Streaming Quantiles

Edo Liberty
Principal Scientist
Amazon Web Services

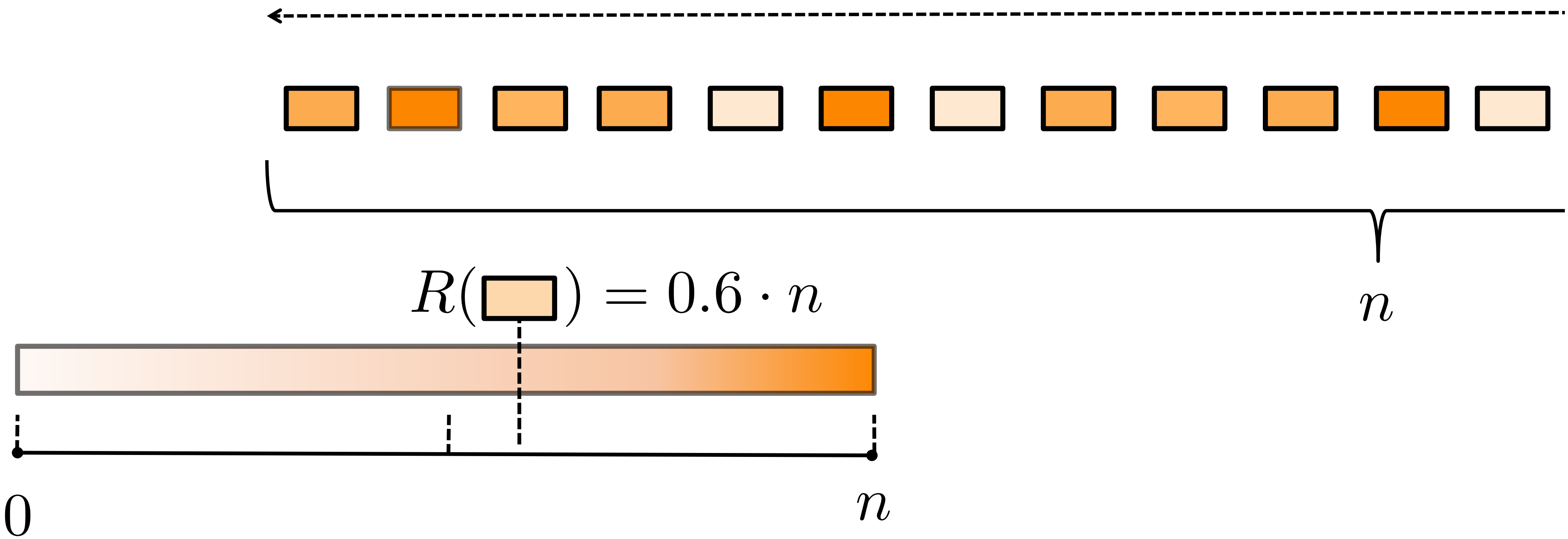
Amazon Kinesis Analytics

Get actionable insights from streaming data in real-time.

Streaming Quantiles

Manku, Rajagopalan, Lindsay. Random sampling techniques for space efficient online computation of order statistics of large datasets.
Munro, Paterson. Selection and sorting with limited storage.
Greenwald, Khanna. Space-efficient online computation of quantile summaries.
Wang, Luo, Yi, Cormode. Quantiles over data streams: An experimental study.
Greenwald, Khanna. Quantiles and equidepth histograms over streams.
Agarwal, Cormode, Huang, Phillips, Wei, Yi. Mergeable summaries.
Felber, Ostrovsky. A randomized online quantile summary in $O((1/\varepsilon)\log(1/\varepsilon))$ words.
Lang, Karnin, Liberty, Optimal Quantile Approximation in Streams.

Problem Definition



Create a sketch for R' such that $|R'(x) - R(x)| \leq \varepsilon n$

Solutions

- Uniform sampling

  ✔ Fast and simple ✔ Fully mergeable ✘ Space $\tilde{O}(1/\varepsilon^2)$

- Greenwald Khanna (GK) sketch

  ✘ Slow, complex ✘ Not mergeable ✔✘ Space $\log(n)/\varepsilon$

- Felber-Ostrovsky, combines sampling and GK (2015)

  ✘ Slow, complex ✘ Not mergeable ✔ Space $\log(1/\varepsilon)/\epsilon$

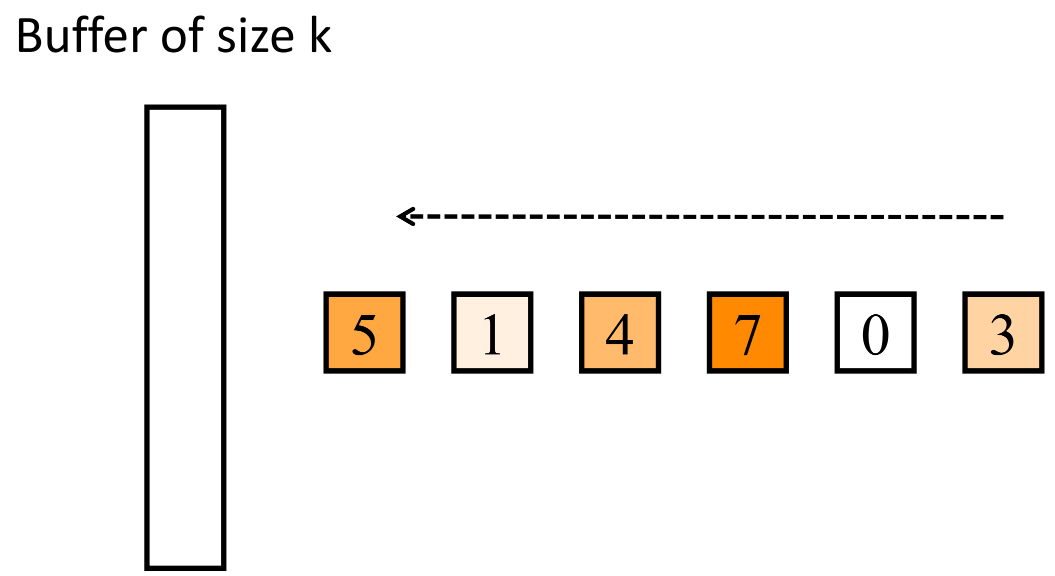
Previously conjectured space optimal for all algorithms.
$\log(1/\varepsilon)/\epsilon$ lower bound for *deterministic* algorithms by Hung and Ting 2010.

Solutions cont'

- Uniform sampling
  ✓ Fast, simple ✓ Fully mergeable ✗ Space $\tilde{O}(1/\varepsilon^2)$
- Greenwald Khanna (GK) sketch
  ✗ Slow, complex ✗ Not mergeable ✓✗ Space $\log(n)/\varepsilon$
- Felber-Ostrovsky, combines sampling and GK (2015)
  ✗ Slow, complex ✗ Not mergeable ✓ Space $\log(1/\varepsilon)/\epsilon$
- Manku-Rajagopalan-Lindsay (MRL)
  ✓ Fast simple ✓ Fully mergeable ✓✗✗ Space $\log^2(n)/\varepsilon$
- Agarwal, Cormode, Huang, Phillips, Wei, Yi
  ✓✗ Fast, complex ✓ Fully mergeable ✓✗ Space $\log^{3/2}(1/\varepsilon)/\varepsilon$

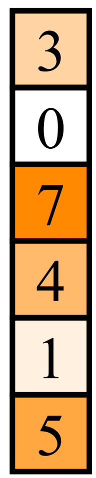
Buffer based solutions

The basic buffer idea



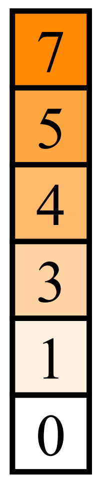
The basic buffer idea

Stores k stream entries



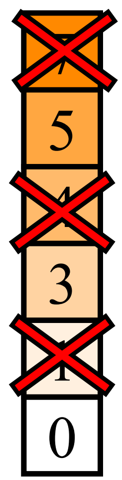
The basic buffer idea

The buffer sorts k stream entries



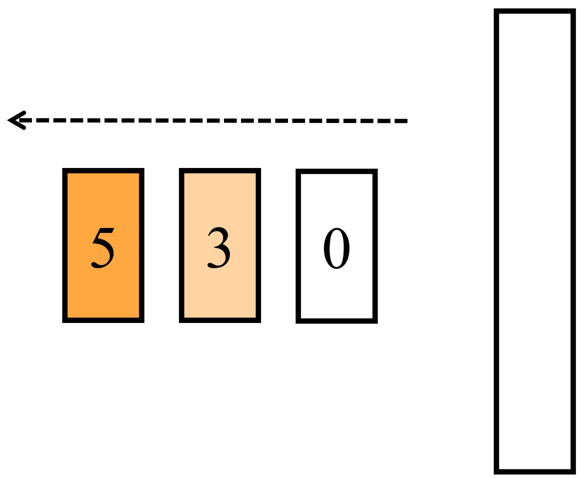
The basic buffer idea

Deletes every other item

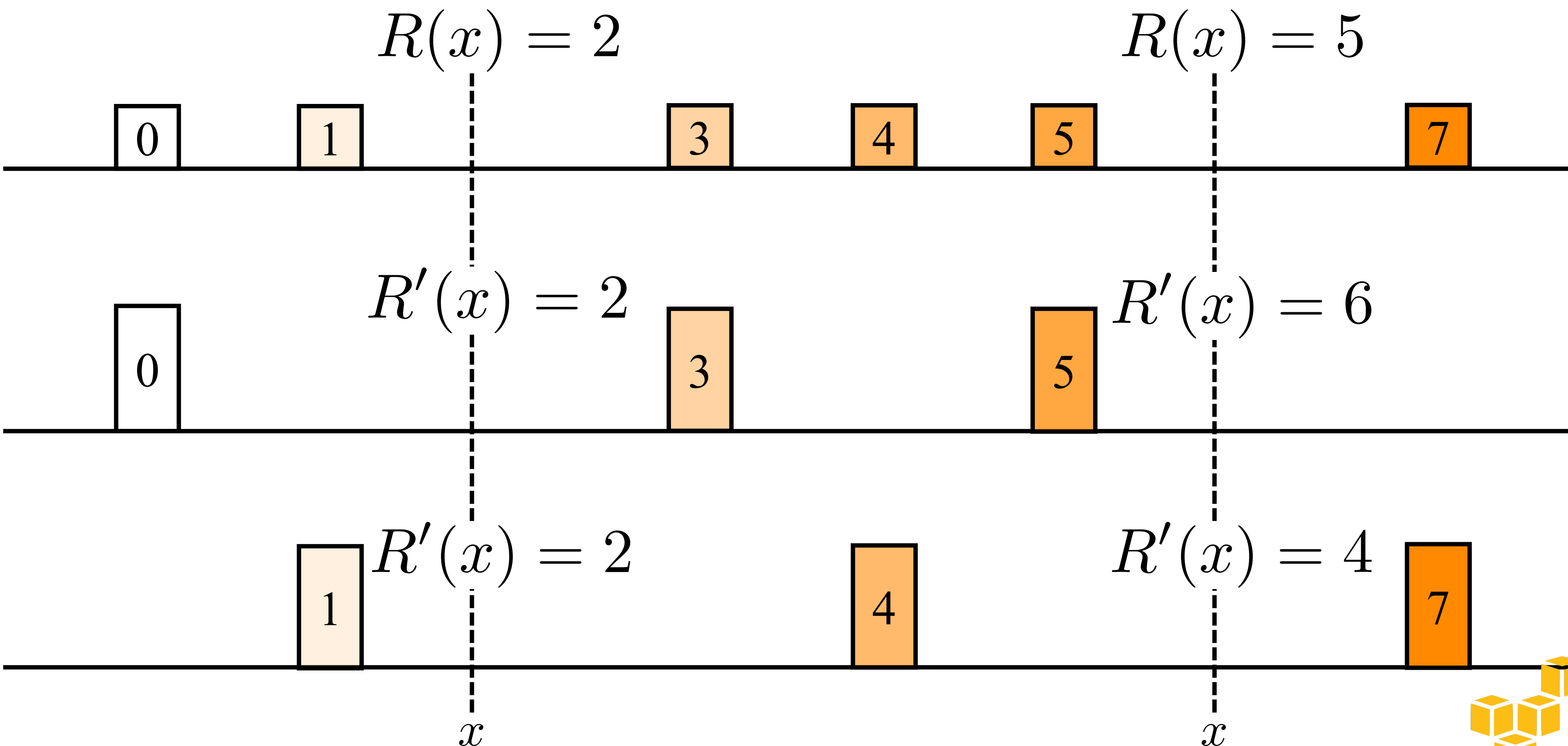


The basic buffer idea

And outputs the rest
with double the weight

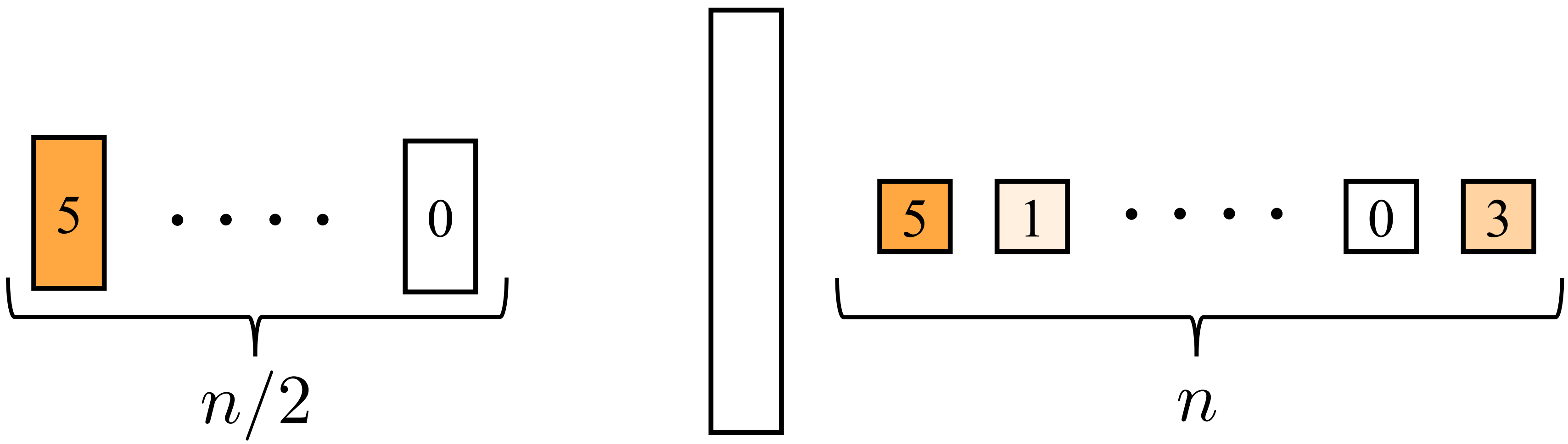


The basic buffer idea



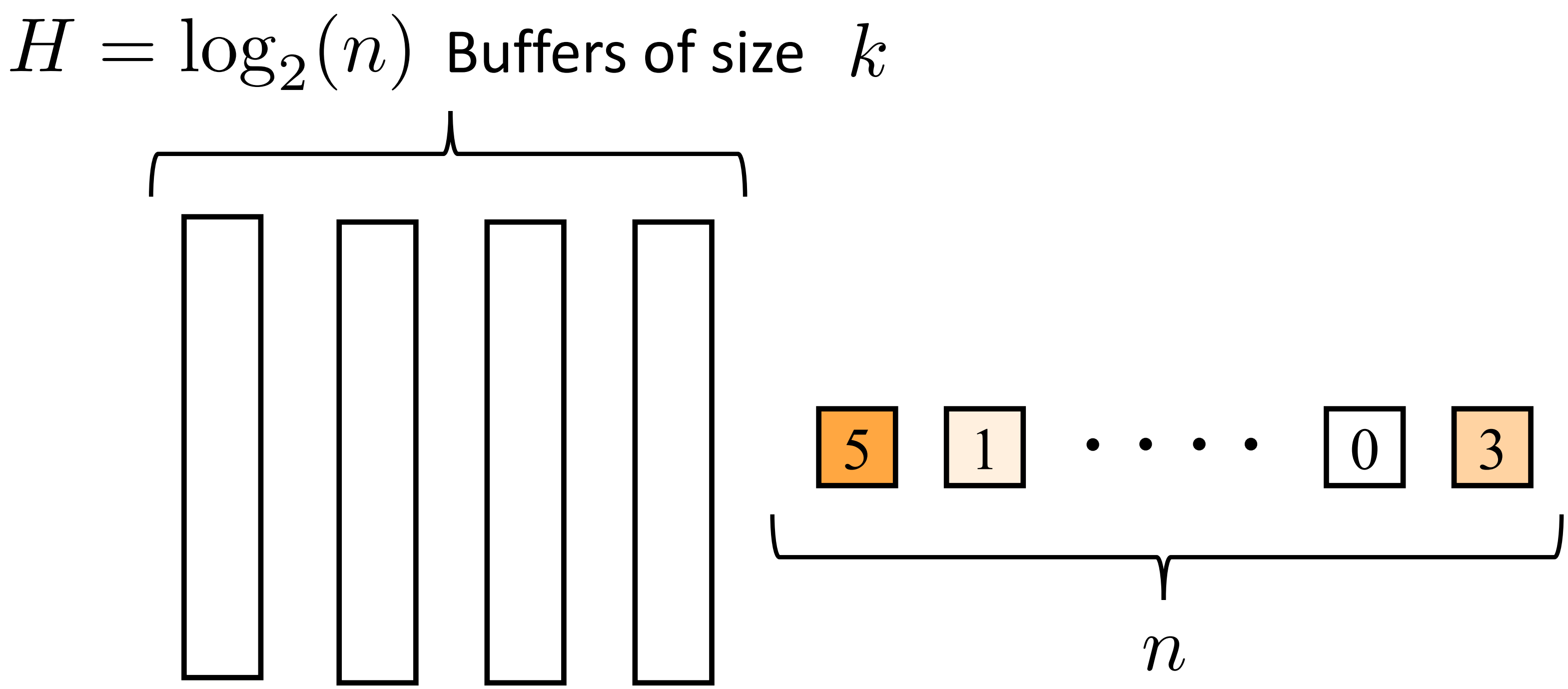
The basic buffer idea

Repeat n/k time until the end of the stream



$$|R'(x) - R(x)| < n/k$$

Manku-Rajagopalan-Lindsay (MRL) sketch



$$|R'(x) - R(x)| \leq n/k \cdot \log_2(n)$$

Manku-Rajagopalan-Lindsay (MRL) sketch

If we set $k = \log_2(n)/\varepsilon$ we get $|R'(x) - R(x)| \leq \varepsilon n$

while maintaining at most $H \cdot k \leq \log_2^2(n)/\varepsilon$ stream items.

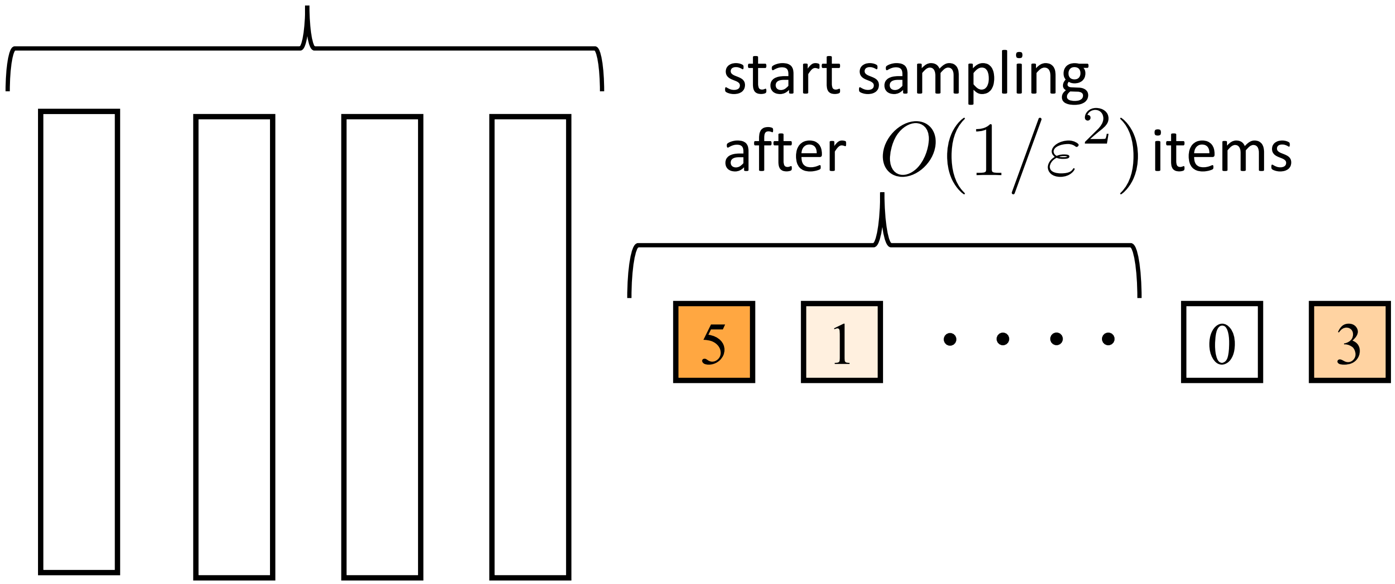
Manku-Rajagopalan-Lindsay (MRL) sketch

- Fast, Simple
- Fully mergeable
- Space

Agarwal, Cormode, Huang, Phillips, Wei, Yi (1)

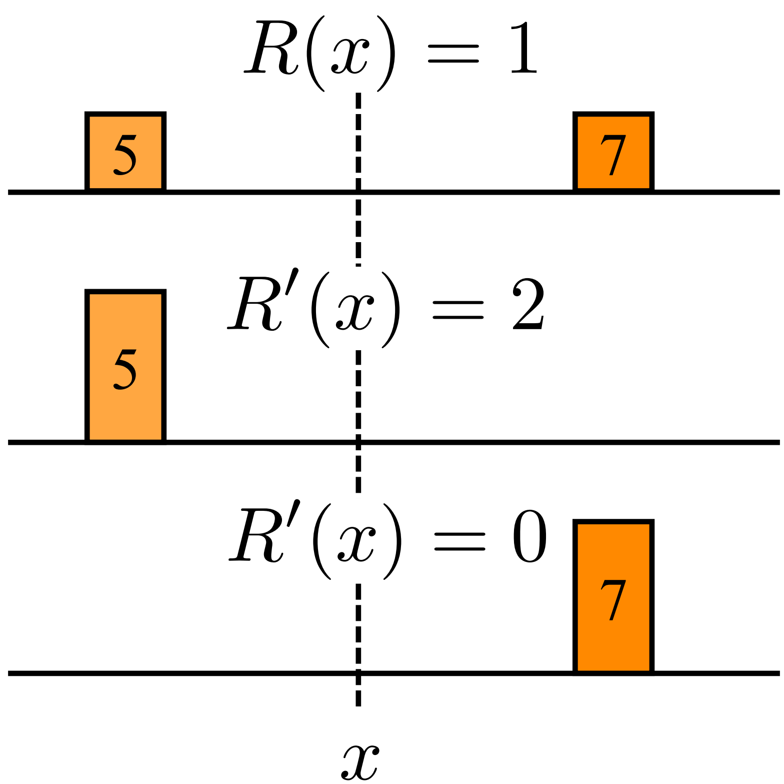
$\log(1/\varepsilon)$ Buffers of size k

start sampling after $O(1/\varepsilon^2)$ items



Reduces space usage to $\log^2(1/\varepsilon)/\varepsilon$ items from the stream.

Agarwal, Cormode, Huang, Phillips, Wei, Yi (2)



$R'(x)$ is a random variable now and

$$E[R'(x)] = R(x)$$

Reduces space usage to $\log^{3/2}(1/\varepsilon)/\varepsilon$ items from the stream.

Recap

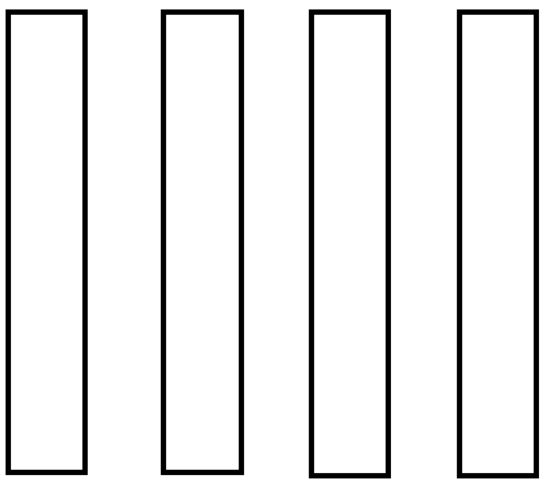
- Uniform sampling
  ✓ Fast, simple ✓ Fully mergeable ✗ Space $\tilde{O}(1/\varepsilon^2)$
- Greenwald Khanna (GK) sketch
  ✗ Slow, complex ✗ Not mergeable ✓✗ Space $\log(n)/\varepsilon$
- Felber-Ostrovsky, combines sampling and GK (2015)
  ✗ Slow, complex ✗ Not mergeable ✓ Space $\log(1/\varepsilon)/\epsilon$
- Manku-Rajagopalan-Lindsay (MRL)
  ✓ Fast, simple ✓ Fully mergeable ✓✗✗ Space $\log^2(n)/\varepsilon$
- Agarwal, Cormode, Huang, Phillips, Wei, Yi
  ✓✗ Fast, complex ✓ Fully mergeable ✓✗ Space $\log^{3/2}(1/\varepsilon)/\varepsilon$

Our goal

- Uniform sampling
  ✓ Fast, simple ✓ Fully mergeable ✗ Space $\tilde{O}(1/\varepsilon^2)$
- Greenwald Khanna (GK) sketch
  ✗ Slow, complex ✗ Not mergeable ✓✗ Space $\log(n)/\varepsilon$
- Felber-Ostrovsky, combines sampling and GK (2015)
  ✗ Slow, complex ✗ Not mergeable ✓ Space $\log(1/\varepsilon)/\epsilon$
- Manku-Rajagopalan-Lindsay (MRL)
  ✓ Fast, simple ✓ Fully mergeable ✓✗✗ Space $\log^2(n)/\varepsilon$
- Agarwal, Cormode, Huang, Phillips, Wei, Yi
  ✓✗ Fast, complex ✓ Fully mergeable ✓✗ Space $\log^{3/2}(1/\varepsilon)/\varepsilon$
- $\boxed{\text{Karnin, Lang, Liberty}}$
  ✓ Fast, simple ✓ Fully mergeable ✓ Space $\log(1/\varepsilon)/\epsilon$

Observation

The first buffers contribute very little to the error. They are “too good”.



$h = H$ .... $h = 2$ $h = 1$

$$w_h = 2^{h-1}$$

Weight of items in the level

$$m_h = 2^{H-h-1}$$

Number of compactions

Idea

TBD later

Let buffers shrink at-most-exponentially

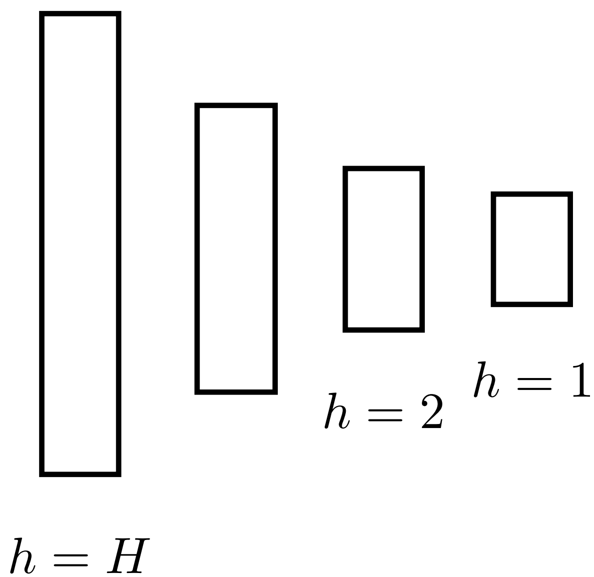
$$k_h \geq kc^{H-h}$$

$$w_h = 2^{h-1}$$

$$H \leq \log(n/ck) + 2$$

$$m_h \leq (2/c)^{H-h-1}$$

Number of compactions



Analysis

$R(h, x)$ the rank of x among

1. The items yielded by the compactor at height h
2. All the items stored in the compactors of heights $h' \leq h$

Claim, for $C = c^2(2c - 1)$

$$\Pr\left[|R(x, H') - R(x)| \geq \varepsilon n\right] \leq 2\exp\left(-C\varepsilon^2 k^2 2^{2(H-H')}\right)$$

Proof

Use Hoeffding's inequality on $\sum_{h=1}^{H}[R(x, h) - R(x, h-1)]$

Solution 1

Set $c = 2/3$ and $k_h = \lceil kc^{H-h} \rceil + 1$

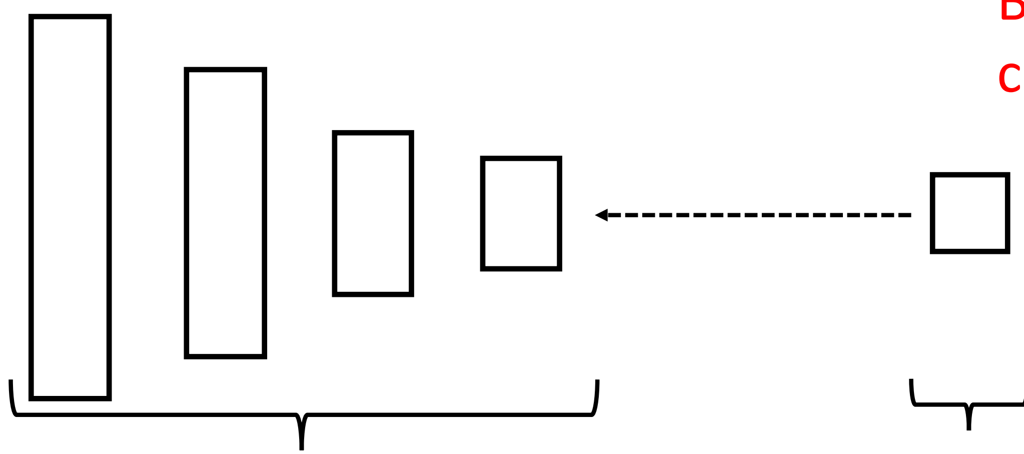
- Karnin, Lang, Liberty (1)

Fast, simple

Fully mergeable

Space $\sqrt{\log(1/\varepsilon)}/\varepsilon$

Better than previously conjectured optimal!



$\log(n)$ exponentially decreasing capacity buffers

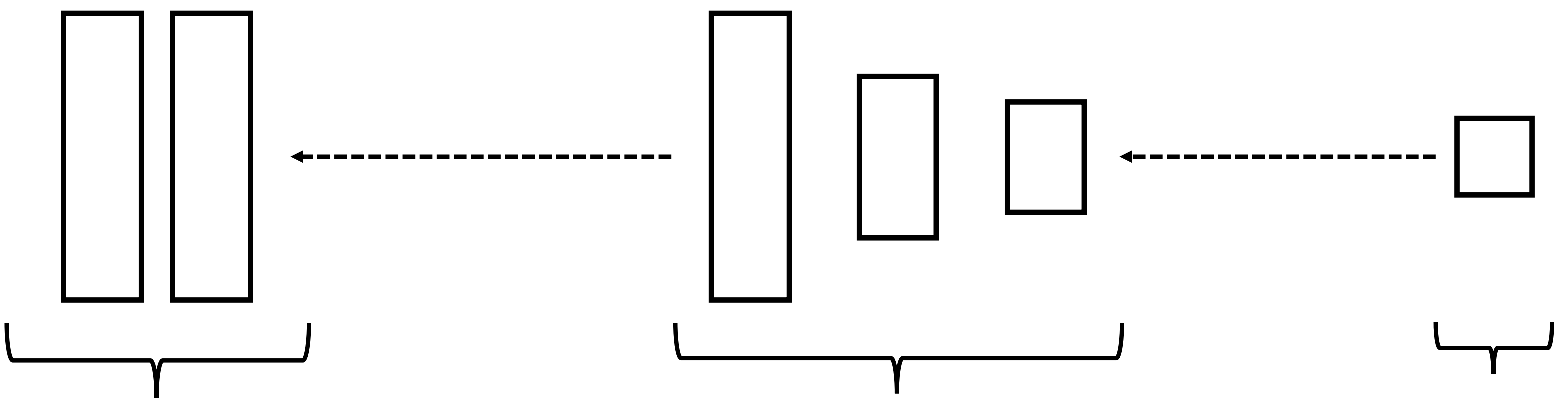
sampler replaces all buffers of size 2

Solution 2 (KLL + MRL)

Set $c = 2/3$ and $k_h = \lceil kc^{H-h} \rceil + 1$ except that the top $\log\log(1/\varepsilon)$ buffers all have capacity k.

- Karnin, Lang, Liberty (2)

Fast, simple Fully mergeable Space $\log^2\log(1/\varepsilon)/\varepsilon$



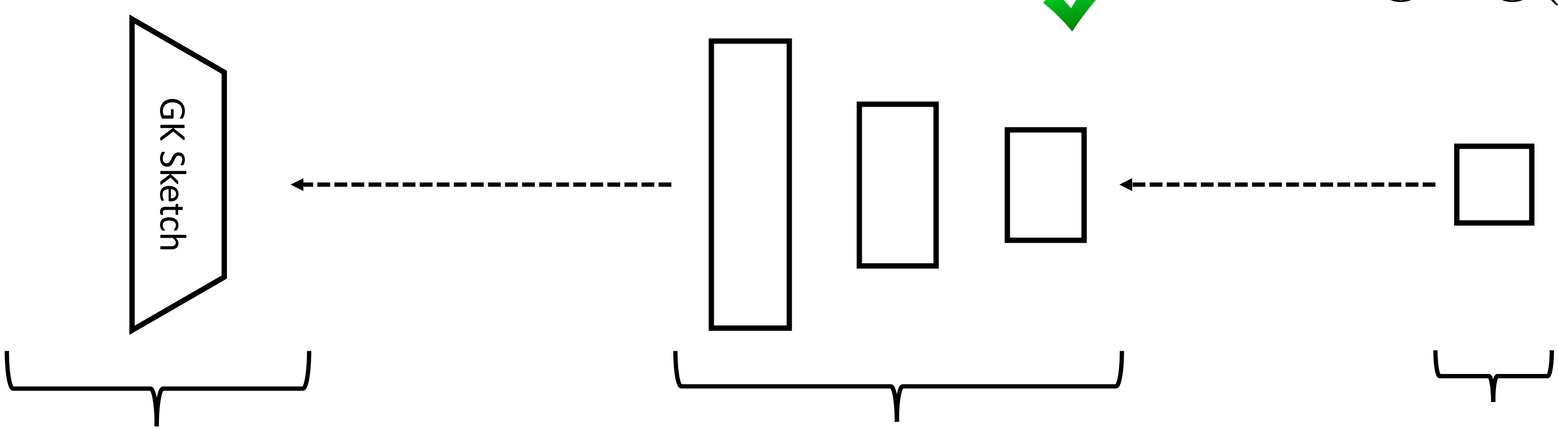
Solution 3 (KLL + GK)

Set $c = 2/3$ and $k_h = \lceil k c^{H-h} \rceil + 1$ replace the top $\log\log(1/\varepsilon)$ with a GK sketch k

Space Optimal!

- Karnin, Lang, Liberty (3)

  Fast, simple Fully mergeable Space $\log\log(1/\varepsilon)/\varepsilon$

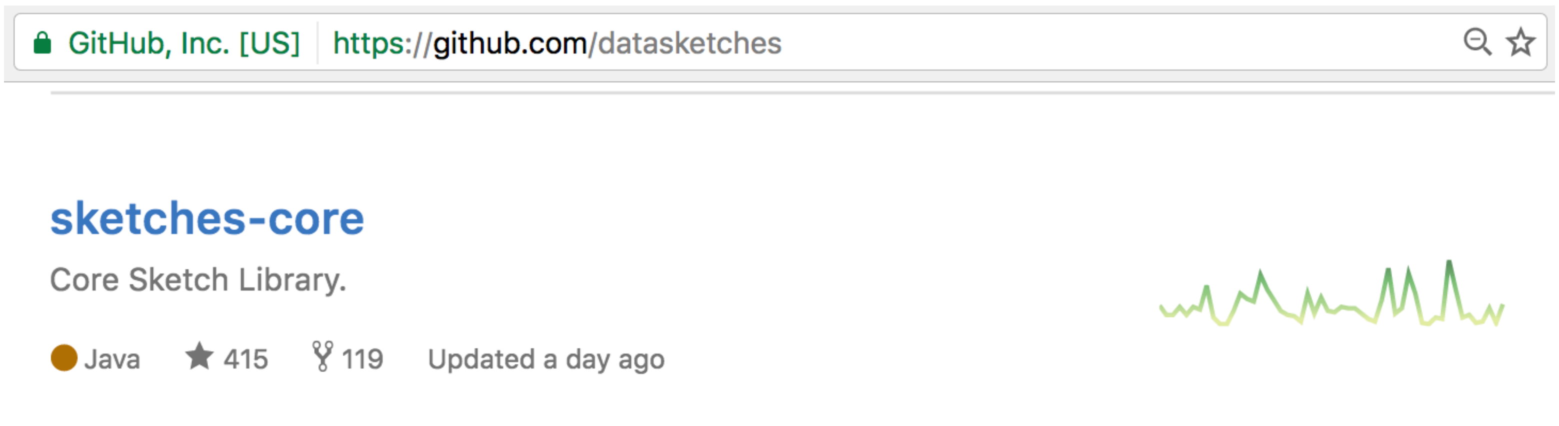


GK sketch replaces top $\log\log(1/\varepsilon)$ levels

$\log(n)$ exponentially decreasing capacity buffers

sampler replaces all buffers of size 2

Count Distinct
(Demo Only)



Assume you need to estimate the distribution of numbers in a file

```
$ head data.csv
0
1
0
3
0
2
3
7
3
2
```

In this one, row i tasks a value from [0,i] uniformly at random.

Some stats: there are 10,000,000 such numbers in this ~76Mb file.

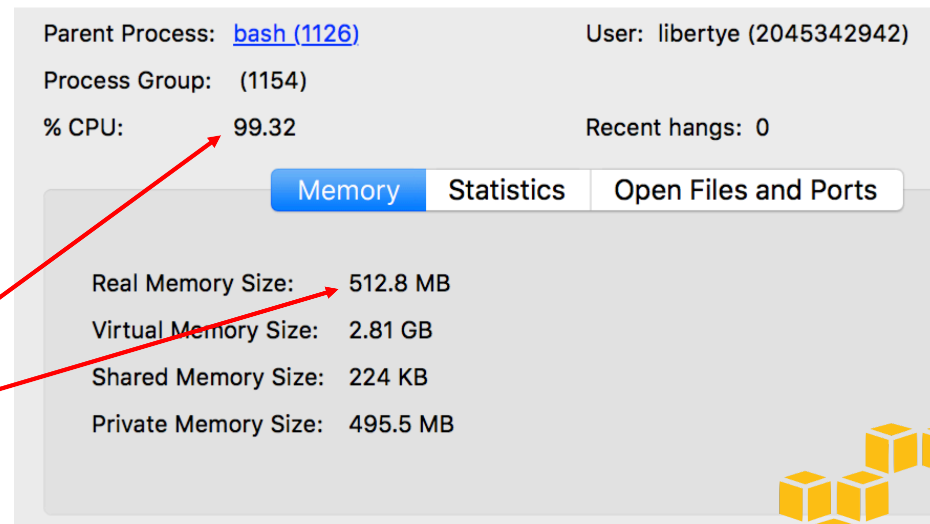
```
$ time wc -lc data.csv
 10000000 76046666 data.csv



real 0m0.101s
user 0m0.072s
sys 0m0.021s
```

Reading the file take ~1/10 seconds. We don’t foresee IO being an issue.

In python it looks like this:

```
$ cat quantiles.py
import sys
ints = sorted([int(x) for x in sys.stdin])
for i in range(0,len(ints),int(len(ints)/100)):
    print(str(ints[i]))
```

```
$ time cat data.csv |
python quantiles.py >
/dev/null

real 0m13.406s
user 0m12.937s
sys 0m0.407s
```



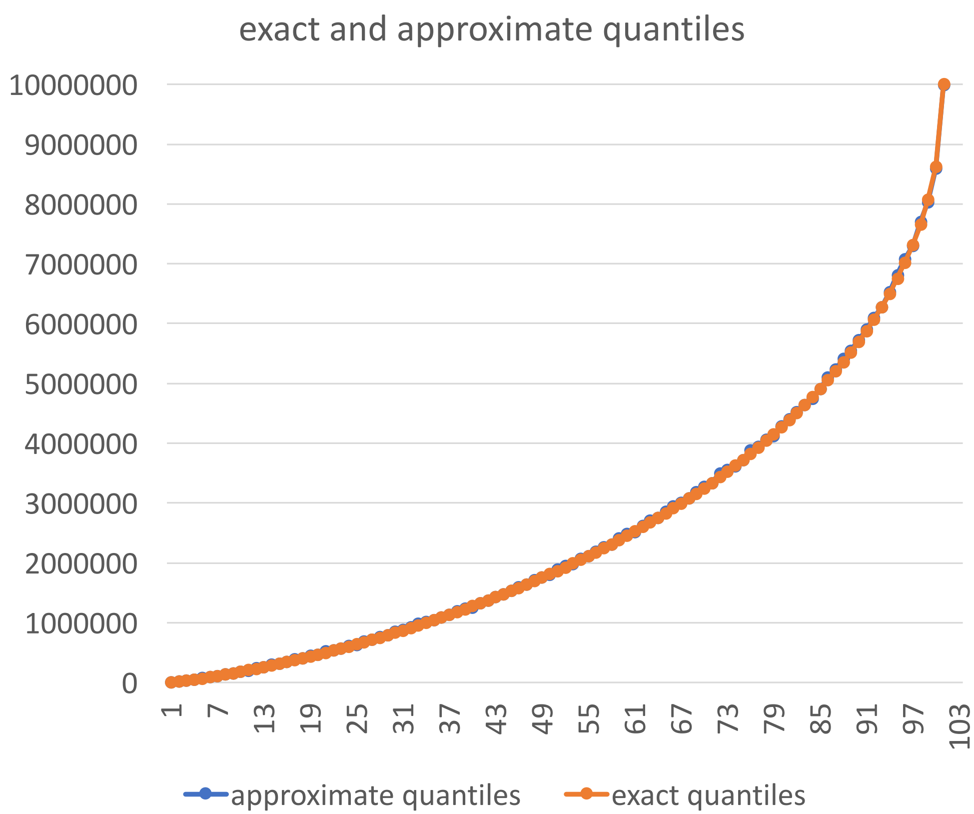
This is the way to do this with the sketching library

```
$ time cat data.csv | sketch rank
```

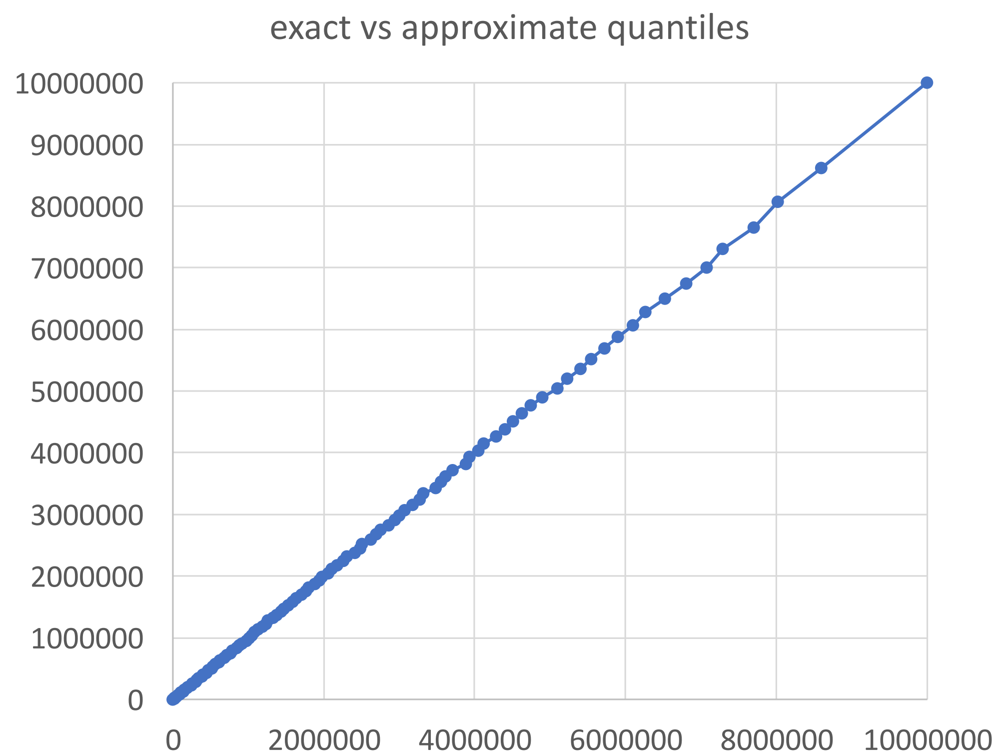
```
$ time cat data.csv |
sketch rank > /dev/null

real 0m1.495s
user 0m1.878s
sys 0m0.141s
```

Too fast to use the system monitor UI...

It uses ~ 4k of memory!

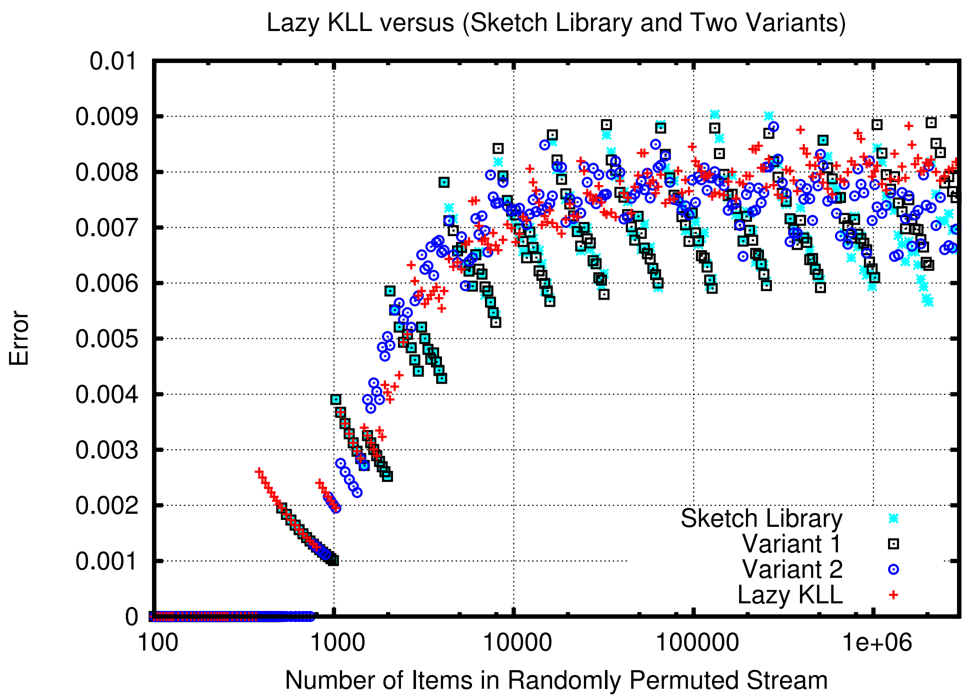


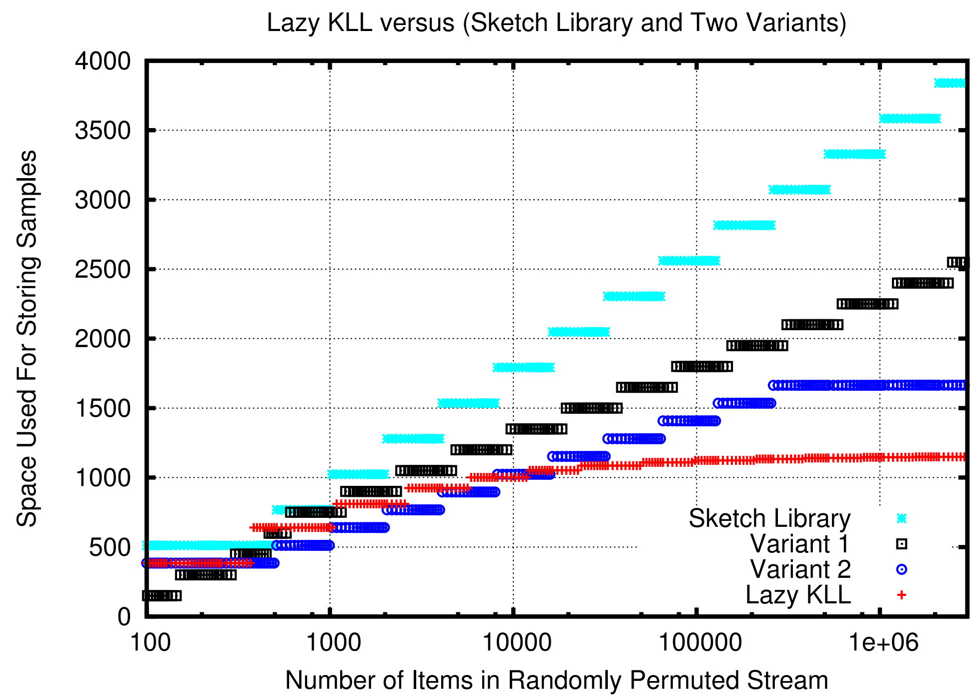
exact and approximate quantiles
10000000
9000000
8000000
7000000
6000000
5000000
4000000
3000000
2000000
1000000
0
1
7
13
19
25
31
37
43
49
55
61
67
73
79
85
91
97
103
approximate quantiles
exact quantiles



exact vs approximate quantiles
10000000
9000000
8000000
7000000
6000000
5000000
4000000
3000000
2000000
1000000
0
0
2000000
4000000
6000000
8000000
10000000

Some experimental results





Thank you!