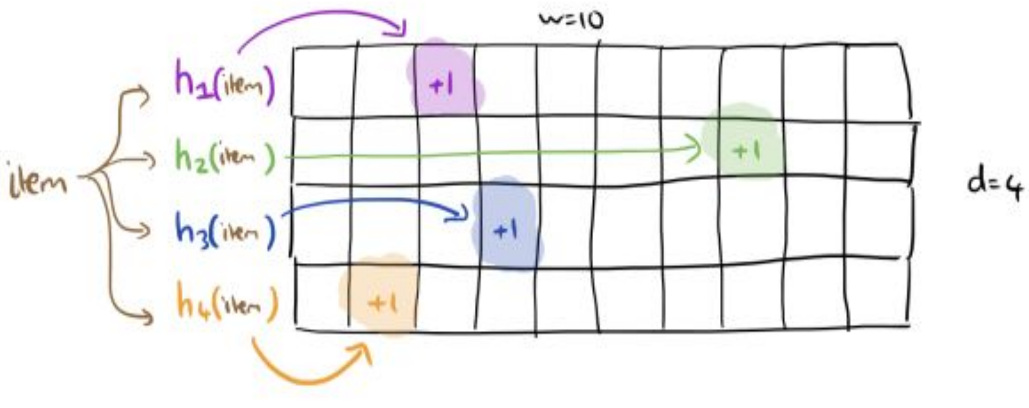
- Find all items with count $\geq \varphi N$, none with count $< (\varphi - \varepsilon)N$
- Error $0 < \varepsilon < 1$, e.g., $\varepsilon = 1/1000$
- Related problem: **estimate each frequency with error $\pm\varepsilon N$**



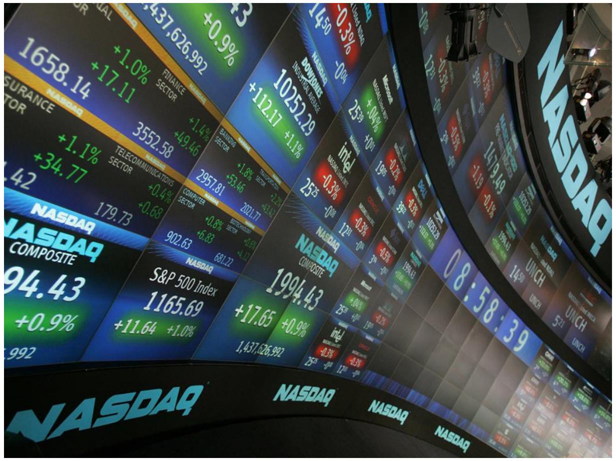
Picture taken from: http://dmac.rutgers.edu/Workshops/WGUnifyingTheory/Slides/cormode.pdf.

Sketch data structures

- A sketch is a **compact representation** of a data stream.
- It is typically **lossy**.
- It is useful to **approximately answer analytical questions** about data stream. E.g.,
    - Heavy hitters
    - Quantile queries
    - Inner-product queries

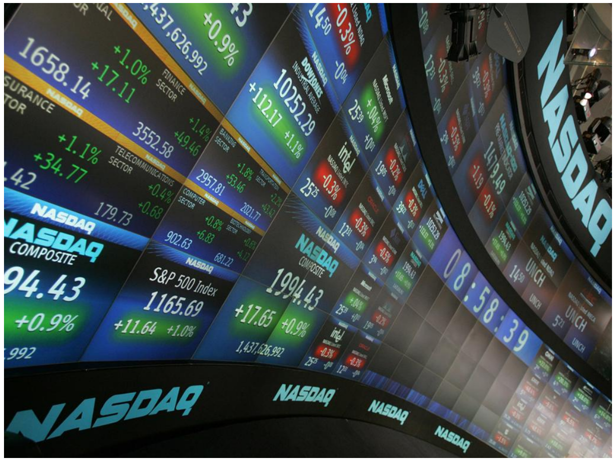


Sketches are at the heart of stream analyses



Financial market

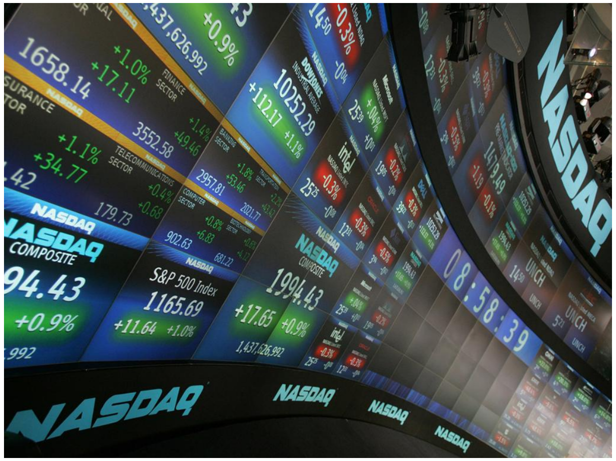
Sketches are at the heart of stream analyses



Financial market

Sensor networks

Sketches are at the heart of stream analyses

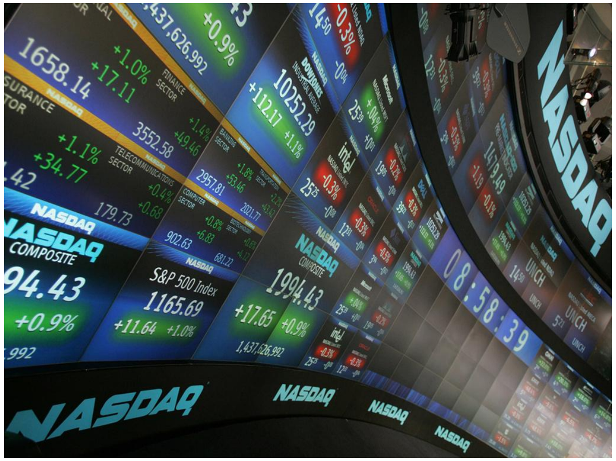


Financial market

IP traffic

Sensor networks

Sketches are at the heart of stream analyses



Financial market

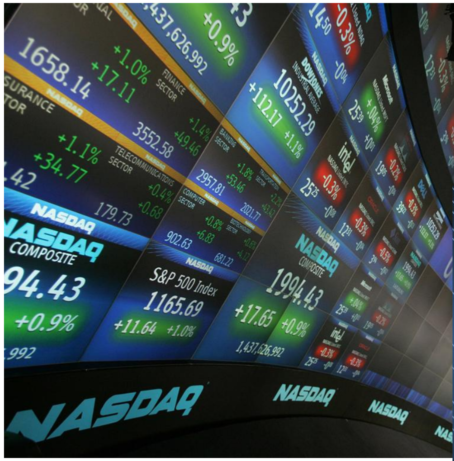
IP traffic

Sensor networks

automated data mining survey
responses co ter transcripts
qualatativ root cause
classificati text analysis insights
ad-hoc an is product
reviews se t vo f the
customer dashboards consume
trends ad-hoc analysis early warning

Text analysis

Sketches are at the heart of stream analyses



Financial mark

IP traffic

Cyber security

Sensor networks

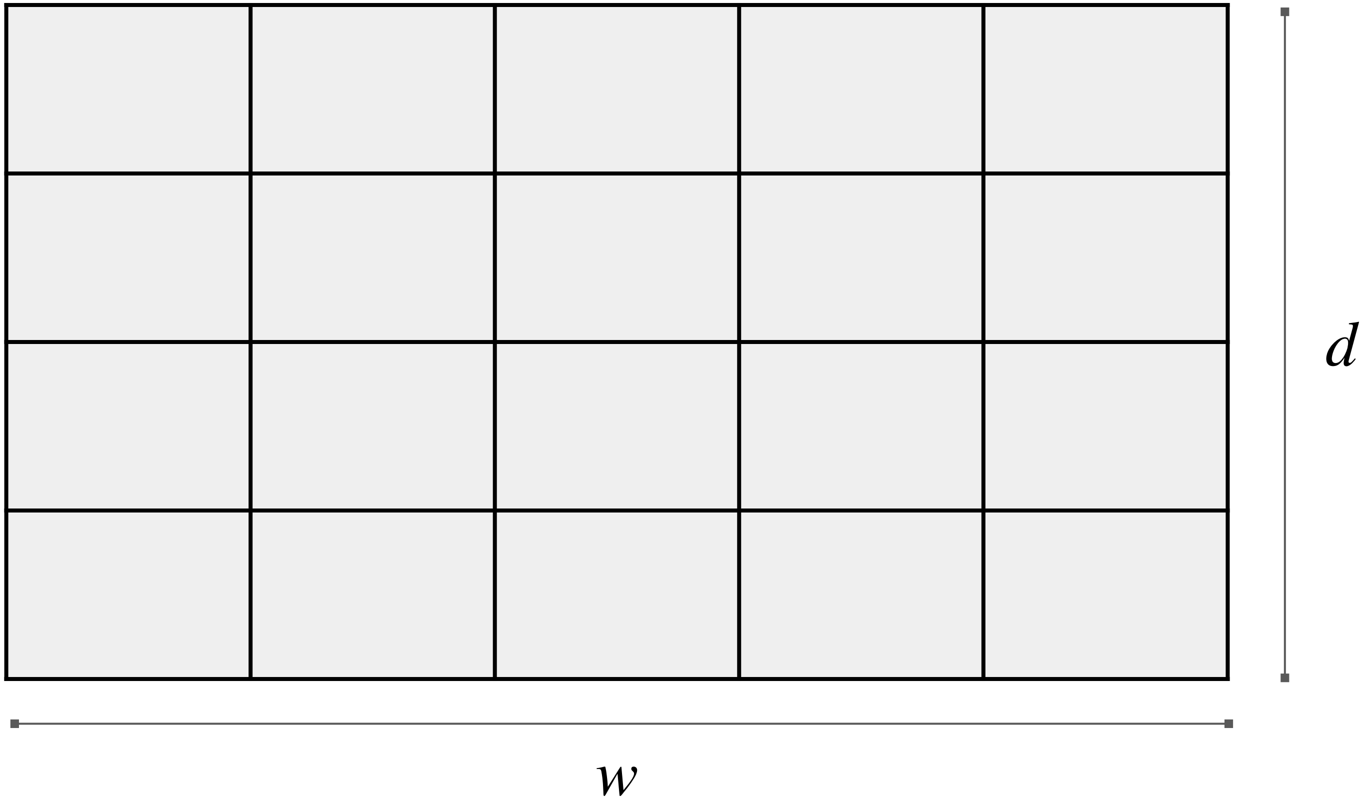
mated data mining survey
responses co ter transcripts
qualatativ root cause
classificati text analysis insights
ad-hoc an is product
reviews se t vo f the
customer dashboards consume
trends ad-hoc analysis early warning

Text analysis

In this talk:

- The **buffered count-min sketch (BCMS)**, an SSD-based sketch data structure.
  - The BCMS scales efficiently to large datasets keeping the total estimation error bounded.
- **Theoretical analysis** of the BCMS for:
  - Update and estimate times on SSD
  - Bounded error
- **Experimental** evaluation of the BCMS.

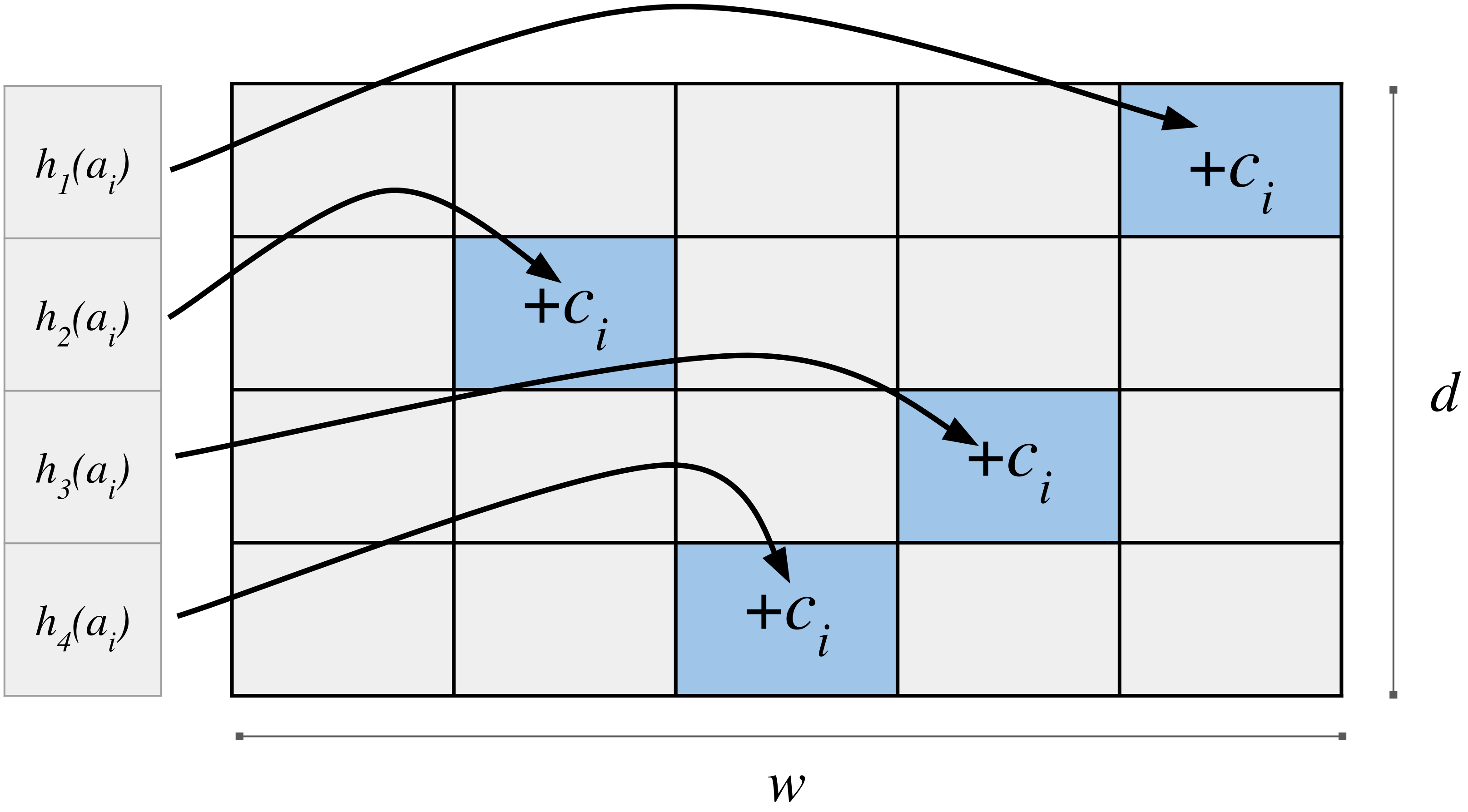
Count-min sketch (CMS)[1]



A CMS consists of a 2-D counter-array of depth *d* and width *w* and *d* hash functions.

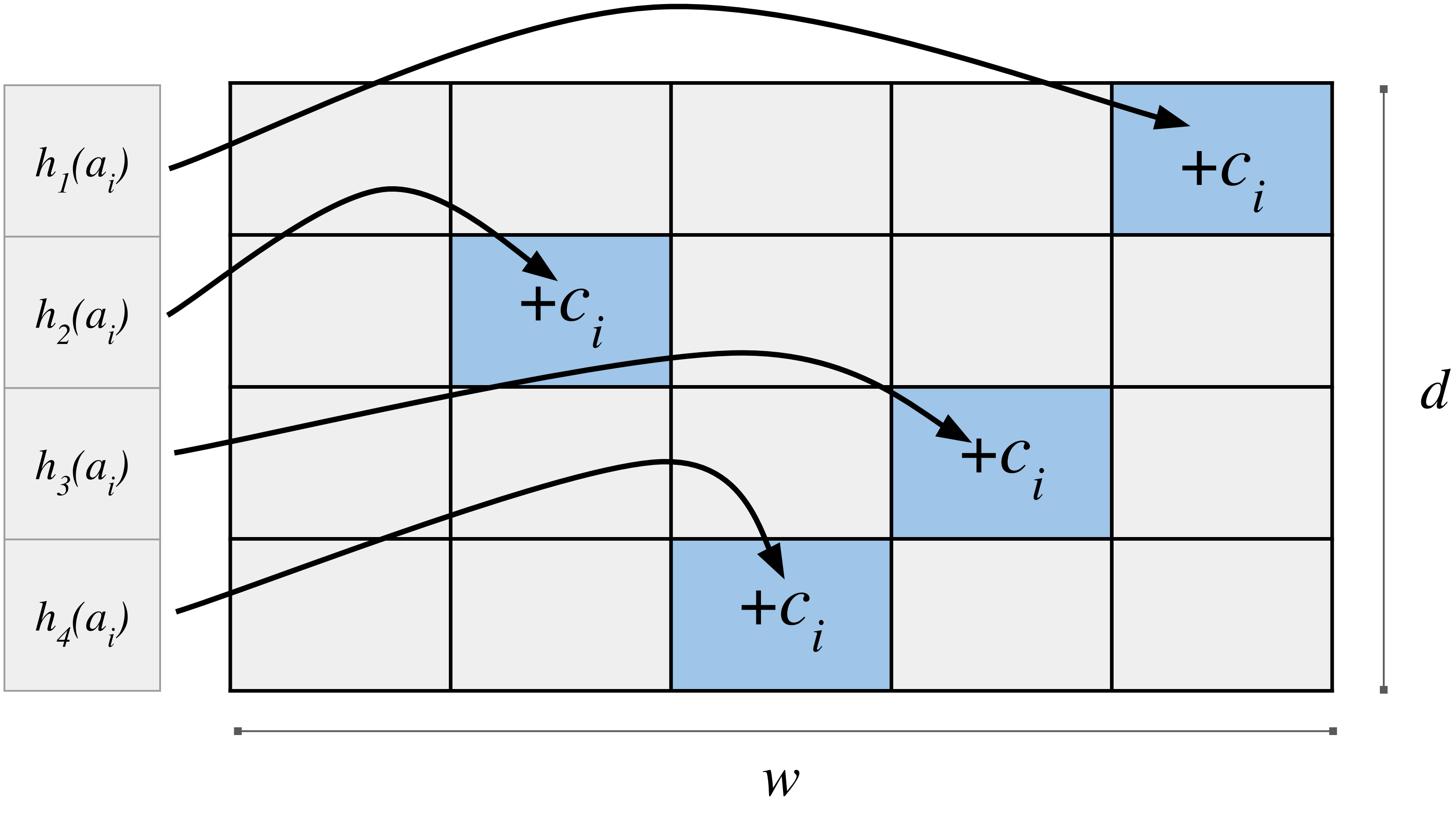
[1]. Graham Cormode and S. Muthukrishnan. An improved data stream summary: The count-min sketch and its applications. Journal. of Algorithms.

Count-min sketch: update



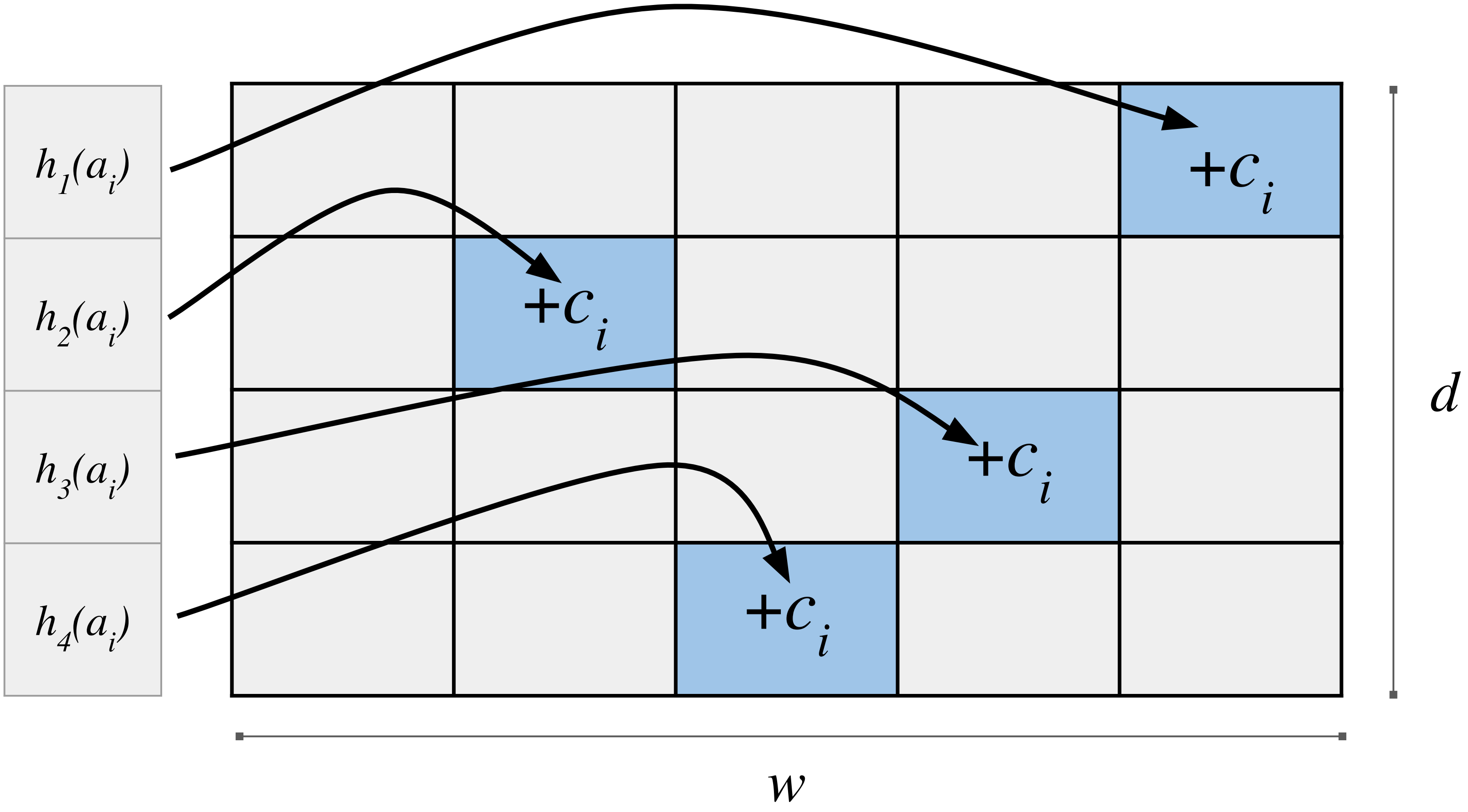
UPDATE(a_i, c_i)

Count-min sketch: estimate



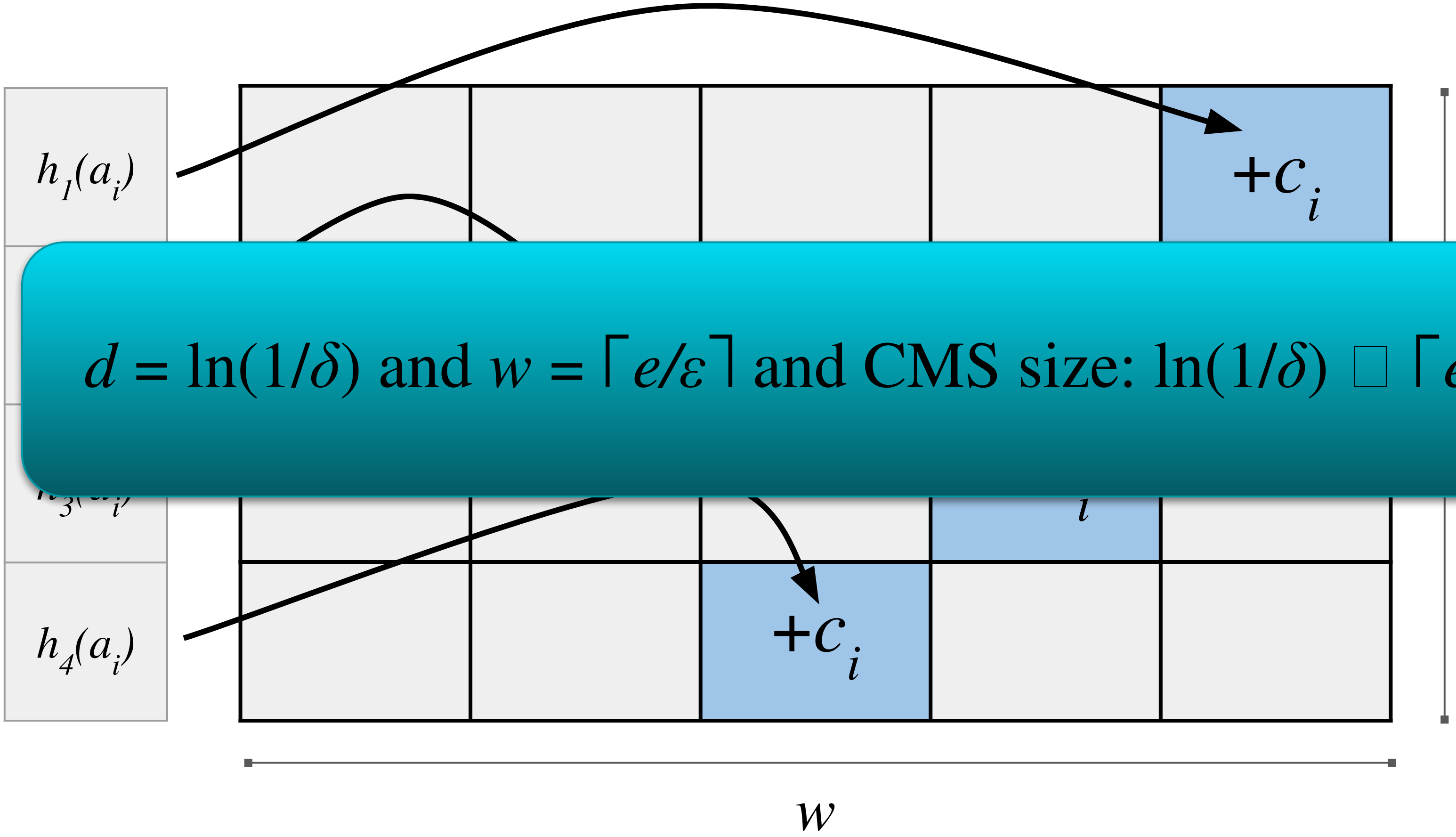
$$\text{ESTIMATE}(a_i) = \text{MIN}_{1 \le i \le d}(C_i)$$

Count-min sketch: analysis



We want estimation error within the range εN, with probability at least $1 - \delta$, i.e., $\Pr[\text{Error}(q) > \varepsilon N] \leq \delta$.

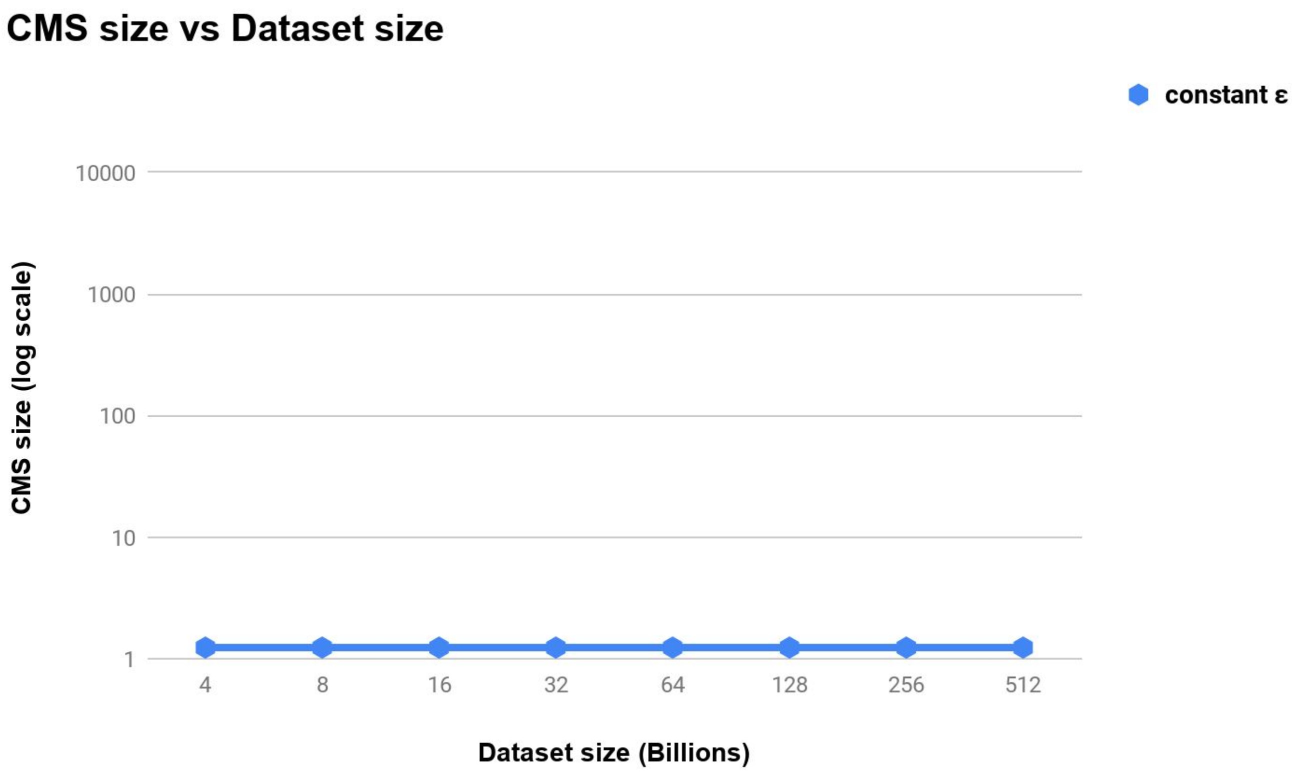
Count-min sketch: analysis



$d = \ln(1/\delta)$ and $w = \lceil e/\varepsilon \rceil$ and CMS size: $\ln(1/\delta)$ □ $\lceil e/\varepsilon \rceil$

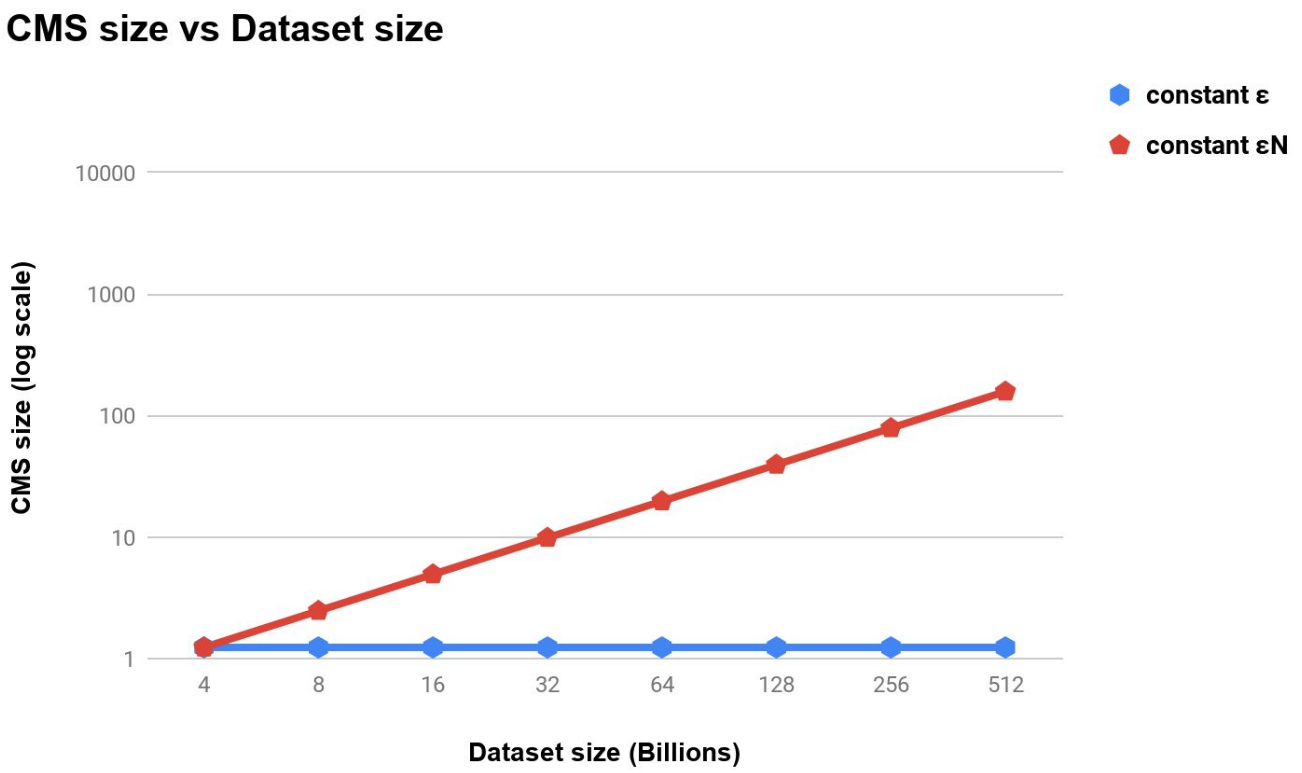
We want estimation error within the range εN, with probability at least $1 - \delta$, i.e., $\Pr[\text{Error}(q) > \varepsilon N] \leq \delta$.

CMS size vs dataset size when ε is constant



CMS size remains constant when ε is constant.

CMS size vs dataset size when εN is constant



CMS size grows linearly with dataset size.

The count-min sketch size grows with data set size

Consider the example,

N = 2^{30}, where overestimate 512 (ε = 2^{-21}) with 99.9% certainty (δ = 0.001), then

The count-min sketch size grows with data set size

Consider the example,

$N = 2^{30}$, where overestimate 512 ($\varepsilon = 2^{-21}$) with 99.9% certainty ($\delta = 0.001$), then

CMS size ~3.36 GB (assuming 4 Byte counters).

The count-min sketch size grows with data set size

Consider the example,

$N = 2^{30}$, where overestimate 512 ($\varepsilon = 2^{-21}$) with 99.9% certainty ($\delta = 0.001$), then

CMS size ~3.36 GB (assuming 4 Byte counters).

Now, suppose we want to keep the **overestimate same but double the data set size**, then

The count-min sketch size grows with data set size

Consider the example,

$N = 2^{30}$, where overestimate 512 ($\varepsilon = 2^{-21}$) with 99.9% certainty ($\delta = 0.001$), then

CMS size ~3.36 GB (assuming 4 Byte counters).

Now, suppose we want to keep the **overestimate same but double the data set size**, then

CMS size ~6.72 GB.

The count-min sketch size grows with data set size

Consider the example,

$N = 2^{30}$, where overestimate 512 ($\varepsilon = 2^{-21}$) with 99.9% certainty ($\delta = 0.001$), then

CMS size ~3.36 GB (assuming 4 Byte counters).

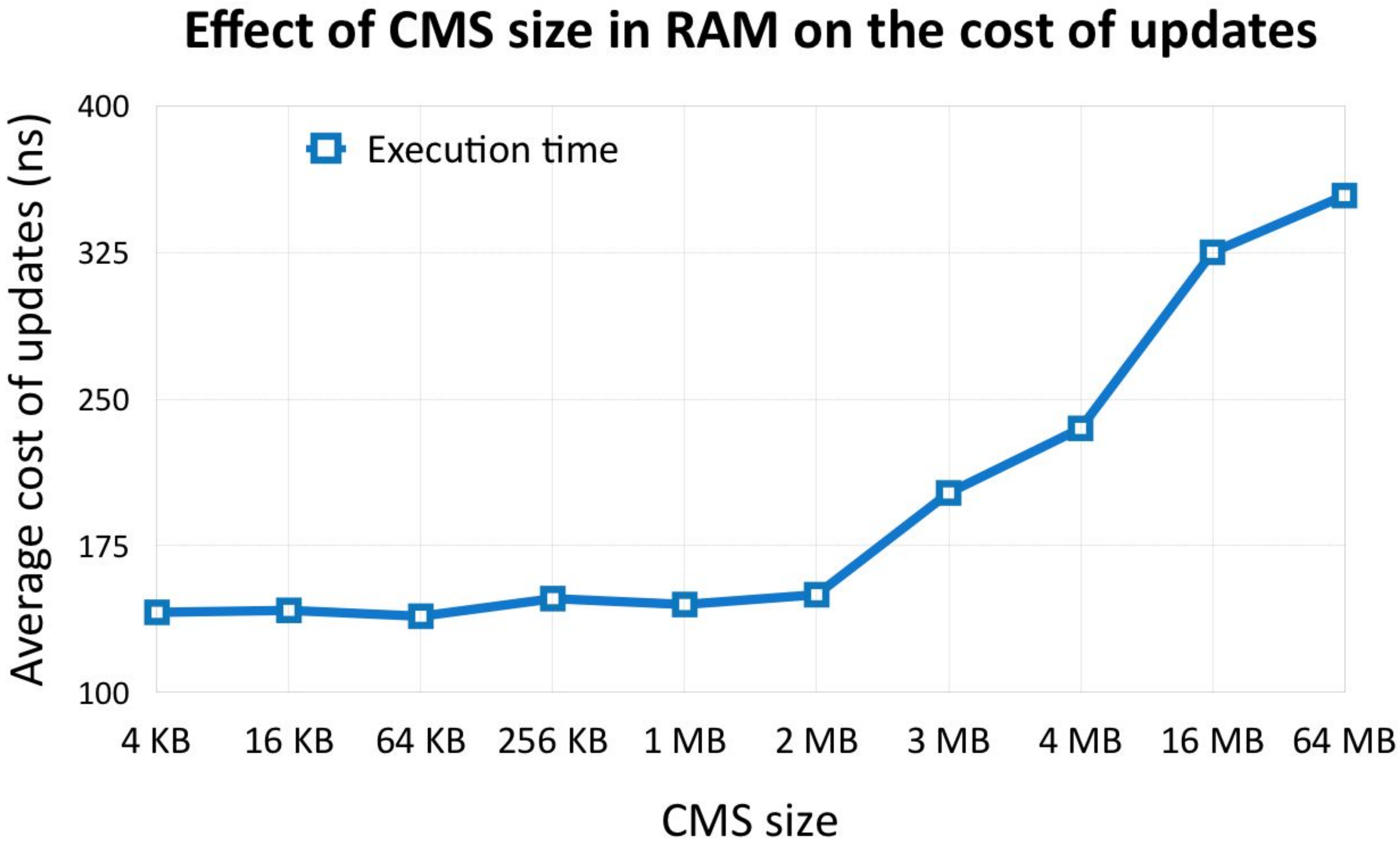
Now, suppose we want to keep the **overestimate same but double the data set size**, then

CMS size ~6.72 GB.

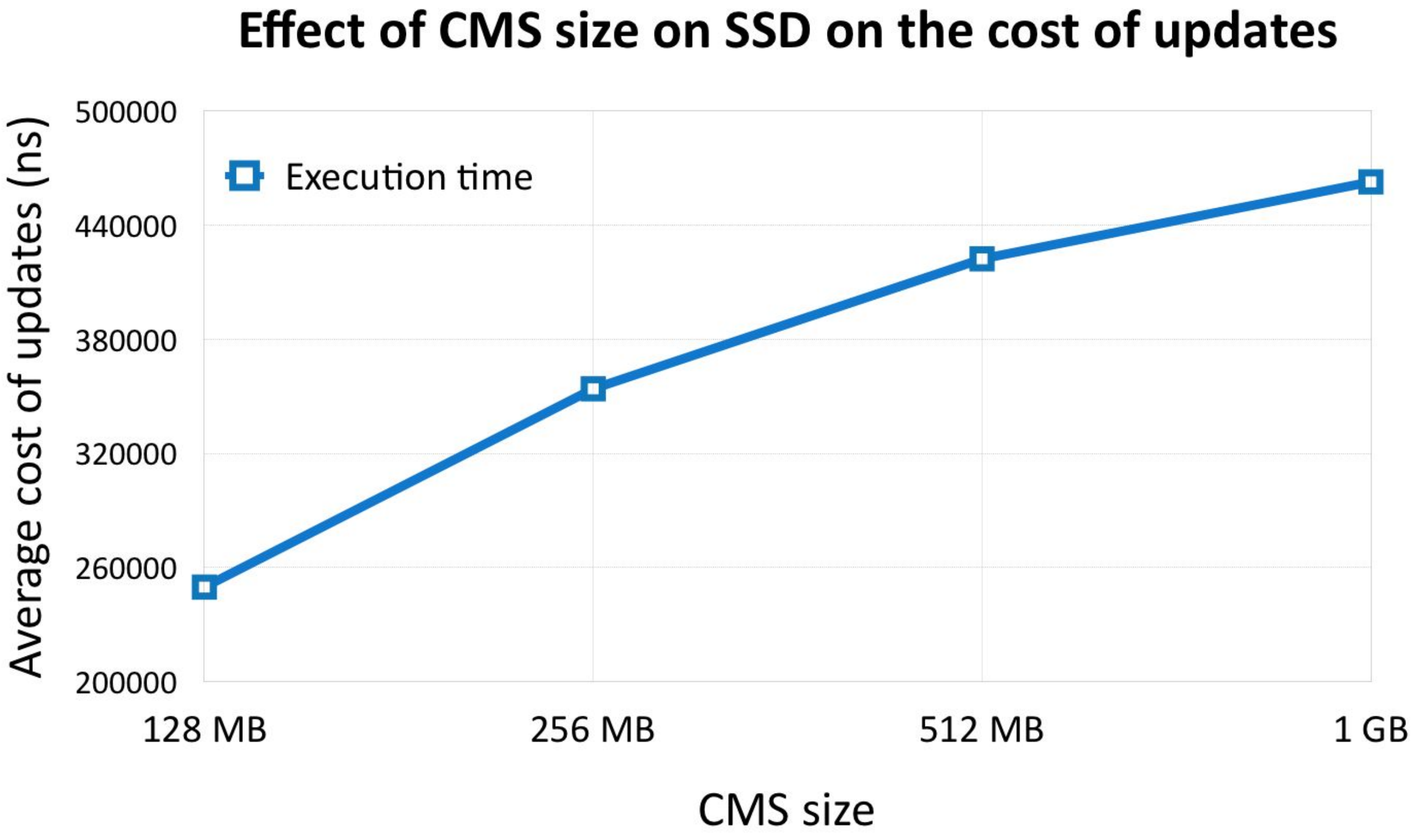
For example, in a word-similarity application[1], for 90 GB of web data the count-min sketch size is 8 GB.

[1]. Amit Goyal, Jagadeesh Jagarlamudi, Hal Daumé, III, and Suresh Venkatasubramanian. Sketch techniques for scaling distributional similarity to the web. GEMS, 2010.

Update performance degrades when the CMS grows out of Cache



Update performance worsens when the CMS grows out of RAM



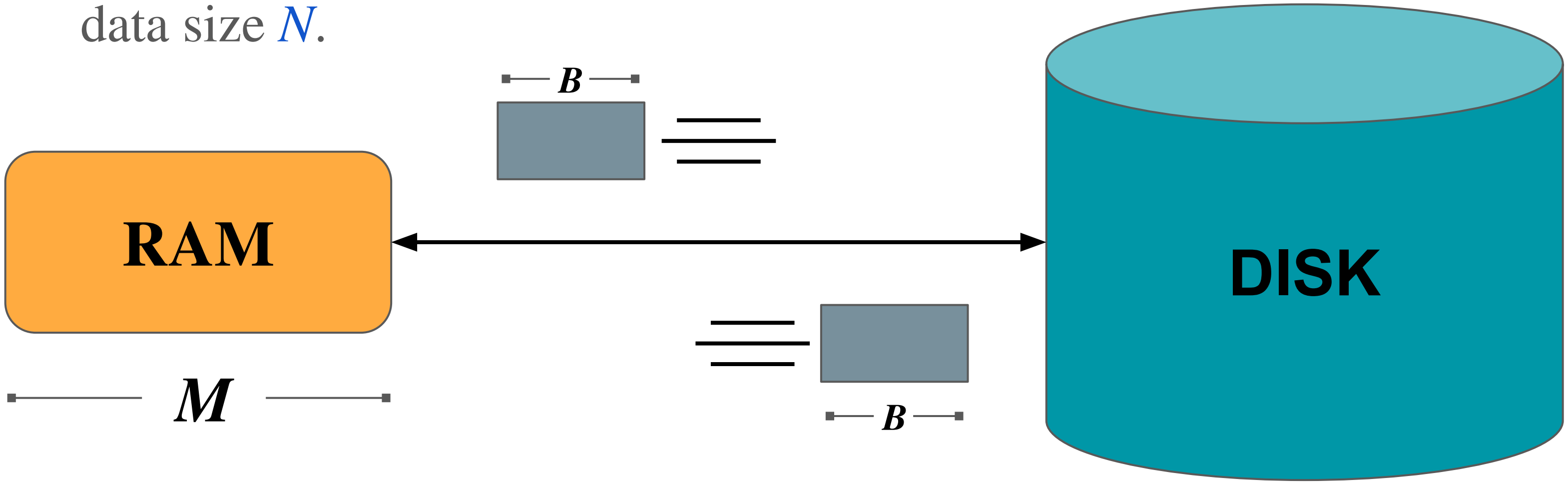
Buffered count-min sketch (BCMS)

- Theoretical:
  - The buffered CMS is **asymptotically faster for estimate operation** than the plain CMS on SSD.
  - The buffered CMS requires **less than 1 I/O per update** operation for most practical configurations on SSD.
  - The buffered CMS offers similar error guarantees as the plain CMS:

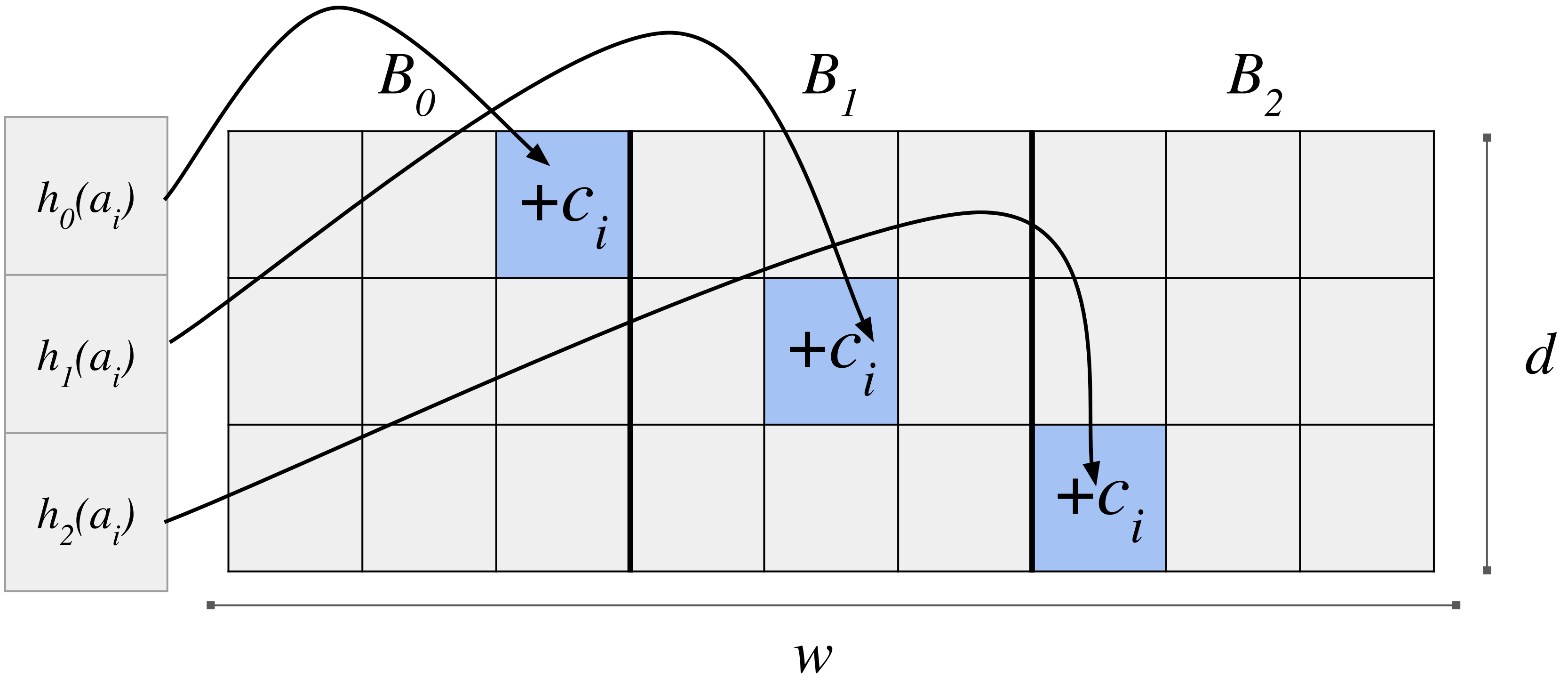
$$\mathbf{Pr[Error(q) > \varepsilon \mathit{N}\ (1 + o(1))] \leq \delta + o(1).}$$

I/O in the disk access machine (DAM) model

- **How computations works:**
  - Data is transferred in blocks between RAM and disk.
  - The # of block transfers dominate the running time.
- **Goal: Minimize # of block transfers**
  - Performance bounds are parameterized by block size *B*, memory size *M*, data size *N*.

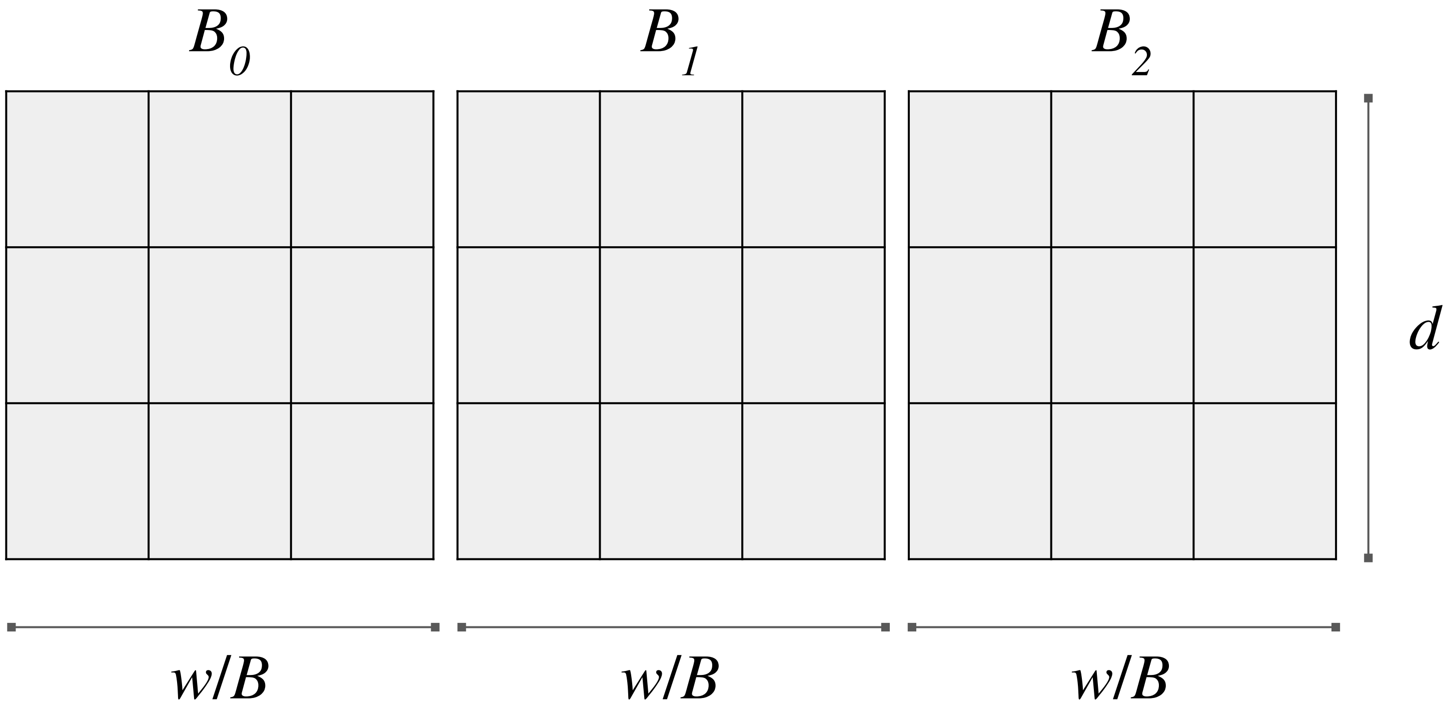


Plain count-min sketch on SSD

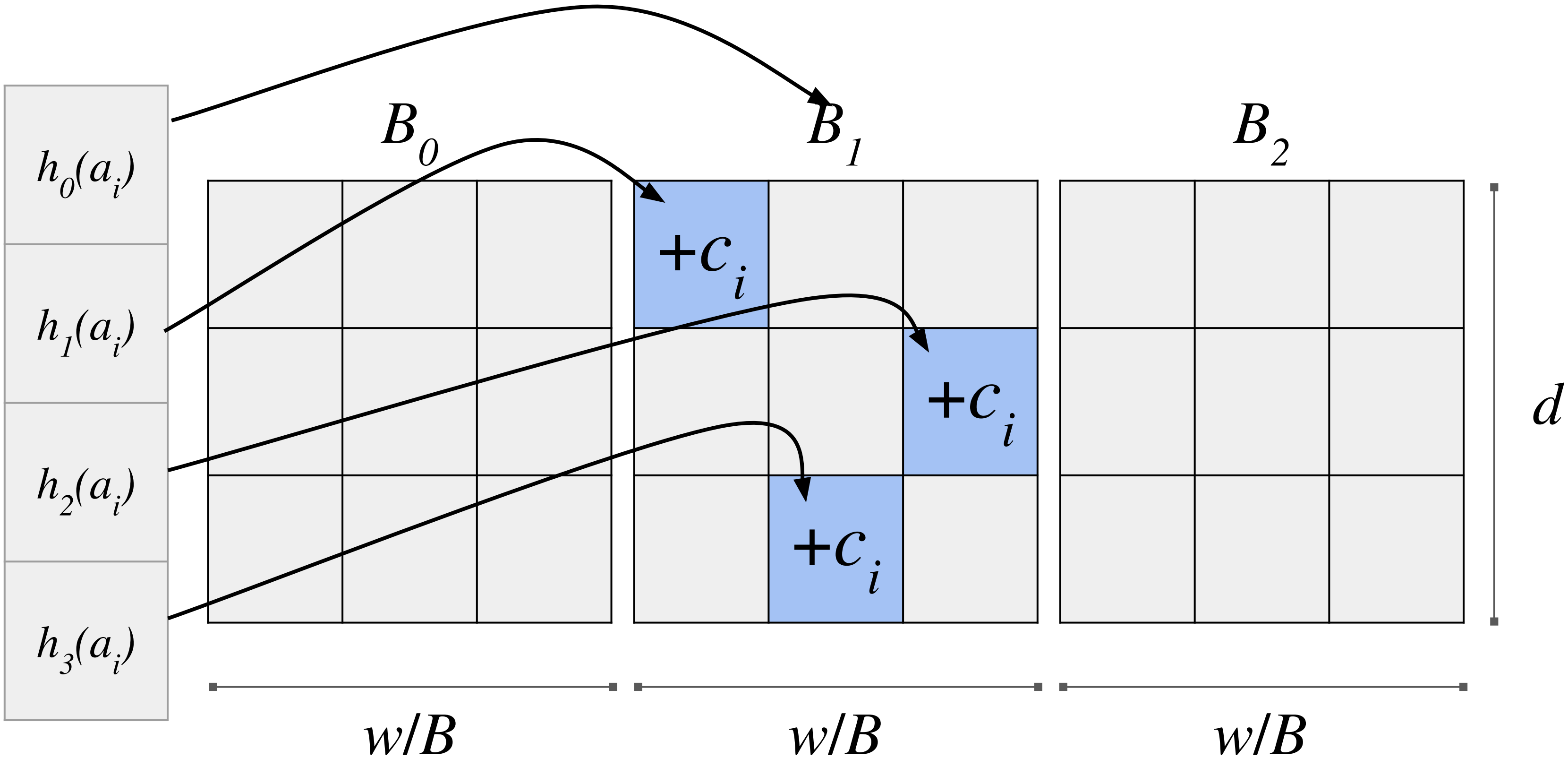


O(d) random I/Os for each operation.

Buffered count-min sketch: hash localization

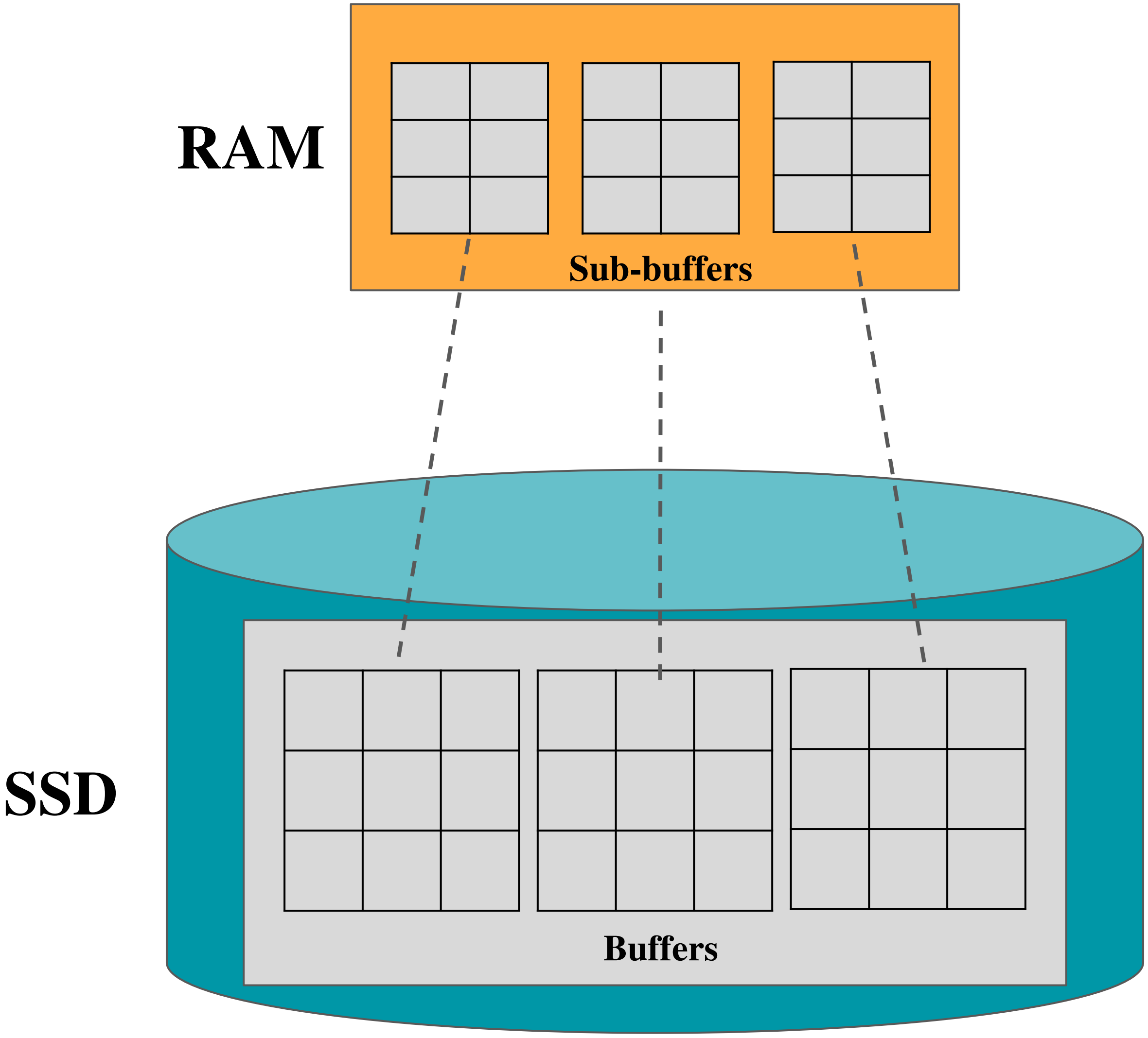


Buffered count-min sketch: hash localization

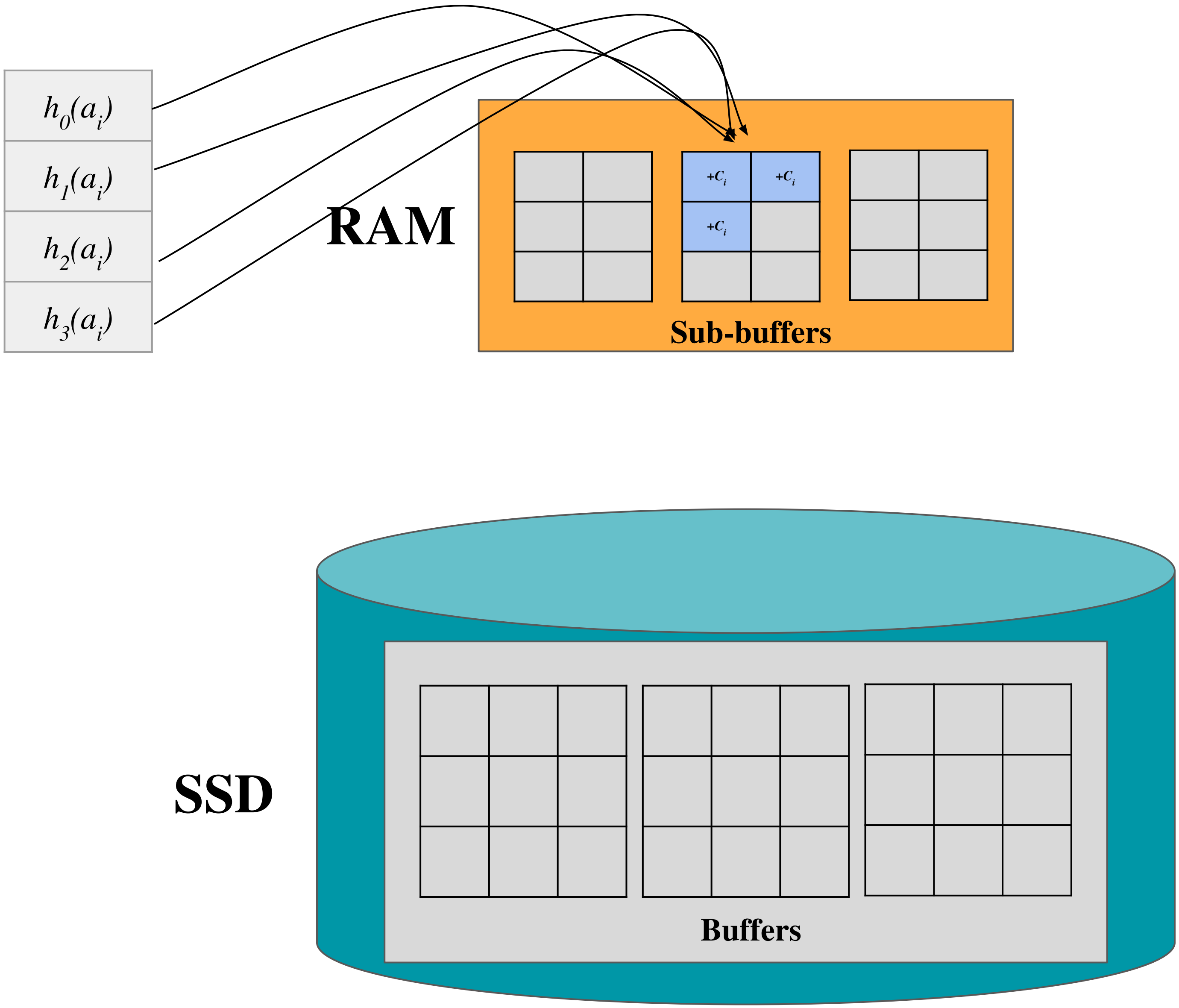


1 I/O per estimate operation.

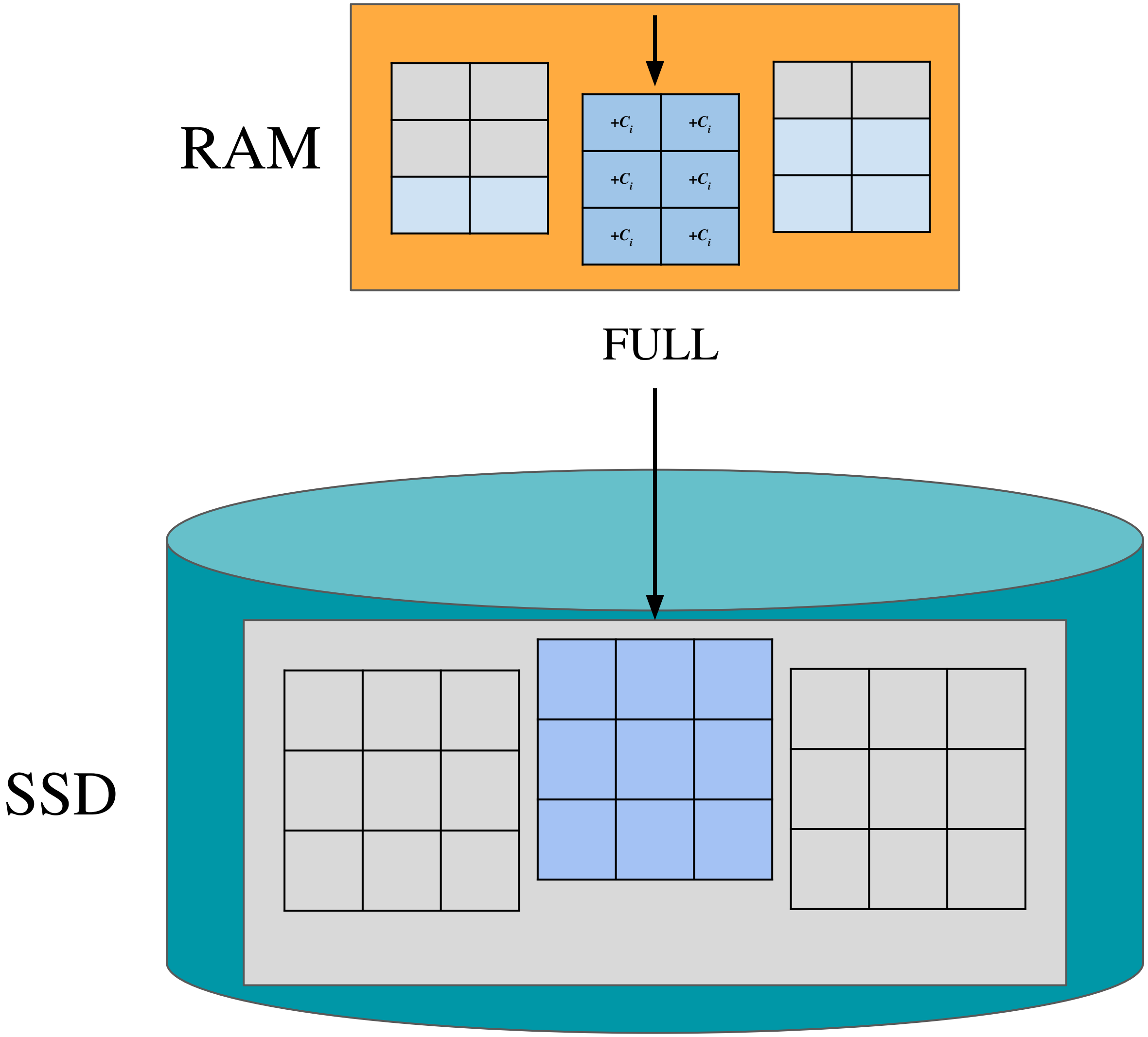
Buffered count-min sketch: buffering



Buffered count-min sketch: buffering



Buffered count-min sketch: buffering

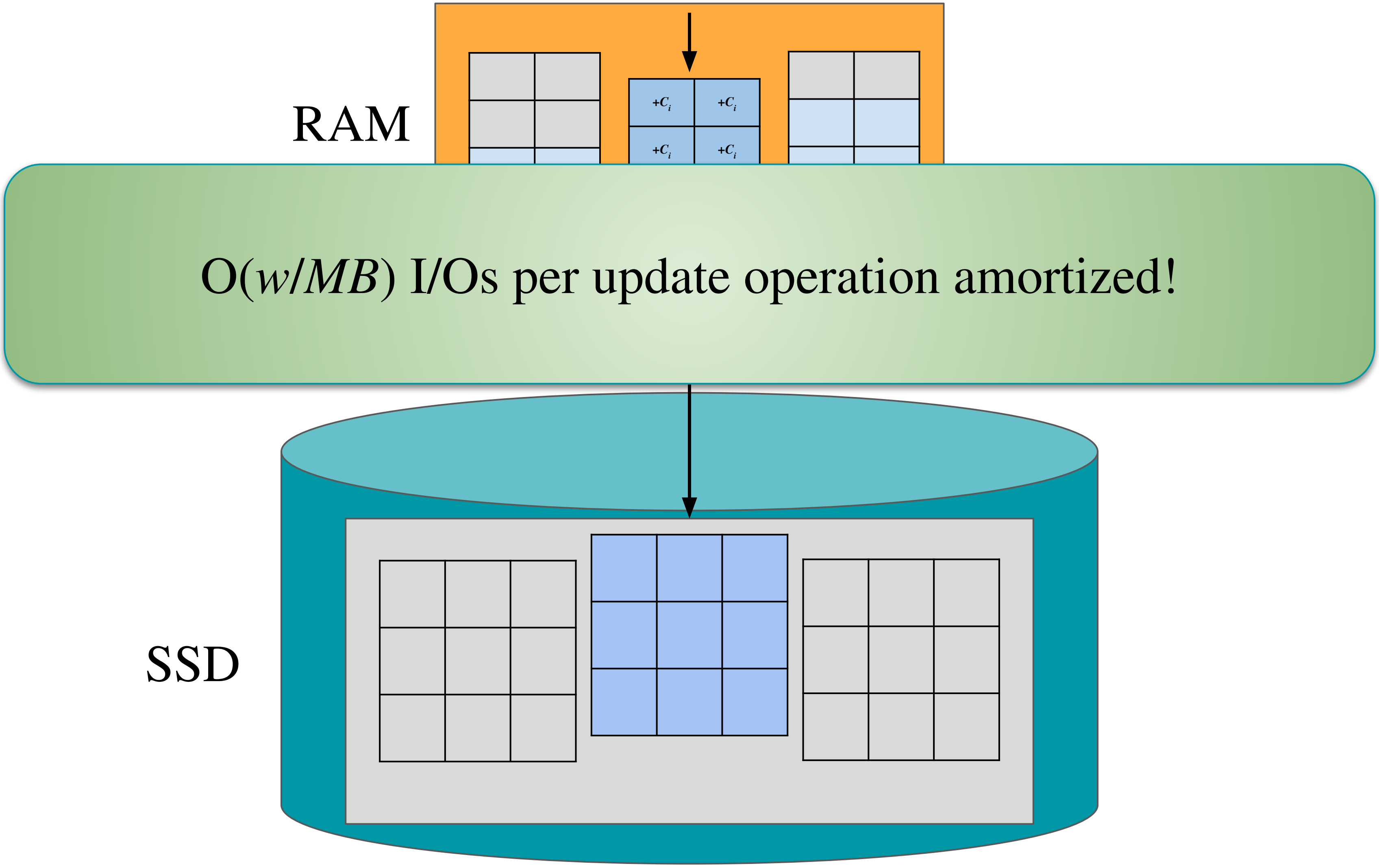


RAM

FULL

SSD

Buffered count-min sketch: buffering



O(*w*/*MB*) I/Os per update operation amortized!

Plain count-min sketch: error analysis

- In a stream of size N and a plain CMS of width w and depth d, for a query item i
  - E[# items colliding with i in a row (or error)] = N/w.

Plain count-min sketch: error analysis

- In a stream of size N and a plain CMS of width w and depth d, for a query item i
  - E[# items colliding with i in a row (or error)] = N/w.
- Using Markov's inequality
  - Probability of seeing more than e times the expected error is $1/e$, i.e., $P[\text{Error} > e\, N/w] \leq 1/e$.

Plain count-min sketch: error analysis

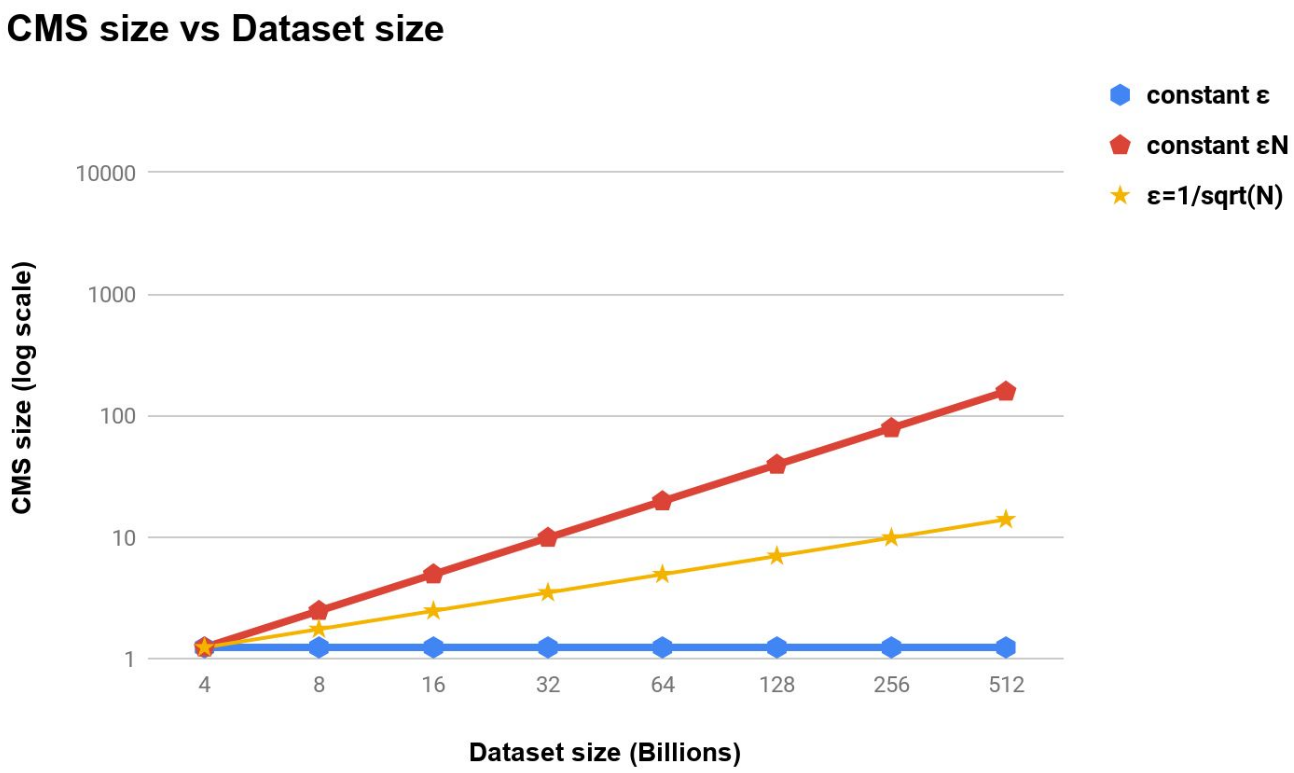
- In a stream of size N and a plain CMS of width w and depth d, for a query item i
  - E[# items colliding with i in a row (or error)] = N/w.
- Using Markov's inequality
  - Probability of seeing more than e times the expected error is $1/e$, i.e., P[Error > $e\ N/w$] ≤ $1/e$.
- Each row has an independent hash function
  - Probability that the expected error is more than e times in each row is $(1/e)^d$.

Plain count-min sketch: error analysis

- In a stream of size N and a plain CMS of width w and depth d, for a query item i
    - E[# items colliding with i in a row (or error)] = N/w.
- Using Markov's inequality
    - Probability of seeing more than e times the expected error is $1/e$, i.e., P[Error > $e\ N/w$] ≤ $1/e$.
- Each row has an independent hash function
    - Probability that the expected error is more than e times in each row is $(1/e)^d$.

For $w = \lceil e/\varepsilon \rceil$ and $d = \ln(1/\delta)$, expected error = δ.

Assumption: ε is subconstant in N ($\lim_{N\to\infty} \varepsilon(N) = 0$)



CMS size grows much slowly with dataset size.

Buffered count-min sketch: error analysis

- Each item i from the stream hashes into a buffer
  - To determine WHP number of items in a buffer we use **balls-and-bins analysis** where, *# balls >> # bins*.

- Error for a query q in different **rows are no longer independent**
  - A high error in one row implies more elements were hashed by h_0 to the same bucket.

Buffered count-min sketch: evaluation

- Empirical:
  - The buffered CMS is **3.7X--4.7X faster on update operation** compared to the plain CMS on SSD.
  - The buffered CMS is **4.3X faster on estimate operation** compared to the plain CMS on SSD.

Evaluation parameters

- Update throughput
- Estimate throughput
- Effect of hash localization on estimation error
- Effect of changing RAM-to-sketch-size ratio

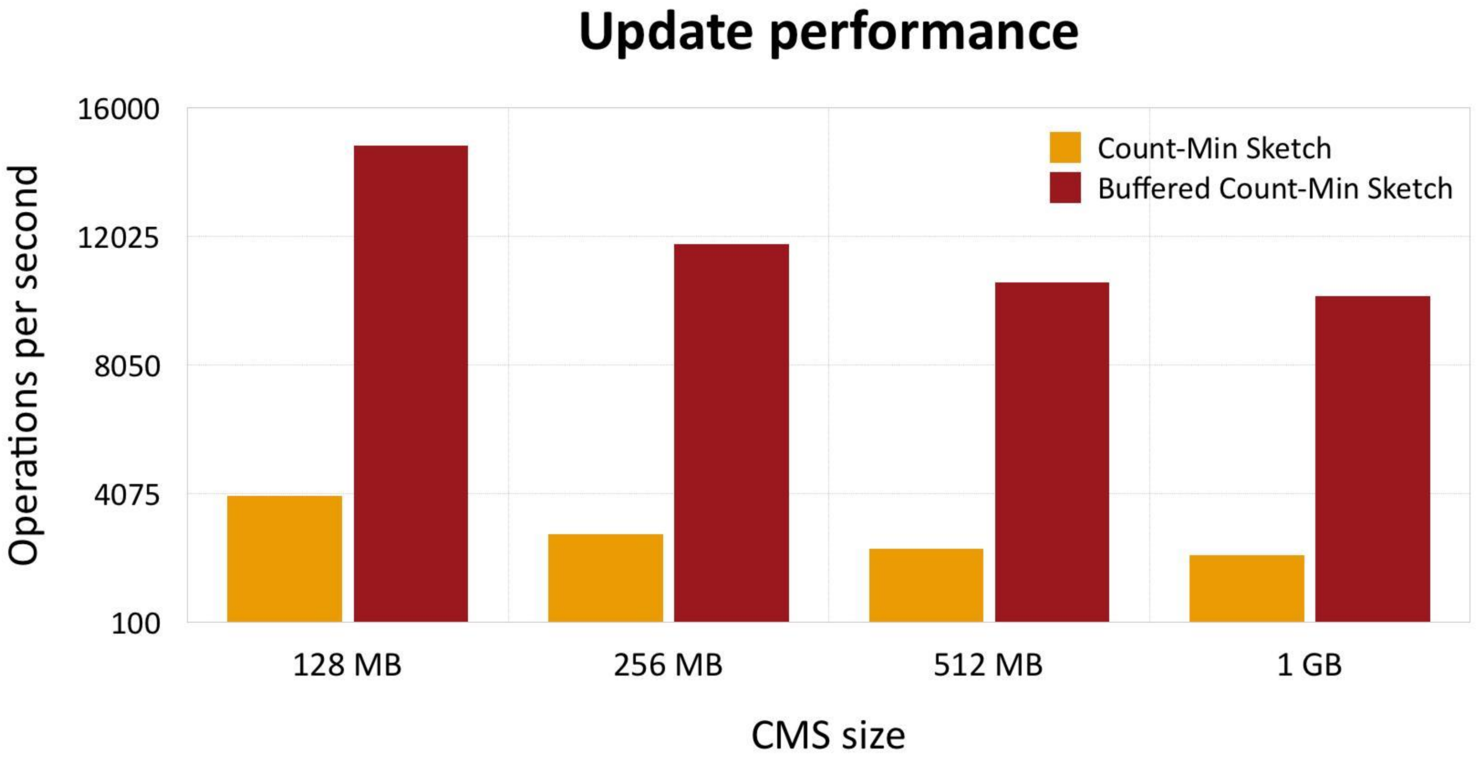
Evaluation setup

Size	Ratio	Width	Depth	#elements
128MB	2	3355444	5	9875188
256MB	4	6710887	5	19750377
512MB	8	13421773	5	39500754
1GB	16	26843546	5	79001508

In all our experiments, $\delta = 0.01$ and overestimate (εN) = 8.

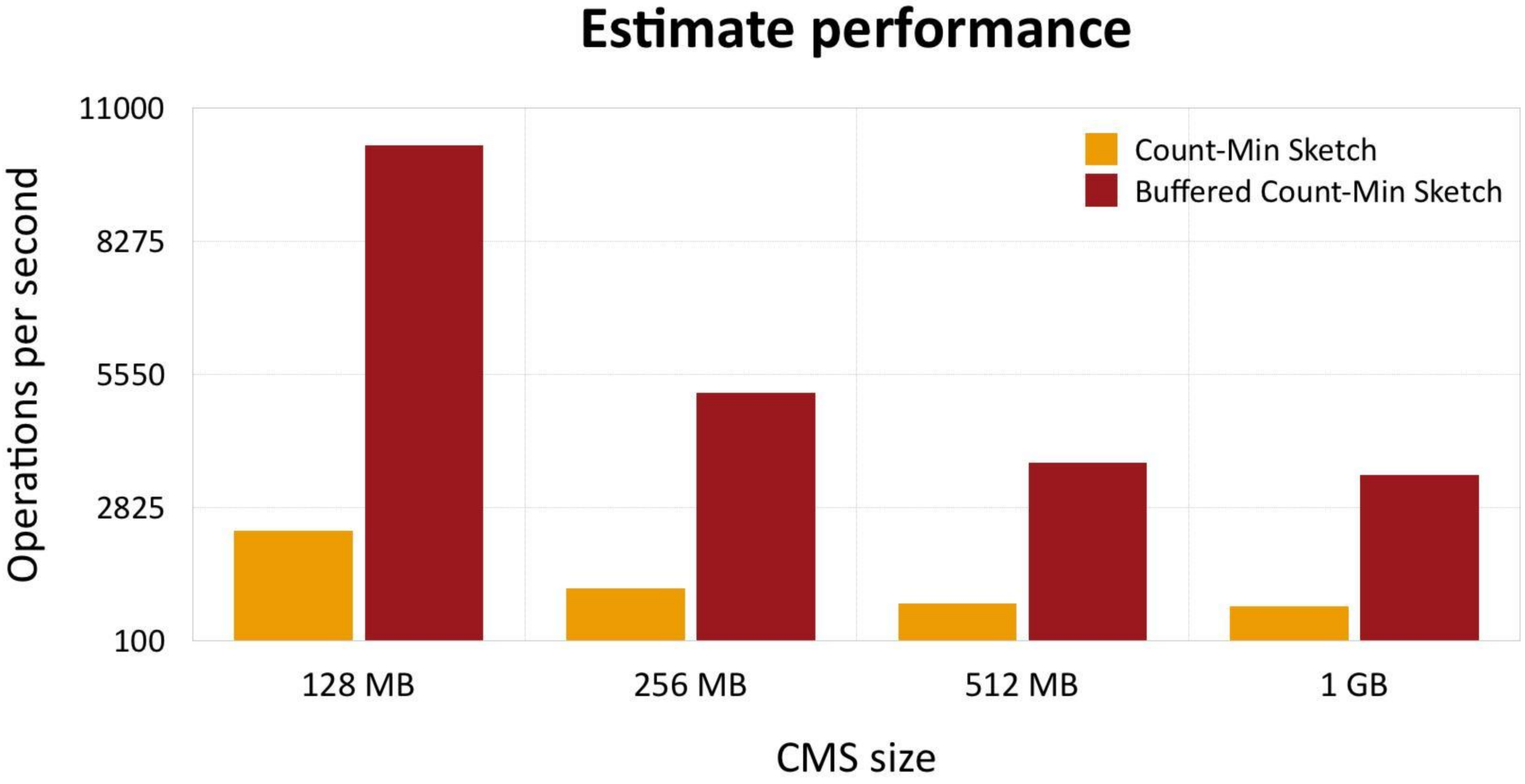
RAM size: 64 MB

Update performance



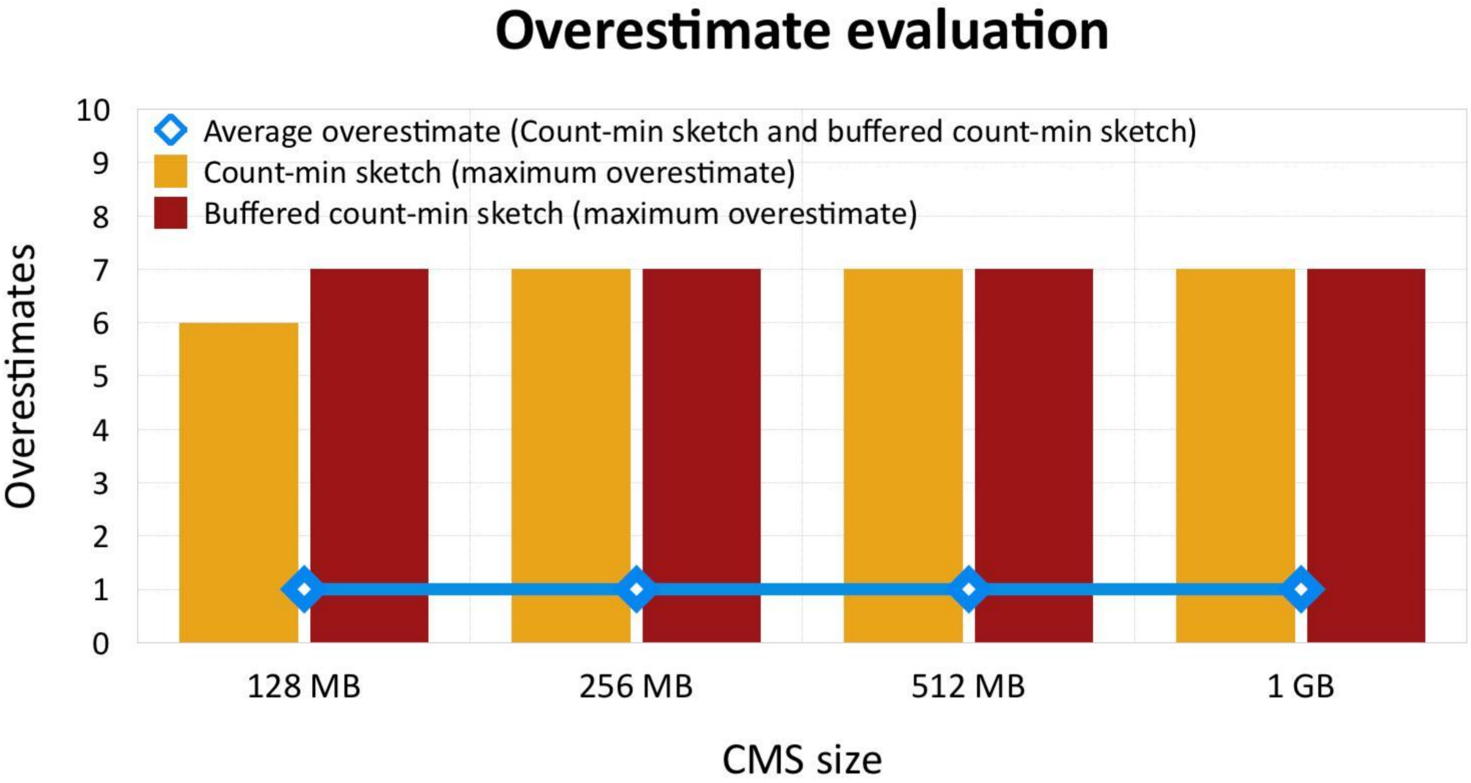
The BCMS is 3.7X--4.7X faster on update operation compared to the plain CMS.

Estimate performance



The BCMS is 4.3X faster on estimate operation compared to the plain CMS.

Accuracy



Overestimates in the BCMS are similar to the plain CMS.

Conclusion

- Techniques like hash localization and buffering can be applied to scale the plain CMS to SSDs.
- We showed both theoretically and empirically, if we keep the ε-error subconstant in N then the hash localization has trivial (or no effect) on the overestimate.
- We leave the question of deriving update lower bounds and/or a SSD-based data structure with faster update time for future work.