

## Quantiles and Selection

Lecture 16

October 20, 2020

# Part I

## Introduction

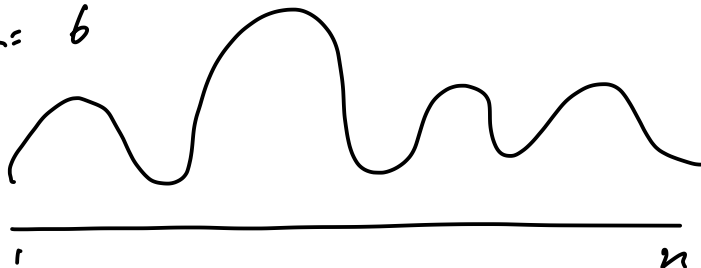
# Selection

**Selection:** Given a sequence of numbers  $a_1, a_2, \dots, a_n$  and integer  $k \in [n]$  want to find the rank  $k$  element (the  $k$ 'th element after sorting)

**Median:** rank  $n/2$  element

10, 100, 1, 5, 95, 26, ---, 5

$k = 6$



# Selection

**Selection:** Given a sequence of numbers  $a_1, a_2, \dots, a_n$  and integer  $k \in [n]$  want to find the rank  $k$  element (the  $k$ 'th element after sorting)

**Median:** rank  $n/2$  element

## Offline solutions:

- Sort and pick the  $k$ 'th element.  $O(n \log n)$  time. Can find all ranks in constant time after sorting.
- $O(n)$  time algorithm for Selection of given rank  $k$ . Randomized QuickSelect or deterministic Median-of-Medians algorithm (clever but slow).

# Selection in Streaming

**Question:** Suppose  $a_1, a_2, \dots, a_n$  arrive in a stream. Can we do Selection in small space?

# Selection in Streaming

**Question:** Suppose  $a_1, a_2, \dots, a_n$  arrive in a stream. Can we do Selection in small space?

**Exact Selection** in one pass requires  $\Omega(n)$  space. Need to store all elements so trivial solution is optimal.

# Selection in Streaming

**Question:** Suppose  $a_1, a_2, \dots, a_n$  arrive in a stream. Can we do Selection in small space?

**Exact Selection** in one pass requires  $\Omega(n)$  space. Need to store all elements so trivial solution is optimal.

## Relaxations:

- Approximate selection. Recall sampling to find  $\epsilon$ -approximate median using  $O(\frac{1}{\epsilon^2} \log(1/\delta))$  samples. Can do this in streaming with reservoir sampling.
- *Multiple* passes.
- Assume random order arrival of elements.

# Selection in Multiple Passes

**Multipass model:** See same stream  $p$  times for some  $p \geq 1$ . With larger  $p$  one can do more with same memory bound.

Initially motivated by database applications where random access main memory is small and large external memory (such as tapes) that allow for reasonably fast sequential scans.



# Selection in Multiple Passes

**Multipass model:** See same stream  $p$  times for some  $p \geq 1$ . With larger  $p$  one can do more with same memory bound.

Initially motivated by database applications where random access main memory is small and large external memory (such as tapes) that allow for reasonably fast sequential scans.

Selection in multiple passes:

- $\Theta(n)$  space allows **1** pass.
- $O(1)$  space. How many passes?

# Selection in Multiple Passes

**Multipass model:** See same stream  $p$  times for some  $p \geq 1$ . With larger  $p$  one can do more with same memory bound.

Initially motivated by database applications where random access main memory is small and large external memory (such as tapes) that allow for reasonably fast sequential scans.

Selection in multiple passes:

- $\Theta(n)$  space allows **1** pass.
- $O(1)$  space. How many passes?  $O(\log n)$  suffices. Implement Quick Select in  $O(1)$  space. *whp*

# Selection in Multiple Passes

**Multipass model:** See same stream  $p$  times for some  $p \geq 1$ . With larger  $p$  one can do more with same memory bound.

Initially motivated by database applications where random access main memory is small and large external memory (such as tapes) that allow for reasonably fast sequential scans.

Selection in multiple passes:

- $\Theta(n)$  space allows **1** pass.
- $O(1)$  space. How many passes?  $O(\log n)$  suffices. Implement Quick Select in  $O(1)$  space.
- $p$  passes?

# Selection in Multiple Passes

**Multipass model:** See same stream  $p$  times for some  $p \geq 1$ . With larger  $p$  one can do more with same memory bound.

Initially motivated by database applications where random access main memory is small and large external memory (such as tapes) that allow for reasonably fast sequential scans.

Selection in multiple passes:

- $\Theta(n)$  space allows **1** pass.
- $O(1)$  space. How many passes?  $O(\log n)$  suffices. Implement Quick Select in  $O(1)$  space.
- $p$  passes?  $O(n^{1/p} \text{polylog}(n))$  space suffices. Hence  $O(\sqrt{n} \log n)$  for **2** passes. [Munro-Paterson 1980]

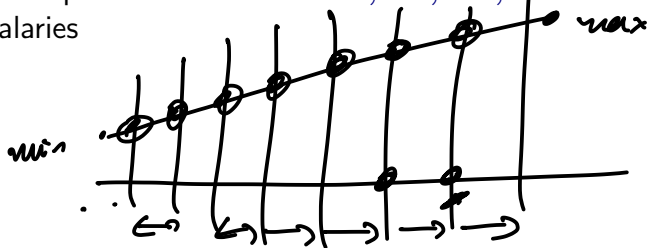
# Quantiles

Large numerical/ordered data: say heights/weights/salaries of the population of the country.

Exact selection is not as interesting as high-level summary. Pick some granularity and bucket data into groups of roughly equal size.

**Example:** For  $\alpha = 1, 2, \dots, 100$  want  $\alpha$  percentile salaries

More precision: For  $\alpha = 0.1, 0.2, \dots, 100$  want  $\alpha$  percentile salaries



# Quantiles

Large numerical/ordered data: say heights/weights/salaries of the population of the country.

Exact selection is not as interesting as high-level summary. Pick some granularity and bucket data into groups of roughly equal size.

**Example:** For  $\alpha = 1, 2, \dots, 100$  want  $\alpha$  percentile salaries

More precision: For  $\alpha = 0.1, 0.2, \dots, 100$  want  $\alpha$  percentile salaries

In terms of Selection:

- want rank  $k$  element for  $k = \frac{\alpha}{100}n$  for each  $\alpha$
- allows for  $\epsilon$ -approximate Selection (additive error  $\epsilon n$  where  $\epsilon$  is granularity in quantile)

# Quantile Summaries or Approximate Selection in Streaming

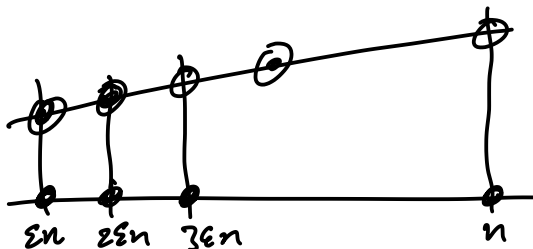
See stream of numbers  $a_1, a_2, \dots, a_n$ .

Parameter  $\epsilon \in (0, 1)$

Maintain a small space summary such that given any  $k \in [n]$  can output number  $a$  from stream such that

$$k - \epsilon n \leq \text{rank}(a) \leq k + \epsilon n$$

$\frac{1}{\epsilon}$



# Quantile Summaries or Approximate Selection in Streaming

See stream of numbers  $a_1, a_2, \dots, a_n$ .

Parameter  $\epsilon \in (0, 1)$

Maintain a small space summary such that given any  $k \in [n]$  can output number  $a$  from stream such that

$$k - \epsilon n \leq \text{rank}(a) \leq k + \epsilon n$$

Offline: can do with  $O(1/\epsilon)$  space. Store rank  $\epsilon i/n$  elements for  $i = 1, 2, \dots, 1/\epsilon$

**Q:** Can we do it in streaming and how much space do we need?



# Quantile Summaries or Approximate Selection in Streaming

See stream of numbers  $a_1, a_2, \dots, a_n$  Parameter  $\epsilon \in (0, 1)$

Maintain a small space summary such that given any  $k \in [n]$  can output number  $a$  from stream such that

$$k - \epsilon n \leq \text{rank}(a) \leq k + \epsilon n$$

**Q:** Can we do it in streaming and how much space do we need?

- $O(\frac{1}{\epsilon} \log^2 n)$  space using merge and reduce approach
- Involved  $O(\frac{1}{\epsilon} \log(n/\epsilon))$  space algorithm that is near optimal

Both are deterministic algorithms. Can be used to derive Munro-Paterson multi-pass Selection algorithm

## Part II

# Approximate Quantiles in Streaming

# Quantile Summary

See stream of numbers  $a_1, a_2, \dots, a_n$ . Parameter  $\epsilon \in (0, 1)$

**Note:** Items can be from any ordered set, use only comparison

What should we store?

# Quantile Summary

See stream of numbers  $a_1, a_2, \dots, a_n$ . Parameter  $\epsilon \in (0, 1)$

**Note:** Items can be from any ordered set, use only comparison

What should we store? Take cue from offline solution. Equally spaced  $1/\epsilon$  elements from sorted list.

# Quantile Summary

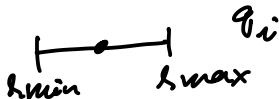
See stream of numbers  $a_1, a_2, \dots, a_n$ . Parameter  $\epsilon \in (0, 1)$

**Note:** Items can be from any ordered set, use only comparison

What should we store? Take cue from offline solution. Equally spaced  $1/\epsilon$  elements from sorted list.

## Quantile Summary:

- $Q = \{q_1, q_2, \dots, q_\ell\}$  where each  $q_i$  is an element of stream. Wlog  $q_1 < q_2 < \dots < q_\ell$  and  $q_1$  is smallest and  $q_\ell$  is largest in stream
- For each  $q_i \in Q$  an interval  $I(q_i) = [\text{rmin}_Q(q_i), \text{rmax}_Q(q_i)]$  where  $\text{rmin}_Q(q_i) \leq \text{rank}(q_i) \leq \text{rmax}_Q(q_i)$



# Quantile Summary

## Quantile Summary:

- $Q = \{q_1, q_2, \dots, q_\ell\}$ . Also  $q_1 < q_2 < \dots < q_\ell$  and  $q_1$  is smallest and  $q_\ell$  is largest
- For each  $q_i \in Q$  an interval  $I(q_i) = [\text{rmin}_Q(q_i), \text{rmax}_Q(q_i)]$  where  $\text{rmin}_Q(q_i) \leq \text{rank}(q_i) \leq \text{rmax}_Q(q_i)$

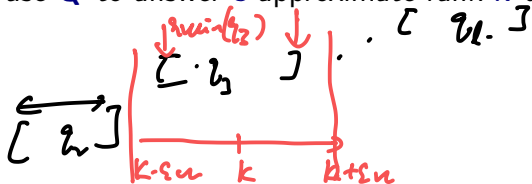
Given  $k \in [n]$  want to use  $Q$  to answer  $\epsilon$ -approximate rank  $k$  query.  
How?

$$k = 1000$$

want any  
#  $a \in S$

$$[q_i]$$

$$k - \epsilon n \leq \text{rank}(a) \leq k + \epsilon n$$



# Quantile Summary

## Quantile Summary:

- $Q = \{q_1, q_2, \dots, q_\ell\}$ . Also  $q_1 < q_2 < \dots < q_\ell$  and  $q_1$  is smallest and  $q_\ell$  is largest
- For each  $q_i \in Q$  an interval  $I(q_i) = [\text{rmin}_Q(q_i), \text{rmax}_Q(q_i)]$  where  $\text{rmin}_Q(q_i) \leq \text{rank}(q_i) \leq \text{rmax}_Q(q_i)$

Given  $k \in [n]$  want to use  $Q$  to answer  $\epsilon$ -approximate rank  $k$  query. How?

Suppose  $I(q_i) \subseteq [k - \epsilon n, k + \epsilon n]$  then it is clear that  $q_i$  is good to output since

$$k - \epsilon n \leq \text{rmin}(q_i) \leq \text{rank}(q_i) \leq \text{rmax}(q_i) \leq k + \epsilon n.$$

# $\epsilon$ -Approximate Quantile Summary

## Quantile Summary:

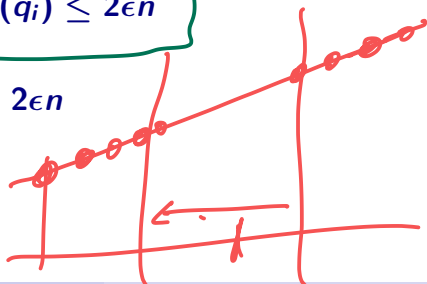
- $Q = \{q_1, q_2, \dots, q_\ell\}$ . Also  $q_1 < q_2 < \dots < q_\ell$  and  $q_1$  is smallest and  $q_\ell$  is largest
- For each  $q_i \in Q$  an interval  $I(q_i) = [\text{rmin}_Q(q_i), \text{rmax}_Q(q_i)]$  where  $\text{rmin}_Q(q_i) \leq \text{rank}(q_i) \leq \text{rmax}_Q(q_i)$

Maintain key invariant: For each  $i$ ,

$$\rightarrow \boxed{\text{rmax}(q_{i+1}) - \text{rmin}(q_i) \leq 2\epsilon n}$$

also implies  $\text{rank}(q_{i+1}) - \text{rank}(q_i) \leq 2\epsilon n$

$$\begin{array}{l} \overbrace{[q_{i+1}]} \\ [q_i] \\ \text{rmin}(q_i) \quad \text{rmax}(q_i) \end{array} \leq \underline{\underline{2\epsilon n}}$$





# $\epsilon$ -Approximate Quantile Summary

## Quantile Summary:

- $Q = \{q_1, q_2, \dots, q_\ell\}$ . Also  $q_1 < q_2 < \dots < q_\ell$  and  $q_1$  is smallest and  $q_\ell$  is largest
- For each  $q_i \in Q$  an interval  $I(q_i) = [\text{rmin}_Q(q_i), \text{rmax}_Q(q_i)]$  where  $\text{rmin}_Q(q_i) \leq \text{rank}(q_i) \leq \text{rmax}_Q(q_i)$

**Maintain key invariant:** For each  $i$ ,

$$\text{rmax}(q_{i+1}) - \text{rmin}(q_i) \leq 2\epsilon n$$

also implies  $\text{rank}(q_{i+1}) - \text{rank}(q_i) \leq 2\epsilon n$

## Lemma

*With invariant quantile summary can be used to answer  $\epsilon$ -approximate rank queries.*

# Proof of Lemma

**Maintain key invariant:** For each  $i$ ,

$$\text{rmax}(q_{i+1}) - \text{rmin}(q_i) \leq 2\epsilon n$$

**Claim:** There exists  $q_j$  such that  $I(q_j) \subseteq [k - \epsilon n, k + \epsilon n]$

# Proof of Lemma

**Maintain key invariant:** For each  $i$ ,

$$\text{rmax}(q_{i+1}) - \text{rmin}(q_i) \leq 2\epsilon n$$

**Claim:** There exists  $q_j$  such that  $I(q_j) \subseteq [k - \epsilon n, k + \epsilon n]$

- If  $k \geq (1 - \epsilon)n$  then  $q_\ell$  satisfies condition.

# Proof of Lemma

**Maintain key invariant:** For each  $i$ ,

$$\text{rmax}(q_{i+1}) - \text{rmin}(q_i) \leq 2\epsilon n$$

**Claim:** There exists  $q_j$  such that  $I(q_j) \subseteq [k - \epsilon n, k + \epsilon n]$

- If  $k \geq (1 - \epsilon)n$  then  $q_\ell$  satisfies condition.
- Let  $j$  be smallest index such that  $\text{rmax}(q_j) \geq k + \epsilon n$  (exists since  $\text{rmax}(q_\ell) = n$  and  $k < (1 - \epsilon)n$ ).

# Proof of Lemma

**Maintain key invariant:** For each  $i$ ,

$$\mathbf{rmax}(q_{i+1}) - \mathbf{rmin}(q_i) \leq 2\epsilon n$$

**Claim:** There exists  $q_j$  such that  $I(q_j) \subseteq [k - \epsilon n, k + \epsilon n]$

- If  $k \geq (1 - \epsilon)n$  then  $q_\ell$  satisfies condition.
- Let  $j$  be smallest index such that  $\mathbf{rmax}(q_j) \geq k + \epsilon n$  (exists since  $\mathbf{rmax}(q_\ell) = n$  and  $k < (1 - \epsilon)n$ ).
- $q_{j-1}$  satisfies condition. Suppose not. By choice of  $j$ ,  $\mathbf{rmax}(q_{j-1}) < k + \epsilon n$ . Since condition is not satisfied by  $q_{j-1}$ ,  $\mathbf{rmin}(q_{j-1}) < k - \epsilon n$  but then

$$\mathbf{rmax}(q_j) - \mathbf{rmin}(q_{j-1}) > k + \epsilon n - (k - \epsilon n) > 2\epsilon n$$

contradiction to invariant.

# Maintaining $\epsilon$ -Approx Quantile Summary in Streaming

**Question:** How to maintain  $\epsilon$ -approximate quantile summary in small space in streaming setting?

**Merge and Reduce/Prune Framework**  
(also useful in other settings)

**Merge:** given  $\epsilon_1$ -approx  $Q_1$  for multiset  $S_1$  and  $\epsilon_2$ -approx  $Q_2$  for multiset  $S_2$  obtain approx  $Q$  for  $S = S_1 \cup S_2$

**Prune:** Given  $\epsilon$ -approx  $Q$  for  $S$  of size  $\ell$ , prune to size  $h$  without increasing error by too much

# Merging Summaries

$Q_1 = \{q_1, q_2, \dots, q_\ell\}$  and intervals  $I_1(q_1), \dots, I_1(q_\ell)$  for multiset  $S_1$  with  $n_1 = |S_1|$   *$\epsilon_1$ -approx*

$Q_2 = \{s_1, s_2, \dots, s_m\}$  and intervals  $I_2(s_1), \dots, I_2(s_m)$  for multiset  $S_2$  with  $n_2 = |S_2|$   *$\epsilon_2$ -approx*

$Q = \{z_1, z_2, \dots, z_{\ell+m}\}$  which is sorted version of  $\{q_1, q_2, \dots, q_\ell, s_1, \dots, s_m\}$  for multiset  $S = S_1 \uplus S_2$  with  $n = n_1 + n_2$   *$=$*

# Merging Summaries

$Q_1 = \{q_1, q_2, \dots, q_\ell\}$  and intervals  $l_1(q_1), \dots, l_1(q_\ell)$  for multiset  $S_1$  with  $n_1 = |S_1|$

$Q_2 = \{s_1, s_2, \dots, s_m\}$  and intervals  $l_2(s_1), \dots, l_2(s_m)$  for multiset  $S_2$  with  $n_2 = |S_2|$

$Q = \{\underline{z_1}, \underline{z_2}, \dots, \underline{z_{\ell+m}}\}$  which is sorted version of  $\{q_1, q_2, \dots, q_\ell, s_1, \dots, s_m\}$  for multiset  $S = S_1 \uplus S_2$  with  $n = n_1 + n_2$

How do we find intervals for  $Q$  while maintaining key invariant?

Consider  $z_i$  and assume wlog that  $\underline{z_i} = \underline{q_j}$  for some  $1 \leq j \leq \ell$



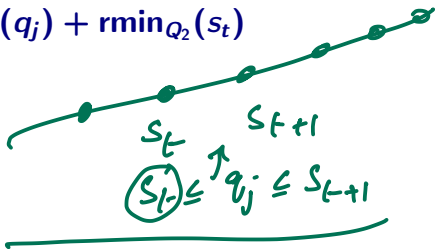
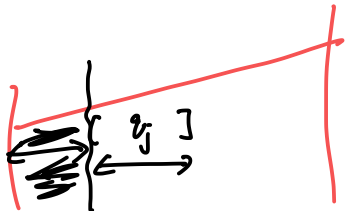
# Merging

Consider  $z_i$  and assume wlog that  $z_i = q_j$  for some  $1 \leq j \leq \ell$

Find  $s_t, s_{t+1}$  such that  $s_t \leq q_j \leq s_{t+1}$  (ignore corner cases)

We know that  $\text{rmin}_{Q_1}(q_j)$  elements in  $S_1$  are smaller than  $q_j$  and also  $\text{rmin}_{Q_2}(s_t)$  elements in  $S_2$  are smaller than  $q_j$ . Hence it safe to set

$$\underline{\underline{\text{rmin}_Q(z_i) = \text{rmin}_{Q_1}(q_j) + \text{rmin}_{Q_2}(s_t)}}$$



# Merging

Consider  $z_i$  and assume wlog that  $z_i = q_j$  for some  $1 \leq j \leq \ell$

Find  $s_t, s_{t+1}$  such that  $s_t \leq q_j \leq s_{t+1}$  (ignore corner cases)

We know that  $\text{rmin}_{Q_1}(q_j)$  elements in  $S_1$  are smaller than  $q_j$  and also  $\text{rmin}_{Q_2}(s_t)$  elements in  $S_2$  are smaller than  $q_j$ . Hence it safe to set

$$\text{rmin}_Q(z_i) = \text{rmin}_{Q_1}(q_j) + \text{rmin}_{Q_2}(s_t)$$

Similarly it is safe to set

$$\text{rmax}_Q(z_i) = \text{rmax}_{Q_1}(q_j) + \text{rmax}_{Q_2}(s_{t+1}) - 1$$

# Merging

## Lemma

If  $Q_1$  is an  $\epsilon_1$ -approx quantile summary for  $S_1$  and  $Q_2$  is an  $\epsilon_2$ -approx quantile summary for  $S_2$  then  $Q$  is an  $\epsilon = \max\{\epsilon_1, \epsilon_2\}$ -approx quantile summary for  $S = S_1 \uplus S_2$ .

Hence error does not increase but  $|Q| = |Q_1| + |Q_2|$ .

For proof need to verify key invariant.  $Q = \{z_1, z_2, \dots, z_{\ell+m}\}$ .  
Need to show that

$$\text{rmax}_Q(z_{i+1}) - \text{rmin}_Q(z_i) \leq 2\epsilon(n_1 + n_2).$$

# Merging Analysis

Need to show that

$$\text{rmax}_Q(z_{i+1}) - \text{rmin}_Q(z_i) \leq 2\epsilon(n_1 + n_2).$$

**Case 1:**  $z_i, z_{i+1}$  in same summary, say  $Q_1$  wlog. Then  $z_i = q_j$  and  $z_{i+1} = q_{j+1}$  for some  $j$ .

# Merging Analysis

Need to show that

$$\text{rmax}_Q(z_{i+1}) - \text{rmin}_Q(z_i) \leq 2\epsilon(n_1 + n_2).$$

**Case 1:**  $z_i, z_{i+1}$  in same summary, say  $Q_1$  wlog. Then  $z_i = q_j$  and  $z_{i+1} = q_{j+1}$  for some  $j$ .

This implies that there are  $s_t, s_{t+1}$  in  $Q_2$  such that  $s_t \leq q_j < q_{j+1} \leq s_{t+1}$ .

$$\begin{aligned} \text{Hence } & \text{rmax}_Q(z_{i+1}) - \text{rmin}_Q(z_i) \\ &= \text{rmax}_{Q_1}(q_{j+1}) + \text{rmax}_{Q_2}(s_{t+1}) - 1 - (\text{rmin}_{Q_1}(q_j) + \text{rmin}_{Q_2}(s_t)) \\ &\leq (\text{rmax}_{Q_1}(q_{j+1}) - \text{rmin}_{Q_1}(q_j)) + (\text{rmax}_{Q_2}(s_{t+1}) - \text{rmin}_{Q_2}(s_t)) \\ &\leq 2\epsilon n_1 + 2\epsilon n_2 \leq 2\epsilon(n_1 + n_2) \end{aligned}$$

# Merging Analysis

**Case 2:**  $z_i, z_{i+1}$  in different summaries, say  $Q_1, Q_2$  wlog. Then  $z_i = q_j$  and  $z_{i+1} = s_{t+1}$  for some  $j, t$ .

# Merging Analysis

**Case 2:**  $z_i, z_{i+1}$  in different summaries, say  $Q_1, Q_2$  wlog. Then  $z_i = q_j$  and  $z_{i+1} = s_{t+1}$  for some  $j, t$ .

This implies that  $s_t \leq q_j \leq s_{t+1} \leq q_{j+1}$  (ignoring corner cases)

Hence  $\text{rmax}_Q(z_{i+1}) - \text{rmin}_Q(z_i)$

$$\begin{aligned} &= \text{rmax}_{Q_1}(q_{j+1}) + \text{rmax}_{Q_2}(s_{t+1}) - 1 - (\text{rmin}_{Q_1}(q_j) + \text{rmin}_{Q_2}(s_t)) \\ &\leq (\text{rmax}_{Q_1}(q_{j+1}) - \text{rmin}_{Q_1}(q_j)) + (\text{rmax}_{Q_2}(s_{t+1}) - \text{rmin}_{Q_2}(s_t)) \\ &\leq 2\epsilon n_1 + 2\epsilon n_2 \leq 2\epsilon(n_1 + n_2) \end{aligned}$$

# Pruning/Reducing Summary

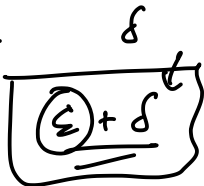
Merging keeps accuracy but increases summary size.

**Reduce/Prune:** reduce size at expense of accuracy.

## Lemma

Given  $\epsilon$ -approx quantile  $Q$  and integer  $h \geq 3$  can find  $Q'$  such that  $|Q'| \leq h + 1$  and  $Q'$  is  $\epsilon'$ -approximate for  $\epsilon' \leq \epsilon + \frac{1}{2h}$ .

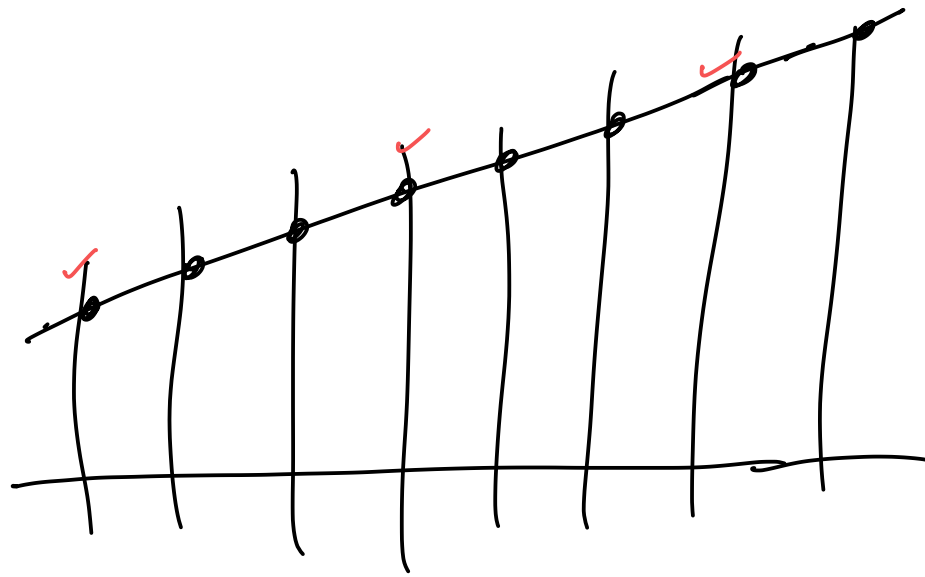
$\epsilon$ -approx



$O\left(\frac{1}{\delta}\right)$

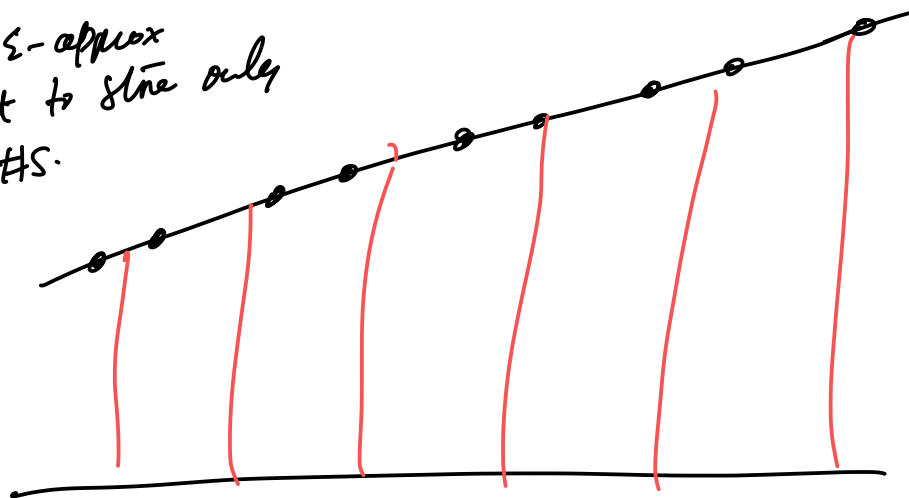
The diagram consists of a hand-drawn box. Inside the box, there is a circle containing the Greek letter epsilon ( $\epsilon$ ), followed by a plus sign and the Greek letter delta ( $\delta$ ). Above the box, the Greek letter delta ( $\delta$ ) is written. Below the box, there are two parallel horizontal lines. To the right of the box, the expression  $O\left(\frac{1}{\delta}\right)$  is written, with two parallel horizontal lines underneath it.





$$\epsilon = \frac{1}{100} \quad 100 \text{ points}$$

$\mathcal{Q} = \epsilon$ -approx  
 want to stay only  
 15 #s.



# Pruning/Reducing Summary

Merging keeps accuracy but increases summary size.

**Reduce/Prune:** reduce size at expense of accuracy.

## Lemma

Given  $\epsilon$ -approx quantile  $Q$  and integer  $h \geq 3$  can find  $Q'$  such that  $|Q'| \leq h + 1$  and  $Q'$  is  $\epsilon'$ -approximate for  $\epsilon' \leq \epsilon + \frac{1}{2h}$ .

$Q = \{q_1, q_2, \dots, q_\ell\}$  and wlog assume  $\ell > h + 1$ .

Query  $Q$  for ranks 1,  $n/h$ ,  $2n/h, \dots, n$ .



Create  $Q'$  from output of queries. Use same intervals as those in  $Q$ .

# Pruning/Reducing Analysis

$Q = \{q_1, q_2, \dots, q_\ell\}$  and wlog assume  $\ell > h + 1$ .

Query  $Q$  for ranks  $1, \frac{n/h}{2}, \dots, n$ .

$Q' = \{q'_1, q'_2, \dots, q'_{h+1}\}$

Suppose  $q'_i = q_a$  and  $q'_{i+1} = q_b$  for some  $a < b$ .

$I(q_a) \subseteq [in/h - \epsilon n, in/h + \epsilon n]$  and

$I(q_b) \subseteq [(i+1)n/h - \epsilon n, (i+1)n/h + \epsilon n]$

# Pruning/Reducing Analysis

$Q = \{q_1, q_2, \dots, q_\ell\}$  and wlog assume  $\ell > h + 1$ .

Query  $Q$  for ranks  $1, \frac{n/h}{2}, \dots, n$ .

$Q' = \{q'_1, q'_2, \dots, q'_{h+1}\}$

Suppose  $q'_i = q_a$  and  $q'_{i+1} = q_b$  for some  $a < b$ .

$I(q_a) \subseteq [in/h - \epsilon n, in/h + \epsilon n]$  and

$I(q_b) \subseteq [(i+1)n/h - \epsilon n, (i+1)n/h + \epsilon n]$

Therefore,

$$\begin{aligned} r_{\max_{Q'}(q'_{i+1})} - r_{\min_{Q'}(q'_i)} &\leq (i+1)n/h + \epsilon n - (in/h - \epsilon n) \\ &\leq 2\epsilon n + n/h \\ &\leq 2(\epsilon + 1/(2h))n. \end{aligned}$$

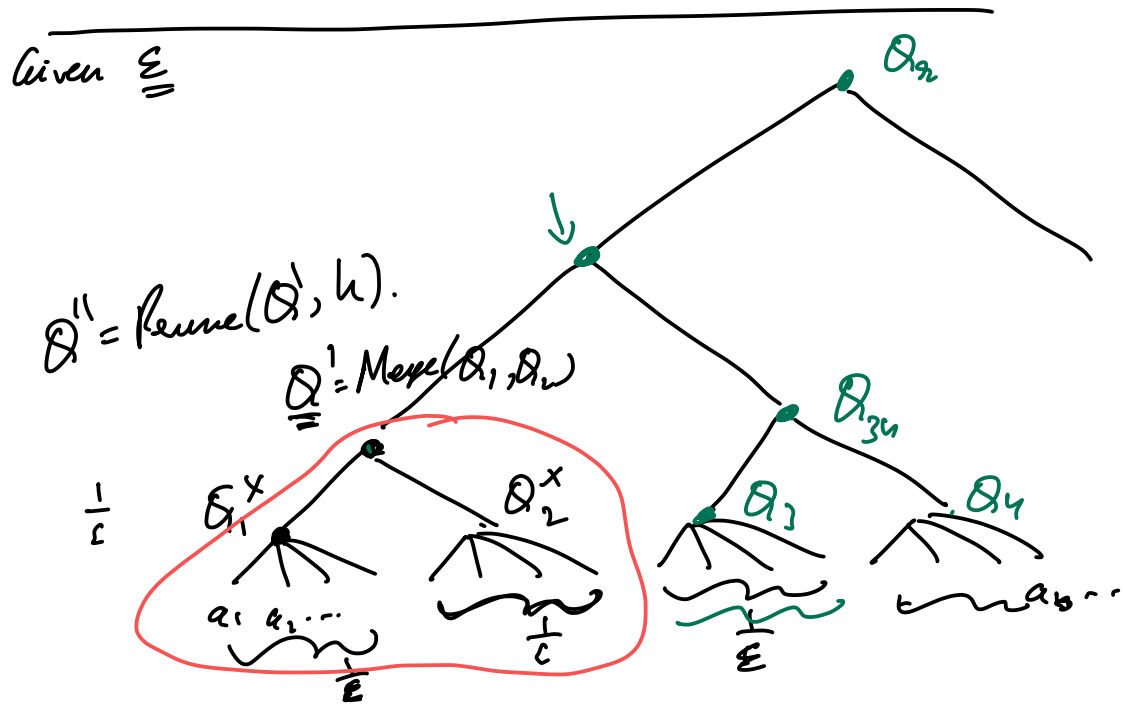
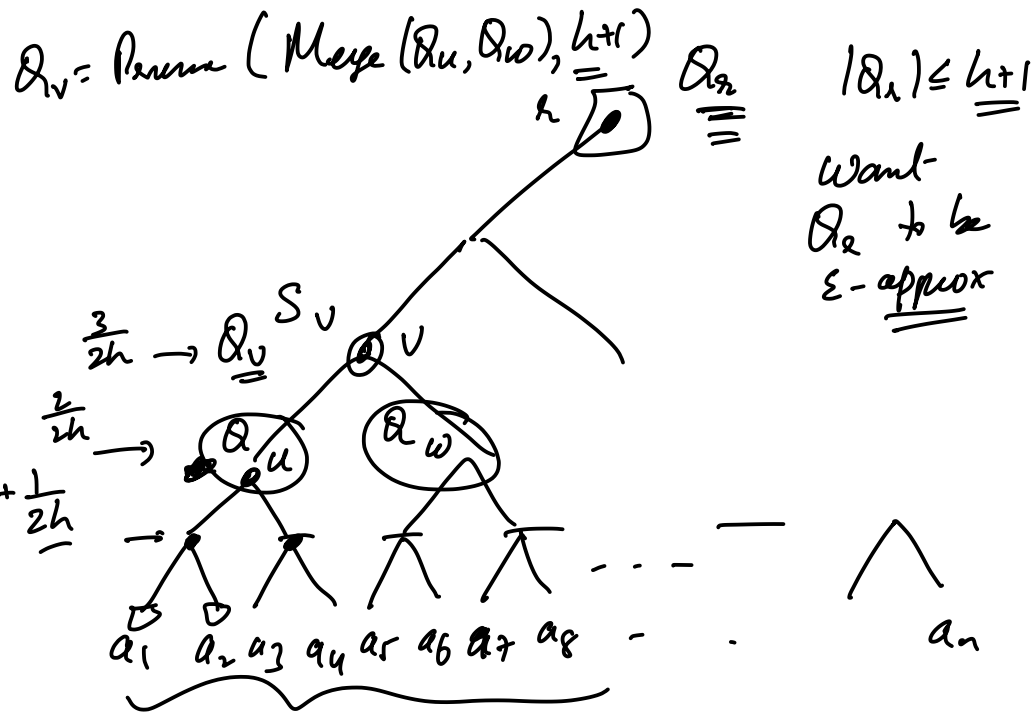
# Merge and Reduce Streaming Quantiles

Stream:  $a_1, a_2, \dots, a_n$  and given  $\epsilon \in (0, 1)$

Want to maintain  $\epsilon$ -approximate quantile summary.

$O(\frac{1}{\epsilon} \log^2 n)$  space algorithm based on reduce and merge.

- Come up with a solution as if the whole stream is available offline
- Show how it can implemented in small space in streaming setting.



Suppose we fix  $h$ .

$$|Q_R| \leq h+1.$$

Every summary has size  $\leq (h+1)$

$$\text{Total space} = O(\log n \cdot \underline{(h+1)}).$$

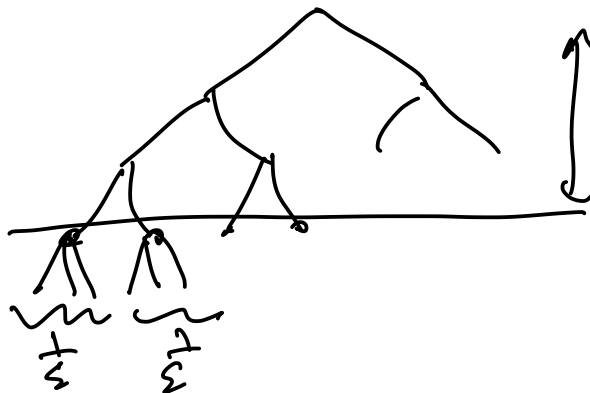
What about accuracy.

$$\log n \cdot \frac{\log n}{\varepsilon} = \boxed{\frac{1}{\varepsilon} \log^2 n}$$

$$\varepsilon_n = \varepsilon_{\text{leaves}} + d \cdot \frac{1}{2h} \leq \varepsilon.$$

~~h~~       $\downarrow$   $\frac{\varepsilon}{2}$        $\downarrow$   $\frac{\varepsilon}{2}$

$$d = \log n \cdot \frac{1}{2h} \leq \frac{\varepsilon}{2}$$
$$\Rightarrow h = \frac{2 \log n}{\varepsilon}$$

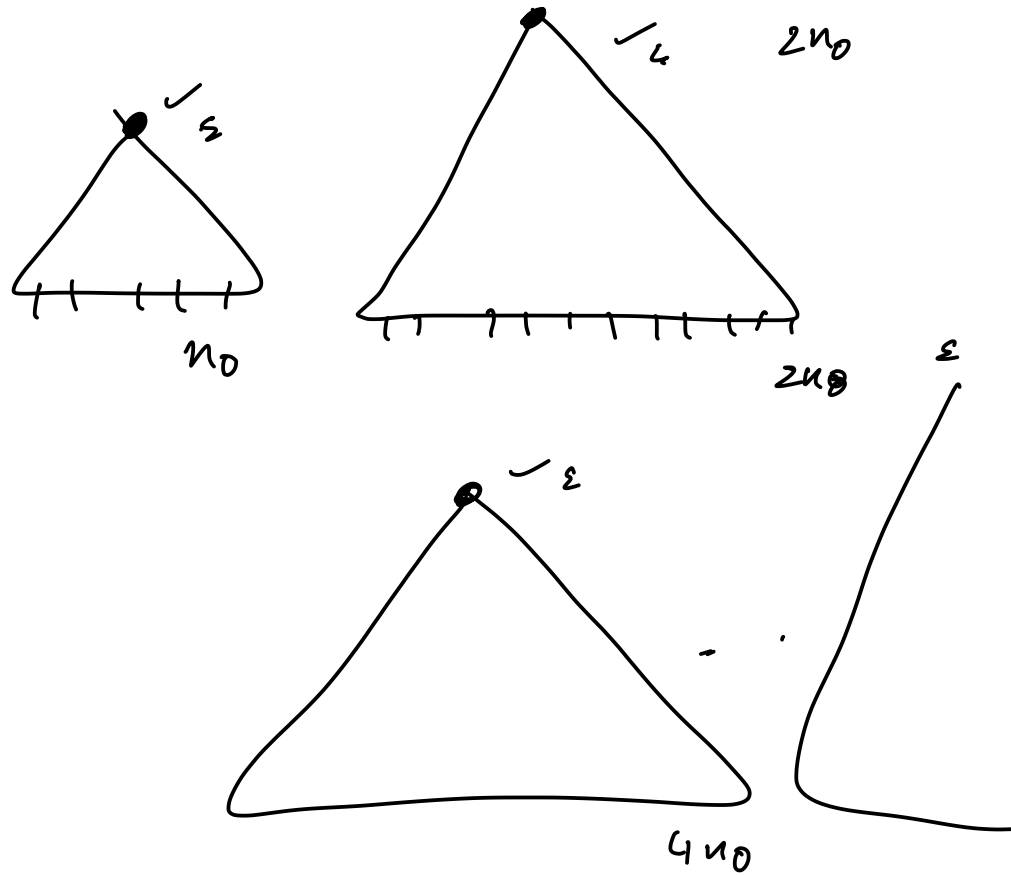


$$\log \varepsilon n$$
$$O\left(\frac{1}{\varepsilon} \log^2 \varepsilon n\right)$$

doubling width

$\epsilon$ : fixed

start with some constant  $\circ$   $n_0 = \underline{\underline{1000}}$



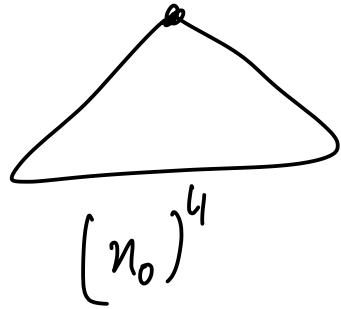
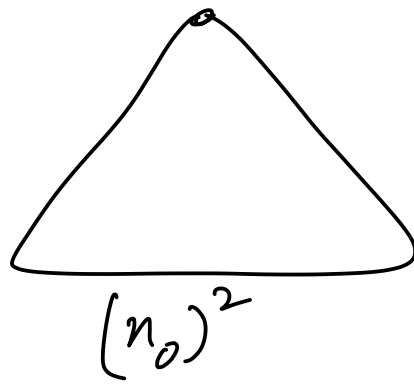
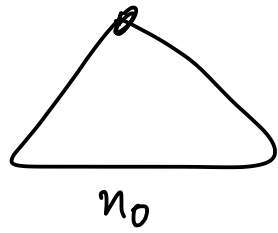
$$\frac{1}{\epsilon} \log^2 n_0 + \frac{1}{\epsilon} \log^2 2n_0 + \frac{1}{\epsilon} \log^2 4n_0$$

$$+ \dots + \frac{1}{\epsilon} \log^2 \underbrace{2^k \cdot n_0}$$

$$= n.$$

$$\leq \underbrace{\log n}_{\epsilon} \left( \frac{1}{\epsilon} \log^2 n \right) \frac{1}{\epsilon} \log^2 n.$$





$$2^{2^i} n_0$$

...

$$O\left(\frac{1}{\varepsilon} \log^2 n\right).$$

# Merge and Reduce for Streaming Quantiles

Stream:  $a_1, a_2, \dots, a_n$  and given  $\epsilon \in (0, 1)$

- Imagine a rooted binary tree with  $a_1, a_2, \dots, a_n$  as leaves in that order (not sorted)
- At each internal node  $v$  let  $S_v$  be leaves under  $v$ .
- Compute a summary  $Q_v$  for  $S_v$  bottom up.  $Q_r$  is output where  $r$  is root. Summary at leaf is optimal simply stores element.
- To compute  $Q_v$  with children  $a, b$  Merge  $Q_a$  and  $Q_b$  and Prune to size  $h + 1$
- Guarantees that  $Q_r$  has size  $h + 1$

# Merge and Reduce for Streaming Quantiles

Stream:  $a_1, a_2, \dots, a_n$  and given  $\epsilon \in (0, 1)$

- Imagine a rooted binary tree with  $a_1, a_2, \dots, a_n$  as leaves in that order (not sorted)
- At each internal node  $v$  let  $S_v$  be leaves under  $v$ .
- Compute a summary  $Q_v$  for  $S_v$  bottom up.  $Q_r$  is output where  $r$  is root. Summary at leaf is optimal simply stores element.
- To compute  $Q_v$  with children  $a, b$  Merge  $Q_a$  and  $Q_b$  and Prune to size  $h + 1$
- Guarantees that  $Q_r$  has size  $h + 1$

How should we choose  $h$  to ensure  $\epsilon$ -approx  $Q_r$ ?

# Merge and Reduce for Streaming Quantiles

If each leaf summary has error  $\epsilon'$  then Merging does not increase error but Pruning adds  $1/(2h)$  at each level. Hence  $\epsilon_r$  at root with depth  $d$  satisfies

$$\epsilon_r \leq \epsilon' + d/(2h) \leq \epsilon' + \frac{\log n}{2h} \leq \epsilon$$

$\uparrow$  error at leaf       $\uparrow$        $\downarrow$

to do that  
choose  $h = \frac{\epsilon}{\log n}$

# Merge and Reduce for Streaming Quantiles

If each leaf summary has error  $\epsilon'$  then Merging does not increase error but Pruning adds  $1/(2h)$  at each level. Hence  $\epsilon_r$  at root with depth  $d$  satisfies

$$\epsilon_r \leq \epsilon' + d/(2h) \leq \epsilon' + \log n/(2h)$$

To ensure  $\epsilon_r \leq \epsilon$  we set  $h = \Omega(\frac{1}{\epsilon} \log n)$ . Hence each summary size is  $O(\frac{1}{\epsilon} \log n)$  numbers

# Merge and Reduce for Streaming Quantiles

If each leaf summary has error  $\epsilon'$  then Merging does not increase error but Pruning adds  $1/(2h)$  at each level. Hence  $\epsilon_r$  at root with depth  $d$  satisfies

$$\epsilon_r \leq \epsilon' + d/(2h) \leq \epsilon' + \log n/(2h)$$

To ensure  $\epsilon_r \leq \epsilon$  we set  $h = \Omega(\frac{1}{\epsilon} \log n)$ . Hence each summary size is  $O(\frac{1}{\epsilon} \log n)$  numbers

How can we implement offline algorithm in streaming setting and how much space does it require?

# Merge and Reduce for Streaming Quantiles

To ensure  $\epsilon_r \leq \epsilon$  we set  $h = \Omega(\frac{1}{\epsilon} \log n)$ . Hence each summary size is  $O(\frac{1}{\epsilon} \log n)$  numbers

How can we implement offline algorithm in streaming setting and how much space does it require?

Only  $Q_r$  needed so sufficient to keep only those summaries in the “imaginary” binary tree that suffice to create  $Q_r$ . Suffices to keep  $O(d)$  summaries where  $d$  is depth. Hence total space is  $O(\frac{1}{\epsilon} \log^2 n)$ .

# Merge and Reduce for Streaming Quantiles

To ensure  $\epsilon_r \leq \epsilon$  we set  $h = \Omega(\frac{1}{\epsilon} \log n)$ . Hence each summary size is  $O(\frac{1}{\epsilon} \log n)$  numbers

How can we implement offline algorithm in streaming setting and how much space does it require?

Only  $Q_r$  needed so sufficient to keep only those summaries in the “imaginary” binary tree that suffice to create  $Q_r$ . Suffices to keep  $O(d)$  summaries where  $d$  is depth. Hence total space is  $O(\frac{1}{\epsilon} \log^2 n)$ .

Need to know  $n$  in advance to set  $h$ . Otherwise use doubling trick with extra log factor.



# Improvements

Instead of binary tree all the way use at first level  $1/\epsilon$  nodes. Depth goes to  $\log(\epsilon n)$  and hence space improves to  $O(\frac{1}{\epsilon} \log^2(\epsilon n))$ .

# Improvements

Instead of binary tree all the way use at first level  $1/\epsilon$  nodes. Depth goes to  $\log(\epsilon n)$  and hence space improves to  $O(\frac{1}{\epsilon} \log^2(\epsilon n))$ .

[Greenwald-Khanna] gave a more involved scheme that achieves  $O(\frac{1}{\epsilon} \log(\epsilon n))$  space. Near-optimal.



## Part III

# Multipass Selection

# Multipass Selection

Selection in multiple passes:

- 1-pass requires and can be done in  $O(n)$  space
- $O(1)$  space.  $O(\log n)$  suffices. Implement Quick Select in  $O(1)$  space.
- $p$  passes?  $O(n^{1/p} \text{polylog}(n))$  space suffices. Hence  $O(\sqrt{n} \log n)$  for 2 passes. [Munro-Paterson 1980]

# Multipass Selection

Selection in multiple passes:

- 1-pass requires and can be done in  $O(n)$  space
- $O(1)$  space.  $O(\log n)$  suffices. Implement Quick Select in  $O(1)$  space.
- $p$  passes?  $O(n^{1/p} \text{polylog}(n))$  space suffices. Hence  $O(\sqrt{n} \log n)$  for 2 passes. [Munro-Paterson 1980]

**Goal:** Derive  $p$ -pass algorithm via approximate quantile summary

## $p = 2$ case

**Goal:** Selection of rank  $k$  element in 2-passes using  $\tilde{O}(\sqrt{n})$  space

### Pass 1:

- Store  $\epsilon = 1/\sqrt{n}$ -approximate summary. Space is  $\tilde{O}(1/\epsilon) = \tilde{O}(\sqrt{n})$ .
- Summary allows to find two numbers  $a < b$  such that  $\text{rank}(a) \geq k - O(\epsilon)n$  and  $\text{rank}(b) \leq k + O(\epsilon)n$

## $p = 2$ case

**Goal:** Selection of rank  $k$  element in 2-passes using  $\tilde{O}(\sqrt{n})$  space

### Pass 1:

- Store  $\epsilon = 1/\sqrt{n}$ -approximate summary. Space is  $\tilde{O}(1/\epsilon) = \tilde{O}(\sqrt{n})$ .

Summary allows to find two numbers  $a < b$  such that  $\text{rank}(a) \geq k - O(\epsilon)n$  and  $\text{rank}(b) \leq k + O(\epsilon)n$

~~$\frac{1}{\sqrt{n}}$~~   
 $\frac{1}{\sqrt{n}}$   $\ln n$

### Pass 2:

- Store all numbers between  $a$  and  $b$ ;  $O(\sqrt{n})$  numbers.
- Compute exact rank of  $a$  and  $b$ . How?
- Find rank  $k$  element from stored elements and knowing rank of  $a, b$ . How?

# General $p$

**Goal:** Selection of rank  $k$  element in  $p$ -passes using  $\tilde{O}(n^{1/p})$  space

## Pass 1:

- Store  $\epsilon = 1/n^{1/p}$ -approximate summary. Space is  $\tilde{O}(1/\epsilon) = \tilde{O}(n^{1/p})$ .
- Summary allows to find two numbers  $a < b$  such that  $\text{rank}(a) \geq k - O(n^{1-1/p})$  and  $\text{rank}(b) \leq k + O(n^{1-1/p})$
- In subsequent passes one can restrict attention to numbers between  $a$  and  $b$ . Only  $n^{1-1/p}$  of them. Hence in one pass reduce to  $n^{1-1/p}$  numbers.



$$p = 3 \\ n^{1/3}$$



# General $p$

**Goal:** Selection of rank  $k$  element in  $p$ -passes using  $\tilde{O}(n^{1/p})$  space

## Pass 1:

- Store  $\epsilon = 1/n^{1/p}$ -approximate summary. Space is  $\tilde{O}(1/\epsilon) = \tilde{O}(n^{1/p})$ .
- Summary allows to find two numbers  $a < b$  such that  $\text{rank}(a) \geq k - O(n^{1-1/p})$  and  $\text{rank}(b) \leq k + O(n^{1-1/p})$
- In subsequent passes one can restrict attention to numbers between  $a$  and  $b$ . Only  $n^{1-1/p}$  of them. Hence in one pass reduce to  $n^{1-1/p}$  numbers.

After  $(p - 1)$  passes we have  $n^{1/p}$  numbers left and we can store all of them in  $p$ 'th pass and solve exactly.

# Random Order Streams

$\Omega(n)$  lower bound for Selection in adversarial setting. Can we do better if we assume non-worst case input?

## Random Order Stream Model:

- Adversary picks some input.
- Algorithm sees a random permutation of the input. Adversary power is weakened.
- Several interesting results in this model.

For Exact Selection in random order streams.

- $O(\sqrt{n})$  space in 1-pass suffices with high probability.  
[Munro-Paterson]
- $O(\log \log n)$  passes suffice with  $O(\text{poly}(\log n))$  space whp.  
[Guha-MacGregor]