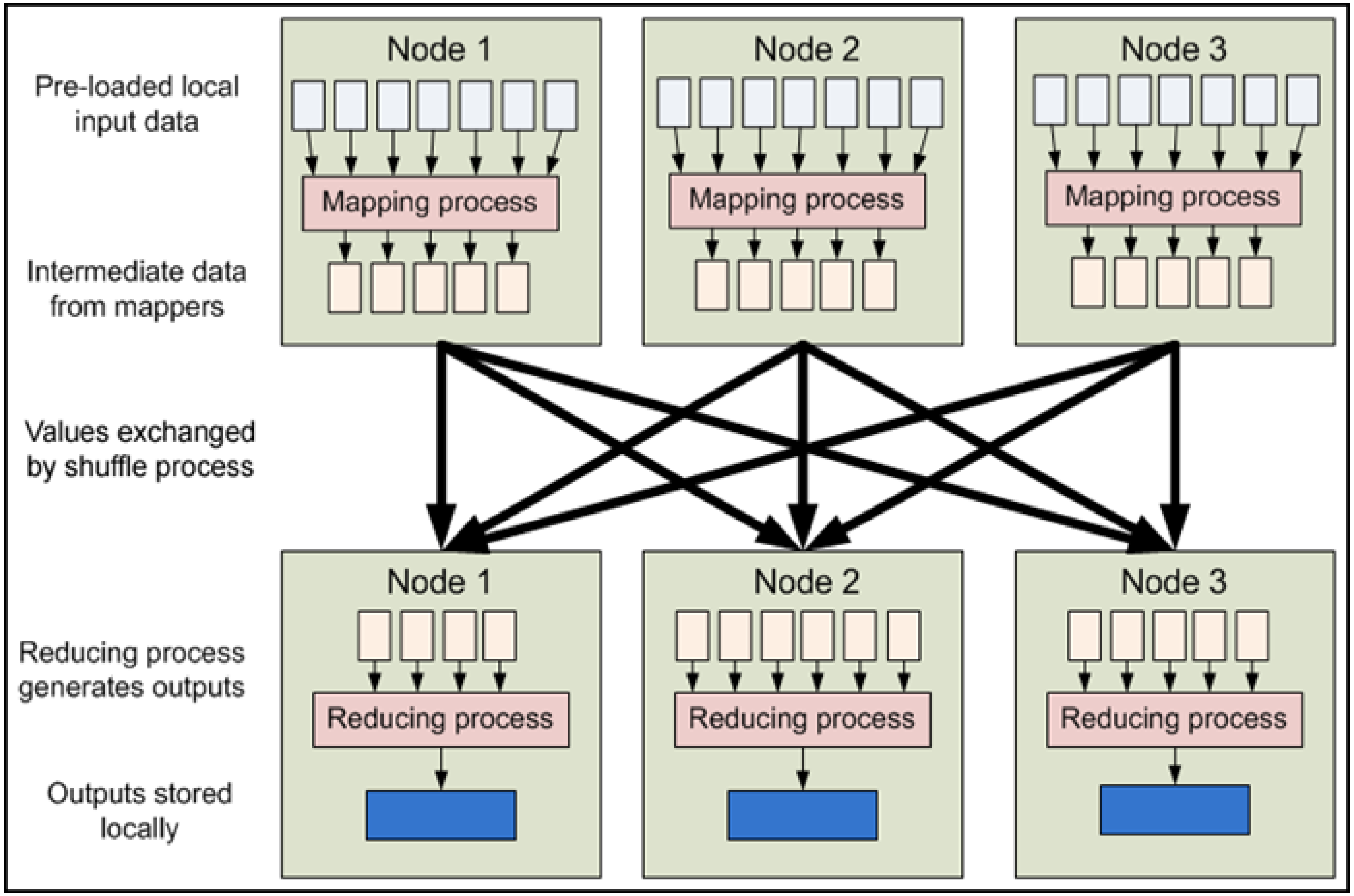
Computing Statistical Summaries over Massive Distributed Data

Ke Yi

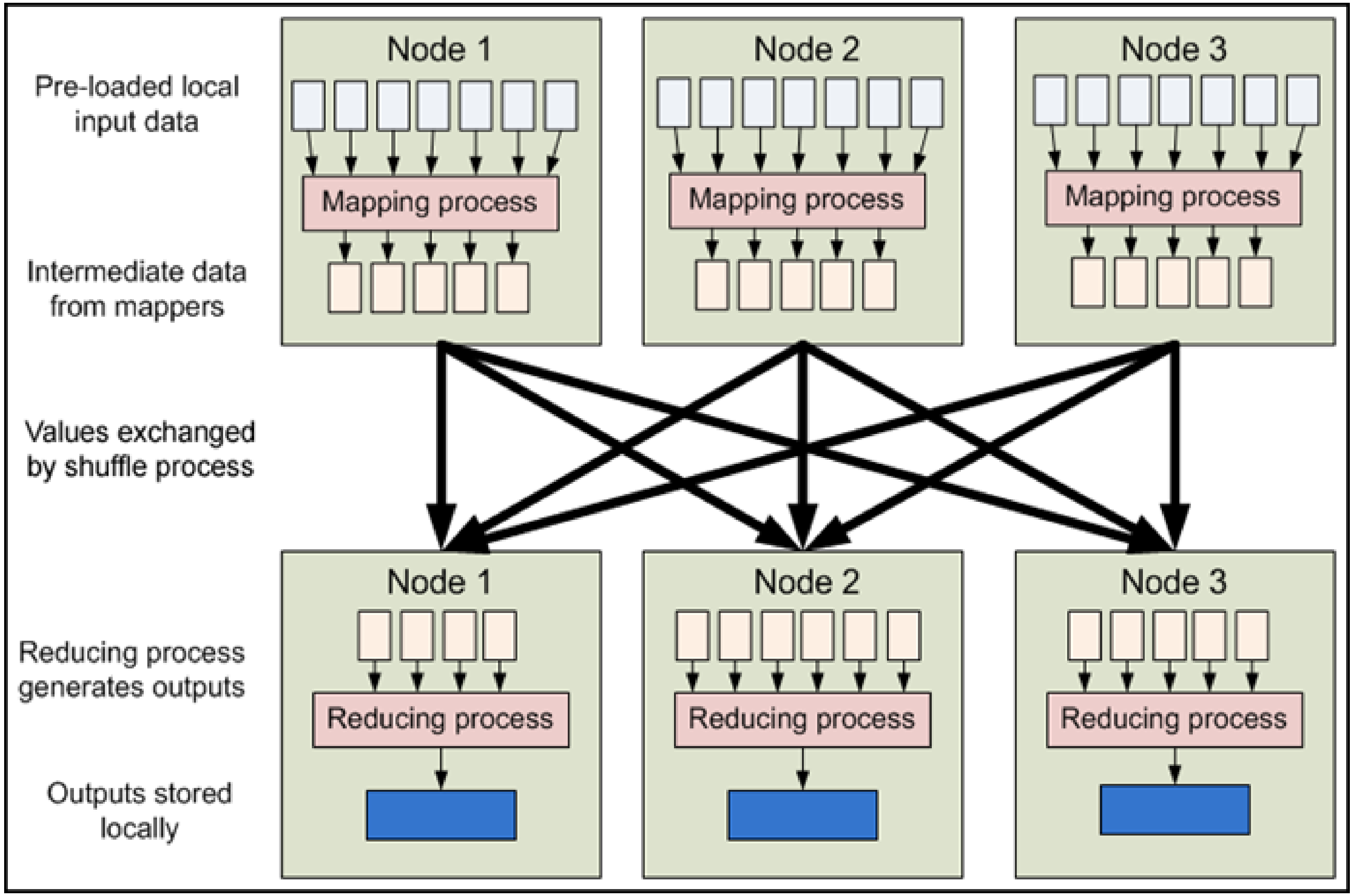
Hong Kong University of Science and Technology

Distributed Systems for Massive Data: MapReduce



Open sourece implementation: Hadoop

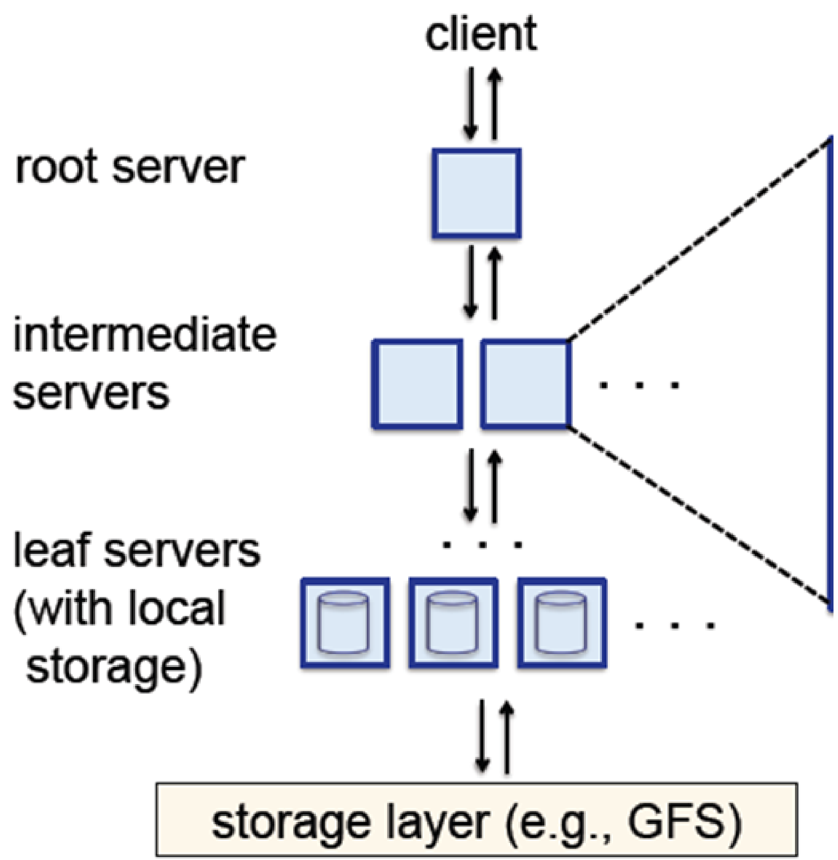
Distributed Systems for Massive Data: MapReduce



Open sourece implementation: Hadoop

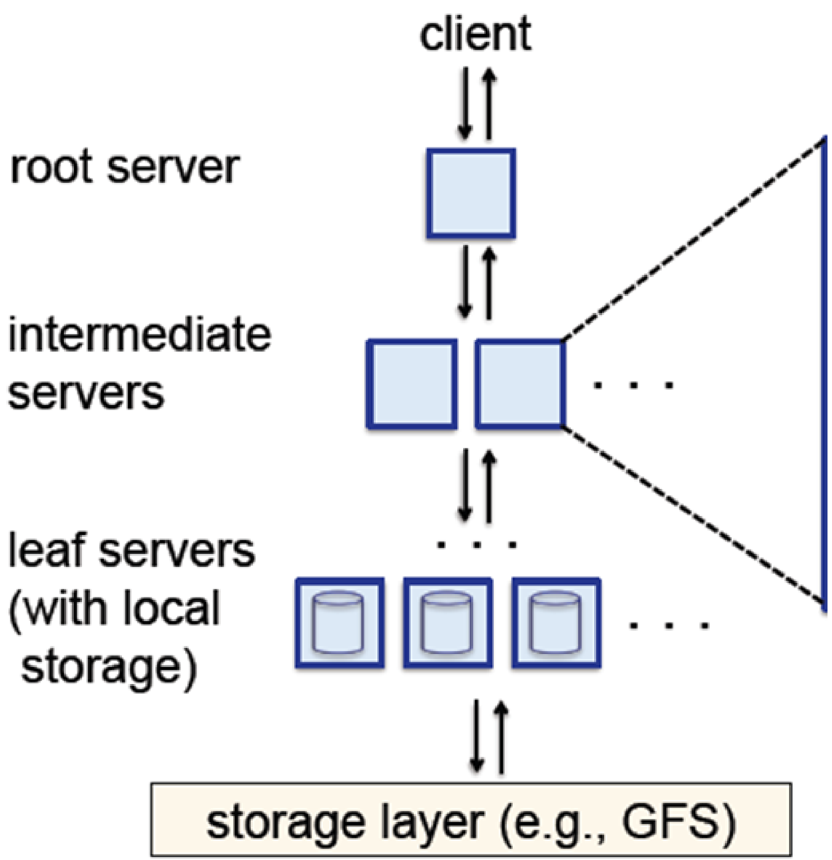
Suitable for batch processing (e.g., index construction)

Distributed Systems for Massive Data: Dremel



No open source implementation yet

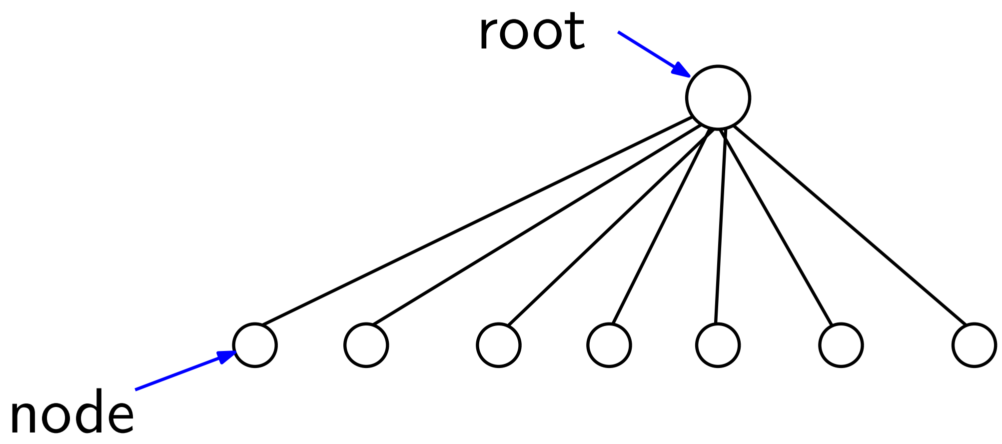
Distributed Systems for Massive Data: Dremel



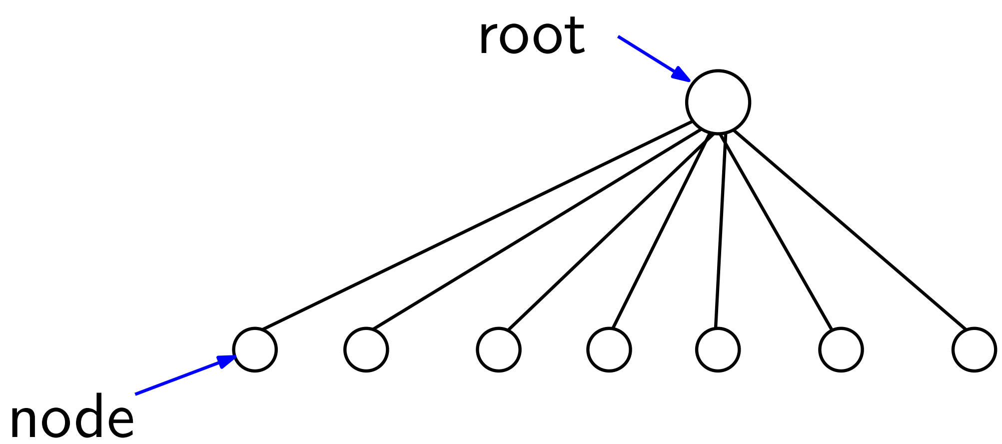
No open source implementation yet

Suitable for analytical queries (e.g., extracting a summary)

(Simplified) Model of Computation

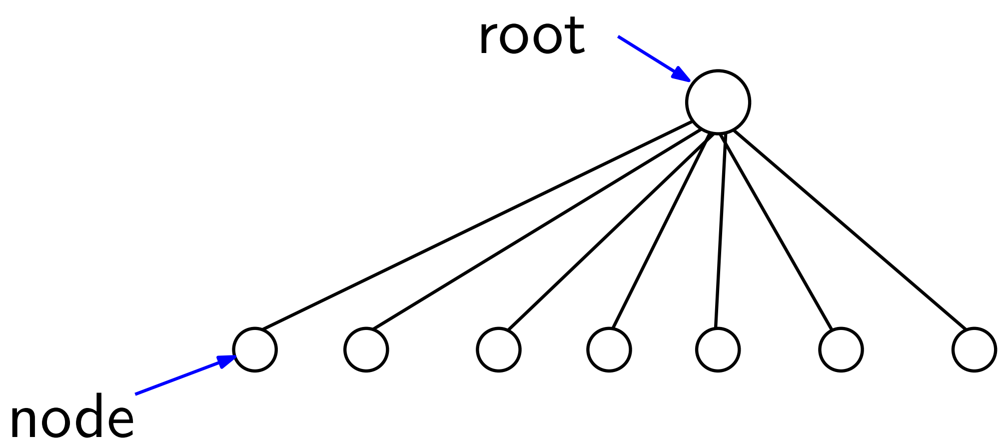


(Simplified) Model of Computation



- The root broadcasts a message to initialize computation
- Each node computes a summary on its local data
- The root combines the summaries to produce a global summary

(Simplified) Model of Computation

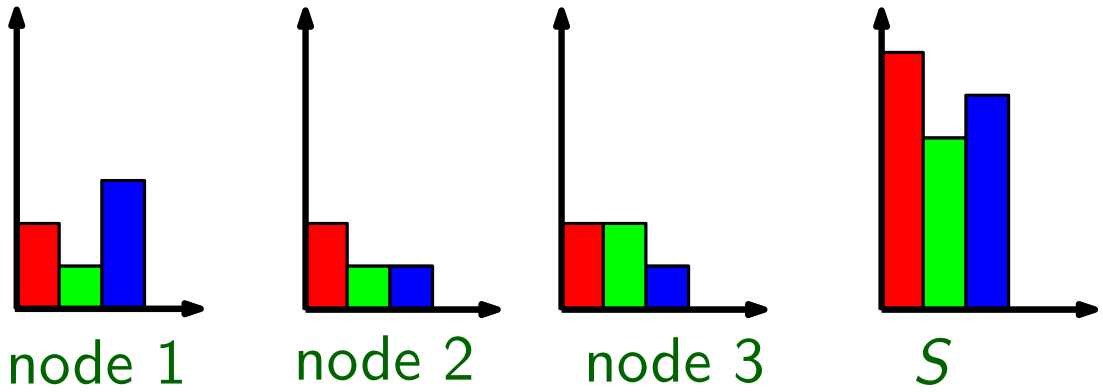


- The root broadcasts a message to initialize computation
- Each node computes a summary on its local data
- The root combines the summaries to produce a global summary
- Using minimum communication (and load balancing)

Outline

- Model of computation
- Frequency estimation (heavy hitters)
- Quantiles (order statistics)
- Other problems

Problem: Frequency Estimation



- Input: Multiset S of N items drawn from the universe $[u] = \{1 \ldots u\}$

  For example, all IP addresses

- Each node $j \in [k]$ holds a subset of S

  For any item $i \in [u]$

  x_{ij}: total number of i's in node j (local count)

  $y_i = \sum_{j=1}^{k} x_{ij}$ (global count)

- Compute y_i for each i

Frequency Estimation: Possible Solutions

- Compute exactly: send everything

Frequency Estimation: Possible Solutions

- Compute exactly: send everything
- Approximate each y_i within addtive error ϵN

Frequency Estimation: Possible Solutions

- Compute exactly: send everything
- Approximate each y_i within addtive error ϵN
- Sketching: Each node computes a sketch of its own data and sends it to the coordinator.

  Count-min sketch, MG sketch, Space saving, etc.

  Sketch size: $O(1/\varepsilon)$

  Communication cost: $O(k/\varepsilon)$

Frequency Estimation: Possible Solutions

- Compute exactly: send everything
- Approximate each y_i within addtive error ϵN
- Sketching: Each node computes a sketch of its own data and sends it to the coordinator.

  Count-min sketch, MG sketch, Space saving, etc.

  Sketch size: $O(1/\varepsilon)$

  Communication cost: $O(k/\varepsilon)$

- Random sampling

  Uniformly randomly sample a subset of size $O(1/\varepsilon^2)$

Frequency Estimation: Possible Solutions

- Compute exactly: send everything
- Approximate each y_i within addtive error ϵN
- Sketching: Each node computes a sketch of its own data and sends it to the coordinator.

  Count-min sketch, MG sketch, Space saving, etc.

  Sketch size: $O(1/\varepsilon)$

  Communication cost: $O(k/\varepsilon)$
- Random sampling

  Uniformly randomly sample a subset of size $O(1/\varepsilon^2)$
- We can achieve: $O(\sqrt{k}/\varepsilon)$

  Typical values of $\varepsilon = 10^{-3} \sim 10^{-6}, k = 10^2 \sim 10^4$
  We assume $k < 1/\varepsilon^2$

HT estimator [Horvitz and Thompson 56]

Each node holds a set of (item, count) pairs

(item, count)

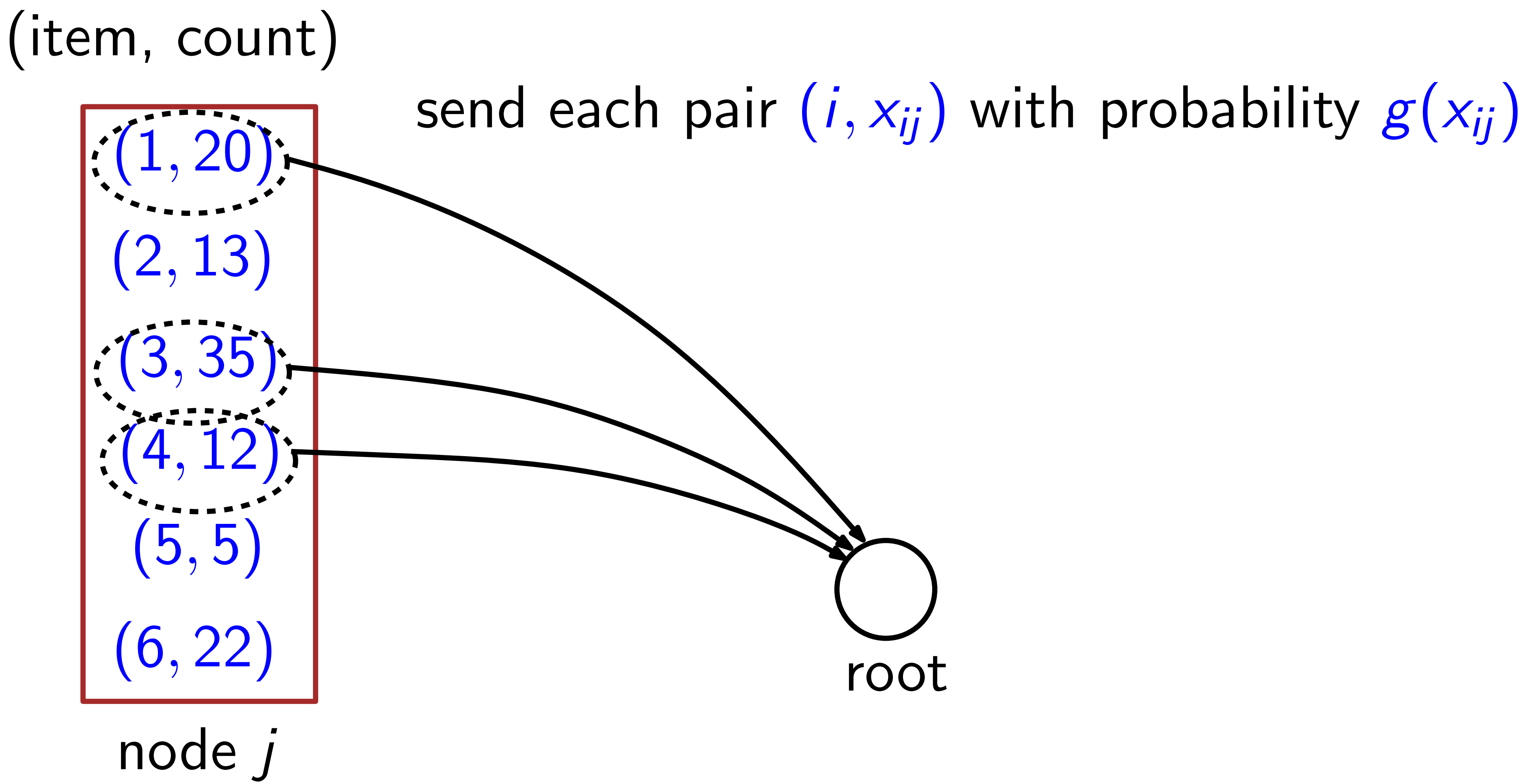
$(1, 20)$
$(2, 13)$
$(3, 35)$
$(4, 12)$
$(5, 5)$
$(6, 22)$

node j

send each pair (i, x_{ij}) with probability $g(x_{ij})$

HT estimator [Horvitz and Thompson 56]

Each node holds a set of (item, count) pairs



HT estimator [Horvitz and Thompson 56]

HT estimator for x_{ij}:

$$Y_{i,j} = \frac{x_{i,j}}{g(x_{i,j})} \text{ if it is sampled, otherwise } 0$$

This is an unbiased estimator

Estimator for y_i:

$$Y_i = Y_{i,1} + \cdots + Y_{i,n}$$

HT estimator [Horvitz and Thompson 56]

HT estimator for x_{ij}:

$$Y_{i,j} = \frac{x_{i,j}}{g(x_{i,j})} \text{ if it is sampled, otherwise } 0$$

This is an unbiased estimator

Estimator for y_i:

$$Y_i = Y_{i,1} + \cdots + Y_{i,n}$$

$$\begin{aligned} \mathrm{Var}[Y_{i,j}] &= \left(\frac{x_{i,j}}{g(x_{i,j})} - x_{i,j}\right)^2 g(x_{i,j}) + (x_{i,j})^2 (1 - g(x_{i,j})) \\ &= \frac{x_{i,j}^2 (1-g(x_{i,j}))}{g(x_{i,j})} \end{aligned}$$

$$\mathrm{Var}[Y_i] = \sum_{j=1}^{n} \mathrm{Var}[Y_{ij}] = \sum_{j=1}^{n} \frac{x_{i,j}^2 (1-g(x_{i,j}))}{g(x_{i,j})}$$

Sampling Function

Question: What sampling function $g(x)$ should we use

Sampling Function

Question: What sampling function $g(x)$ should we use

Accuracy: standard deviation less than εN

A function is valid, if $\mathrm{Var}[Y_i] \leq (\epsilon N)^2$ for all items i

Sampling Function

Question: What sampling function $g(x)$ should we use

Accuracy: standard deviation less than εN

A function is valid, if $\mathrm{Var}[Y_i] \leq (\epsilon N)^2$ for all items i

Communication cost: $\sum_{i,j} g(x_{ij})$

Sampling Function

Question: What sampling function $g(x)$ should we use

Accuracy: standard deviation less than εN

A function is valid, if $\mathrm{Var}[Y_i] \leq (\epsilon N)^2$ for all items i

Communication cost: $\sum_{i,j} g(x_{ij})$

Optimal valid $g(x)$?

A Worst-Case Optimal Sampling Function

$g_1(x) = \min\{\frac{\sqrt{k}}{\varepsilon N}x, 1\}$

A Worst-Case Optimal Sampling Function

$g_1(x) = \min\{\frac{\sqrt{k}}{\varepsilon N} x, 1\}$

Can show:

- $$\mathrm{Var}[Y_i] = -\left(\frac{y_i}{\sqrt{k}} - \frac{\varepsilon N}{2}\right)^2 + \frac{(\varepsilon N)^2}{4} \le \frac{1}{4}(\varepsilon N)^2,$$

  i.e., $g_1(x)$ is valid

- Communication cost of using $g_1(x)$ is $O(\sqrt{k}/\varepsilon)$

Another Sampling Function

$$g_2(x) = (g_1(x))^2$$

Another Sampling Function

$g_2(x) = (g_1(x))^2$

Can show:

- g_2 is also valid

Another Sampling Function

$g_2(x) = (g_1(x))^2$

Can show:

- g_2 is also valid
- Clearly, $g_1(x) \geq g_2(x)$

  g_1's communication cost is always $\Theta(\sqrt{k}/\varepsilon)$, while g_2 can be much better when there are many small local counts

Another Sampling Function

$g_2(x) = (g_1(x))^2$

Can show:

- g_2 is also valid
- Clearly, $g_1(x) \geq g_2(x)$

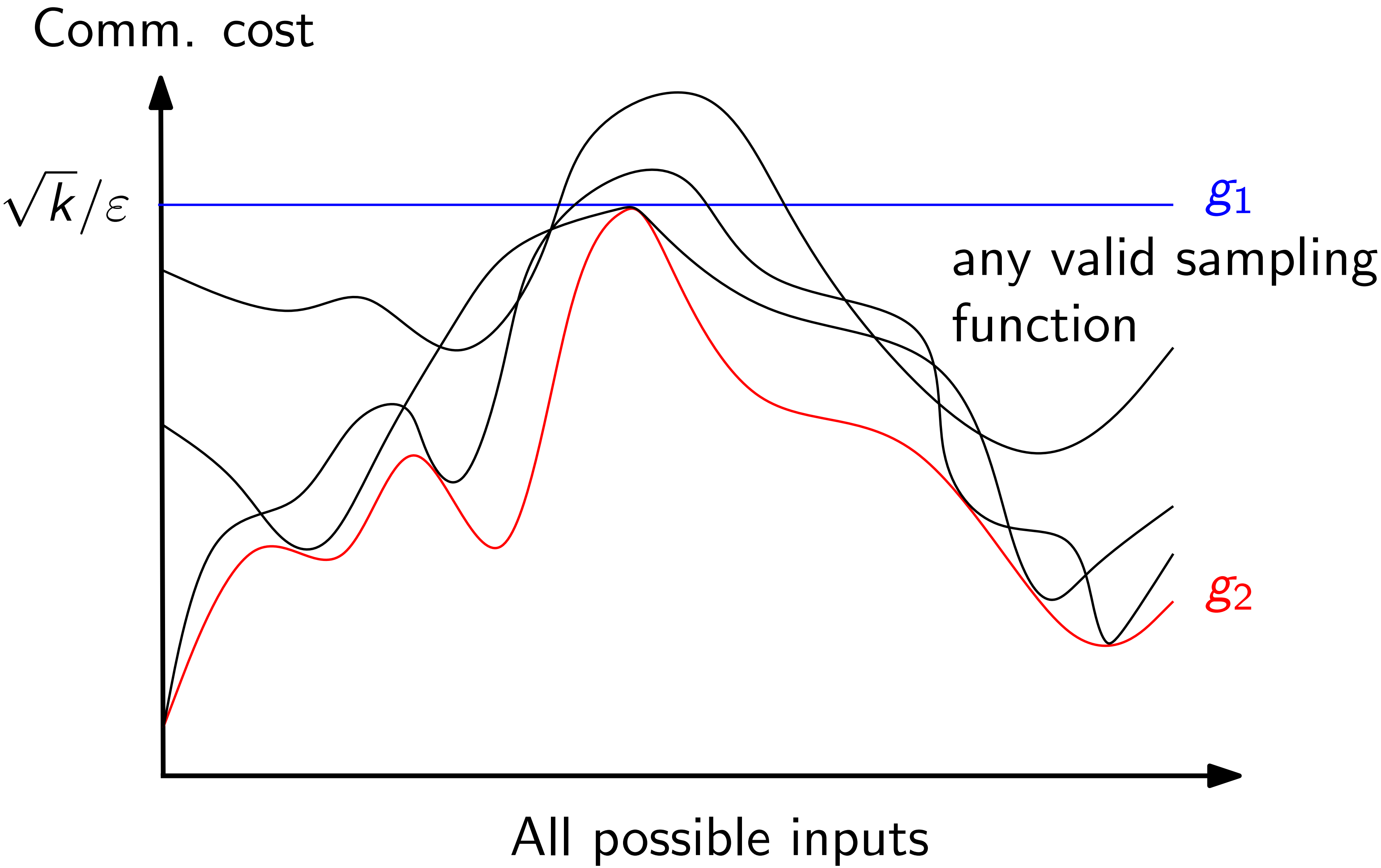
  g_1's communication cost is always $\Theta(\sqrt{k}/\varepsilon)$, while g_2 can be much better when there are many small local counts

- A stronger optimality: $g_2(x)$ is instance-optimal

  Define $opt(I) = \sum_{i,j} g_2(x_{i,j})$ on input $I : \{x_{i,j}\}$

  Can show that on every input I, any valid sampling function must have cost $\Omega(opt(I))$

Instance Optimality



Further Reducing Communication Cost

$g_1(x) = \min\{\frac{\sqrt{k}}{\varepsilon N} x, 1\}$

HT estimator for x_{ij}:

$$Y_{i,j} = \frac{x_{i,j}}{g(x_{i,j})} \text{ if it is sampled, otherwise } 0$$

Estimator for y_i:

$$Y_i = Y_{i,1} + \cdots + Y_{i,n}$$

Further Reducing Communication Cost

$$g_1(x) = \min\{\frac{\sqrt{k}}{\varepsilon N}x, 1\}$$

HT estimator for x_{ij}:

$$Y_{i,j} = \frac{x_{i,j}}{g(x_{i,j})}$$ if it is sampled, otherwise 0

Estimator for y_i:

$$Y_i = Y_{i,1} + \cdots + Y_{i,n}$$

$$Y_i = \frac{\varepsilon N}{\sqrt{k}}(1 + 0 + 1 + 1 + \cdots + 0 + 1)$$

Further Reducing Communication Cost

$$g_1(x) = \min\{\frac{\sqrt{k}}{\varepsilon N}x, 1\}$$

HT estimator for x_{ij}:

$$Y_{i,j} = \frac{x_{i,j}}{g(x_{i,j})} \text{ if it is sampled, otherwise } 0$$

Estimator for y_i:

$$Y_i = Y_{i,1} + \cdots + Y_{i,n}$$

$$Y_i = \frac{\varepsilon N}{\sqrt{k}}(1 + 0 + 1 + 1 + \cdots + 0 + 1)$$

Each site j just needs to tell whether i is sampled or not!

Further Reducing Communication Cost

$g_1(x) = \min\{\frac{\sqrt{k}}{\varepsilon N}x, 1\}$

HT estimator for x_{ij}:

$$Y_{i,j} = \frac{x_{i,j}}{g(x_{i,j})} \text{ if it is sampled, otherwise } 0$$

Estimator for y_i:

$$Y_i = Y_{i,1} + \cdots + Y_{i,n}$$

$$Y_i = \frac{\varepsilon N}{\sqrt{k}}(1 + 0 + 1 + 1 + \cdots + 0 + 1)$$

Each site j just needs to tell whether i is sampled or not!

The set of sampled items can be encoded in a Bloom filter, taking $O(1)$ bits per item

$\Rightarrow$ total cost $= O(\sqrt{k}/\varepsilon)$ bits

Further Reducing Communication Cost

$g_1(x) = \min\{\frac{\sqrt{k}}{\varepsilon N}x, 1\}$

HT estimator for x_{ij}:

$$Y_{i,j} = \frac{x_{i,j}}{g(x_{i,j})} \text{ if it is sampled, otherwise } 0$$

Estimator for y_i:

$$Y_i = Y_{i,1} + \cdots + Y_{i,n}$$

$$Y_i = \frac{\varepsilon N}{\sqrt{k}}(1 + 0 + 1 + 1 + \cdots + 0 + 1)$$

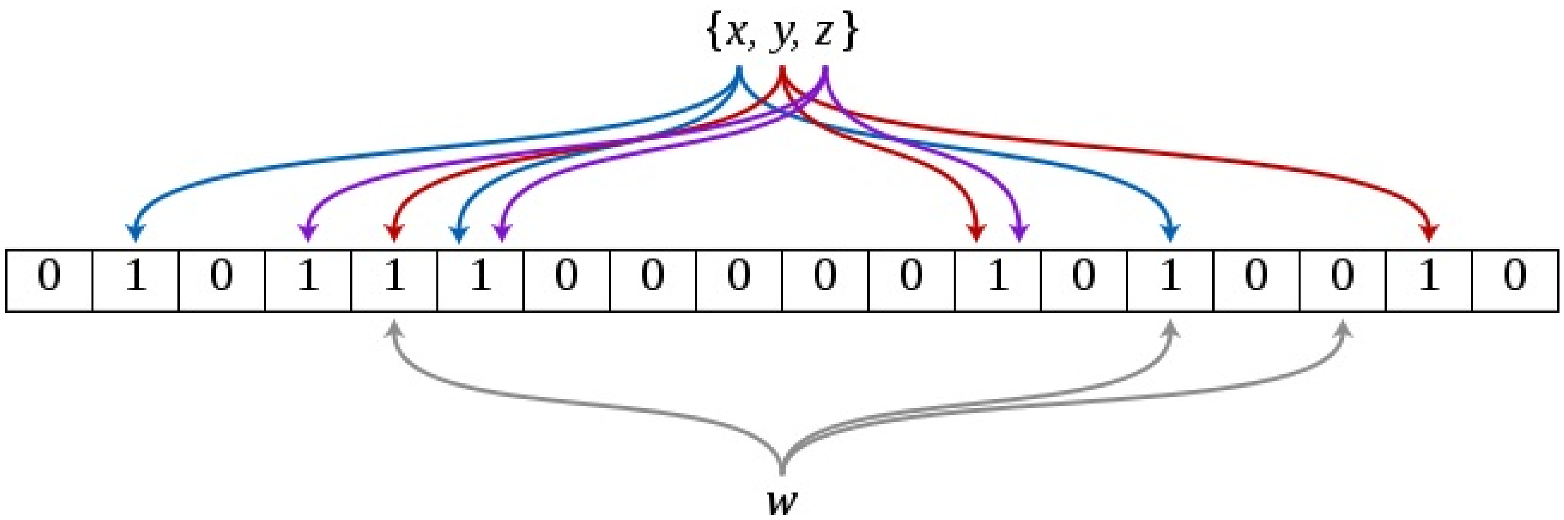
Each site j just needs to tell whether i is sampled or not!

The set of sampled items can be encoded in a Bloom filter, taking $O(1)$ bits per item

$\Rightarrow$ total cost $= O(\sqrt{k}/\varepsilon)$ bits

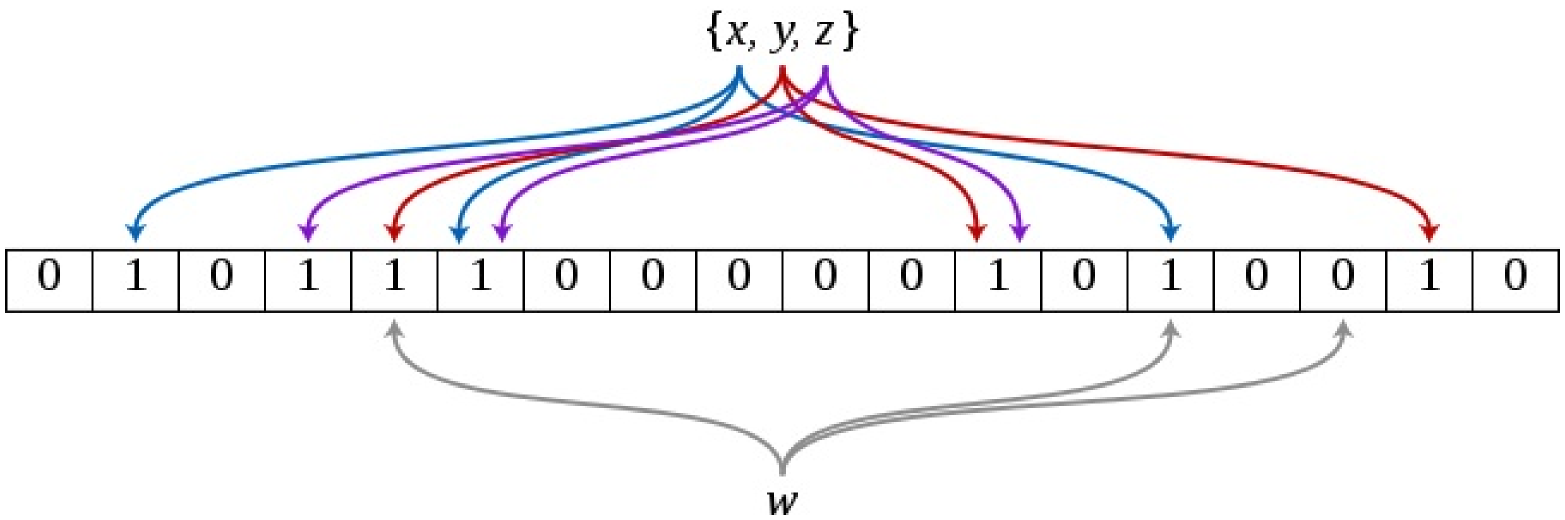
An $\Omega(\sqrt{k}/\varepsilon)$-bit lower bound [Woodruff, Zhang, '12]

Bloom Filters



- A Bloom filter needs $O(\log(1/q))$ bits per item
- No false negatives
- False positive probability $= q$

Bloom Filters



- A Bloom filter needs $O(\log(1/q))$ bits per item
- No false negatives
- False positive probability $= q$

Change the estimator to

$$Y_i = \frac{\varepsilon N}{\sqrt{k}} \cdot \frac{Y_{i,1} + \cdots + Y_{i,k} - kq}{1 - q}$$

Sampling with $g_2(x) = (g_1(x))^2$

- $g_2(x)$ samples $opt(I)$ (item, count) pairs, which may be much smaller than $O(\sqrt{k}/\varepsilon)$ on many inputs
- But it is a nonlinear sampling function

Estimator for y_i:

$$Y_i = \frac{x_{i,1}}{g_2(x_{i,1})} + 0 + 0 + \frac{x_{i,4}}{g_2(x_{i,4})} + \cdots + 0 + \frac{x_{i,k}}{g_2(x_{i,k})}$$

Sampling with $g_2(x) = (g_1(x))^2$

- $g_2(x)$ samples $opt(I)$ (item, count) pairs, which may be much smaller than $O(\sqrt{k}/\varepsilon)$ on many inputs
- But it is a nonlinear sampling function

Estimator for y_i:

$opt(I)$ such terms

$$Y_i = \frac{x_{i,1}}{g_2(x_{i,1})} + 0 + 0 + \frac{x_{i,4}}{g_2(x_{i,4})} + \cdots + 0 + \frac{x_{i,k}}{g_2(x_{i,k})}$$

Sampling with $g_2(x) = (g_1(x))^2$

- $g_2(x)$ samples $opt(I)$ (item, count) pairs, which may be much smaller than $O(\sqrt{k}/\varepsilon)$ on many inputs
- But it is a nonlinear sampling function

Estimator for y_i:

$opt(I)$ such terms

$$Y_i = \frac{x_{i,1}}{g_2(x_{i,1})} + 0 + 0 + \frac{x_{i,4}}{g_2(x_{i,4})} + \cdots + 0 + \frac{x_{i,k}}{g_2(x_{i,k})}$$

- Use $g_2(x)$ to perform the sampling locally
- Then use $g_1(x)$ + Bloom filters to sample the $\frac{x_{i,j}}{g_2(x_{i,j})}$'s

Sampling with $g_2(x) = (g_1(x))^2$

- $g_2(x)$ samples $opt(I)$ (item, count) pairs, which may be much smaller than $O(\sqrt{k}/\varepsilon)$ on many inputs
- But it is a nonlinear sampling function

Estimator for y_i:

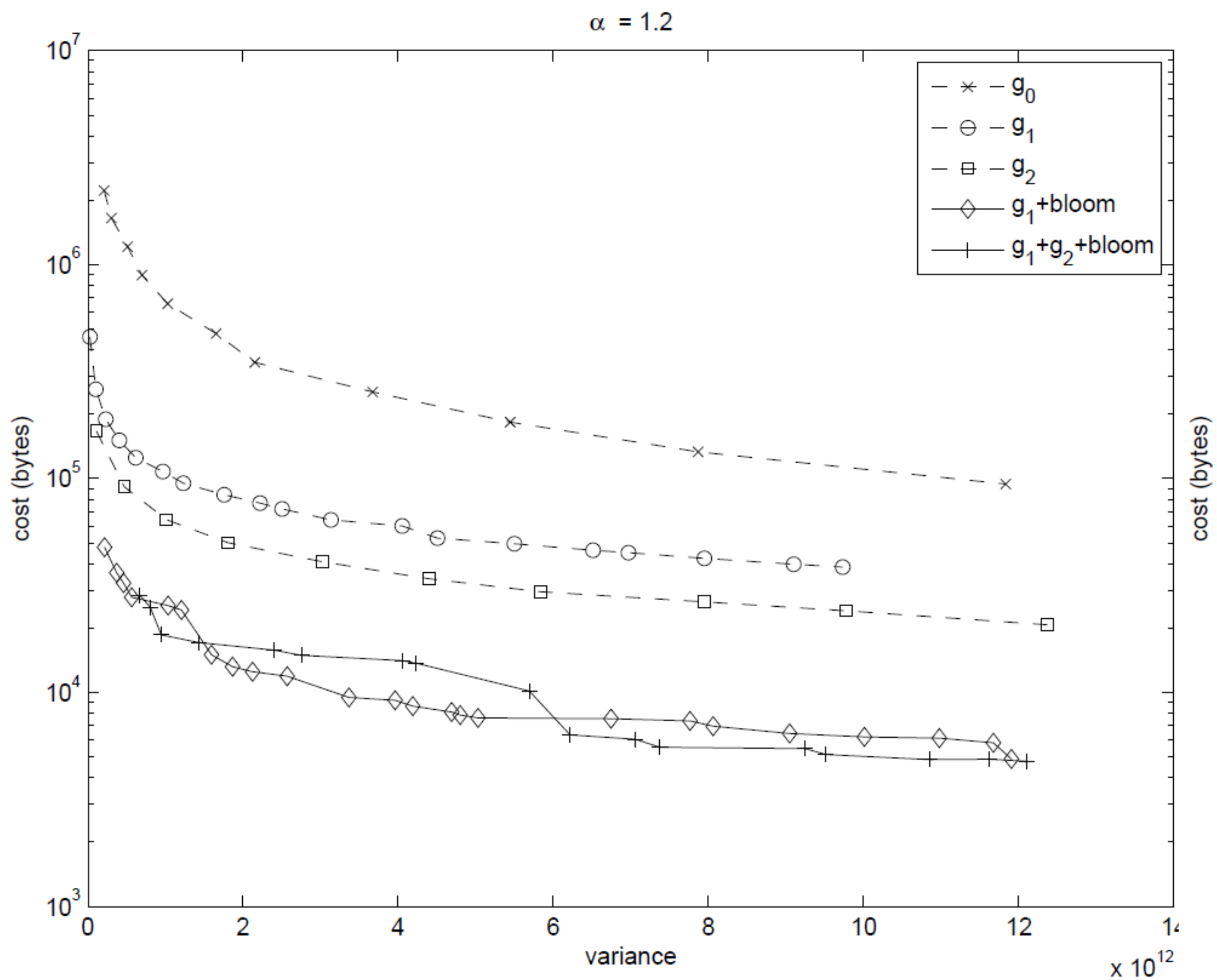
$opt(I)$ such terms

$$Y_i = \frac{x_{i,1}}{g_2(x_{i,1})} + 0 + 0 + \frac{x_{i,4}}{g_2(x_{i,4})} + \cdots + 0 + \frac{x_{i,k}}{g_2(x_{i,k})}$$

- Use $g_2(x)$ to perform the sampling locally
- Then use $g_1(x)$ + Bloom filters to sample the $\frac{x_{i,j}}{g_2(x_{i,j})}$'s

Can show this takes $O\left(opt(I)\log^2\left(\frac{\sqrt{k}}{\varepsilon\, opt(I)}\right)\right)$ bits

Simulation Results



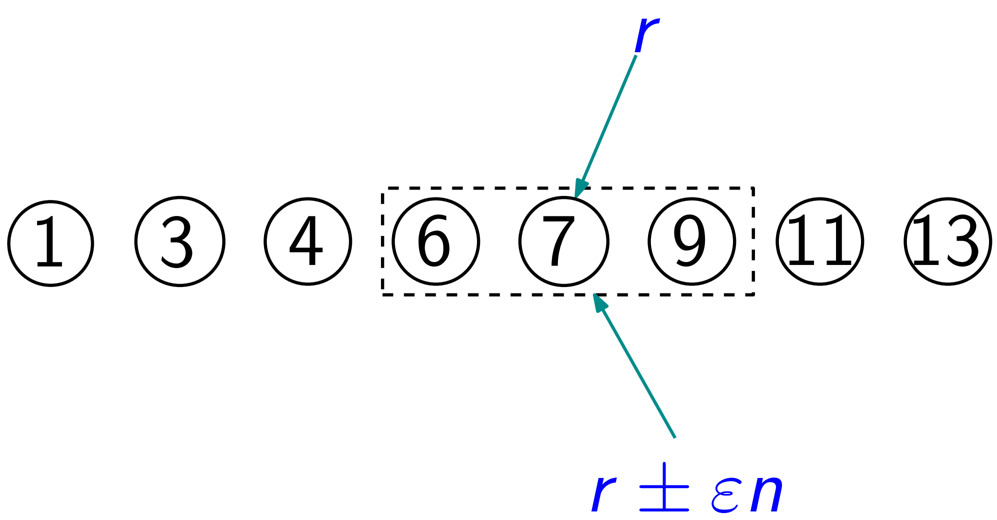
$k = 1000, N = 10^9$ following Zipf distribution with $\alpha = 1.2$. Estimate the frequencies of the 100 most popular items. Variance computed from 100 runs, and take the worst

Outline

- Model of computation
- Frequency estimation (heavy hitters)
- Quantiles (order statistics)
- Other problems

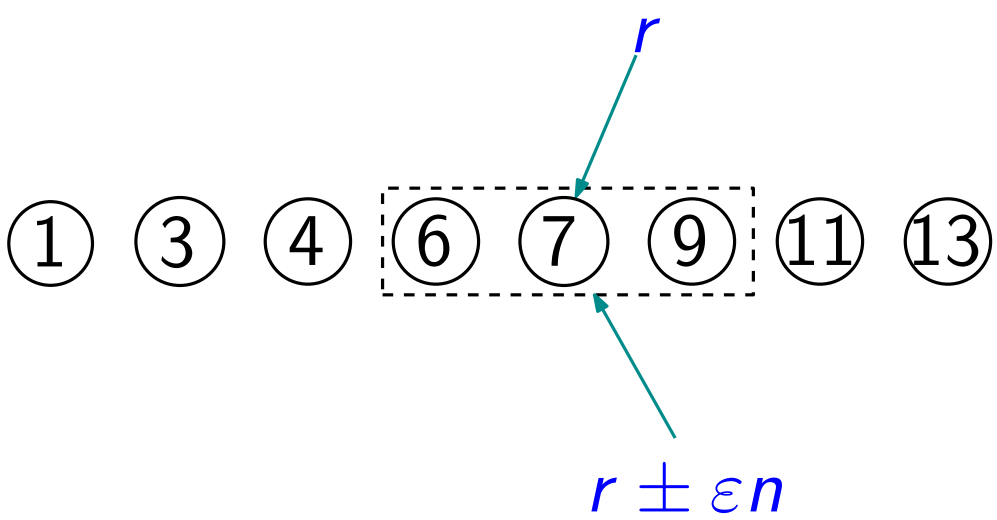
Quantiles

In a set of n values, the (r/n)-quantile is the value ranked at r.
The 0.5-quantile is the median.



Quantiles

In a set of n values, the (r/n)-quantile is the value ranked at r.
The 0.5-quantile is the median.

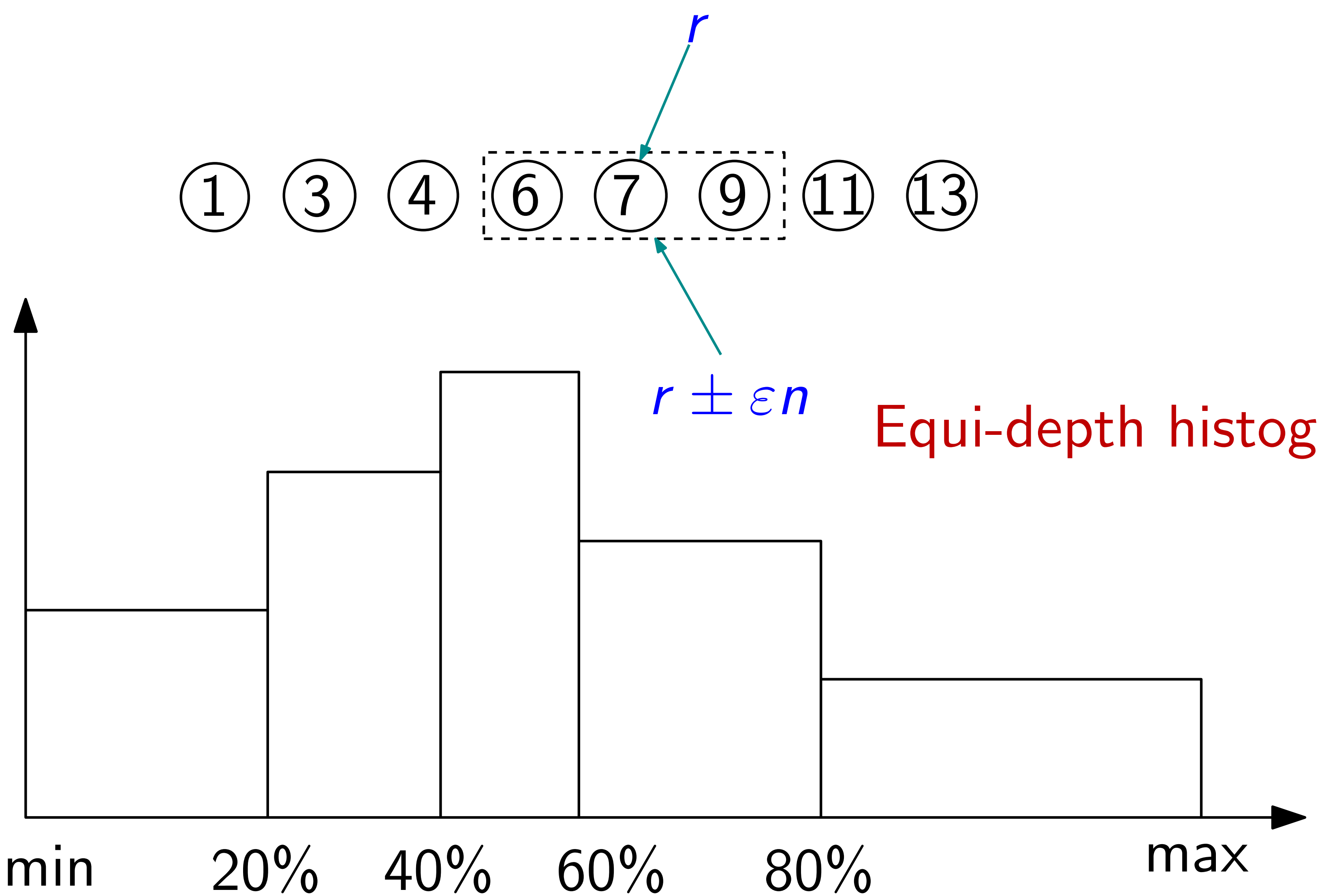


An ε-approximate (r/n)-quantile is any value ranked between $[r - \varepsilon n, r + \varepsilon n]$.

Generalizes the frequency estimation problem.

Quantiles

In a set of n values, the (r/n)-quantile is the value ranked at r.
The 0.5-quantile is the median.



Quantiles: Previous Solutions

- Sketching: Each node computes a sketch of its own data and sends it to the coordinator.

  Sketch size: $O(1/\varepsilon)$

  Communication cost: $O(k/\varepsilon)$

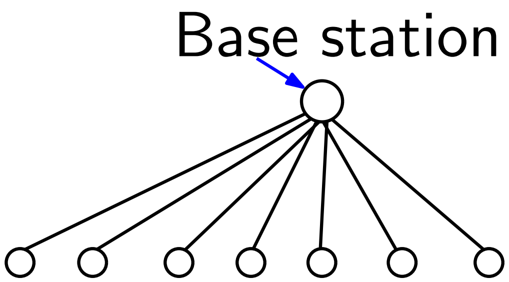
- Random sampling

  Uniformly randomly sample a subset of size $O(1/\varepsilon^2)$

- We can achieve: $O(\sqrt{k}/\varepsilon)$

  Typical values of $\varepsilon = 10^{-3} \sim 10^{-6}$, $k = 10^2 \sim 10^4$
  We assume $k < 1/\varepsilon^2$

The Algorithm



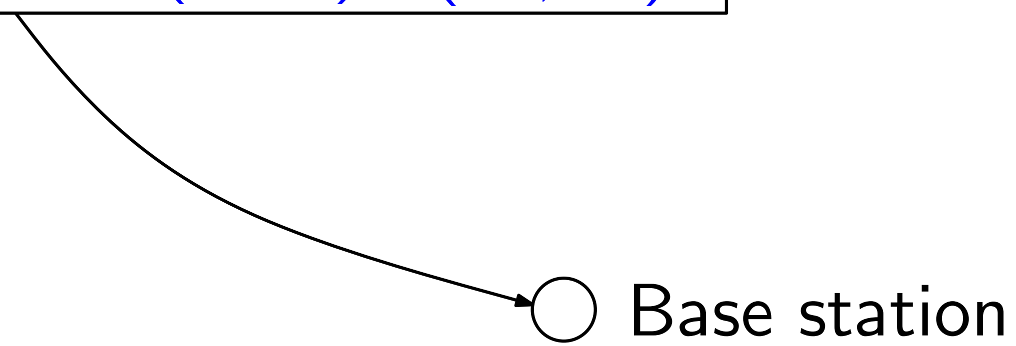
The algorithm for each node

Sample each value with probabiltiy p

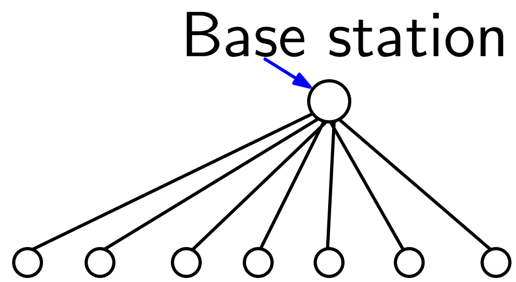
1 3 4 6 7 9 11 13 16 26 21 24

$(3, 2)$ $(7, 5)$ $(13, 8)$ $(26, 10)$

Compute local ranks



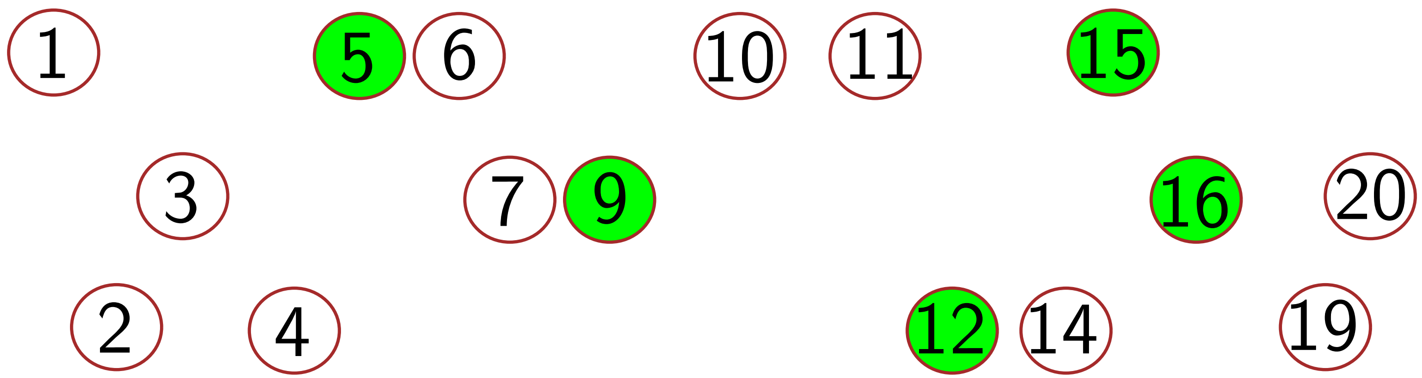
The Algorithm



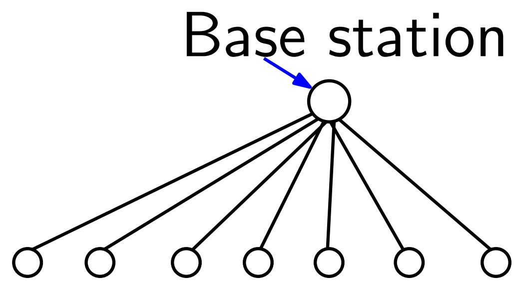
At the base station:

Answering value-to-rank query

Given any value x, estimates its rank $r(x)$



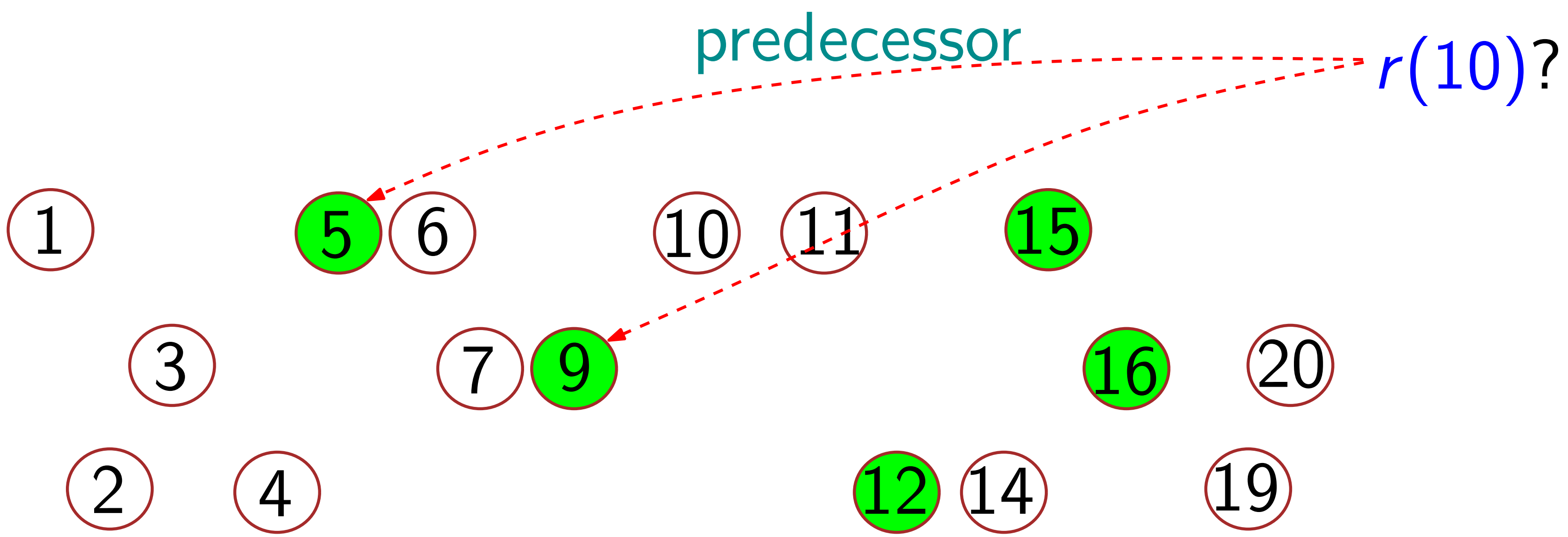
The Algorithm



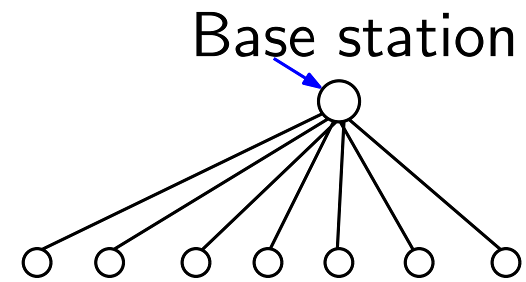
At the base station:

Answering value-to-rank query

Given any value x, estimates its rank $r(x)$



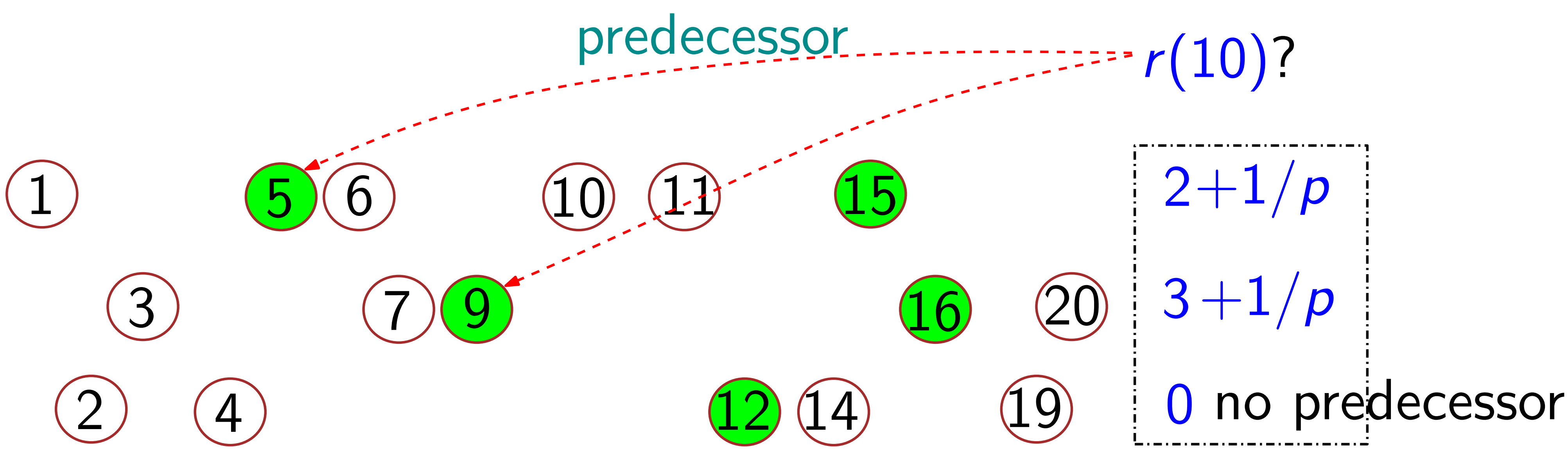
The Algorithm



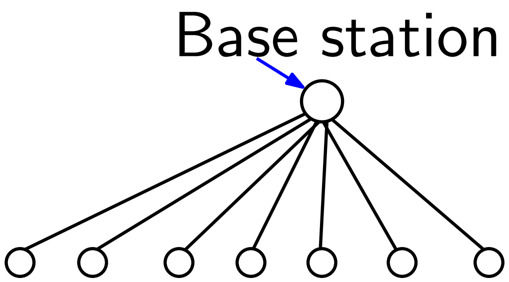
At the base station:

Answering value-to-rank query

Given any value x, estimates its rank $r(x)$



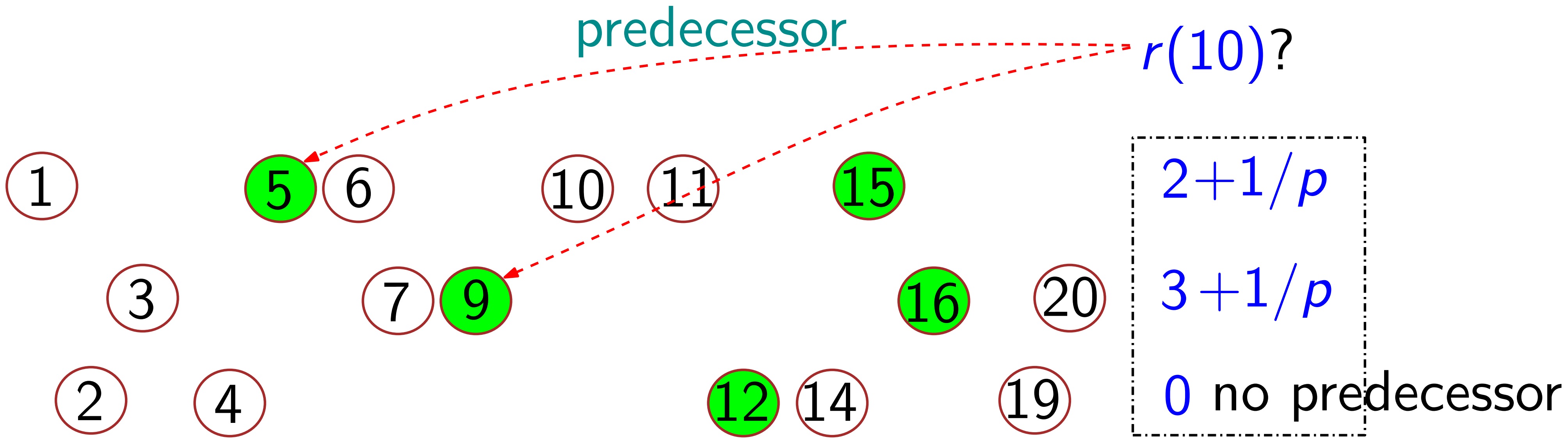
The Algorithm



At the base station:

Answering value-to-rank query

Given any value x, estimates its rank $r(x)$



$\hat{r}(10) = 5 + 2/p$

Correctness

Will show: $\hat{r}(x)$ is an unbiased estimator of $r(x)$ with standard deviation εn.

$r(10)$?

1 5 6 10 11 15

Correctness

Will show: $\hat{r}(x)$ is an unbiased estimator of $r(x)$ with standard deviation εn.

$r(10)$?

5 15

Correctness

Will show: $\hat{r}(x)$ is an unbiased estimator of $r(x)$ with standard deviation εn.

$r(10)$?

5 ? 15

Correctness

Will show: $\hat{r}(x)$ is an unbiased estimator of $r(x)$ with standard deviation εn.

$r(10)$?

5 ? 15

Follows a geometric distribution (almost)

$\mathrm{E}[?] = 1/p$ $\mathrm{Var}[?] \leq 1/p^2$

Correctness

Will show: $\hat{r}(x)$ is an unbiased estimator of $r(x)$ with standard deviation εn.

$r(10)$?

5 ? 15

Follows a geometric distribution (almost)

$\mathrm{E}[?] = 1/p$ $\mathrm{Var}[?] \le 1/p^2$

Set $p = \frac{\sqrt{k}}{\varepsilon n}$

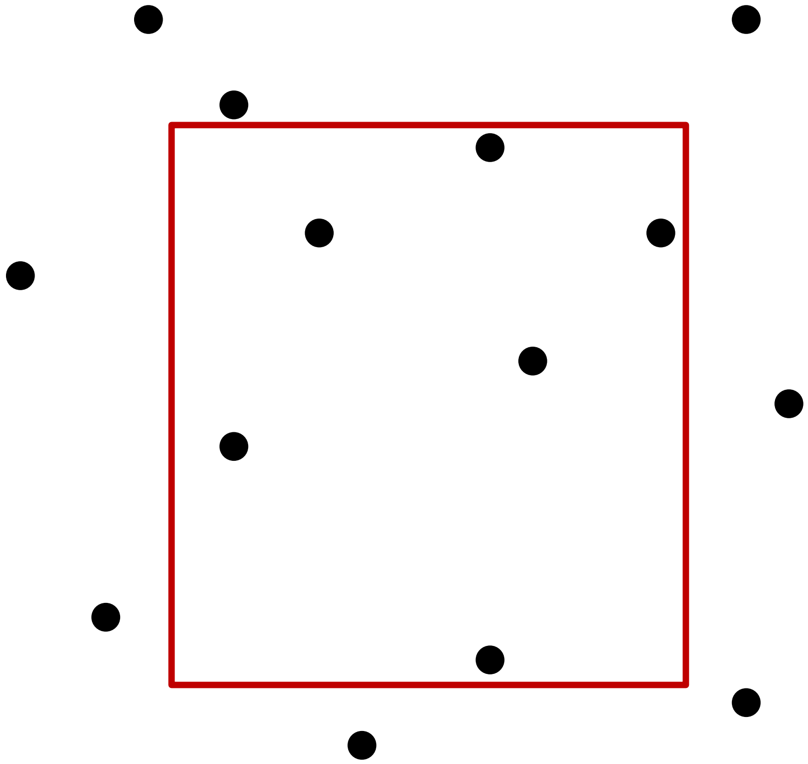
$\mathrm{Var}[\hat{r}(x)] \le k/p^2 = (\varepsilon n)^2$

Total cost: $np = \sqrt{k}/\varepsilon$ in expectation

Outline

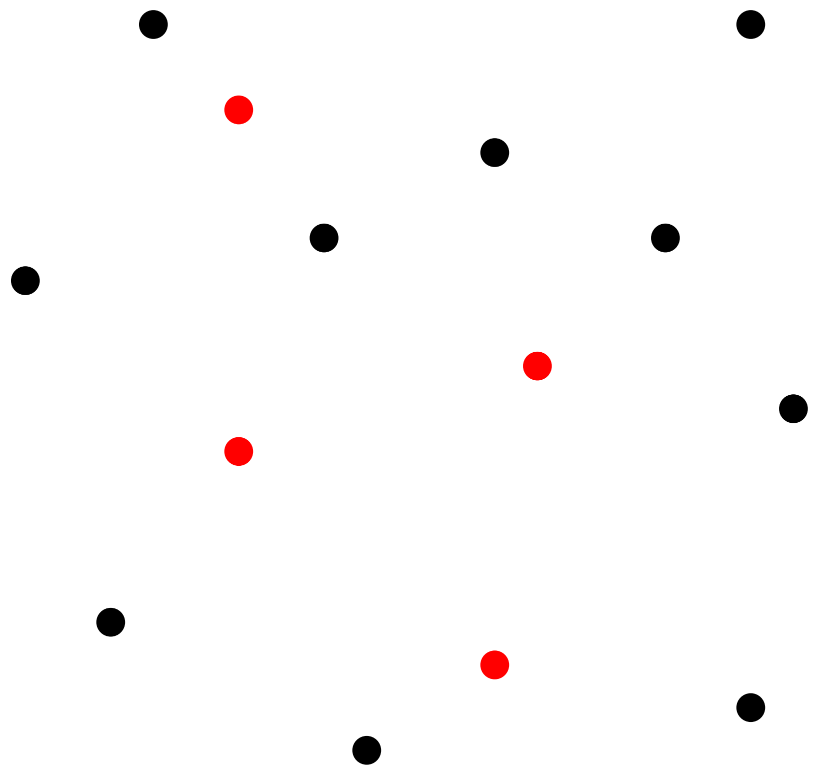
- Model of computation
- Frequency estimation (heavy hitters)
- Quantiles (order statistics)
- Other problems

ε-approximate range counting



Let P be a set of n points in the plane. Compute a summary structure so that, for any range Q (from a certain range space), $|P \cap Q|$ can be extracted with error εn

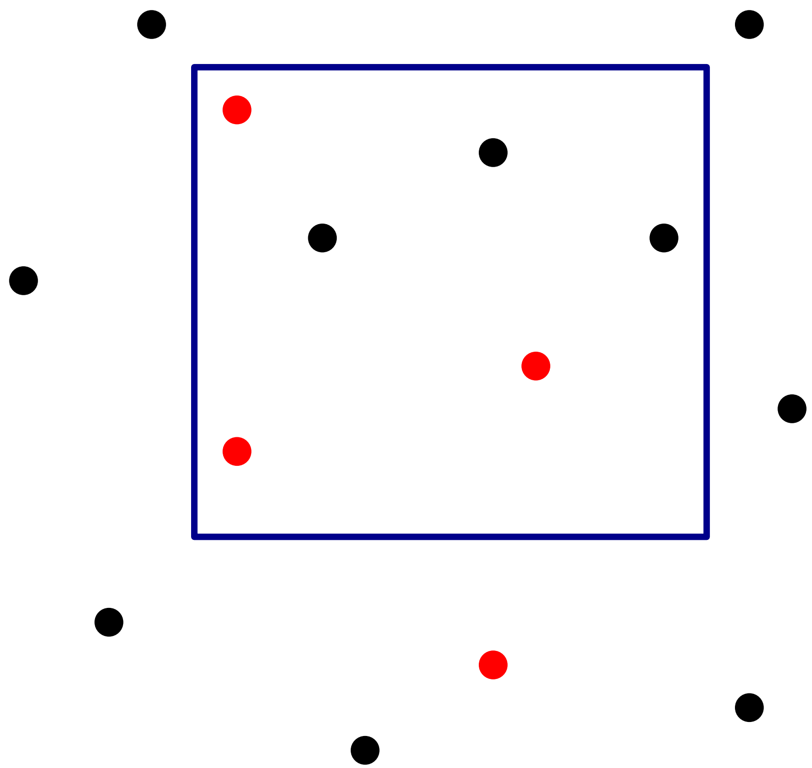
ε-approximations



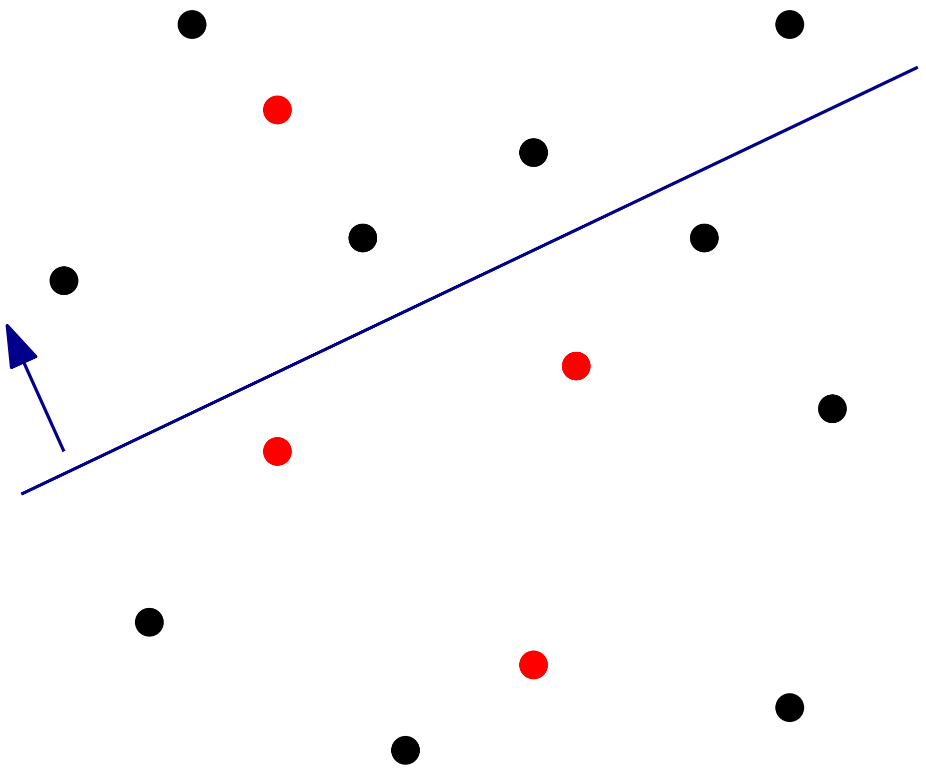
$S \subseteq P$ is an ε-approximation of P if for any Q (from a certain range space),

$$|P \cap Q| = |S \cap Q| \cdot \frac{n}{|S|} \pm \varepsilon n$$

ε-approximations



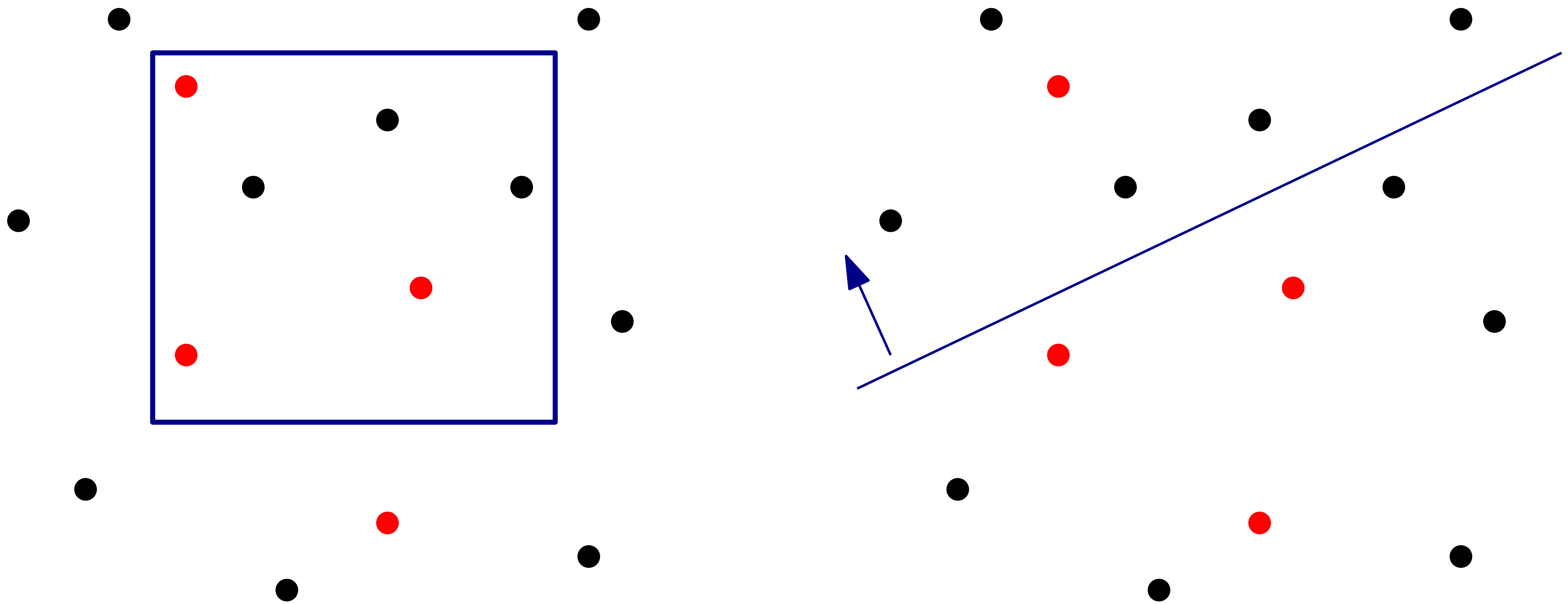
$1/\varepsilon \log^{O(1)}(1/\varepsilon)$



$1/\varepsilon^{4/3}$

Size of ε-approximations

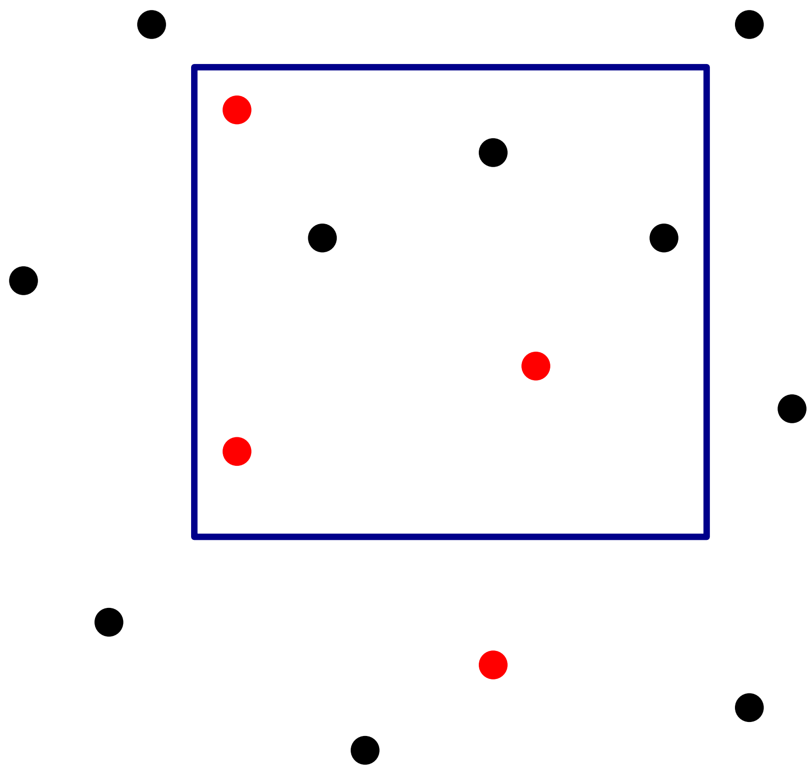
ε-approximations over k distributed data sets

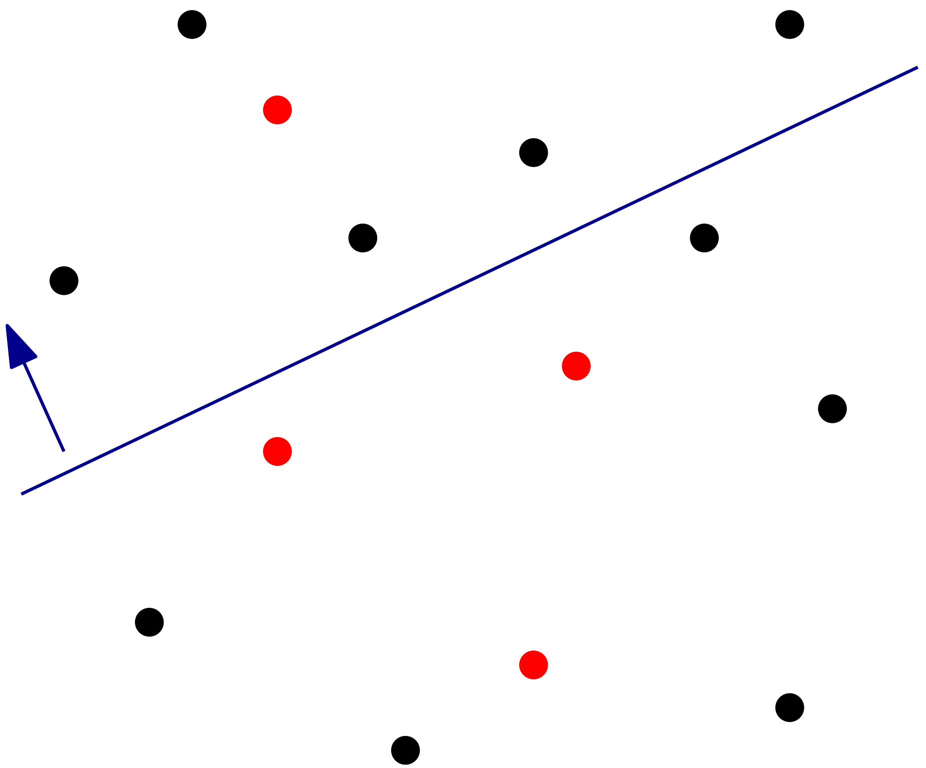


$\sqrt{k} \cdot 1/\varepsilon \log^{O(1)}(1/\varepsilon)$

$k^{1/3} \cdot 1/\varepsilon^{4/3}$

ε-approximations over k distributed data sets





$\sqrt{k} \cdot 1/\varepsilon \log^{O(1)}(1/\varepsilon)$ $\quad$ $k^{1/3} \cdot 1/\varepsilon^{4/3}$

Tight lower bounds (up to polylog factors).

The General Question

For what probems can we do better than $k \times$ sketch size?

The General Question

For what probems can we do better than $k \times$ sketch size?

- Some positive results in this talk

The General Question

For what probems can we do better than $k \times$ sketch size?

- Some positive results in this talk
- Negative results in [Woodruff, Zhang, '12]
  - Number of distinct elements (F_0)
    - Lower bound: $\tilde{\Omega}(k/\varepsilon^2)$
    - Upper bound: the distinct count sketch of size $O(1/\varepsilon^2)$ [Bar-Yossef, Jayram, Kumar, Sivakumar, Trevisan, '02]

The General Question

For what probems can we do better than $k\times$ sketch size?

- Some positive results in this talk
- Negative results in [Woodruff, Zhang, '12]
  - Number of distinct elements (F_0)
    - Lower bound: $\tilde{\Omega}(k/\varepsilon^2)$
    - Upper bound: the distinct count sketch of size $O(1/\varepsilon^2)$ [Bar-Yossef, Jayram, Kumar, Sivakumar, Trevisan, '02]
  - F_2
    - Lower bound: $\tilde{\Omega}(k/\varepsilon^2)$
    - Upper bound: the AMS sketch of size $O(1/\varepsilon^2)$ [Alon, Matias, Szegedy, '96]

References

- Optimal Sampling Algorithms for Frequency Estimation in Distributed Data. Zengfeng Huang, Ke Yi, Yunhao Liu, Guihai Chen. INFOCOM 2011.

- Sampling Based Algorithms for Quantile Computation in Sensor Networks. Zengfeng Huang, Lu Wang, Ke Yi, and Yunhao Liu. SIGMOD 2011.