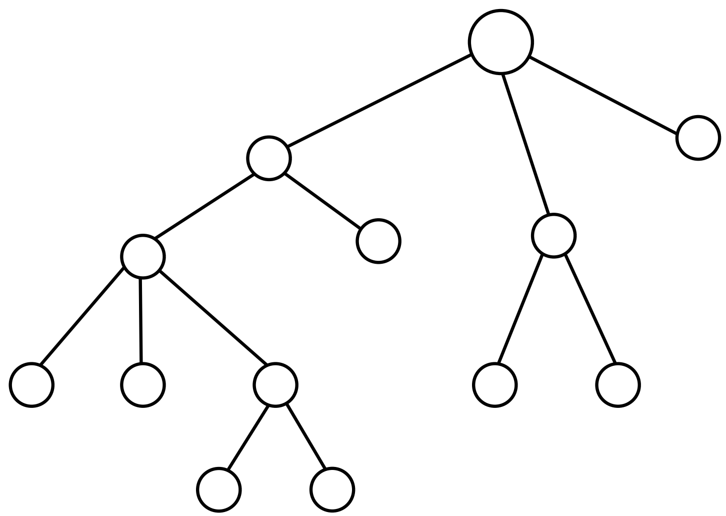
Sampling Based Algorithms for Quantile Computation in Sensor Networks

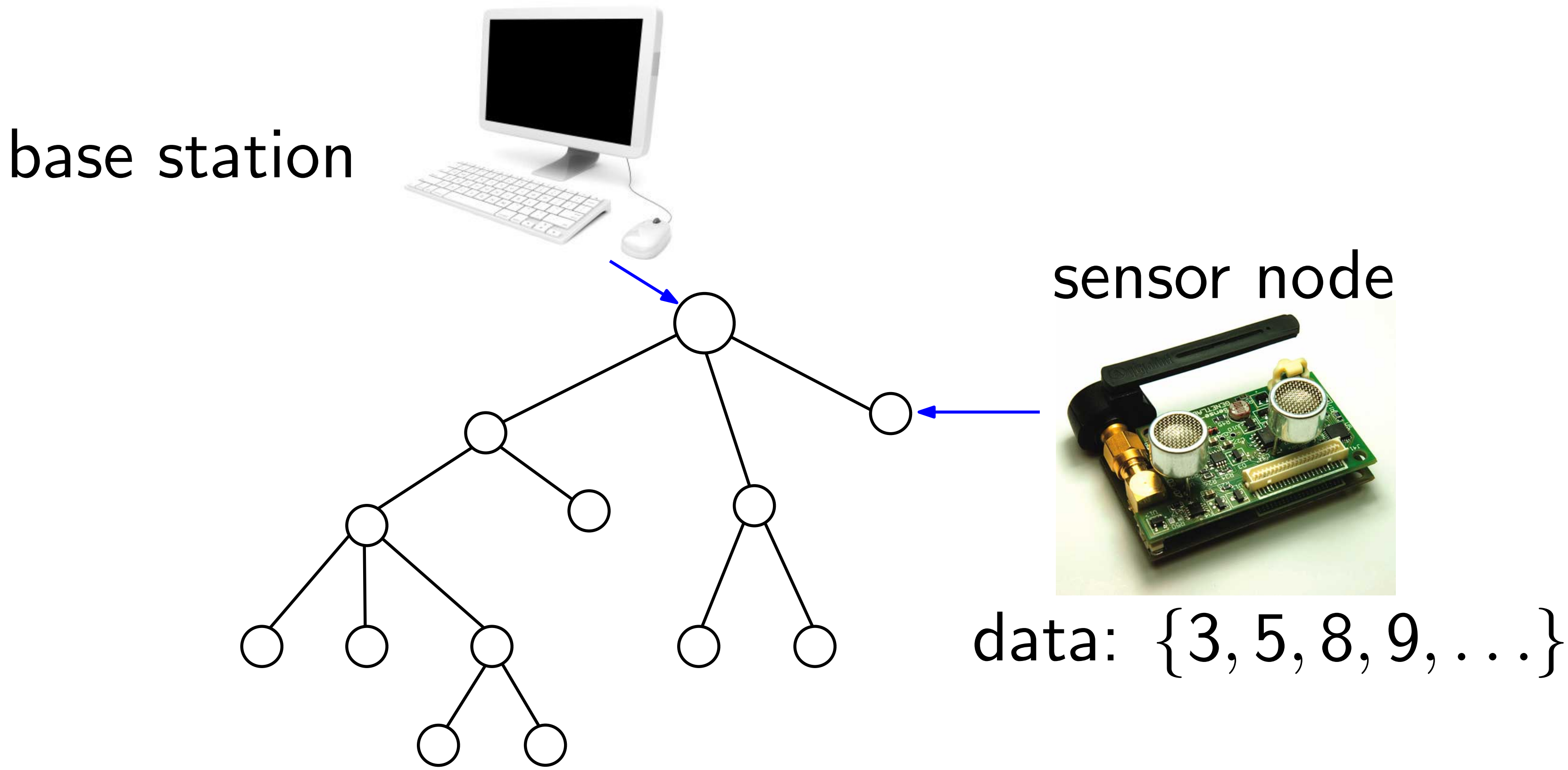


Zengfeng Huang, Lu Wang, Ke Yi, Yunhao Liu

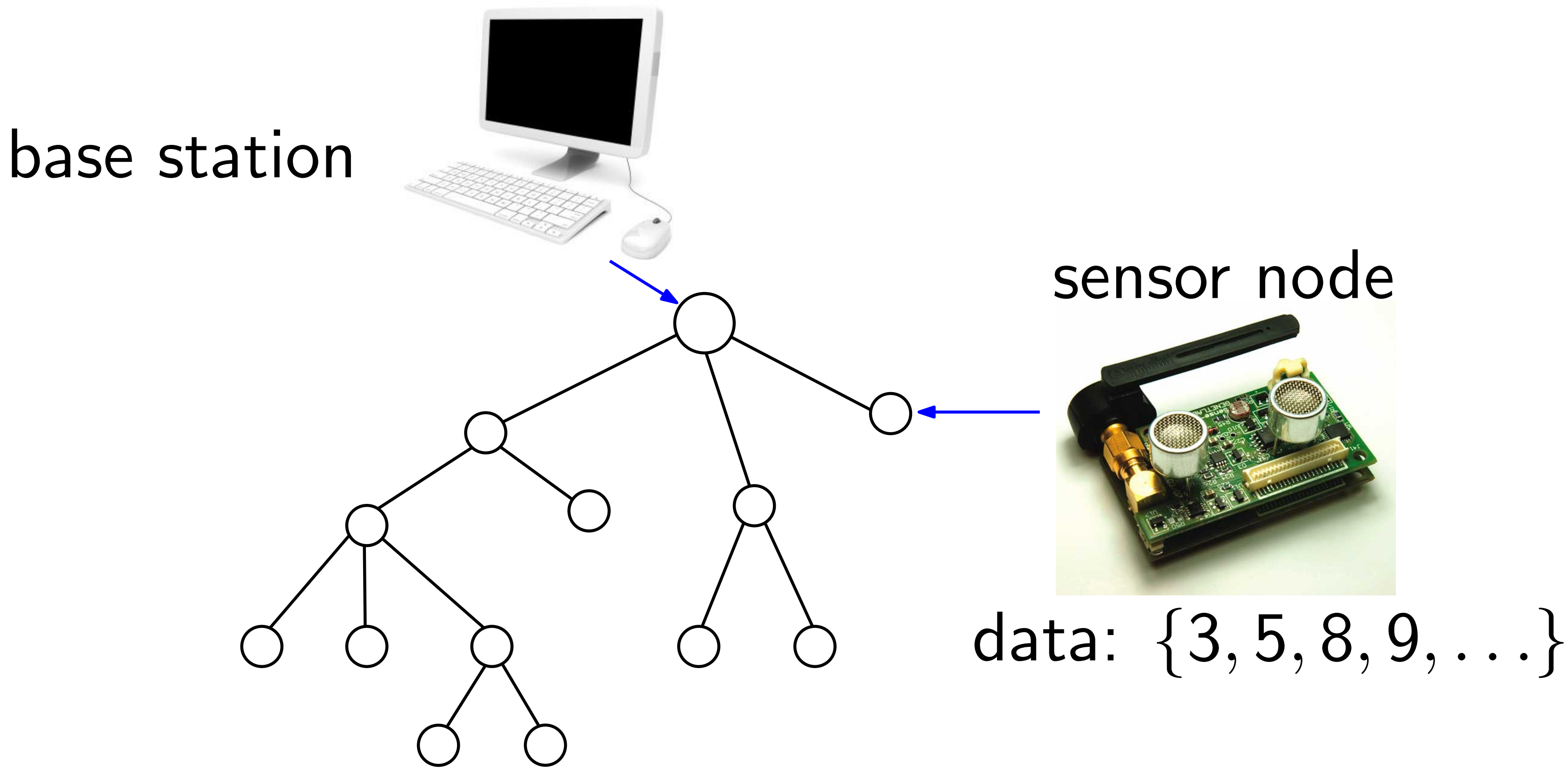
Hong Kong University of Science and Technology

SIGMOD'11

Wireless Sensor Networks



Wireless Sensor Networks



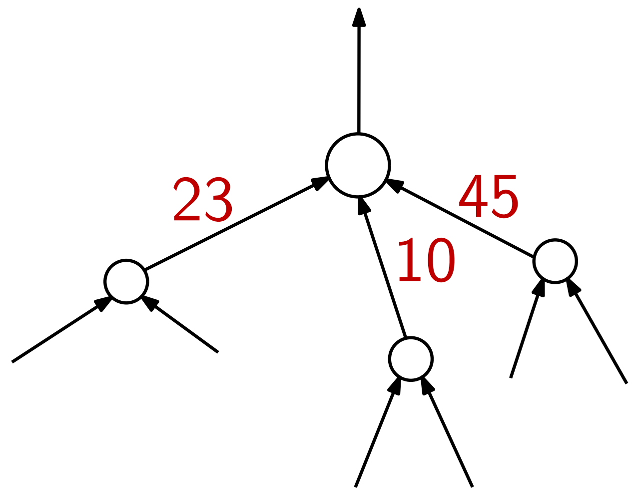
Assume for this talk:

- The network is a tree (may not be balanced).
- The tree has already been built.

Data Aggregation

Goal: Get all the data in an "aggregated form" to save communication.

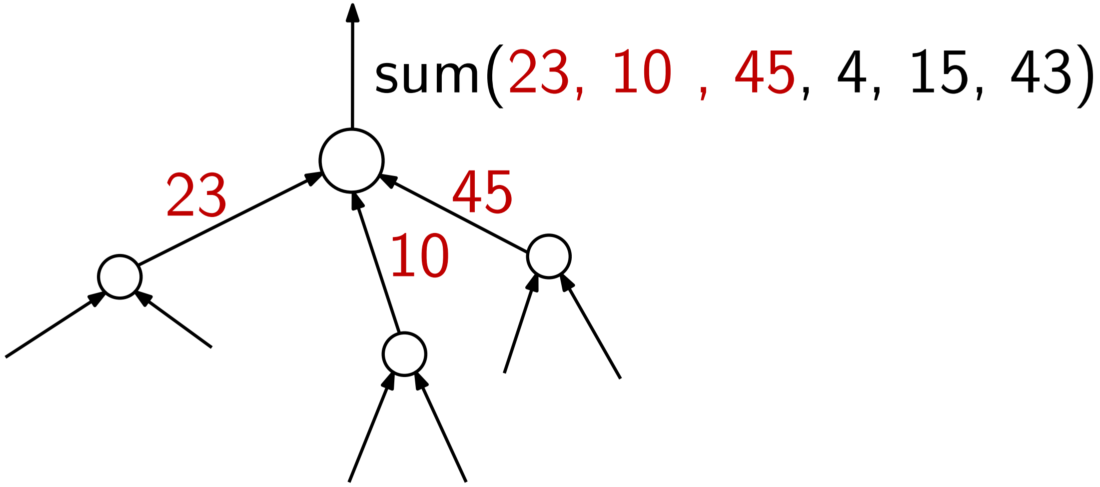
Example: sum



Data Aggregation

Goal: Get all the data in an “aggregated form” to save communication.

Example: sum

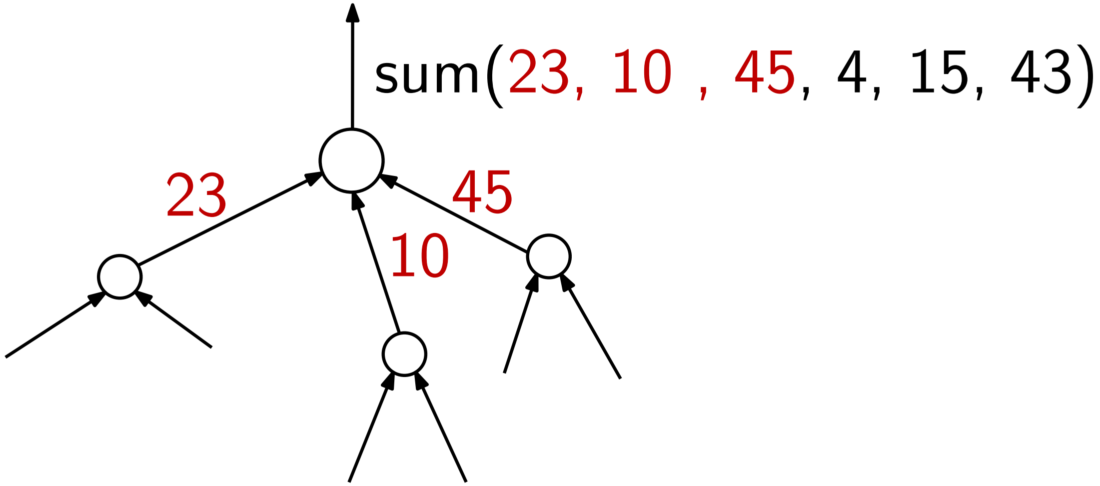


Data Aggregation

Goal: Get all the data in an "aggregated form" to save communication.

Example: sum

Also works for: max
count
average

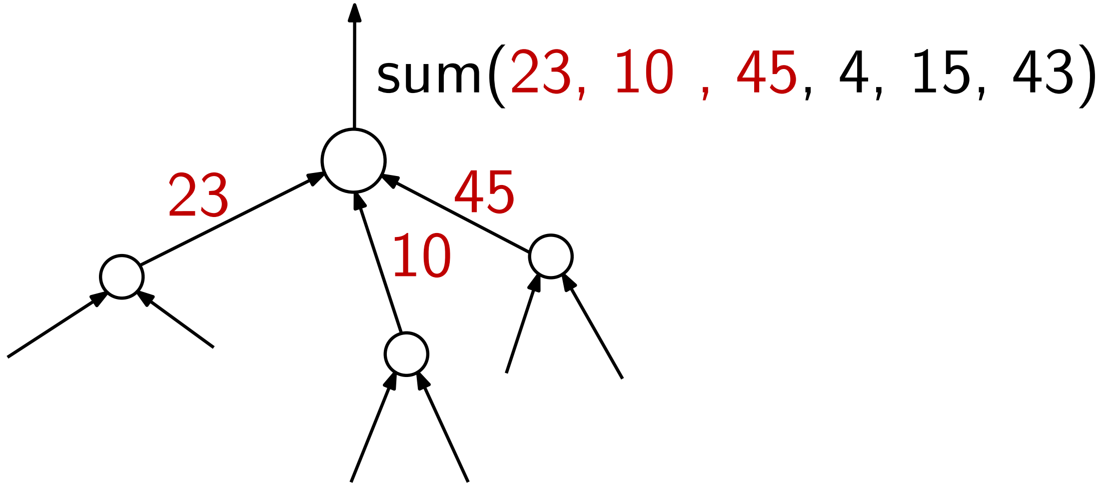


Data Aggregation

Goal: Get all the data in an "aggregated form" to save communication.

Example: sum

Also works for: max
count
average



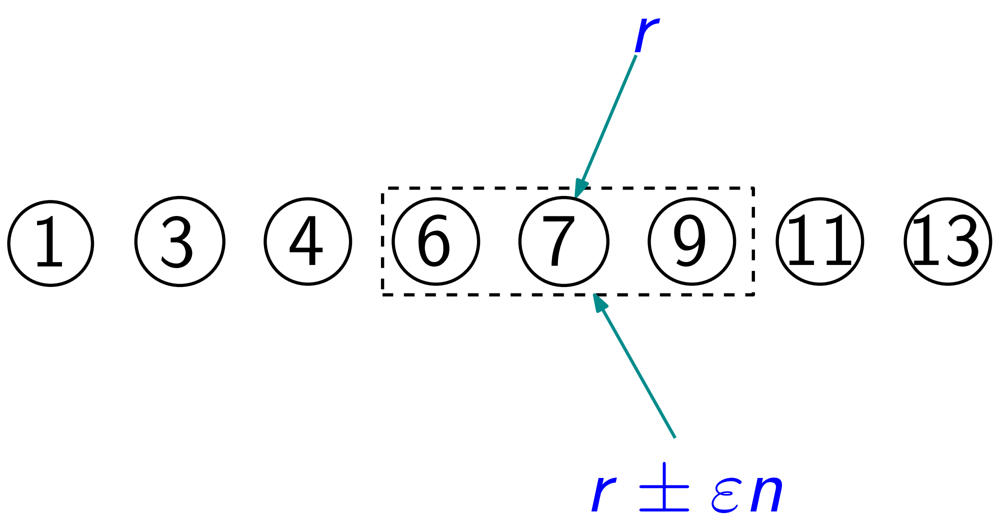
The Decomposable Property

A function f is decomposable if there exists some "combine" function g, such that for any two multisets A, B,

$$f(A \uplus B) = g(f(A), f(B))$$

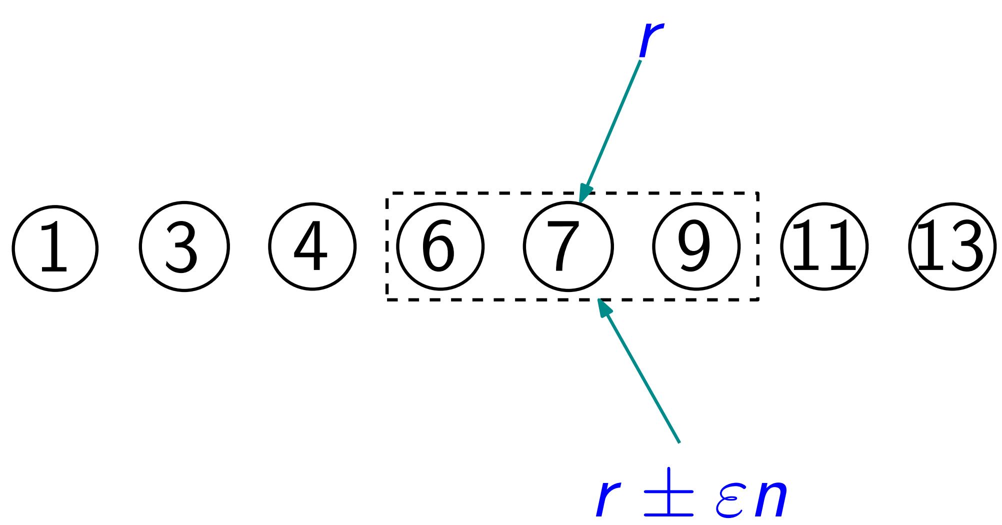
Quantiles

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



Quantiles

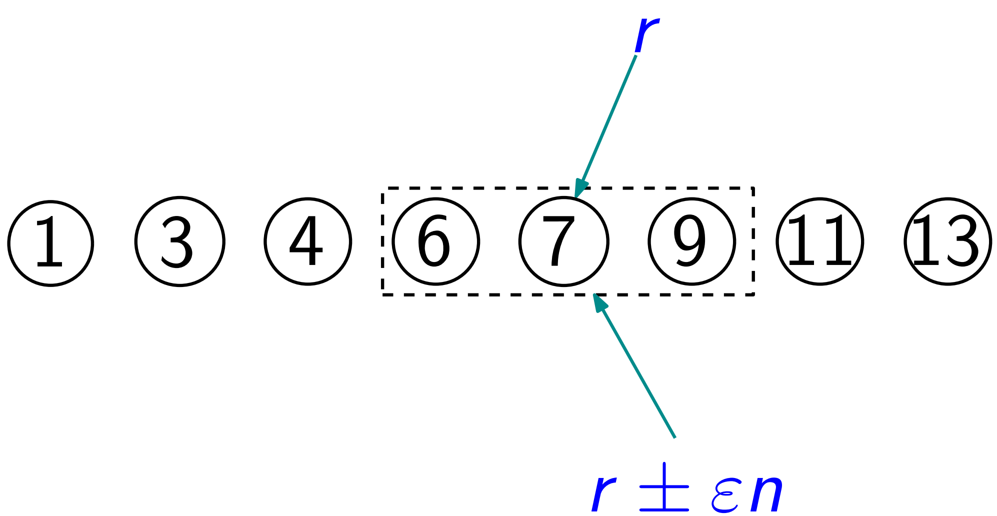
In a set of n values, the (r/n)-quantile is the value ranked at r.
The 0.5-quantile is the median.



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

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 $[r - \varepsilon n, r + \varepsilon n]$.

Problem: Quantiles are NOT decomposable

Quantile Summaries

Previous solution: Design a quantile summary that is decomposable, and any (ε-approximate) quantile can be extracted from this summary

	summary size	
q-digest [Shrivastava et al. '04] 272 citations	$O(\frac{1}{\varepsilon}\log u)$	
GK [Greenwald, Khanna '04] 102 citations	$O(\frac{1}{\varepsilon}\log^2 n)$	

n: total data size.

ε: error. $10^{-2} - 10^{-4}$

k: number of nodes. $100 \sim 10000$

h: height of the tree. $\log k \sim \sqrt{k}$

u: size of universe. $\log u = 32$

Quantile Summaries

Previous solution: Design a quantile summary that is decomposable, and any (ε-approximate) quantile can be extracted from this summary

	summary size	total cost
q-digest [Shrivastava et al. '04] 272 citations	$O(\frac{1}{\varepsilon}\log u)$	$O(\frac{k}{\varepsilon}\log u)$
GK [Greenwald, Khanna '04] 102 citations	$O(\frac{1}{\varepsilon}\log^2 n)$	$O(\frac{k}{\varepsilon}\log^2 n)$

n: total data size.

ε: error. $10^{-2} - 10^{-4}$

k: number of nodes. $100 \sim 10000$

h: height of the tree. $\log k \sim \sqrt{k}$

u: size of universe. $\log u = 32$

Quantile Summaries

Previous solution: Design a quantile summary that is ~~decomposable~~, and any (ε-approximate) quantile can be extracted from this summary

	summary size	total cost
q-digest [Shrivastava et al. '04] 272 citations	$O(\frac{1}{\varepsilon}\log u)$	$O(\frac{k}{\varepsilon}\log u)$
GK [Greenwald, Khanna '04] 102 citations	$O(\frac{1}{\varepsilon}\log^2 n)$	$O(\frac{k}{\varepsilon}\log^2 n)$
New	$O(\frac{1}{\varepsilon}\log k)$	$O(\frac{\sqrt{kh}}{\varepsilon})$

n: total data size.

ε: error. $10^{-2} - 10^{-4}$

k: number of nodes. $100 \sim 10000$

h: height of the tree. $\log k \sim \sqrt{k}$

u: size of universe. $\log u = 32$

Quantile Summaries

Previous solution: Design a quantile summary that is ~~decomposable~~, and any (ε-approximate) quantile can be extracted from this summary

	summary size	total cost
q-digest [Shrivastava et al. '04] 272 citations	$O(\frac{1}{\varepsilon}\log u)$	$O(\frac{k}{\varepsilon}\log u)$
GK [Greenwald, Khanna '04] 102 citations	$O(\frac{1}{\varepsilon}\log^2 n)$	$O(\frac{k}{\varepsilon}\log^2 n)$
New	$O(\frac{1}{\varepsilon}\log k)$	$O(\frac{\sqrt{kh}}{\varepsilon})$ $+k$

n: total data size.

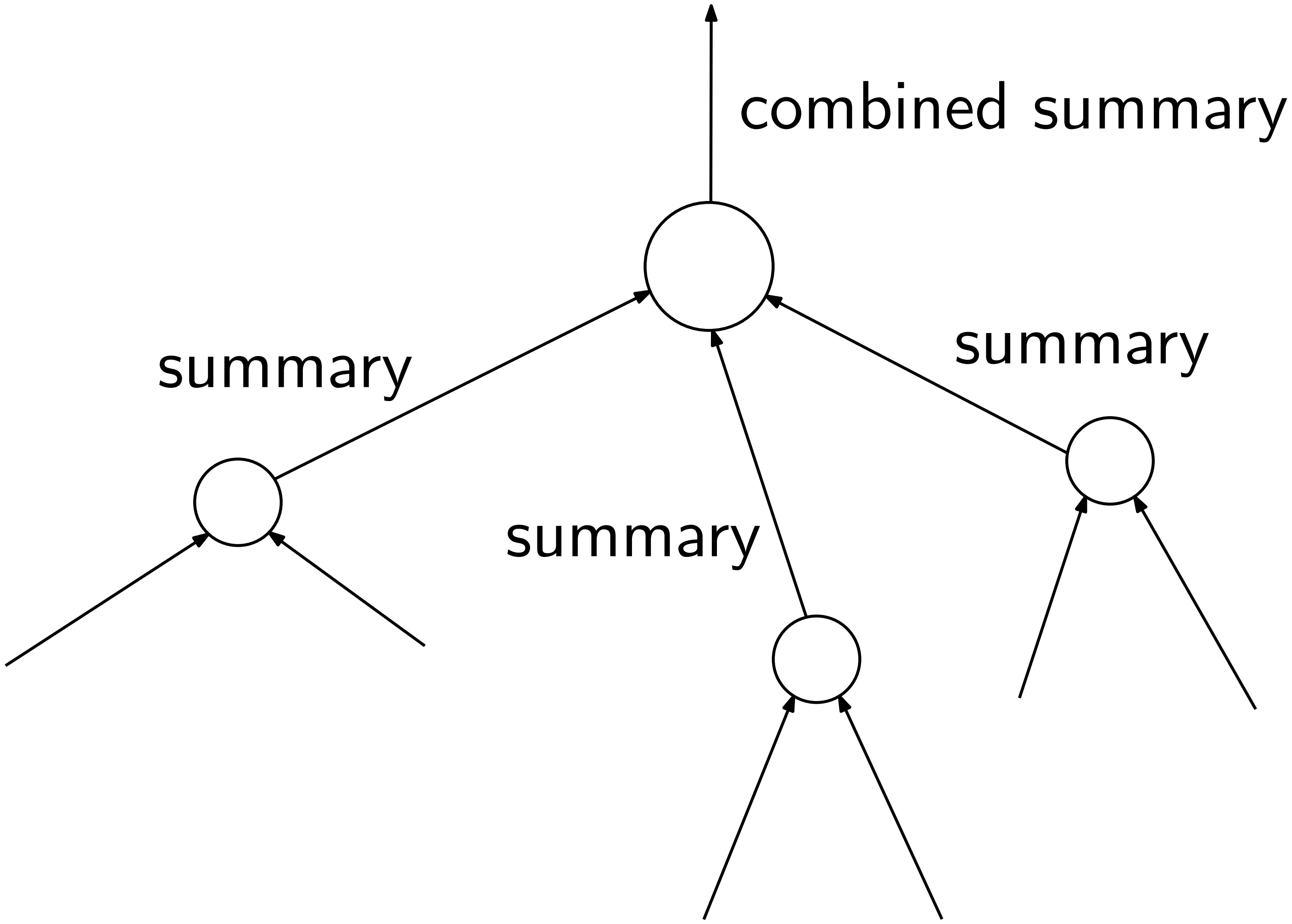
ε: error. $10^{-2} - 10^{-4}$

k: number of nodes. $100 \sim 10000$

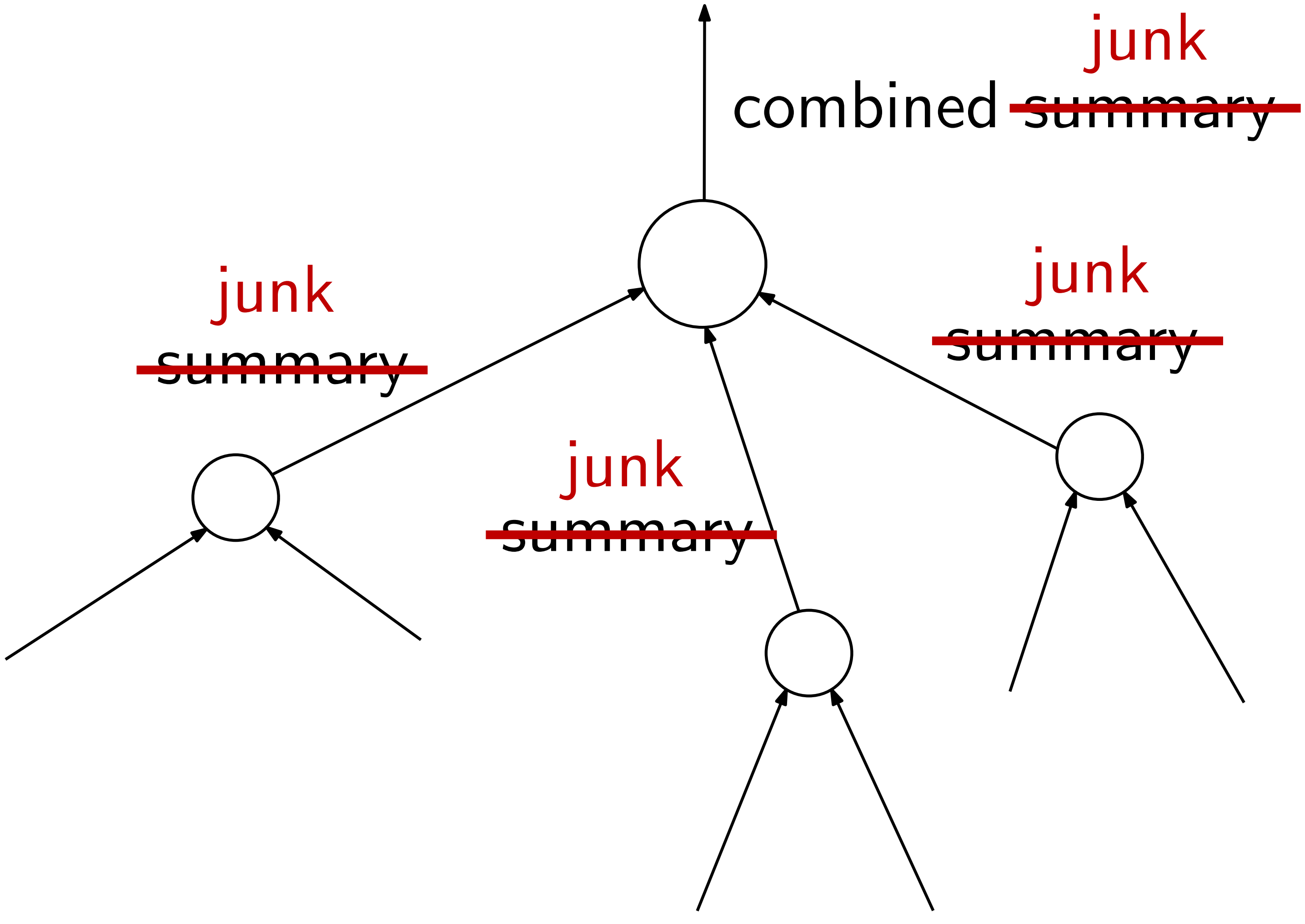
h: height of the tree. $\log k \sim \sqrt{k}$

u: size of universe. $\log u = 32$

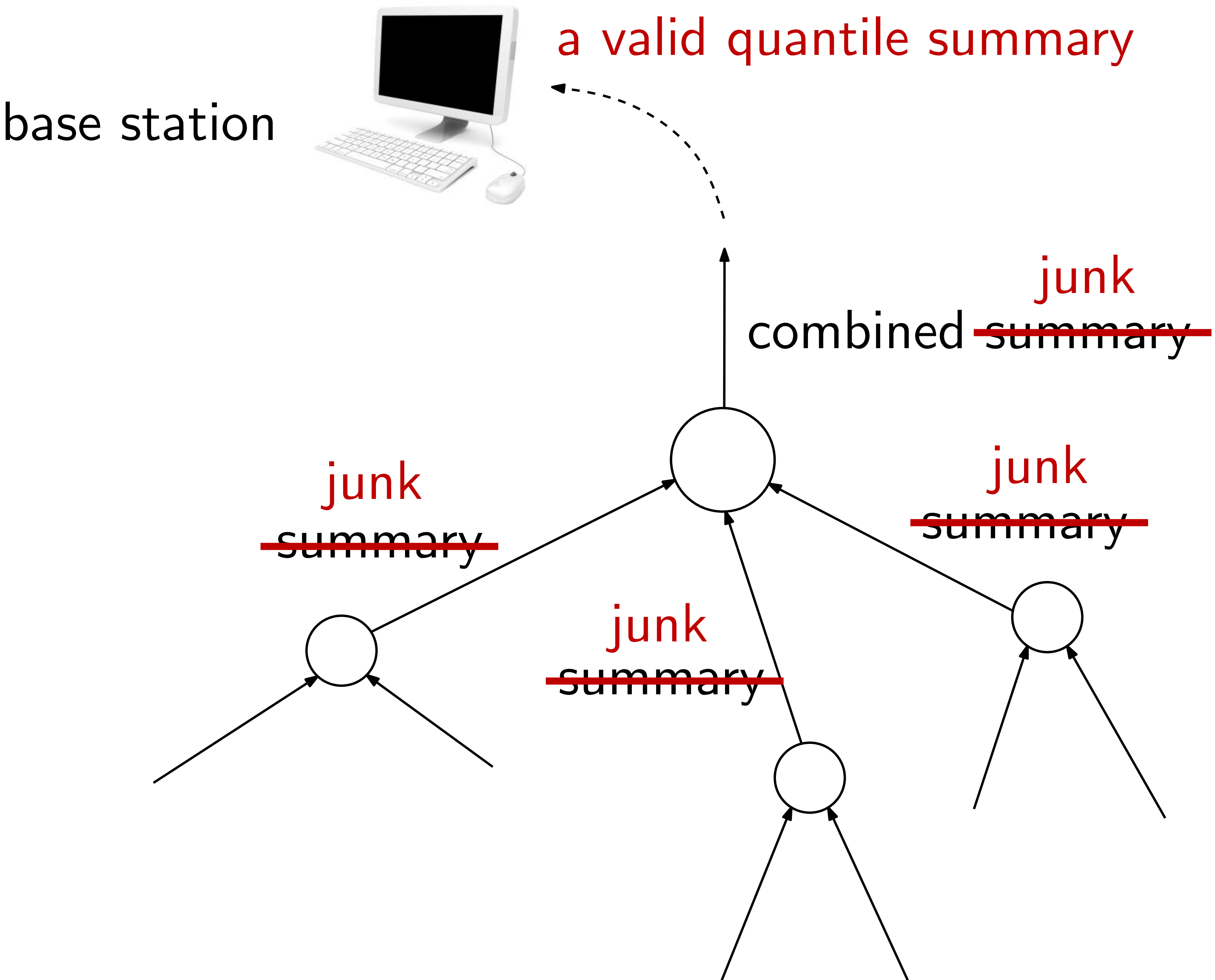
Our Approach



Our Approach

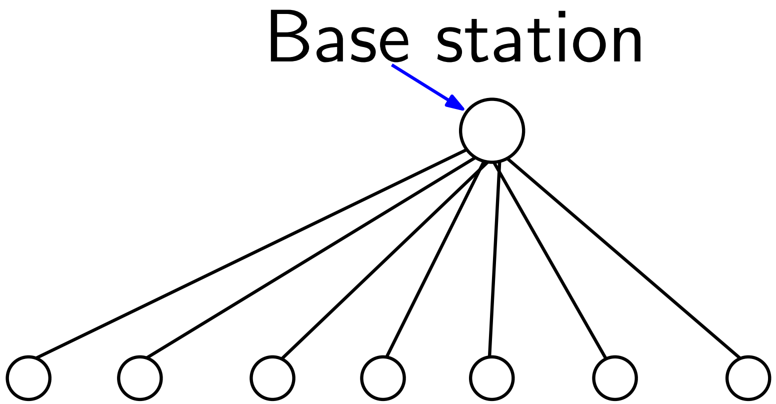


Our Approach

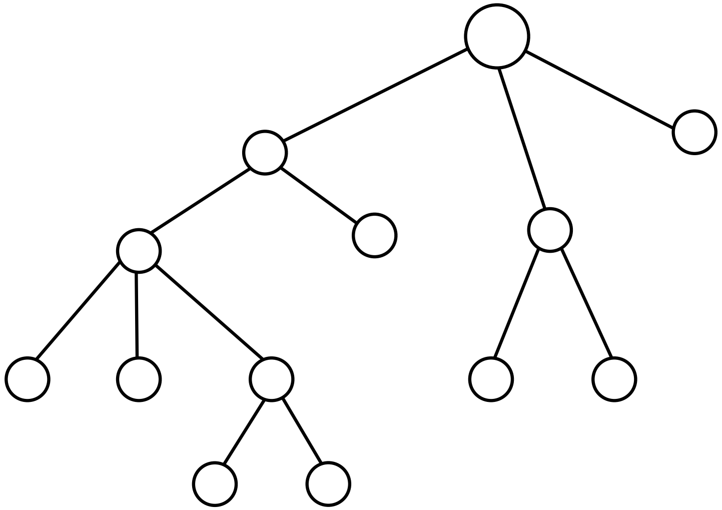


Roadmap

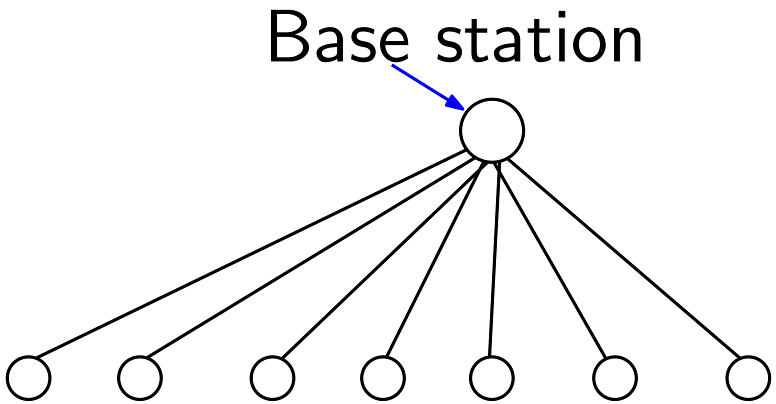
- Flat model



- Tree model

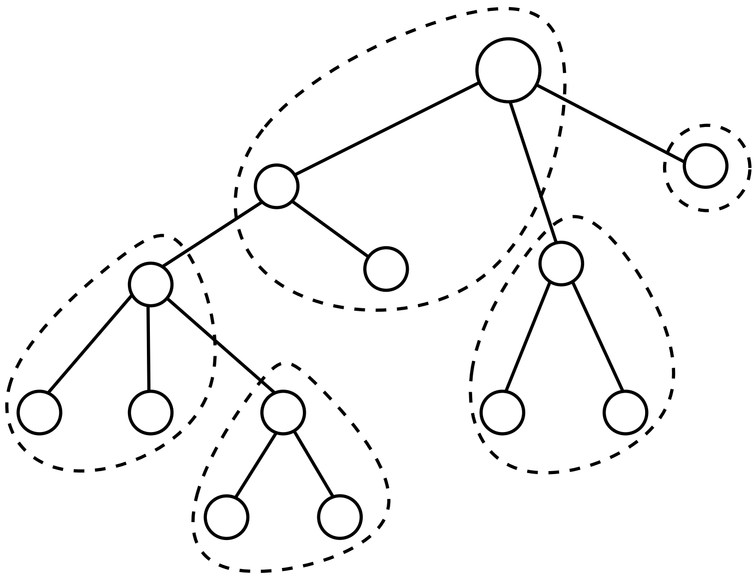


Roadmap



- Flat model

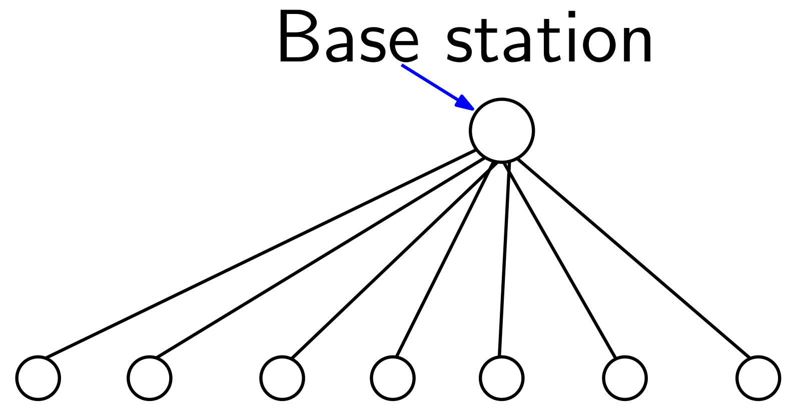
- Tree model



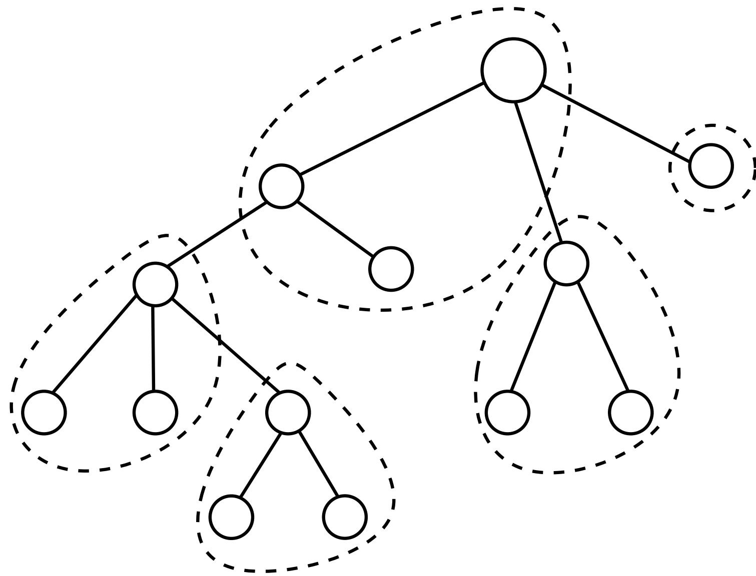
- Partitioned tree

Roadmap

- Flat model



- Tree model



- Partitioned tree

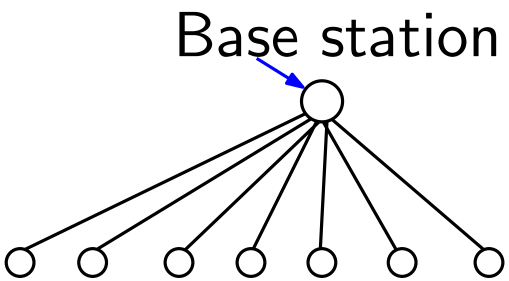
value-to-rank queries

10 $\quad r(10) = 7$

1 3 4 6 7 9 11 13

Return any number within $r(10) \pm \varepsilon n$

The Flat Model: Algorithm



The algorithm for each node

Sample each value with probabiltiy p

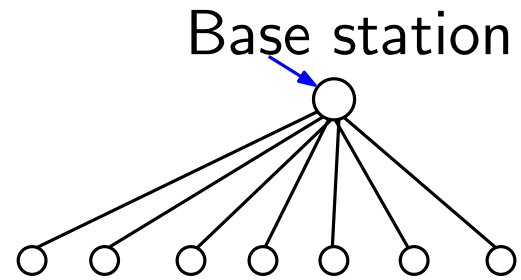
1 3 4 6 7 9 11 13 16 26 21 24

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

Compute local ranks

Base station

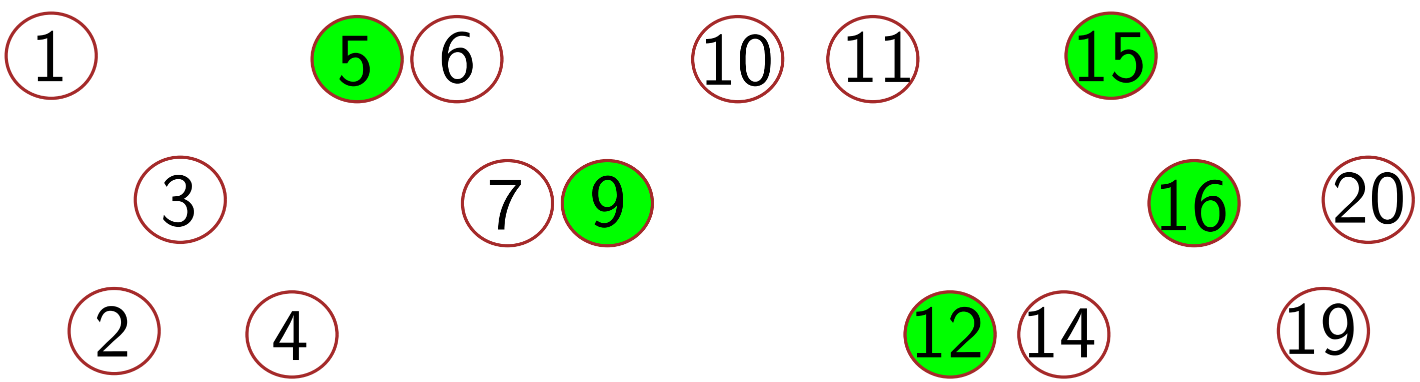
The Flat Model: Algorithm



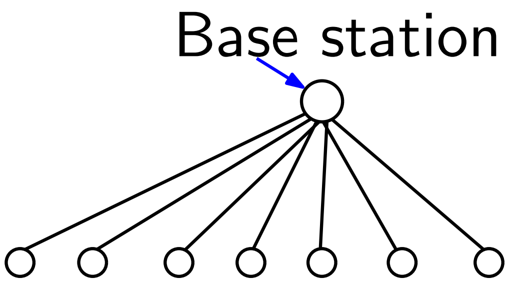
At the base station:

Answering value-to-rank query

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



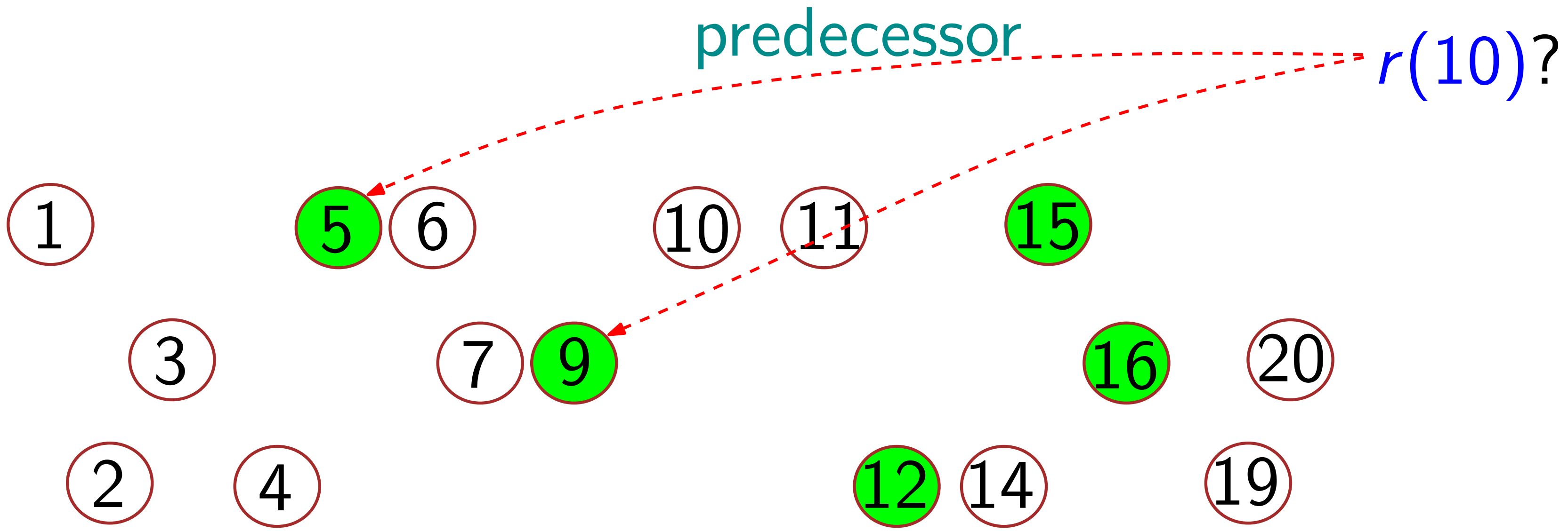
The Flat Model: Algorithm



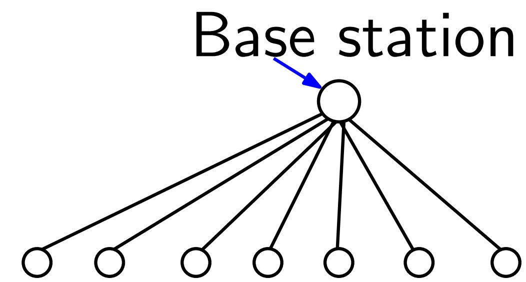
At the base station:

Answering value-to-rank query

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



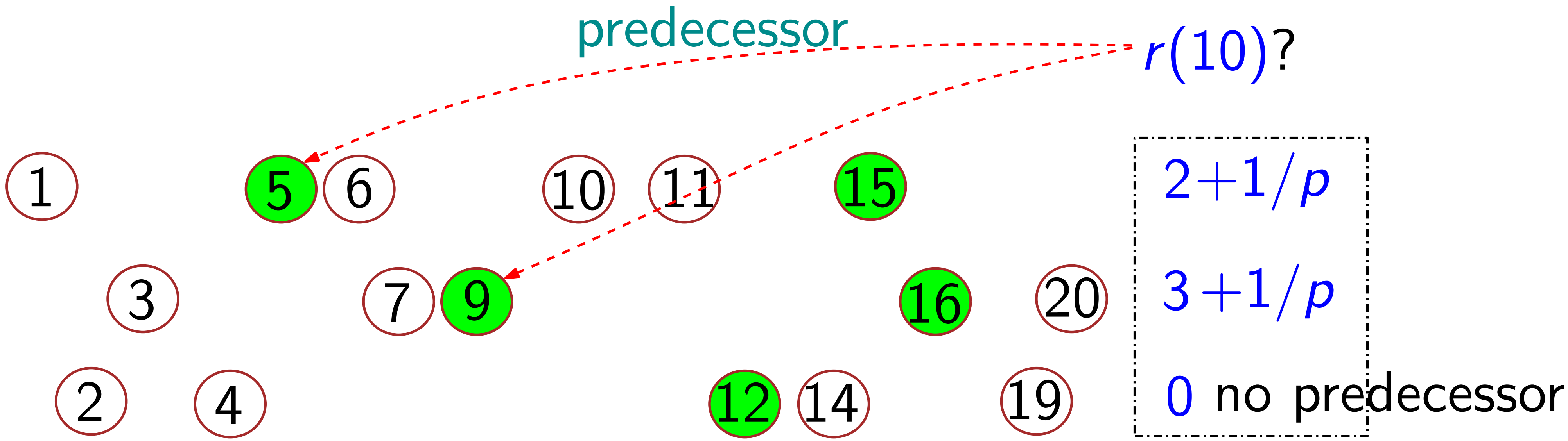
The Flat Model: Algorithm



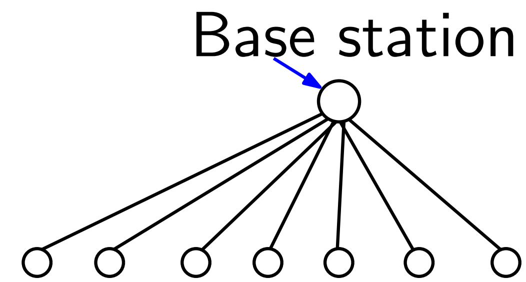
At the base station:

Answering value-to-rank query

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



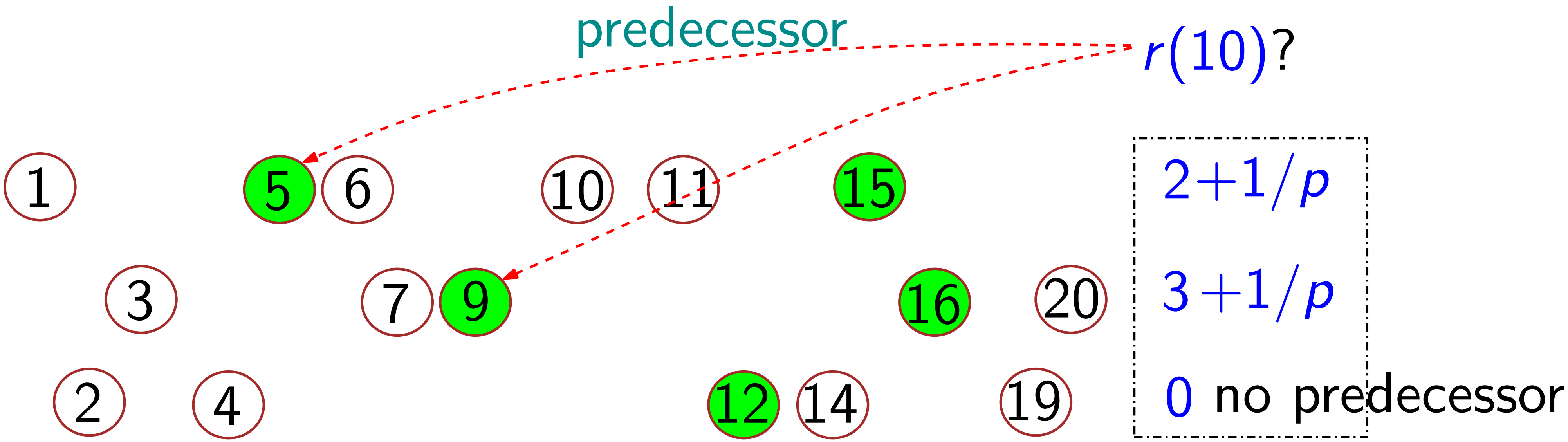
The Flat Model: Algorithm



At the base station:

Answering value-to-rank query

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



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

The Flat Model: 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

The Flat Model: Correctness

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

$r(10)$?

5 15

The Flat Model: Correctness

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

$r(10)$?

5 ? 15

The Flat Model: 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)

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

The Flat Model: 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)

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

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

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

The Flat Model: Communication Cost

- Total cost: $np = \sqrt{k}/\varepsilon$ in expectation
- Max individual node cost: $O(\sqrt{k}/\varepsilon)$

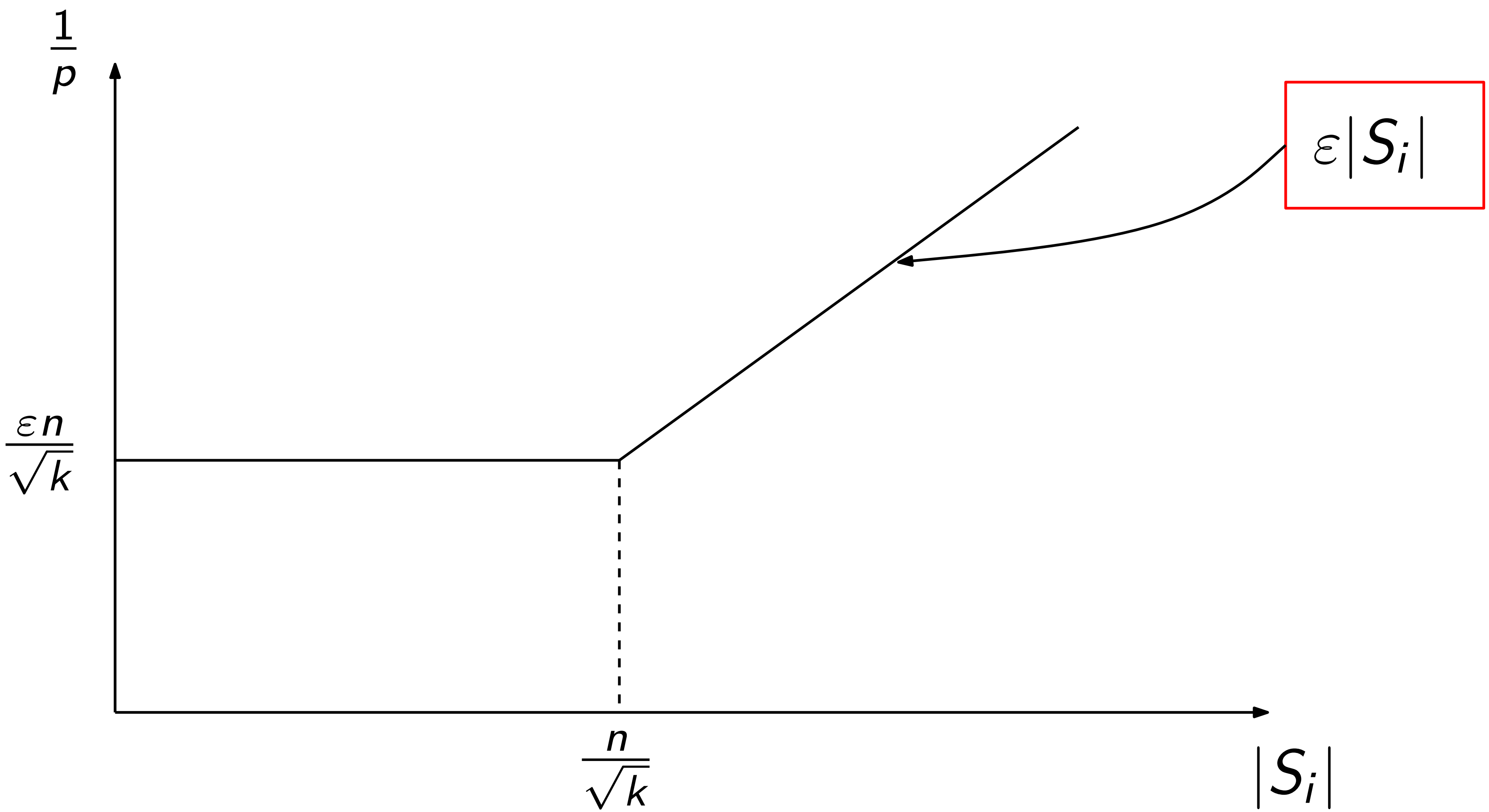
The Flat Model: Communication Cost

- Total cost: $np = \sqrt{k}/\varepsilon$ in expectation
- Max individual node cost: $O(\sqrt{k}/\varepsilon)$

  Reduce to $O(1/\varepsilon)$

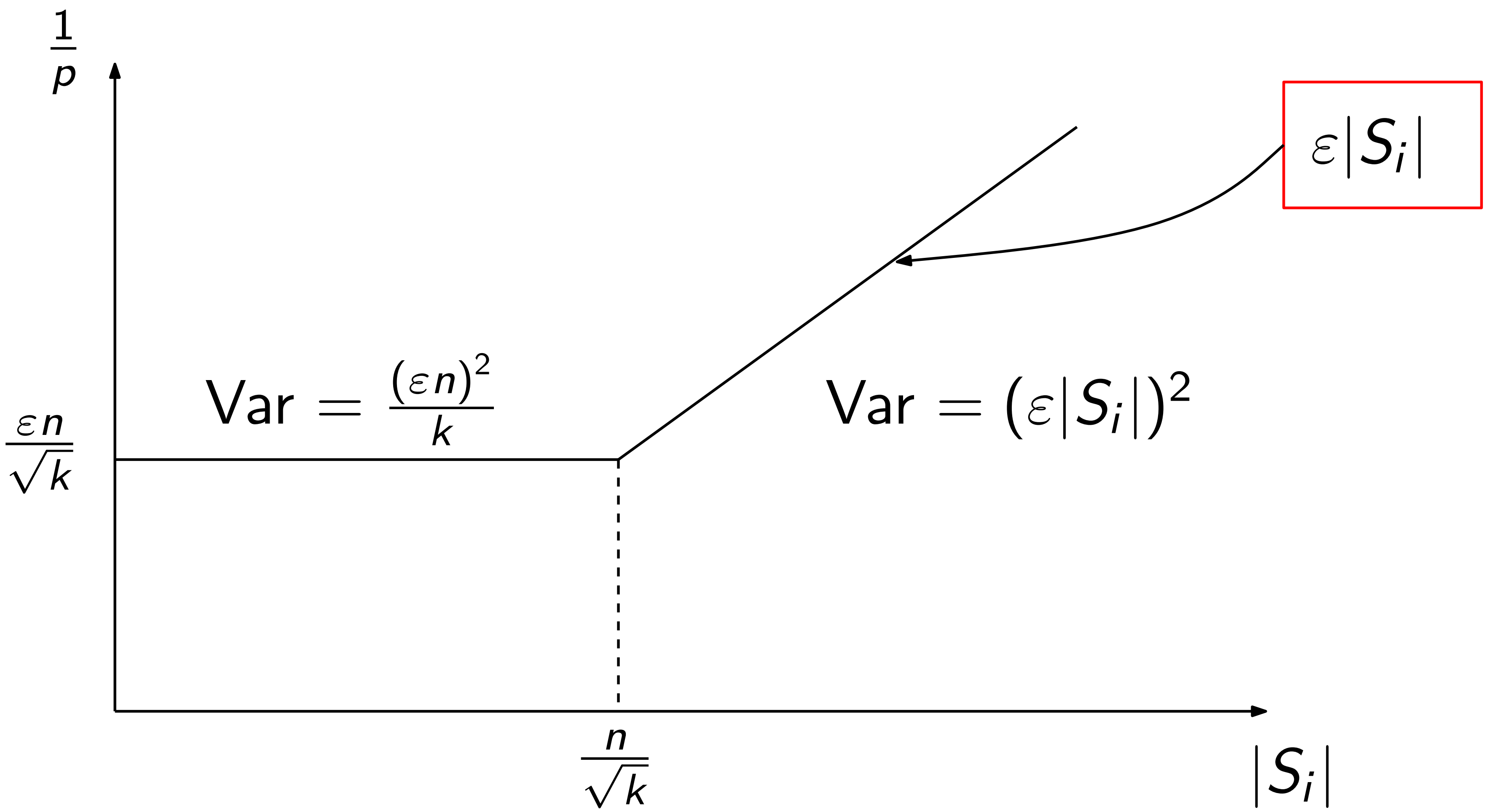
The Flat Model: Communication Cost

Let S_i be the set of data collected at node i



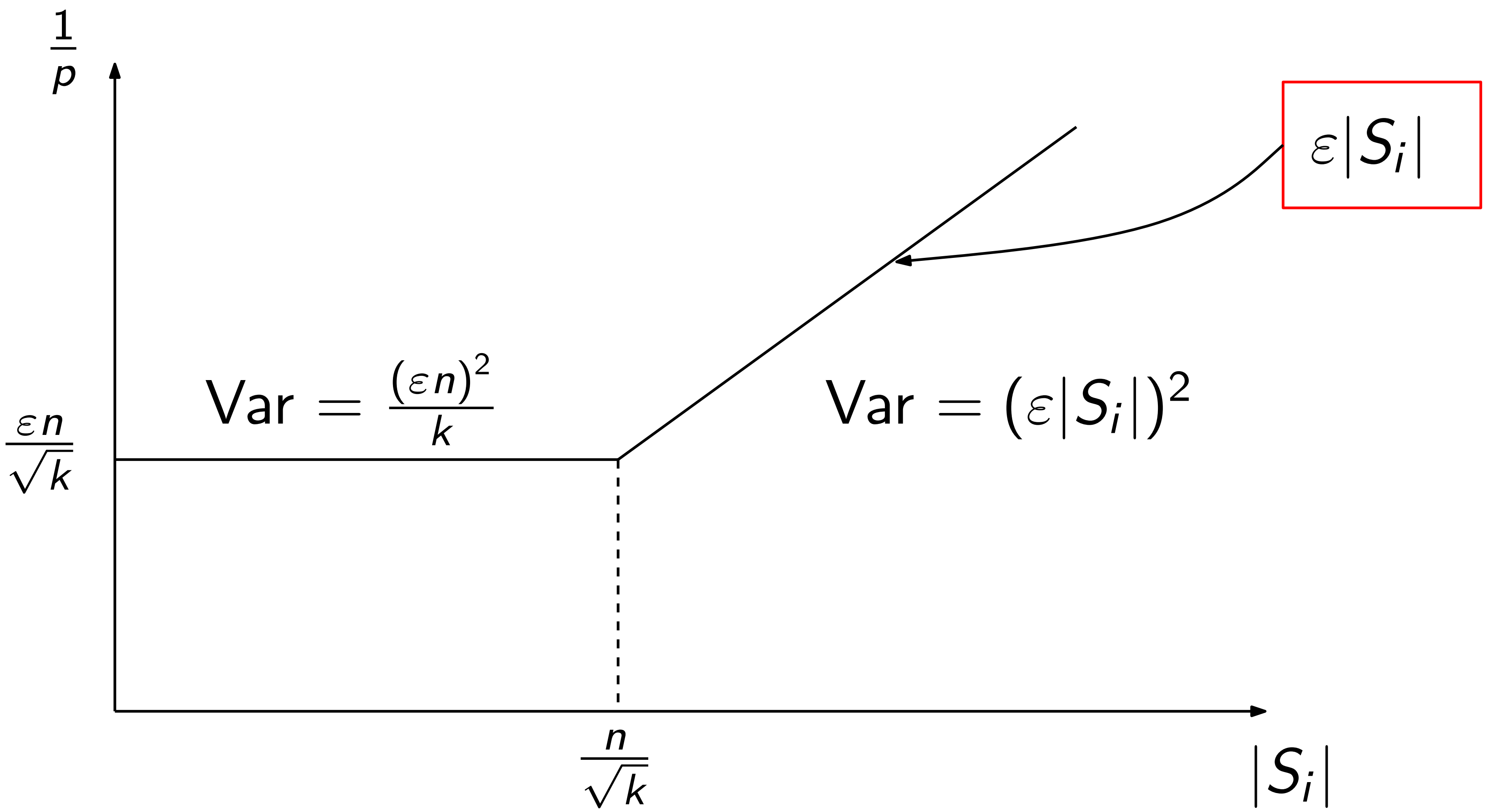
The Flat Model: Communication Cost

Let S_i be the set of data collected at node i



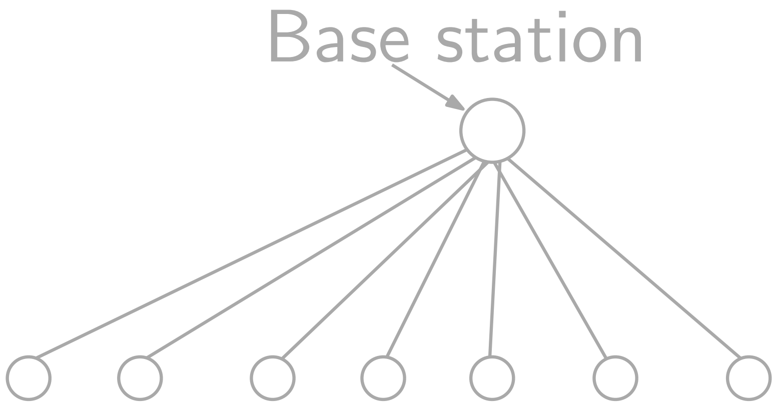
The Flat Model: Communication Cost

Let S_i be the set of data collected at node i



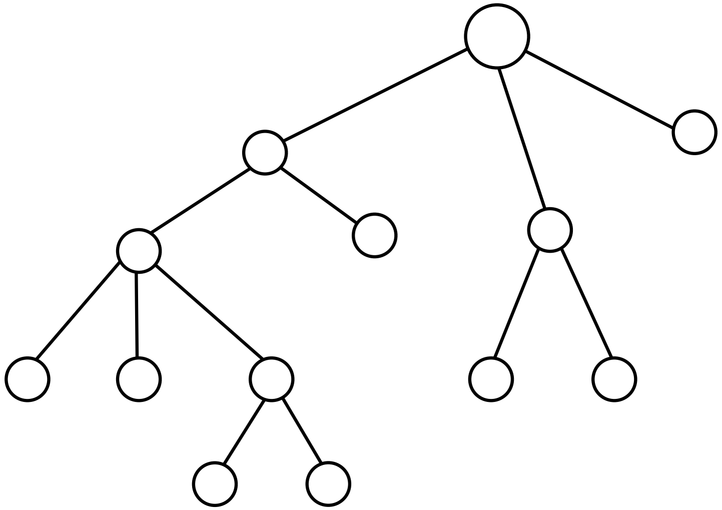
$$\text{Total variance} \leq \sum_i \left(\frac{(\varepsilon n)^2}{k} + (\varepsilon |S_i|)^2\right) \leq 2(\varepsilon n)^2$$

Roadmap



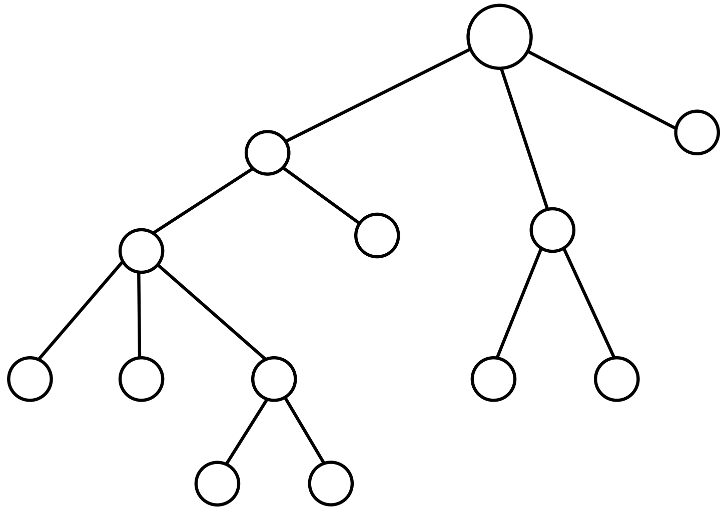
- Flat model

- Tree model



Tree Model: Naive Extension from Flat Model

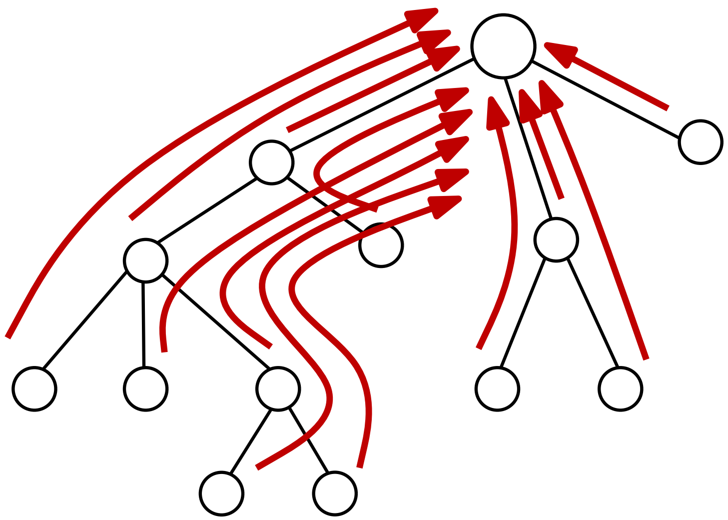
Total cost: $\frac{\sqrt{k}}{\varepsilon}h$ (h is the height of the tree)



Tree Model: Naive Extension from Flat Model

Total cost: $\frac{\sqrt{k}}{\varepsilon}h$ (h is the height of the tree)

Max individual cost: $O(\frac{\sqrt{k}}{\varepsilon})$



Tree Model: Merging Samples

S_i ① ③ ④ ⑥ ⑦ ⑨ ⑪ ⑬ ⑯ ㉖ ㉑ ㉔

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

$$p_i = \begin{cases} \frac{\sqrt{k}}{\varepsilon n}, & |S_i| \le \frac{n}{\sqrt{k}} \\ \frac{1}{\varepsilon |S_i|}, & |S_i| \ge \frac{n}{\sqrt{k}} \end{cases}$$

Tree Model: Merging Samples

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

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

$$p_i = \begin{cases} \frac{\sqrt{k}}{\varepsilon n}, & |S_i| \le \frac{n}{\sqrt{k}} \\ \frac{1}{\varepsilon |S_i|}, & |S_i| \ge \frac{n}{\sqrt{k}} \end{cases}$$

$|S_1|$ p_1 $(3,2)$ $(7,5)$ $(13,8)$ $(26,10)$

$+$

$|S_2|$ p_2 $(5,3)$ $(14,6)$ $(18,8)$ $(24,11)$

$=$

$|S_1| + |S_2|$ p

$\left(> \frac{n}{\sqrt{k}}\right)$

Tree Model: Merging Samples

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

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

$$p_i = \begin{cases} \frac{\sqrt{k}}{\varepsilon n}, & |S_i| \le \frac{n}{\sqrt{k}} \\ \frac{1}{\varepsilon |S_i|}, & |S_i| \ge \frac{n}{\sqrt{k}} \end{cases}$$

$|S_1|$ p_1 $(3,2)$ $(7,5)$ $(13,8)$ $(26,10)$

Sample w.p. p/p_1

$+$

$|S_2|$ p_2 $(5,3)$ $(14,6)$ $(18,8)$ $(24,11)$

$=$

Sample w.p. p/p_2

$|S_1| + |S_2|$ p

$\left(> \frac{n}{\sqrt{k}}\right)$

Tree Model: Merging Samples

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

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

$$p_i = \begin{cases} \frac{\sqrt{k}}{\varepsilon n}, & |S_i| \le \frac{n}{\sqrt{k}} \\ \frac{1}{\varepsilon |S_i|}, & |S_i| \ge \frac{n}{\sqrt{k}} \end{cases}$$

$|S_1|$ p_1 $(3, 2)$ $(7, 5)$ $(13, 8)$ $(26, 10)$

$+$

$|S_2|$ p_2 $(5, 3)$ $(14, 6)$ $(18, 8)$ $(24, 11)$

$=$

Sample w.p. p/p_1

Sample w.p. p/p_2

$|S_1| + |S_2|$ p $(13, 8+?)$

$\left(> \frac{n}{\sqrt{k}}\right)$

Tree Model: Merging Samples

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

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

$$p_i = \begin{cases} \frac{\sqrt{k}}{\varepsilon n}, & |S_i| \le \frac{n}{\sqrt{k}} \\ \frac{1}{\varepsilon |S_i|}, & |S_i| \ge \frac{n}{\sqrt{k}} \end{cases}$$

$|S_1|$ p_1 $(3,2)$ $(7,5)$ $(13,8)$ $(26,10)$

$+$

$|S_2|$ p_2 $(5,3)$ $(14,6)$ $(18,8)$ $(24,11)$

$=$

Sample w.p. p/p_1

Sample w.p. p/p_2

$|S_1| + |S_2|$ p $(13, 8+?)$

$\left(> \frac{n}{\sqrt{k}}\right)$

This is a rank query in the other sample!

Tree Model: Merging Samples

Problem: Variances accumulate

(*Law of total variance*)

$$|S_1| \quad p_1 \qquad (3,2)\ (7,5)\ (13,8)\ (26,10)$$

$$+$$

$$|S_2| \quad p_2 \qquad (5,3)\ (14,6)\ (18,8)\ (24,11)$$

$$=$$

$$(13, 8+3+1/p_2)$$

Tree Model: Merging Samples

Problem: Variances accumulate

(*Law of total variance*)

$$|S_1| \quad p_1 \qquad (3,2)\ (7,5)\ \ (13,8)\ (26,10)$$

$$+$$

$$|S_2| \quad p_2 \qquad (5,3)\ (14,6)\ (18,8)\ (24,11)$$

$$=$$

$$(13, 8+3+1/p_2)$$

Var[local count of 13 in $S_1 \cup S_2$] $=$
Var[local count of 13 in S_1] $+$
Var[local count of 5 in S_2] $+$
$1/p_2^2$

Tree Model: Merging Samples

Problem: Variances accumulate

(*Law of total variance*)

$|S_1| \quad p_1 \qquad (3,2)\ (7,5)\ (13,8)\ (26,10)$

$+$

$|S_2| \quad p_2 \qquad (5,3)\ (14,6)\ (18,8)\ (24,11)$

$=$

$(13, 8+3+1/p_2)$

Var[local count of 13 in $S_1 \cup S_2$] $=$
Var[local count of 13 in S_1] $+$
Var[local count of 5 in S_2] $+$
$1/p_2^2$

Var[merged sample] $=$
Var[sample of S_1] $+$
Var[sample of S_2] $+$
$\max(1/p_1^2, 1/p_2^2)$

Tree Model: Merging Samples

Problem: Variances accumulate

(*Law of total variance*)

$$|S_1| \quad p_1 \qquad (3,2)\ (7,5)\ (13,8)\ (26,10)$$

$$+$$

$$|S_2| \quad p_2 \qquad (5,3)\ (14,6)\ (18,8)\ (24,11)$$

$$=$$

$$(13, 8+3+1/p_2)$$

Var[local count of 13 in $S_1 \cup S_2$] $=$
Var[local count of 13 in S_1] $+$
Var[local count of 5 in S_2] $+$
$1/p_2^2$

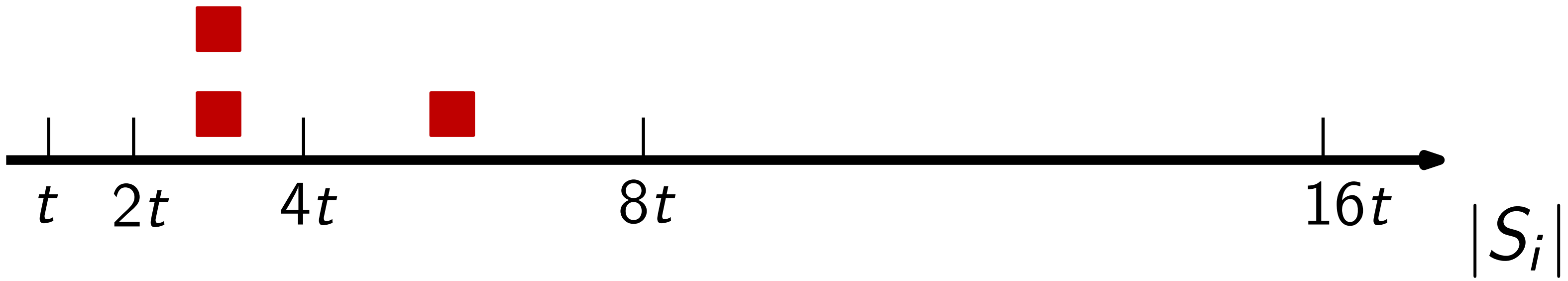
Var[merged sample] $=$
Var[sample of S_1] $+$
Var[sample of S_2] $+$
$\max(1/p_1^2, 1/p_2^2)$

Get "penalized" if we merge two samples of uneven sizes.

Tree Model: Merging Samples

Idea: Only merge samples with $p_1 \approx p_2$, i.e., $|S_1| \approx |S_2|$

$t = \frac{n}{\sqrt{k}}$

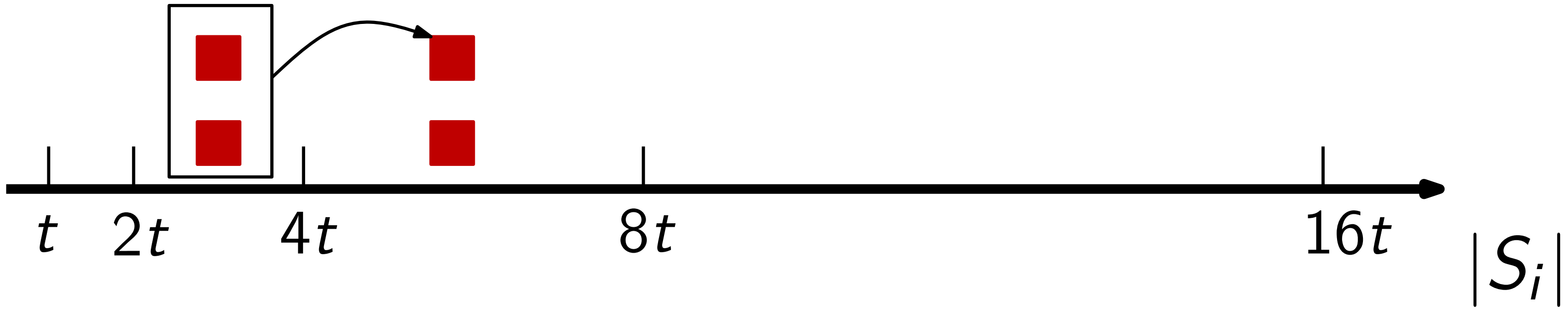


Merge samples in the same bucket

Tree Model: Merging Samples

Idea: Only merge samples with $p_1 \approx p_2$, i.e., $|S_1| \approx |S_2|$

$t = \frac{n}{\sqrt{k}}$

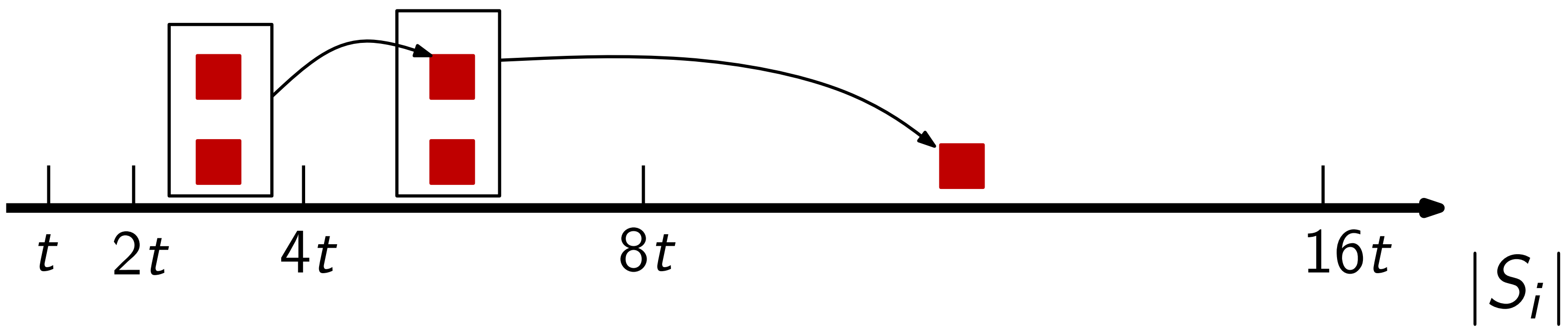


Merge samples in the same bucket

Tree Model: Merging Samples

Idea: Only merge samples with $p_1 \approx p_2$, i.e., $|S_1| \approx |S_2|$

$t = \frac{n}{\sqrt{k}}$



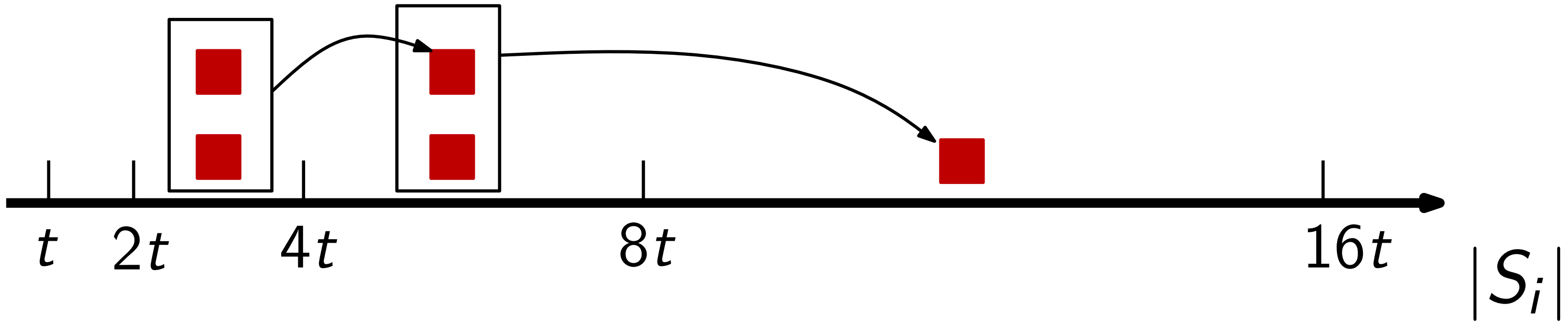
Merge samples in the same bucket

After merging, there are at most $\log \sqrt{k}$ samples
(so, the max individual cost is $O(\frac{1}{\varepsilon} \log k)$)

Tree Model: Merging Samples

Idea: Only merge samples with $p_1 \approx p_2$, i.e., $|S_1| \approx |S_2|$

$t = \frac{n}{\sqrt{k}}$

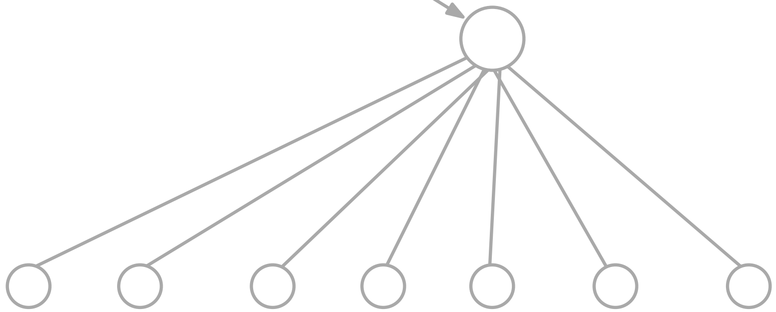


Merge samples in the same bucket

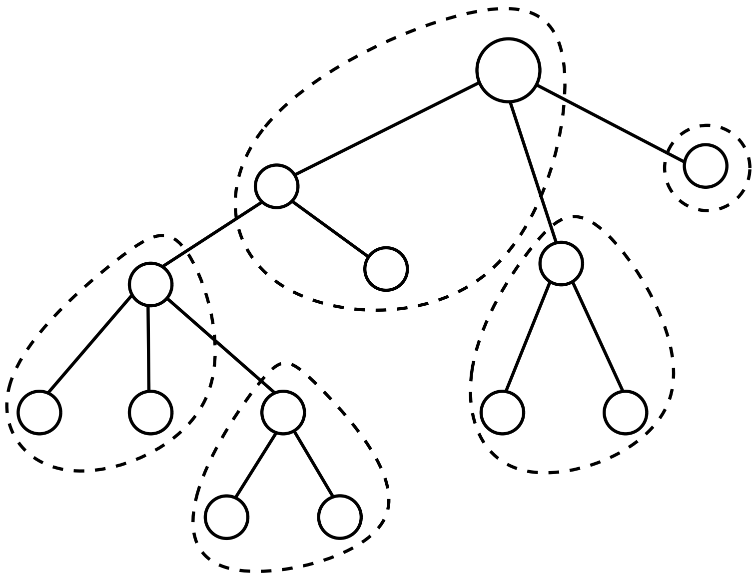
After merging, there are at most $\log \sqrt{k}$ samples
(so, the max individual cost is $O(\frac{1}{\varepsilon} \log k)$)

Can show that the final variance is $O((\varepsilon n)^2)$ (please see paper)

Roadmap



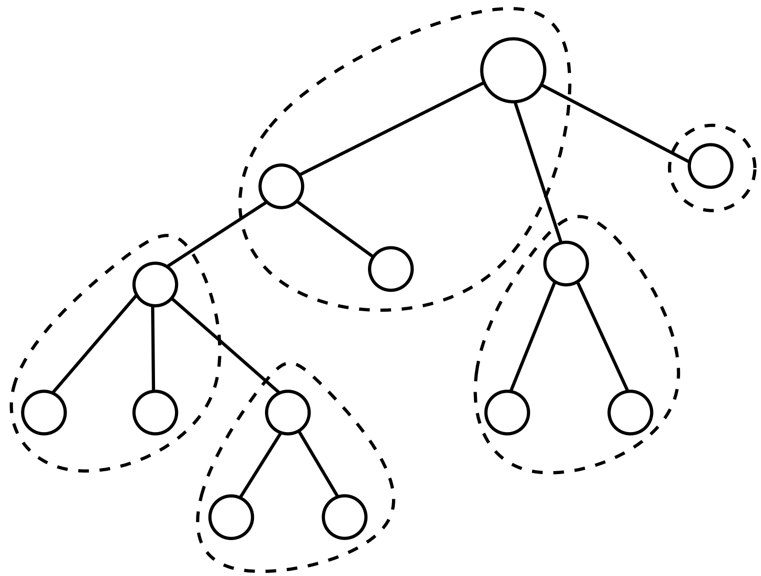
- Flat model
- Tree model



- Partitioned tree

Tree Partitioning

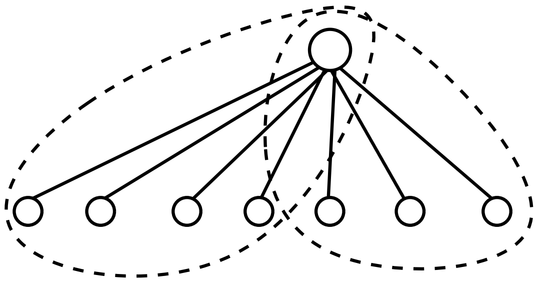
Partition into t connected components with $O(k/t)$ nodes each

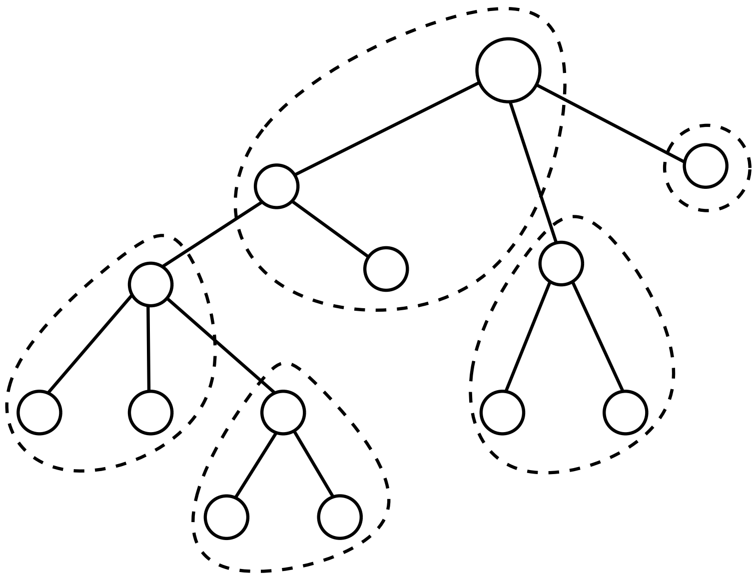


Tree Partitioning

Partition into t connected components with $O(k/t)$ nodes each

May need to share a few ($\leq t$) nodes

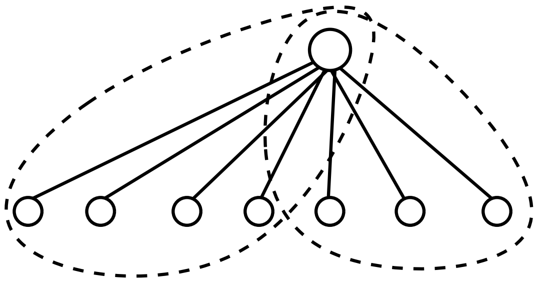


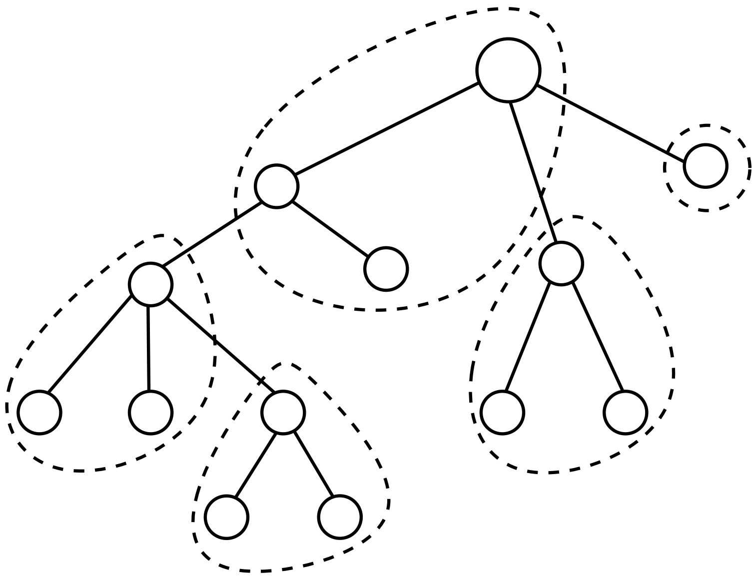


Tree Partitioning

Partition into t connected components with $O(k/t)$ nodes each

May need to share a few ($\leq t$) nodes

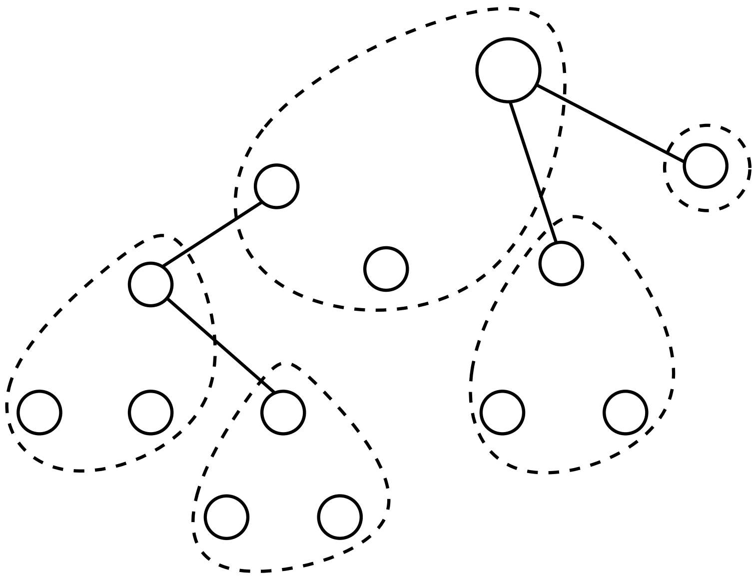




Can be done in linear time and communication.
Algorithm not too difficult
— a nice homework question for an algorithms course?

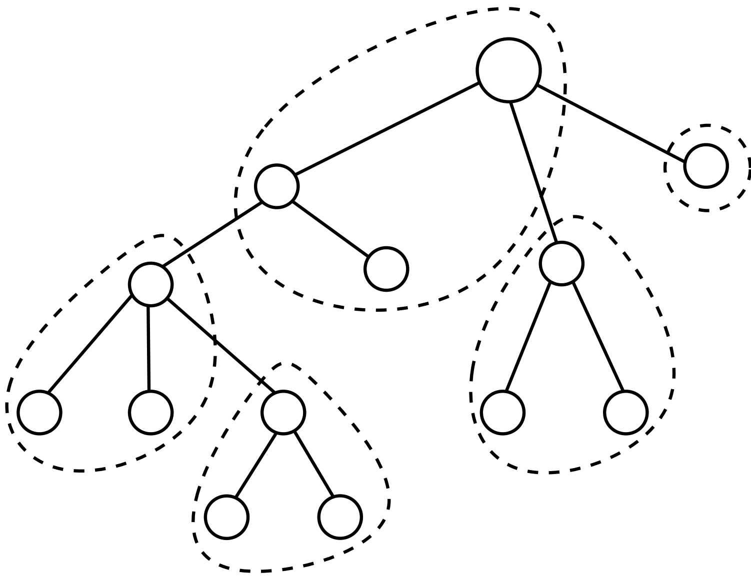
Quantile Computation on a Partitioned Tree

Consider the t components as t “super nodes”
Total sampe size is $O(\sqrt{t}/\varepsilon)$, communication cost $O(\sqrt{t}/\varepsilon \cdot h)$



Quantile Computation on a Partitioned Tree

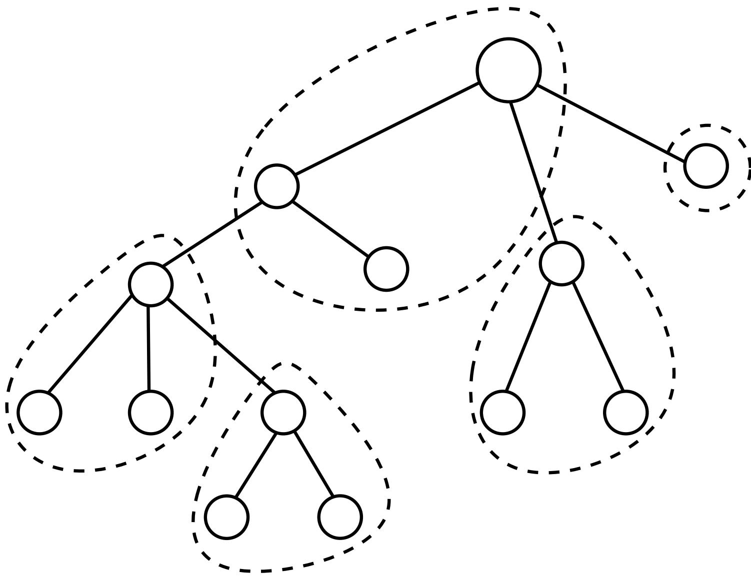
Consider the t components as t "super nodes"
Total sampe size is $O(\sqrt{t}/\varepsilon)$, communication cost $O(\sqrt{t}/\varepsilon \cdot h)$



Computing the "local ranks" of the sampled values, need to broadcast them to all nodes in the super node, costing $O(\sqrt{t}/\varepsilon \cdot k/t)$

Quantile Computation on a Partitioned Tree

Consider the t components as t "super nodes"
Total sampe size is $O(\sqrt{t}/\varepsilon)$, communication cost $O(\sqrt{t}/\varepsilon \cdot h)$

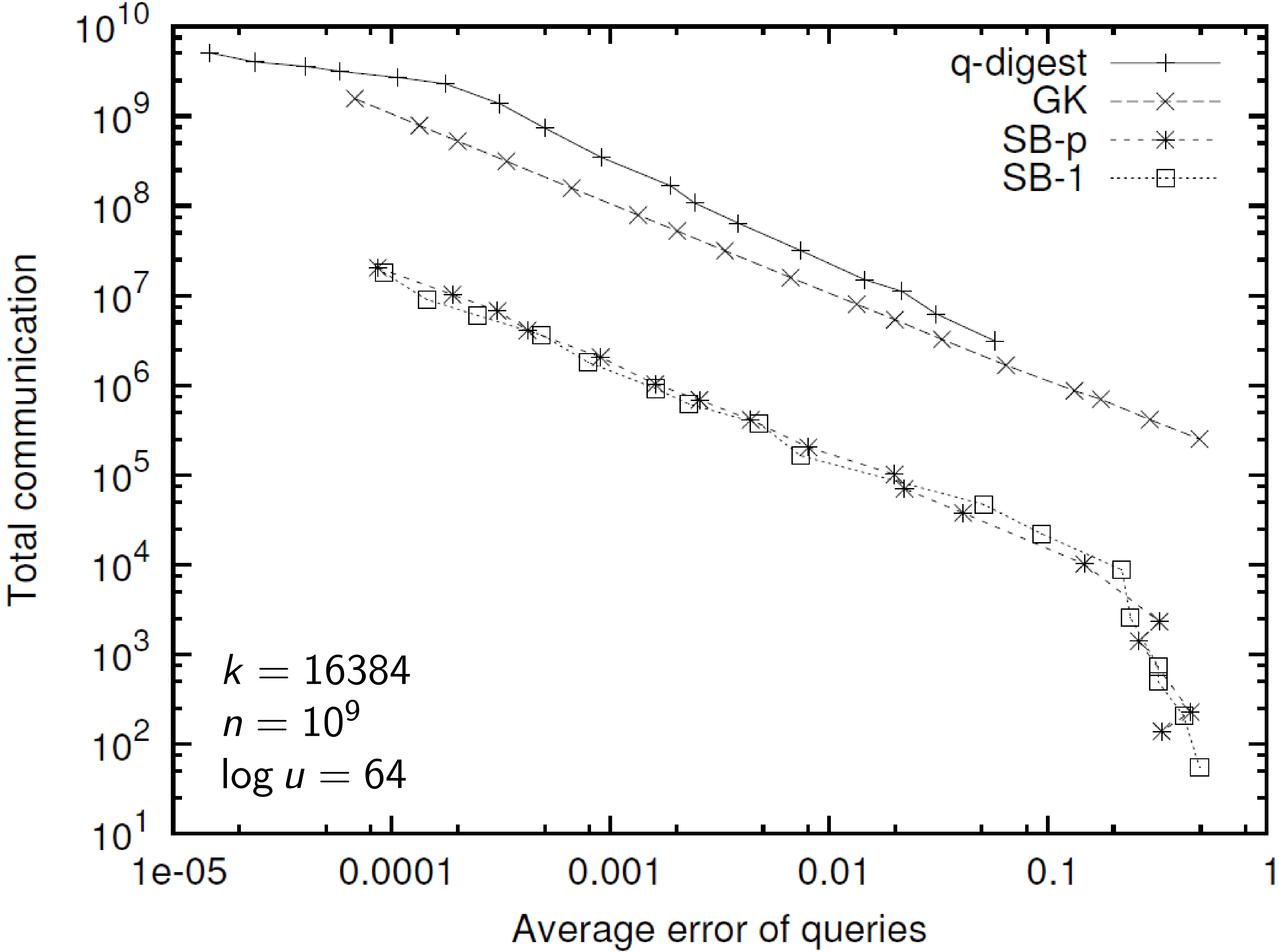


Need $h = k/t$ to balance

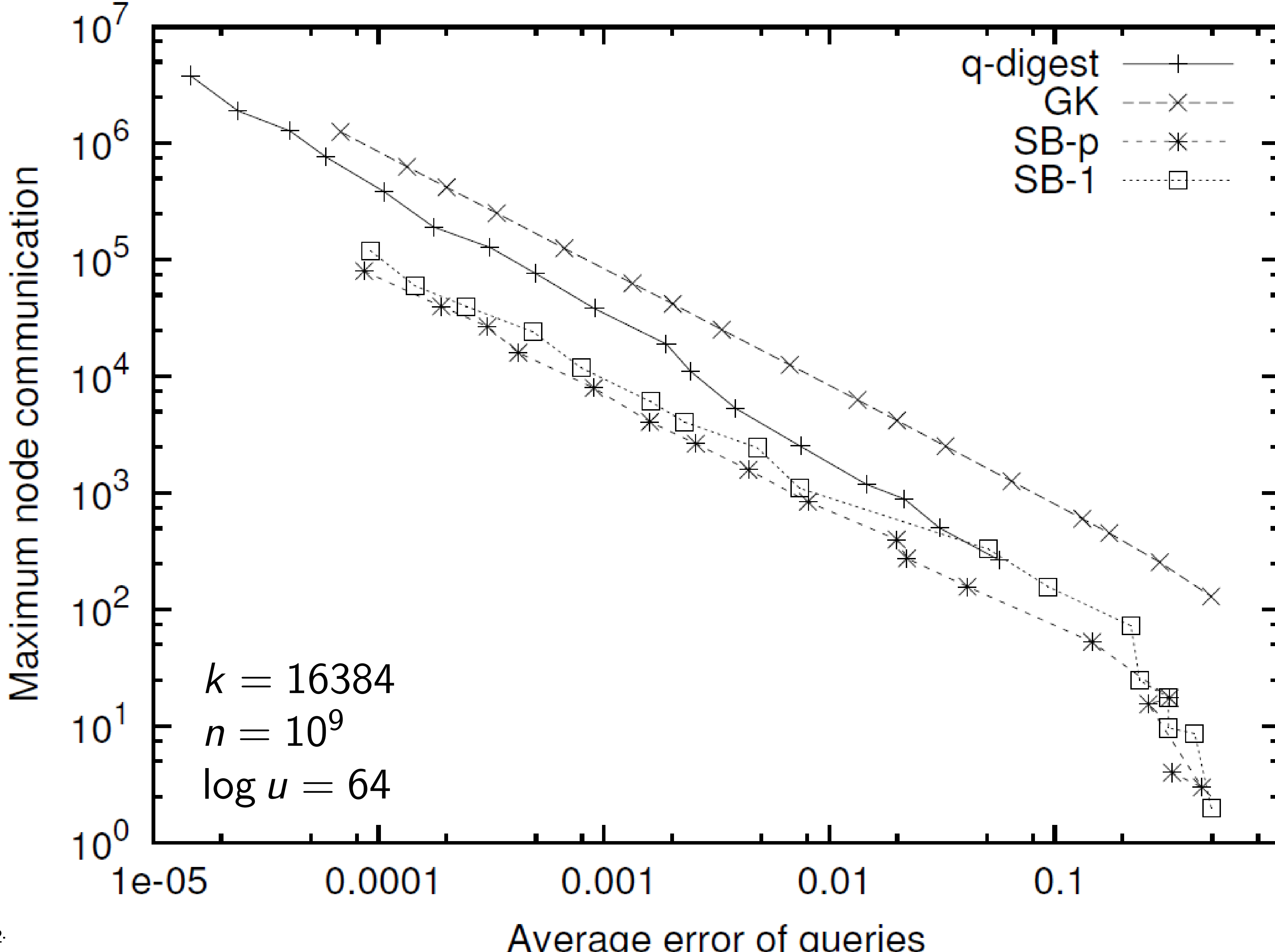
When $t = k/h$, both are $O(\sqrt{kh}/\varepsilon)$

Computing the "local ranks" of the sampled values, need to broadcast them to all nodes in the super node, costing $O(\sqrt{t}/\varepsilon \cdot k/t)$

Experimental Results on Terrain Data



Experimental Results on Terrain Data



Conclusion

- A sampling based algorithm whose total communication cost grows sublinearly as network size
- Deviate from the traditional "decomposable" framework
- Can we solve other data aggregation problems using similar ideas?