Tight Lower Bound for Comparison-Based Quantile Summaries

Pavel Veselý

University of Warwick

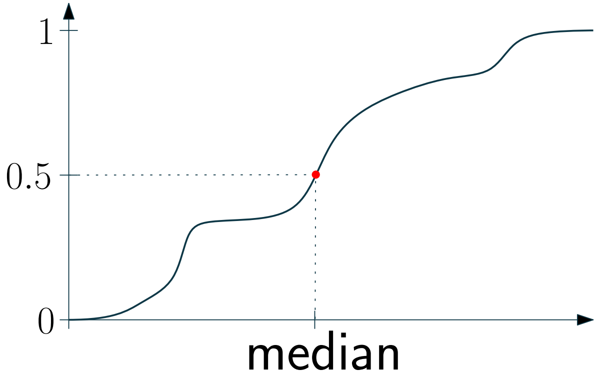
8 April 2020

Based on joint work with Graham Cormode (Warwick)

Powered by BeamerikZ

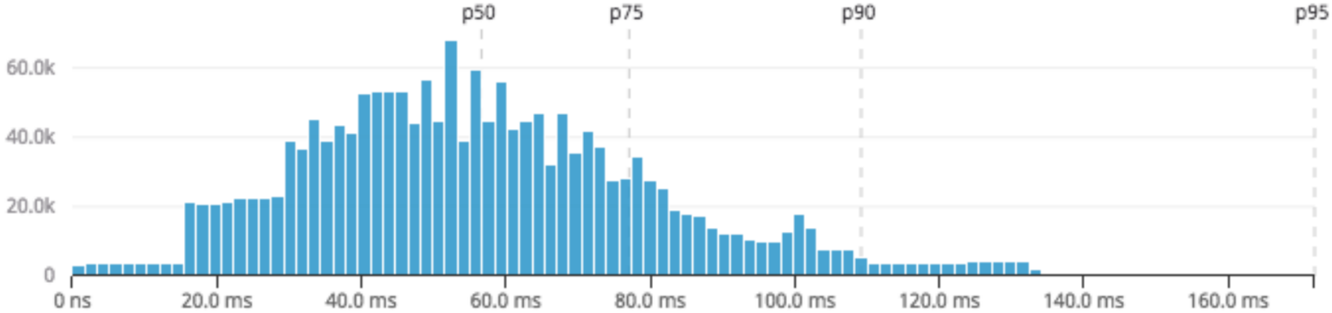
Overview of the talk

Quantiles & Distributions & **Big Data Algorithms**



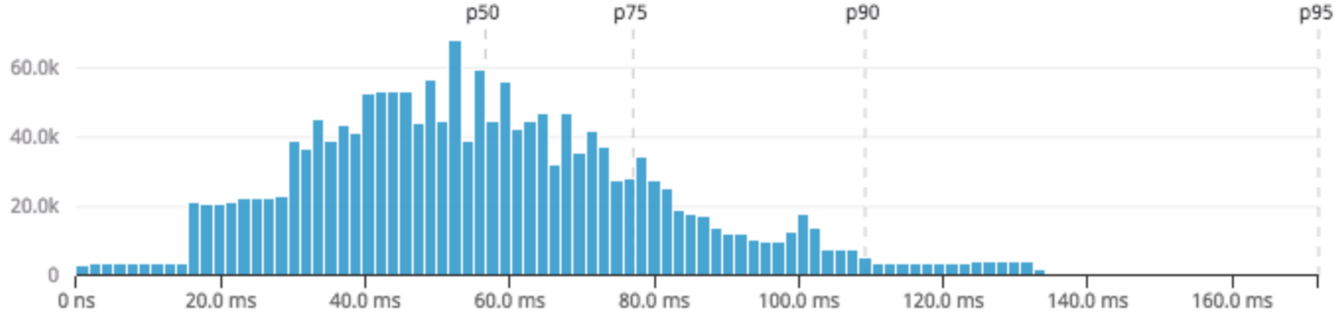
Streaming Model

Motivation: Monitoring Latencies of Web Requests



Source: C. Masson, J.E. Rim, and H.K. Lee. Ddsketch: A fast and fully-mergeable quantile sketch with relative-error guarantees. PVLDB, 12(12):2195–2205, 2019.

Motivation: Monitoring Latencies of Web Requests

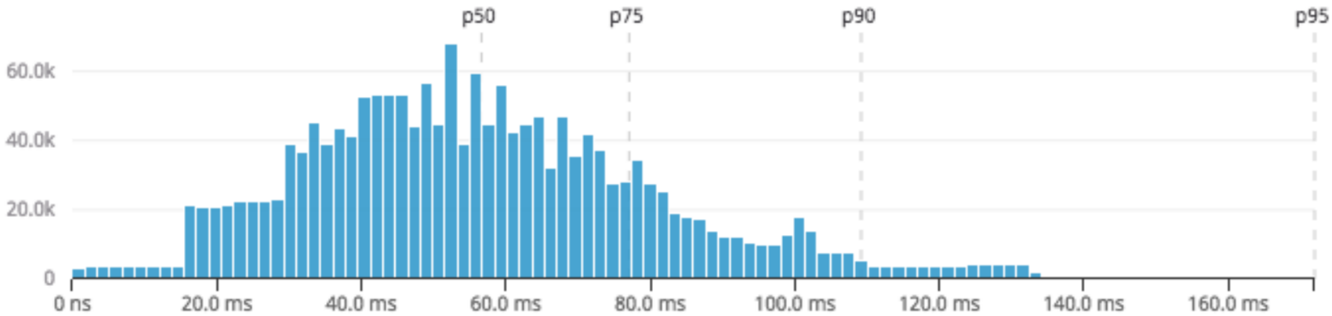


Source: C. Masson, J.E. Rim, and H.K. Lee. Ddsketch: A fast and fully-mergeable quantile sketch with relative-error guarantees. PVLDB, 12(12):2195–2205, 2019.

Millions of observations

- no need to store all observed latencies

Motivation: Monitoring Latencies of Web Requests



Source: C. Masson, J.E. Rim, and H.K. Lee. Ddsketch: A fast and fully-mergeable quantile sketch with relative-error guarantees. PVLDB, 12(12):2195–2205, 2019.

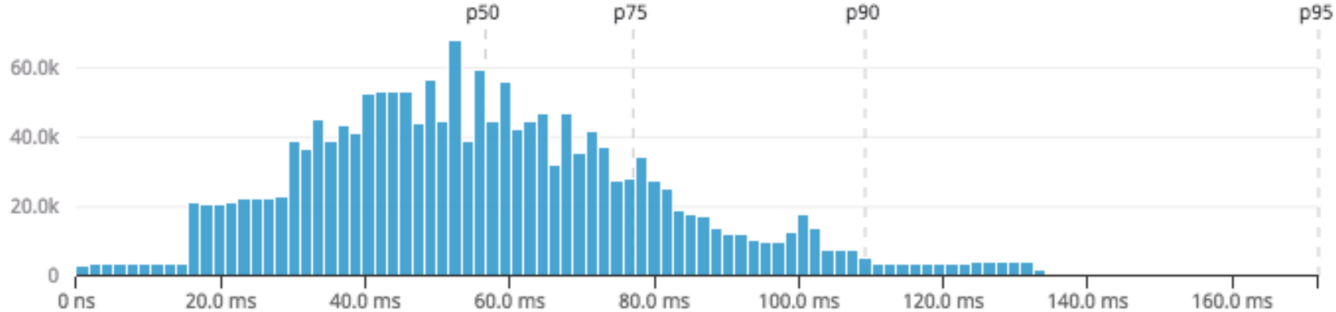
Millions of observations

- no need to store all observed latencies

How does the distribution *look like?*

What is the median latency?

Motivation: Monitoring Latencies of Web Requests



Source: C. Masson, J.E. Rim, and H.K. Lee. Ddsketch: A fast and fully-mergeable quantile sketch with relative-error guarantees. PVLDB, 12(12):2195–2205, 2019.

Millions of observations

- no need to store all observed latencies

How does the distribution *look like?*

What is the median latency?

- Average latency too high due to $\sim 2\%$ of very high latencies

Streaming Model

Motivation: monitoring latencies of requests

Streaming Model

Motivation: monitoring latencies of requests

Streaming model = one pass over data & limited memory

Streaming Model

Motivation: monitoring latencies of requests

Streaming model = one pass over data & limited memory

Streaming algorithm

- receives data in a stream, item by item
- uses memory sublinear in N = stream length
- at the end, computes the answer

Streaming Model

Motivation: monitoring latencies of requests

Streaming model = one pass over data & limited memory

Streaming algorithm

- receives data in a stream, item by item
- uses memory sublinear in N = stream length
- at the end, computes the answer

Challenges:

- N very large & not known
- Data independent
- Stream ordered arbitrarily
- No random access to data

Streaming Model

Motivation: monitoring latencies of requests

Streaming model = one pass over data & limited memory

Streaming algorithm

- receives data in a stream, item by item
- uses memory sublinear in N = stream length
- at the end, computes the answer

Challenges:

- N very large & not known
- Data independent
- Stream ordered arbitrarily
- No random access to data

Main objective: **space**

Streaming Model

Motivation: monitoring latencies of requests

Streaming model = one pass over data & limited memory

Streaming algorithm

- receives data in a stream, item by item
- uses memory sublinear in N = stream length
- at the end, computes the answer

Challenges:
- N very large & not known
- Data independent
- Stream ordered arbitrarily
- No random access to data

Main objective: **space**

How to summarize the input?

Selection Problem & Streaming

- Input: stream of N numbers
- Goal: find the k-th smallest
  - e.g.: the median, 99th percentile
- $\mathcal{O}(N)$ time offline algorithm [Blum *et al.* '73]

Selection Problem & Streaming

- Input: stream of N numbers
- Goal: find the k-th smallest
  - e.g.: the median, 99th percentile
- $\mathcal{O}(N)$ time offline algorithm [Blum *et al.* '73]
- Streaming restrictions:
  - just one pass over the data
  - limited memory: $o(N)$

Selection Problem & Streaming

- Input: stream of N numbers
- Goal: find the k-th smallest
  - e.g.: the median, 99th percentile
- $\mathcal{O}(N)$ time offline algorithm [Blum *et al.* '73]
- Streaming restrictions:
  - just one pass over the data
  - limited memory: $o(N)$

No streaming algorithm for exact selection

$\Omega(N)$ space needed to find the median

[Munro & Paterson '80, Guha & McGregor '07]

Selection Problem & Streaming

- Input: stream of N numbers
- Goal: find the k-th smallest
  - e.g.: the median, 99th percentile
- $\mathcal{O}(N)$ time offline algorithm [Blum *et al.* '73]
- Streaming restrictions:
  - just one pass over the data
  - limited memory: $o(N)$

No streaming algorithm for exact selection

$\Omega(N)$ space needed to find the median

[Munro & Paterson '80, Guha & McGregor '07]

What about finding an approximate median?

Approximate Median & Quantiles

How to define an approximate median?

Approximate Median & Quantiles

How to define an approximate median?

ϕ-quantile $= \lceil \phi \cdot N \rceil$-th smallest element ($\phi \in [0, 1]$)

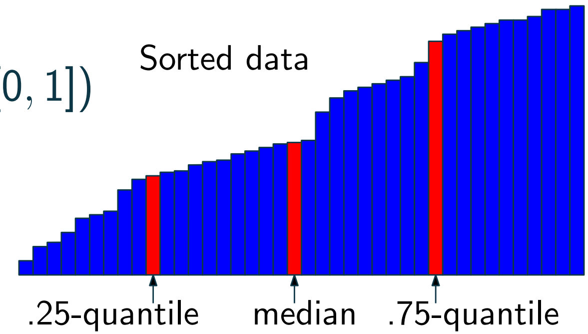
- Median $=$ .5-quantile

Approximate Median & Quantiles

How to define an approximate median?

ϕ-quantile = $\lceil \phi \cdot N \rceil$-th smallest element ($\phi \in [0, 1]$)

- Median = .5-quantile
- Quartiles = .25, .5, and .75-quantiles
- Percentiles = .01, .02, . . . , .99-quantiles

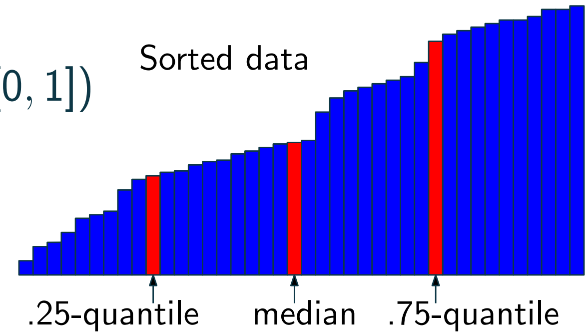


Approximate Median & Quantiles

How to define an approximate median?

ϕ-quantile = $\lceil \phi \cdot N \rceil$-th smallest element ($\phi \in [0, 1]$)

- Median = .5-quantile
- Quartiles = .25, .5, and .75-quantiles
- Percentiles = .01, .02, . . . , .99-quantiles



ε-approximate ϕ-quantile = any ϕ'-quantile for $\phi' = [\phi - \varepsilon, \phi + \varepsilon]$

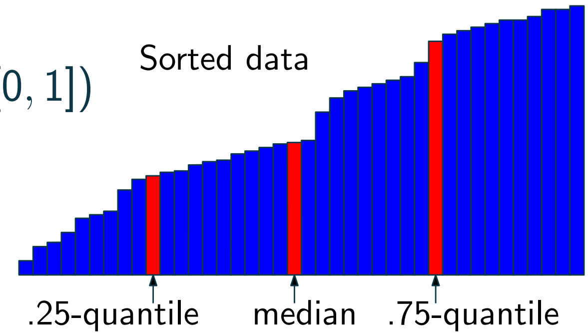
- .01-approximate medians are .49- and .51-quantiles (and items in between)

Approximate Median & Quantiles

How to define an approximate median?

ϕ-quantile $= \lceil \phi \cdot N \rceil$-th smallest element ($\phi \in [0, 1]$)

- Median $=$ .5-quantile
- Quartiles $=$ .25, .5, and .75-quantiles
- Percentiles $=$ .01, .02, ..., .99-quantiles



ε-approximate ϕ-quantile $=$ any ϕ'-quantile for $\phi' = [\phi - \varepsilon, \phi + \varepsilon]$

- .01-approximate medians are .49- and .51-quantiles (and items in between)

ε-approximate selection:

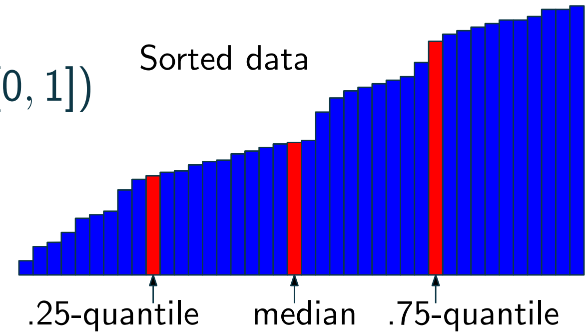
- query k-th smallest $\rightarrow$ return k'-th smallest for $k' = k \pm \varepsilon N$

Approximate Median & Quantiles

How to define an approximate median?

ϕ-quantile = $\lceil \phi \cdot N \rceil$-th smallest element ($\phi \in [0, 1]$)

- Median = .5-quantile
- Quartiles = .25, .5, and .75-quantiles
- Percentiles = .01, .02, . . . , .99-quantiles



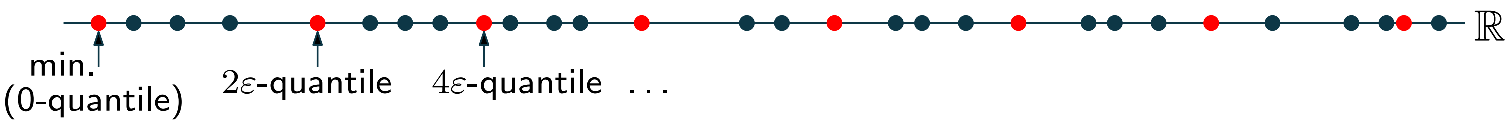
ε-approximate ϕ-quantile = any ϕ'-quantile for $\phi' = [\phi - \varepsilon, \phi + \varepsilon]$

- .01-approximate medians are .49- and .51-quantiles (and items in between)

ε-approximate selection:

- query k-th smallest $\rightarrow$ return k'-th smallest for $k' = k \pm \varepsilon N$

Offline summary: sort data & select $\sim \dfrac{1}{2\varepsilon}$ items



ε-Approximate Quantile Summaries

Data structure with two operations:

- $\textsc{Update}(x)$: $x =$ new item from the stream

ε-Approximate Quantile Summaries

Data structure with two operations:

- $\textsc{Update}(x)$: $x =$ new item from the stream
- $\textsc{Quantile\_Query}(\phi)$: For $\phi \in [0, 1]$, return ε-approximate ϕ-quantile

ε-Approximate Quantile Summaries

Data structure with two operations:

- $\textsc{Update}(x)$: $x =$ new item from the stream
- $\textsc{Quantile\_Query}(\phi)$: For $\phi \in [0, 1]$, return ε-approximate ϕ-quantile

Additional operations:

- $\textsc{Rank\_Query}(x)$:
  - For item x, determine its rank $=$ position in the ordering of the input

ε-Approximate Quantile Summaries

Data structure with two operations:

- $\textsc{Update}(x)$: $x =$ new item from the stream
- $\textsc{Quantile\_Query}(\phi)$: For $\phi \in [0, 1]$, return ε-approximate ϕ-quantile

Additional operations:

- $\textsc{Rank\_Query}(x)$:
  - For item x, determine its rank = position in the ordering of the input
- Merge of two quantile summaries
  - Preserve space bounds, while maintaining accuracy

ε-Approximate Quantile Summaries

Data structure with two operations:

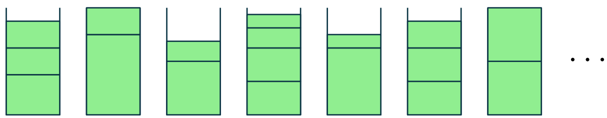
- $\textsc{Update}(x)$: $x =$ new item from the stream
- $\textsc{Quantile\_Query}(\phi)$: For $\phi \in [0, 1]$, return ε-approximate ϕ-quantile

Additional operations:

- $\textsc{Rank\_Query}(x)$:
  - For item x, determine its rank = position in the ordering of the input
- Merge of two quantile summaries
  - Preserve space bounds, while maintaining accuracy

Quantile summaries $\rightarrow$ streaming algorithms for:

- Approximating distributions
- Equi-depth histograms
- Streaming Bin Packing [Cormode & **V.** '20]



ε-Approximate Quantile Summaries

Data structure with two operations:

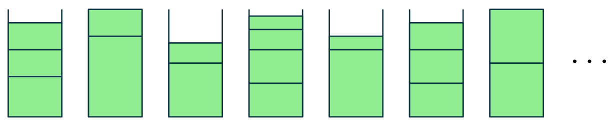
- Update(x): $x =$ new item from the stream
- Quantile_Query(ϕ): For $\phi \in [0, 1]$, return ε-approximate ϕ-quantile

Additional operations:

- Rank_Query(x):
  - For item x, determine its rank = position in the ordering of the input
- Merge of two quantile summaries
  - Preserve space bounds, while maintaining accuracy

Quantile summaries $\rightarrow$ streaming algorithms for:

- Approximating distributions
- Equi-depth histograms
- Streaming Bin Packing [Cormode & **V.** '20]



Bottom line: Finding ε-approximate median in data streams

Approximate Median & Quantiles: Streaming Algorithms

State-of-the-art results

space $\sim$ # of stored items

Approximate Median & Quantiles: Streaming Algorithms

State-of-the-art results

space $\sim$ # of stored items

- $\mathcal{O}\left(\frac{1}{\varepsilon} \cdot \log \varepsilon N\right)$ – deterministic comparison-based [Greenwald & Khanna '01]

  maintains a subset of items + bounds on their ranks

Approximate Median & Quantiles: Streaming Algorithms

State-of-the-art results

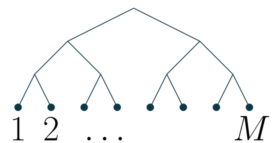
space $\sim$ # of stored items

- $\mathcal{O}\left(\frac{1}{\varepsilon} \cdot \log \varepsilon N\right)$ – deterministic comparison-based [Greenwald & Khanna '01]

  maintains a subset of items + bounds on their ranks

- $\mathcal{O}\left(\frac{1}{\varepsilon} \cdot \log M\right)$ – deterministic for integers $\{1, \ldots, M\}$ [Shrivastava *et al.* '04]

  not for floats or strings



Approximate Median & Quantiles: Streaming Algorithms

State-of-the-art results

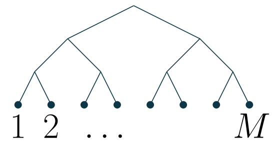
space $\sim$ # of stored items

- $\mathcal{O}\left(\frac{1}{\varepsilon}\cdot \log \varepsilon N\right)$ – deterministic comparison-based [Greenwald & Khanna '01]

  maintains a subset of items + bounds on their ranks

- $\mathcal{O}\left(\frac{1}{\varepsilon}\cdot \log M\right)$ – deterministic for integers $\{1,\ldots,M\}$ [Shrivastava *et al.* '04]

  not for floats or strings



- $\mathcal{O}\left(\frac{1}{\varepsilon}\right)$ – randomized [Karnin *et al.* '16]

  const. probability of violating $\pm\varepsilon N$ error guarantee

Approximate Median & Quantiles: Streaming Algorithms

State-of-the-art results

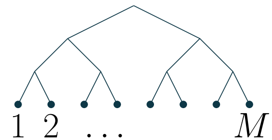
space $\sim$ # of stored items

- $\mathcal{O}\left(\frac{1}{\varepsilon} \cdot \log \varepsilon N\right)$ – deterministic comparison-based [Greenwald & Khanna '01]

  maintains a subset of items + bounds on their ranks

- $\mathcal{O}\left(\frac{1}{\varepsilon} \cdot \log M\right)$ – deterministic for integers $\{1, \ldots, M\}$ [Shrivastava *et al.* '04]

  not for floats or strings



- $\mathcal{O}\left(\frac{1}{\varepsilon}\right)$ – randomized [Karnin *et al.* '16]

  const. probability of violating $\pm\varepsilon N$ error guarantee

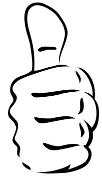
Many more papers: [Munro & Paterson '80, Manku *et al.* '98, Manku *et al.* '99]

[Hung & Ting '10, Agarwal *et al.* '12, Wang *et al.* '13, Felber & Ostrovsky '15, . . .]

Approx. Median & Quantiles: Is There a "Perfect" Algorithm?

Approx. Median & Quantiles: Is There a “Perfect” Algorithm?

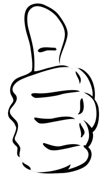
What would be a “perfect” streaming algorithm?



- finds ε-approximate median
- deterministic 

Approx. Median & Quantiles: Is There a "Perfect" Algorithm?

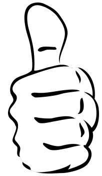
What would be a "perfect" streaming algorithm?



- finds ε-approximate median
- deterministic 
- constant space for fixed ε
  - ideally $\mathcal{O}\left(\frac{1}{\varepsilon}\right)$; or e.g. $\mathcal{O}\left(\frac{1}{\varepsilon^2}\right)$

Approx. Median & Quantiles: Is There a "Perfect" Algorithm?

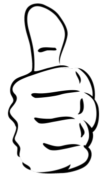
What would be a "perfect" streaming algorithm?



- finds ε-approximate median
- deterministic 
- constant space for fixed ε
  - ideally $\mathcal{O}\left(\frac{1}{\varepsilon}\right)$; or e.g. $\mathcal{O}\left(\frac{1}{\varepsilon^2}\right)$
- no additional knowledge about items
  - comparison-based $\leq$

Approx. Median & Quantiles: Is There a "Perfect" Algorithm?

What would be a "perfect" streaming algorithm?



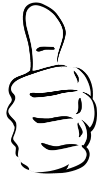
- finds ε-approximate median
- deterministic 
- constant space for fixed ε
  - ideally $\mathcal{O}\left(\frac{1}{\varepsilon}\right)$; or e.g. $\mathcal{O}\left(\frac{1}{\varepsilon^2}\right)$
- no additional knowledge about items
  - comparison-based 

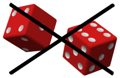
Theorem (Cormode, **V.** '20)

There is **no** perfect streaming algorithm for ε-approximate median

Approx. Median & Quantiles: Is There a "Perfect" Algorithm?

What would be a "perfect" streaming algorithm?



- finds ε-approximate median
- deterministic 
- constant space for fixed ε
  - ideally $\mathcal{O}\left(\frac{1}{\varepsilon}\right)$; or e.g. $\mathcal{O}\left(\frac{1}{\varepsilon^2}\right)$
- no additional knowledge about items
  - comparison-based 

Theorem (Cormode, **V.** '20)

There is **no** perfect streaming algorithm for ε-approximate median

- Optimal space lower bound $\Omega\left(\frac{1}{\varepsilon} \cdot \log \varepsilon N\right)$
  - Matches the result in [Greenwald & Khanna '01]

Approx. Median & Quantiles: Lower Bound Idea

Comparison-based algorithm

$\Rightarrow$ cannot compare with items deleted from the memory

Approx. Median & Quantiles: Lower Bound Idea

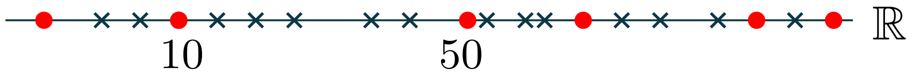
Comparison-based algorithm

$\Rightarrow$ cannot compare with items deleted from the memory

Approx. Median & Quantiles: Lower Bound Idea

Comparison-based algorithm

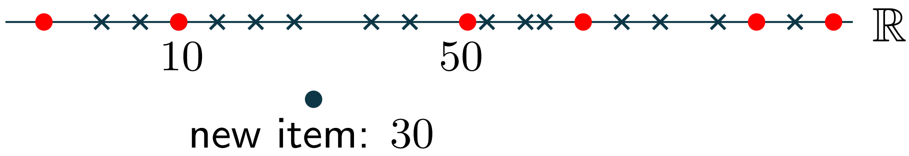
⇒ cannot compare with items deleted from the memory



Approx. Median & Quantiles: Lower Bound Idea

Comparison-based algorithm

$\Rightarrow$ cannot compare with items deleted from the memory

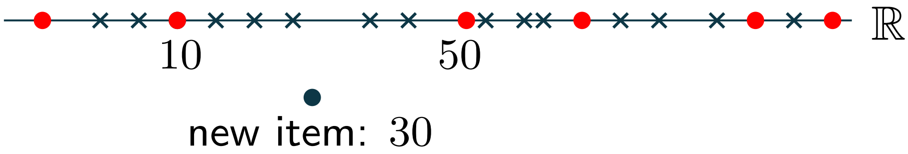


How does 30 compare to discarded items between 10 and 50?

Approx. Median & Quantiles: Lower Bound Idea

Comparison-based algorithm

$\Rightarrow$ cannot compare with items deleted from the memory



How does 30 compare to discarded items between 10 and 50?

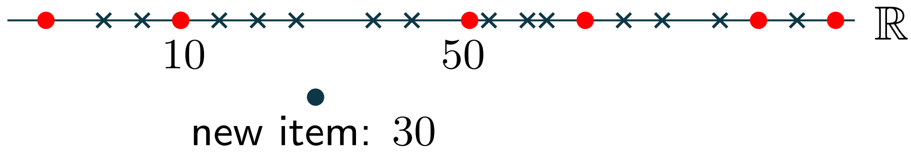
Idea: Introduce uncertainty

- too high uncertainty $\Rightarrow$ not accurate-enough answers

Approx. Median & Quantiles: Lower Bound Idea

Comparison-based algorithm

$\Rightarrow$ cannot compare with items deleted from the memory



How does 30 compare to discarded items between 10 and 50?

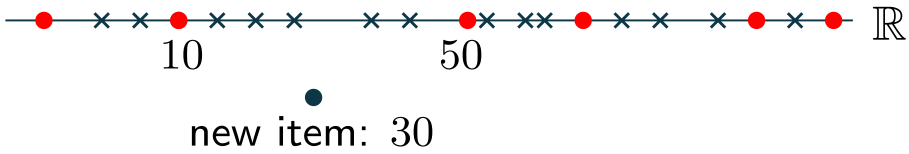
Idea: Introduce uncertainty

- too high uncertainty $\Rightarrow$ not accurate-enough answers
- need to show: low uncertainty $\Rightarrow$ many items stored $\Rightarrow$ large space needed

Approx. Median & Quantiles: Lower Bound Idea

Comparison-based algorithm

$\Rightarrow$ cannot compare with items deleted from the memory



How does 30 compare to discarded items between 10 and 50?

Idea: Introduce uncertainty

- too high uncertainty $\Rightarrow$ not accurate-enough answers
- need to show: low uncertainty $\Rightarrow$ many items stored $\Rightarrow$ large space needed

$\rightarrow$ recursive construction of worst-case stream $\rightarrow$ lower bound $\Omega\left(\frac{1}{\varepsilon} \cdot \log \varepsilon N\right)$

Approximating Median & Quantiles: Conclusions & Open Problems

Problem solved: 

- Deterministic algorithms: space $\Theta\left(\frac{1}{\varepsilon} \cdot \log \varepsilon N\right)$ optimal [Greenwald & Khanna '01] [Cormode, **V.** '20]
- Randomized algorithms: space $\Theta\left(\frac{1}{\varepsilon}\right)$ optimal (const. probability of too high error) [Karnin *et al.* '16]

Approximating Median & Quantiles: Conclusions & Open Problems

Problem solved: $\leq$

- Deterministic algorithms: space $\Theta\left(\frac{1}{\varepsilon} \cdot \log \varepsilon N\right)$ optimal [Greenwald & Khanna '01] [Cormode, **V.** '20]
- Randomized algorithms: space $\Theta\left(\frac{1}{\varepsilon}\right)$ optimal (const. probability of too high error) [Karnin *et al.* '16]

Future work:

- Figure out constant factors

Approximating Median & Quantiles: Conclusions & Open Problems

Problem solved: 

- Deterministic algorithms: space $\Theta\left(\frac{1}{\varepsilon} \cdot \log \varepsilon N\right)$ optimal [Greenwald & Khanna '01] [Cormode, **V.** '20]
- Randomized algorithms: space $\Theta\left(\frac{1}{\varepsilon}\right)$ optimal (const. probability of too high error) [Karnin *et al.* '16]

Future work:

- Figure out constant factors
- Randomized algorithm with good expected space, but guaranteed $\pm\varepsilon N$ error

Approximating Median & Quantiles: Conclusions & Open Problems

Problem solved:

- Deterministic algorithms: space $\Theta\left(\frac{1}{\varepsilon} \cdot \log \varepsilon N\right)$ optimal [Greenwald & Khanna '01] [Cormode, **V.** '20]
- Randomized algorithms: space $\Theta\left(\frac{1}{\varepsilon}\right)$ optimal (const. probability of too high error) [Karnin *et al.* '16]

Future work:

- Figure out constant factors
- Randomized algorithm with good expected space, but guaranteed $\pm\varepsilon N$ error
- A non-trivial lower bound for integers $\{1, \ldots, M\}$?
  - Or can we do better than $\mathcal{O}\left(\frac{1}{\varepsilon} \cdot \log M\right)$?

Approximating Median & Quantiles: Conclusions & Open Problems

Problem solved:

- Deterministic algorithms: space $\Theta\left(\frac{1}{\varepsilon} \cdot \log \varepsilon N\right)$ optimal [Greenwald & Khanna '01] [Cormode, **V.** '20]
- Randomized algorithms: space $\Theta\left(\frac{1}{\varepsilon}\right)$ optimal (const. probability of too high error) [Karnin *et al.* '16]

Future work:

- Figure out constant factors
- Randomized algorithm with good expected space, but guaranteed $\pm\varepsilon N$ error
- A non-trivial lower bound for integers $\{1, \ldots, M\}$?
  - Or can we do better than $\mathcal{O}\left(\frac{1}{\varepsilon} \cdot \log M\right)$?
- Dynamic streams w/ insertions and deletions of items


1
0
Thank You!