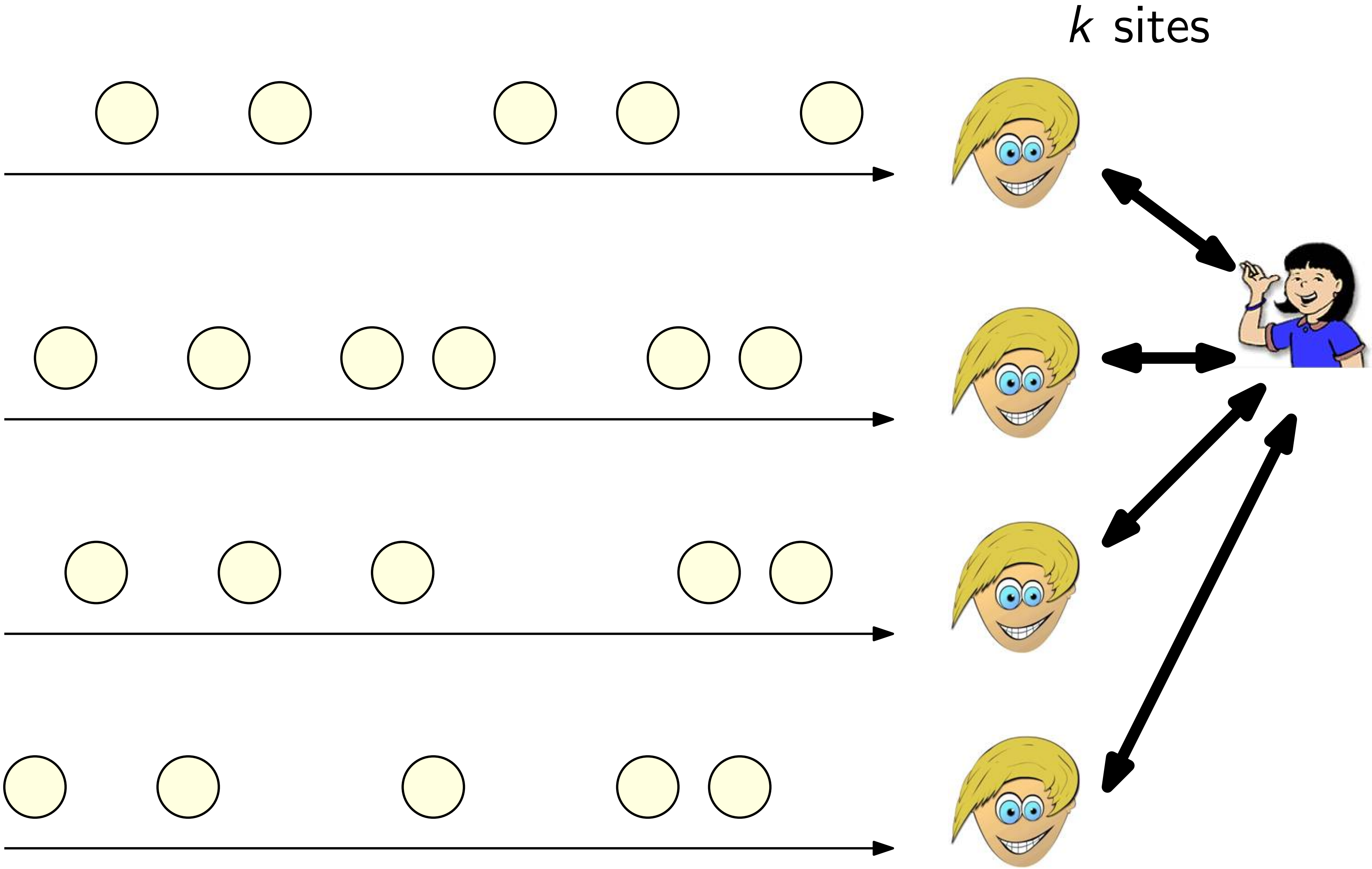
Tracking Distributed Data

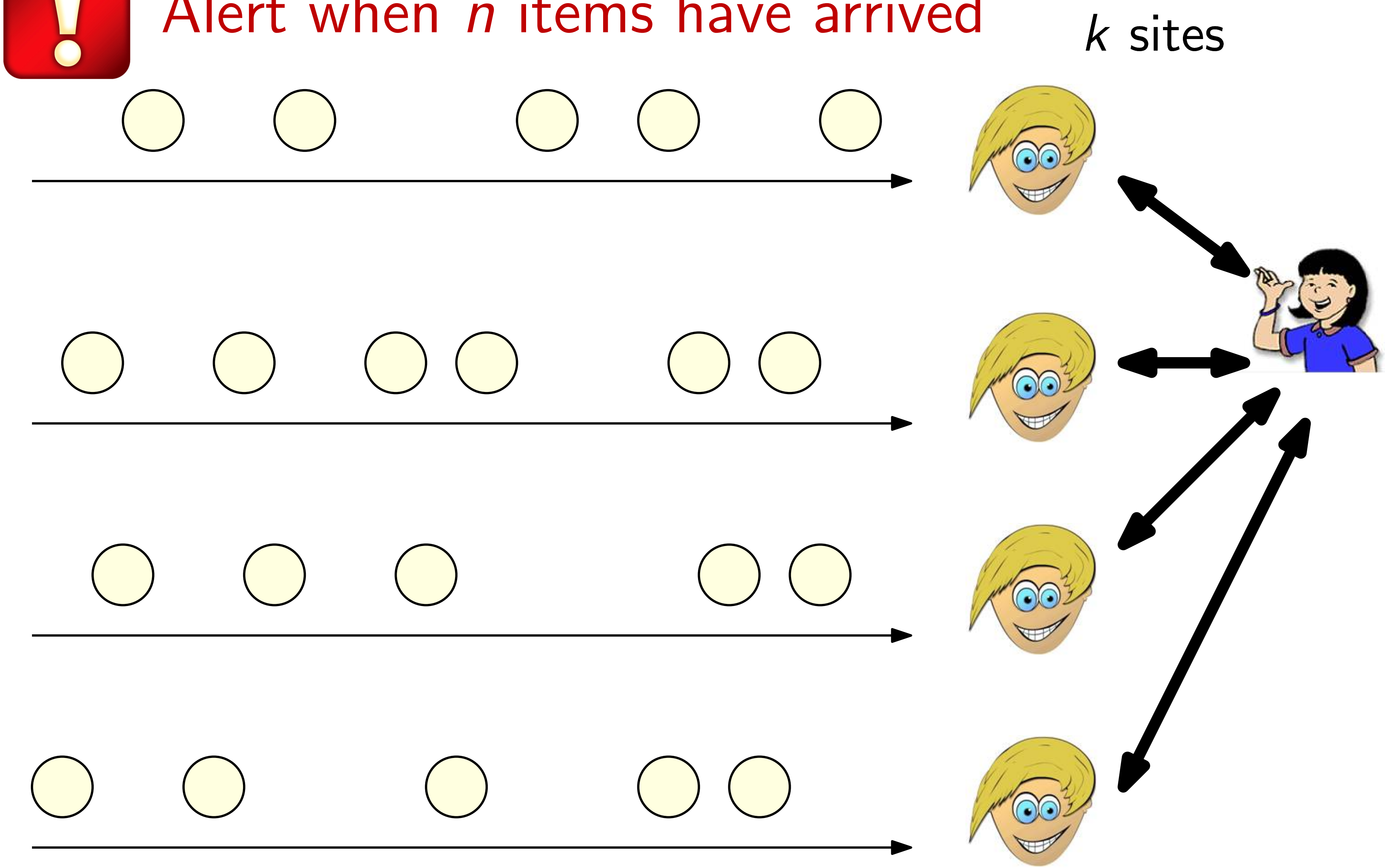
Ke Yi
HKUST

The Distributed Count-Down Problem



The Distributed Count-Down Problem

Alert when n items have arrived



The Count-Down Problem

Naive solution: $O(n)$ communication

[Cormode, Muthukrishnan, Yi, SODA'08]

The Count-Down Problem

Naive solution: $O(n)$ communication

"Safe zone" based approach:

- Set threshold $= n/k$, safe when every local count $< n/k$

[Cormode, Muthukrishnan, Yi, SODA'08]

The Count-Down Problem

Naive solution: $O(n)$ communication

"Safe zone" based approach:

- Set threshold $= n/k$, safe when every local count $< n/k$
- When one local count reaches n/k, broadcast to
  - Compute the current total count
  - Compute new leeway $= n -$ total count
  - Set new threshold $=$ leeway $/\ k$

[Cormode, Muthukrishnan, Yi, SODA'08]

The Count-Down Problem

Naive solution: $O(n)$ communication

"Safe zone" based approach:

- Set threshold $= n/k$, safe when every local count $< n/k$
- When one local count reaches n/k, broadcast to
  - Compute the current total count
  - Comput[…]ount
  - Set new[…]

Analysis

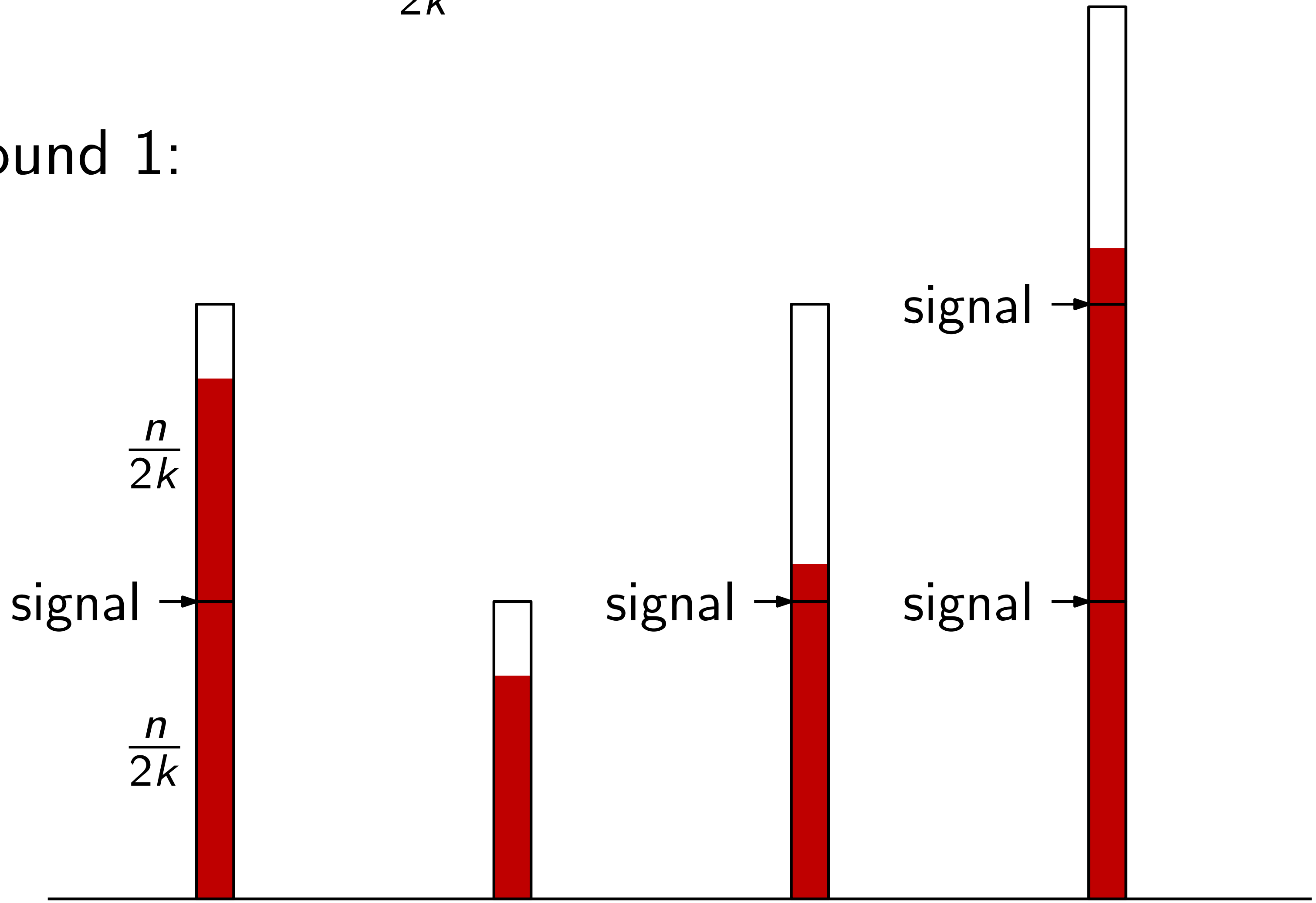
\# rounds: $O(k \log n)$
cost per round: $O(k)$
total cost: $O(k^2 \log n)$

[Cormode, Muthukrishnan, Yi, SODA'08]

The Count-Down Problem

Set threshold $= \frac{n}{2k}$

Round 1:

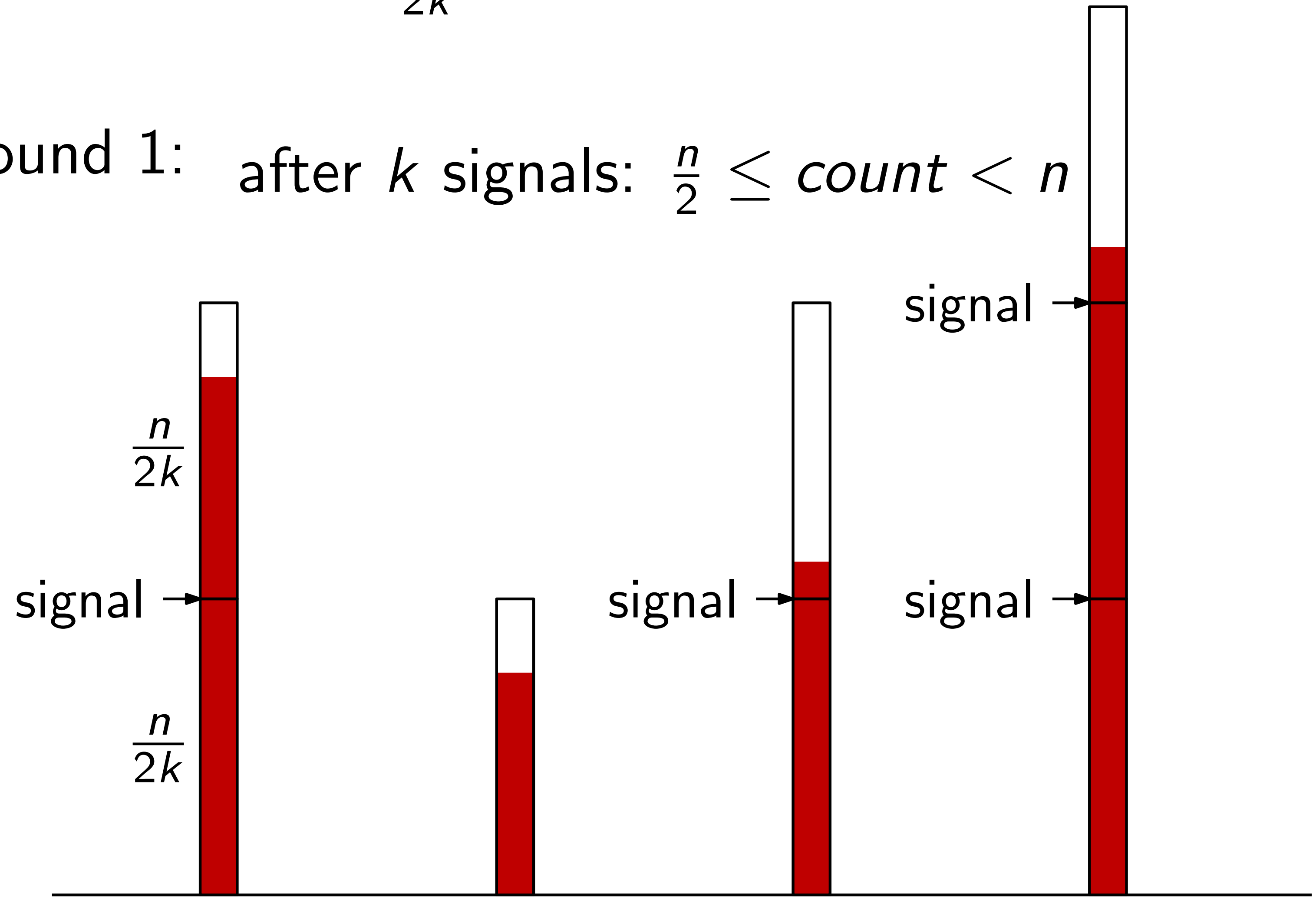


[Cormode, Muthukrishnan, Yi, SODA'08]

The Count-Down Problem

Set threshold $= \frac{n}{2k}$

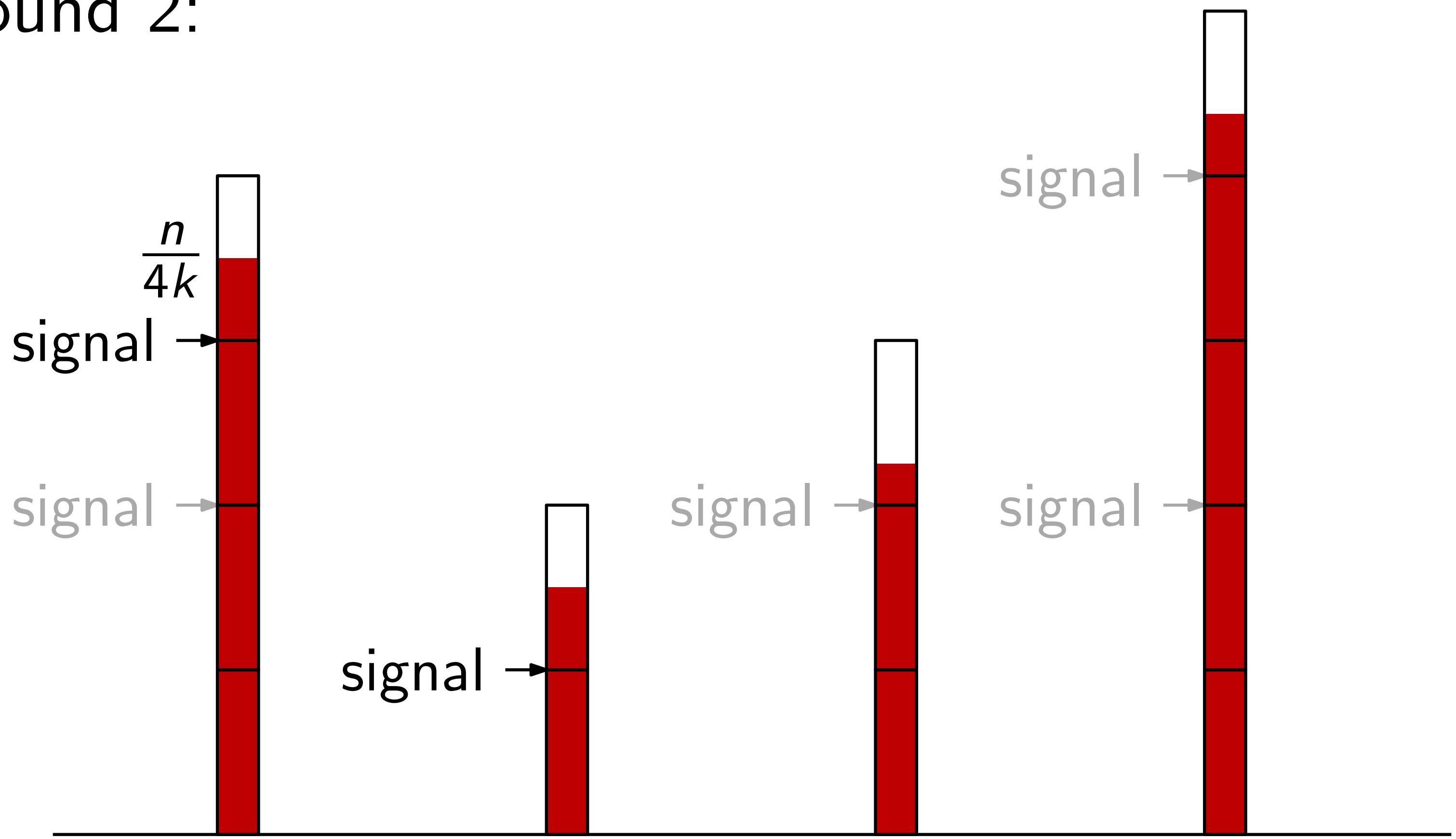
Round 1: after k signals: $\frac{n}{2} \leq count < n$



[Cormode, Muthukrishnan, Yi, SODA'08]

The Count-Down Problem

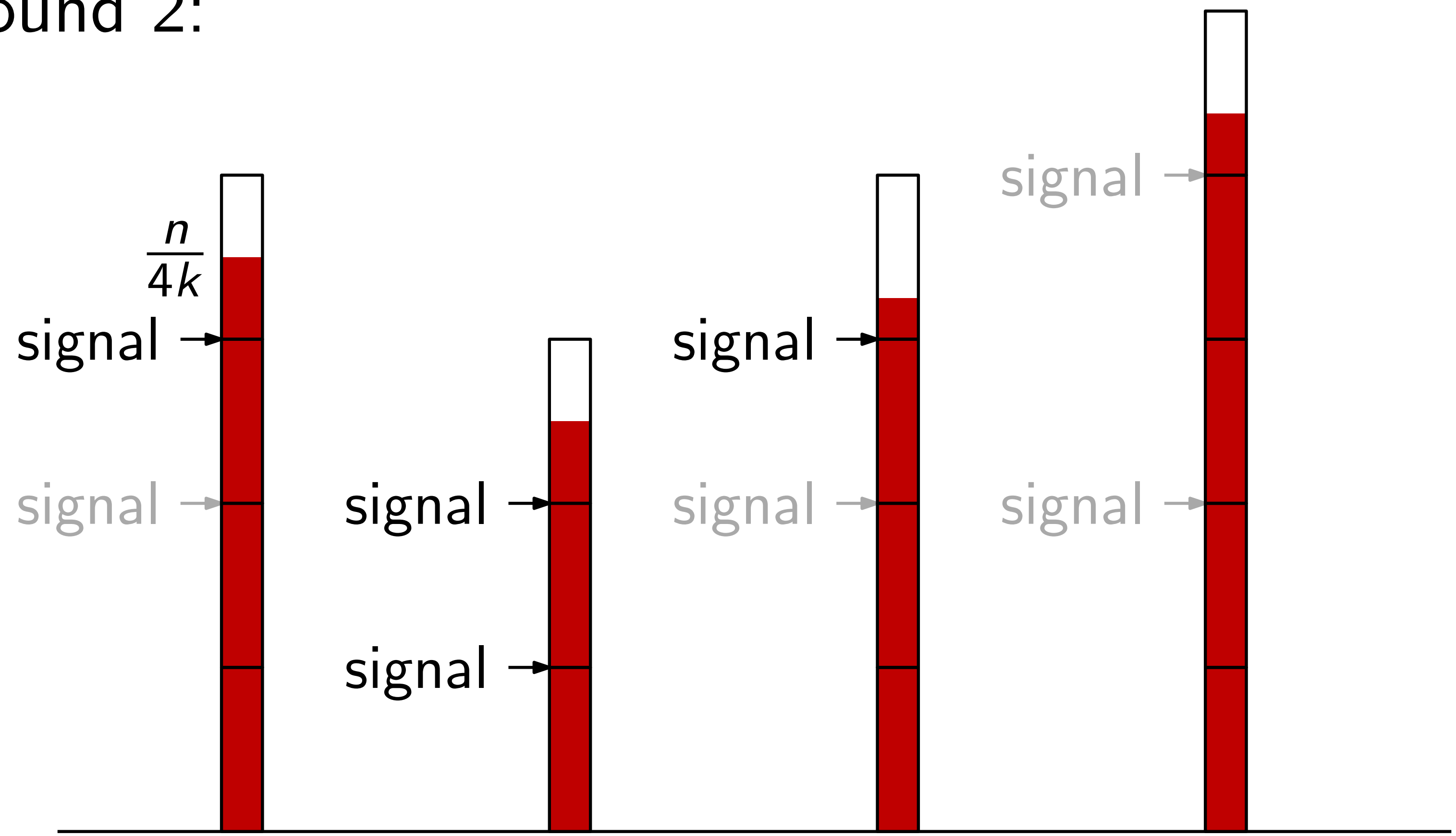
Round 2:



[Cormode, Muthukrishnan, Yi, SODA'08]

The Count-Down Problem

Round 2:

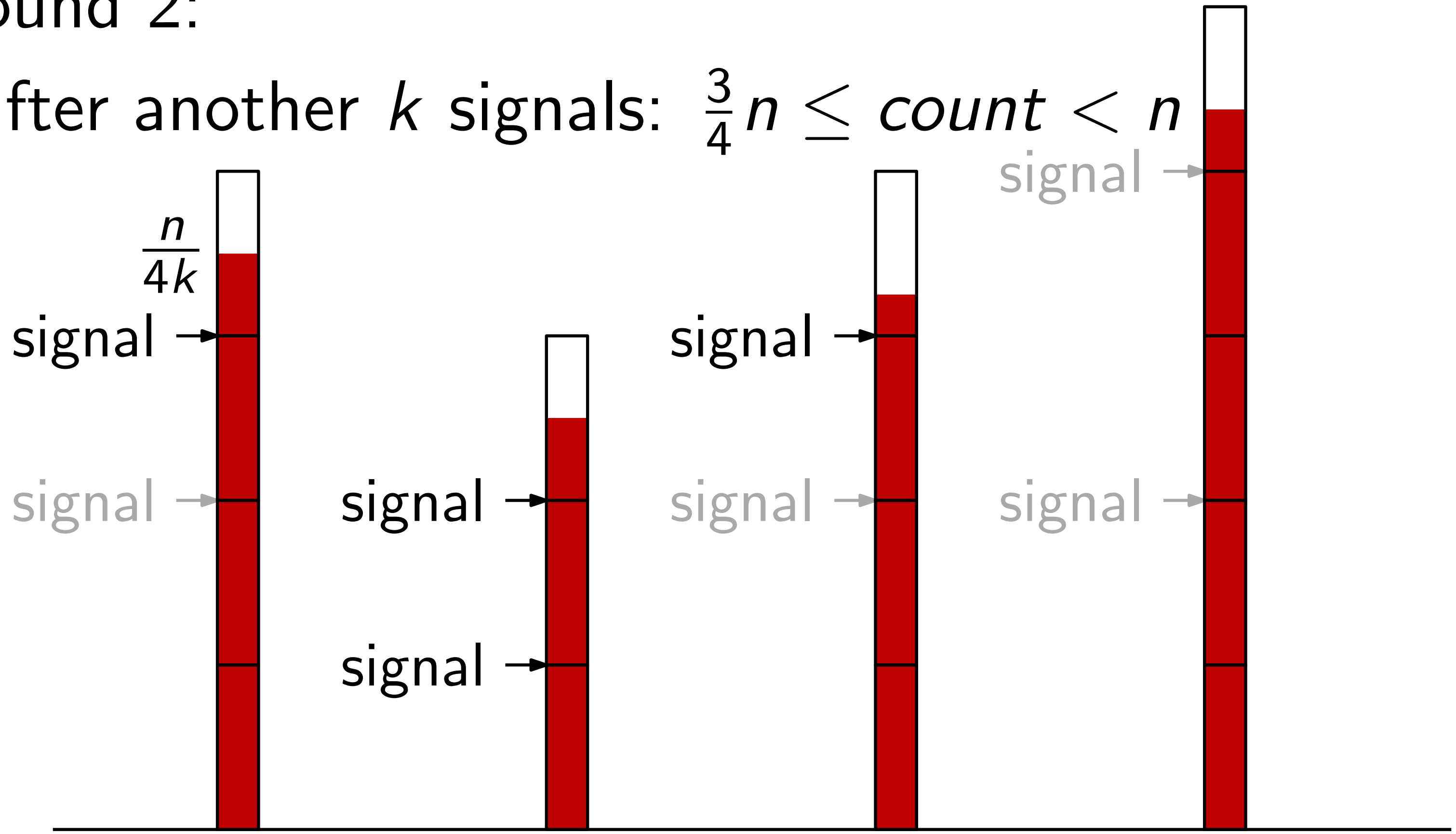


[Cormode, Muthukrishnan, Yi, SODA'08]

The Count-Down Problem

Round 2:

after another k signals: $\frac{3}{4}n \leq count < n$

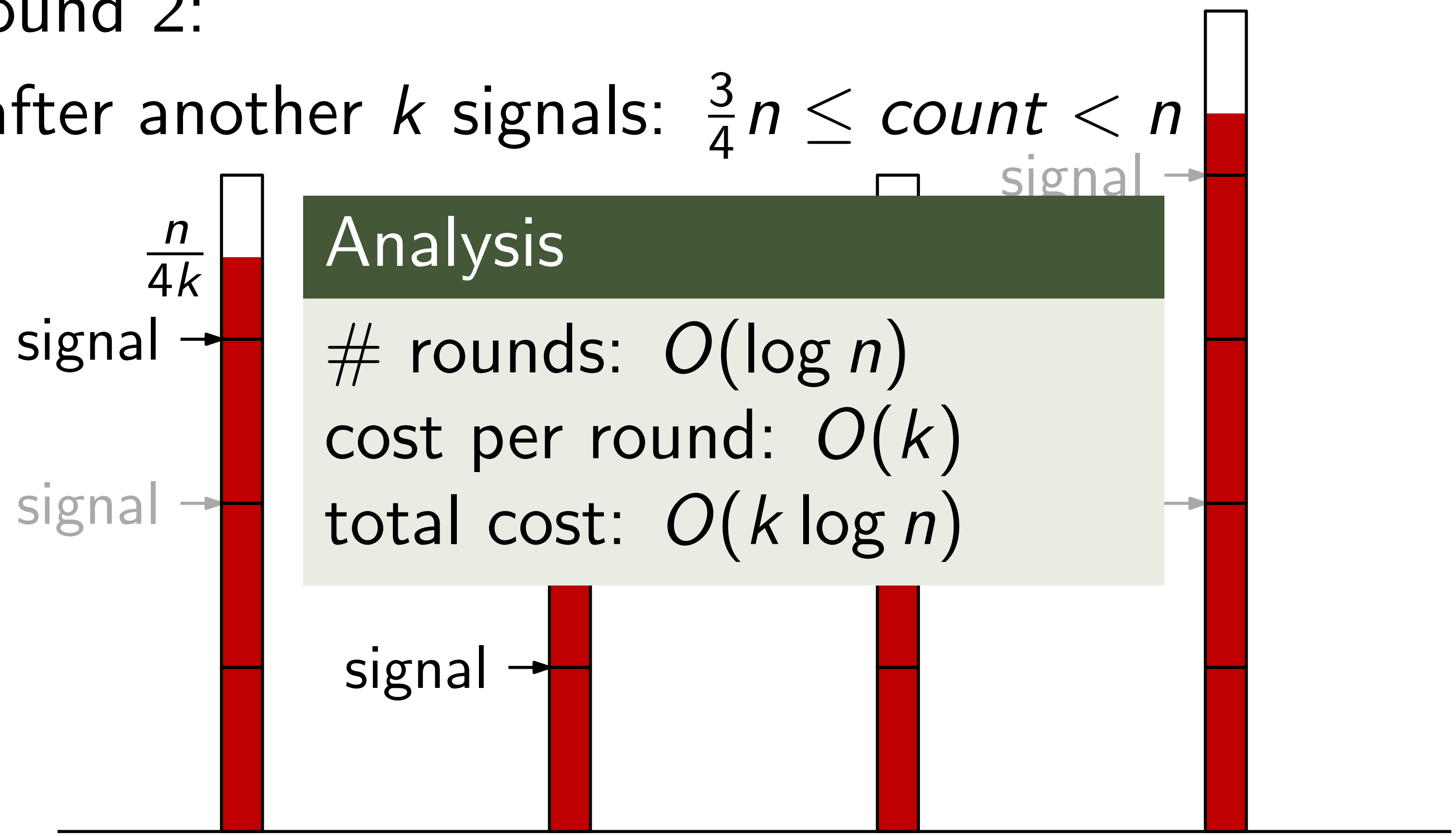


[Cormode, Muthukrishnan, Yi, SODA'08]

The Count-Down Problem

Round 2:

after another k signals: $\frac{3}{4}n \leq count < n$



Analysis

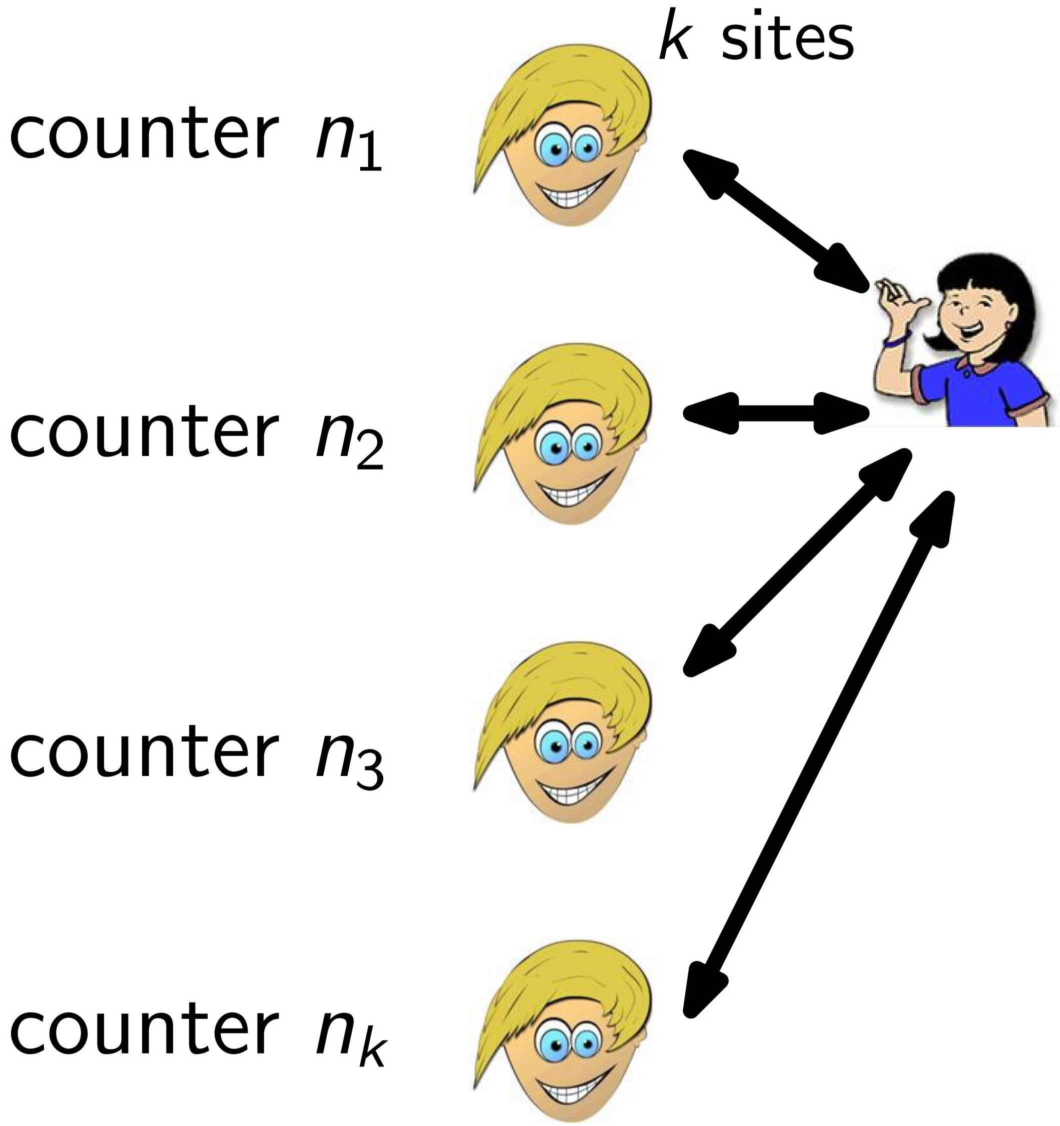
\# rounds: $O(\log n)$

cost per round: $O(k)$

total cost: $O(k \log n)$

[Cormode, Muthukrishnan, Yi, SODA'08]

The Count-Tracking Problem



Counters increment over time

The Count-Tracking Problem

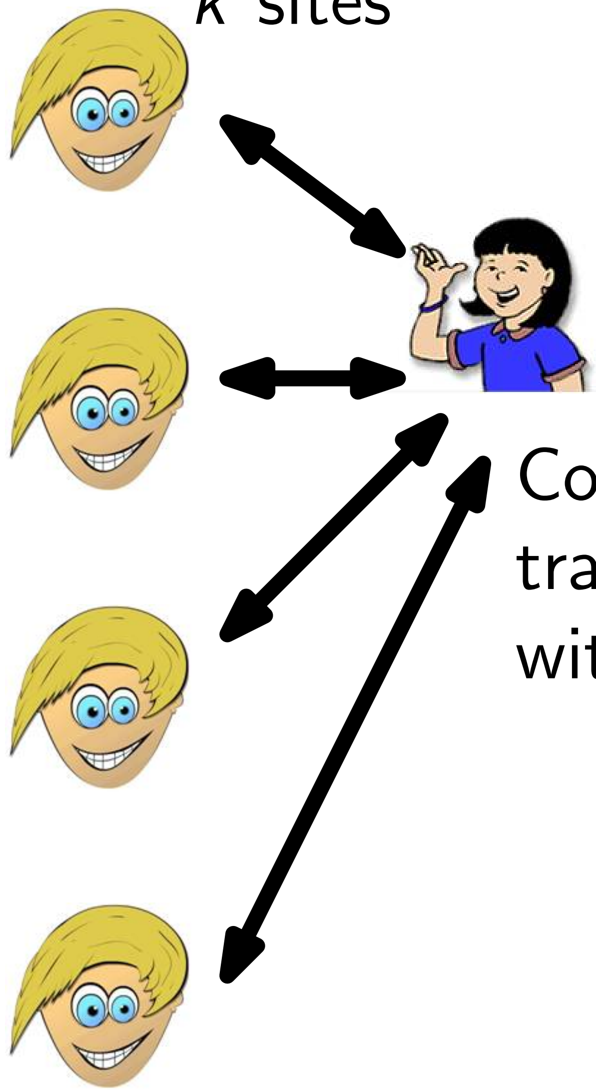
k sites

counter n_1

counter n_2

counter n_3

counter n_k



Coordinator wants to track $n = \sum n_i$ with relative ε-error

Counters increment over time

Deterministic Algorithm

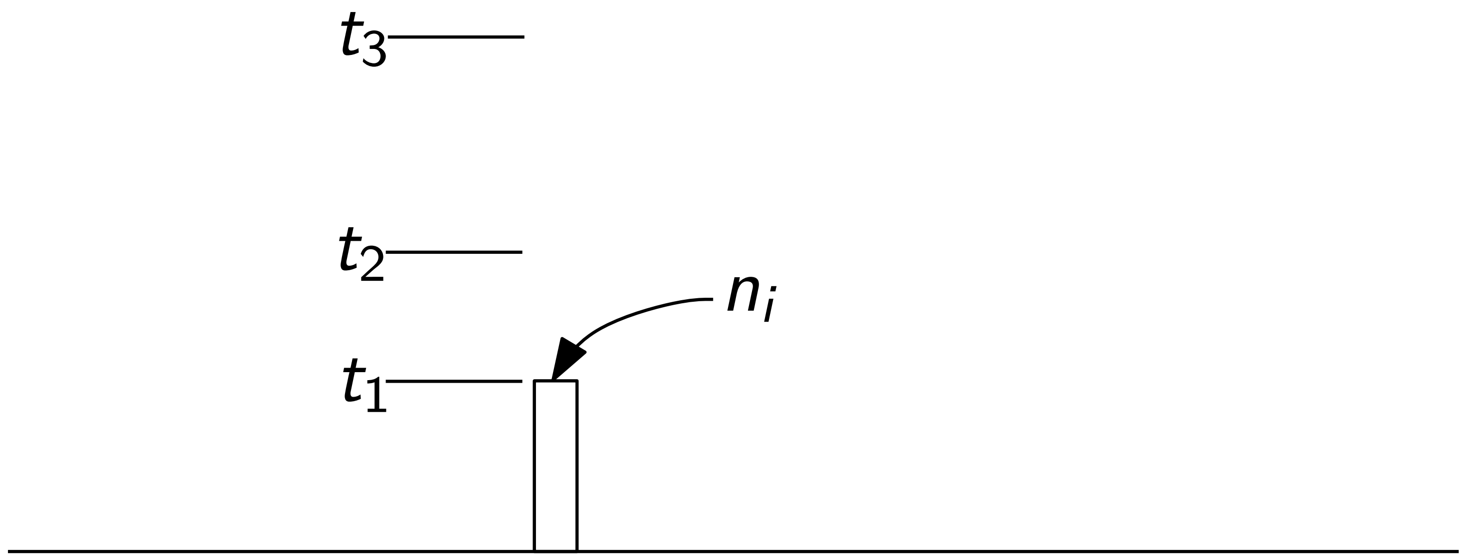
Every site uses a series of thresholds:
$t_0 = 1, t_1 = 1 + \varepsilon, t_2 = (1 + \varepsilon)^2, \ldots$

Sends a message when n_i reaches a threshold

Deterministic Algorithm

Every site uses a series of thresholds:
$t_0 = 1, t_1 = 1 + \varepsilon, t_2 = (1 + \varepsilon)^2, \ldots$

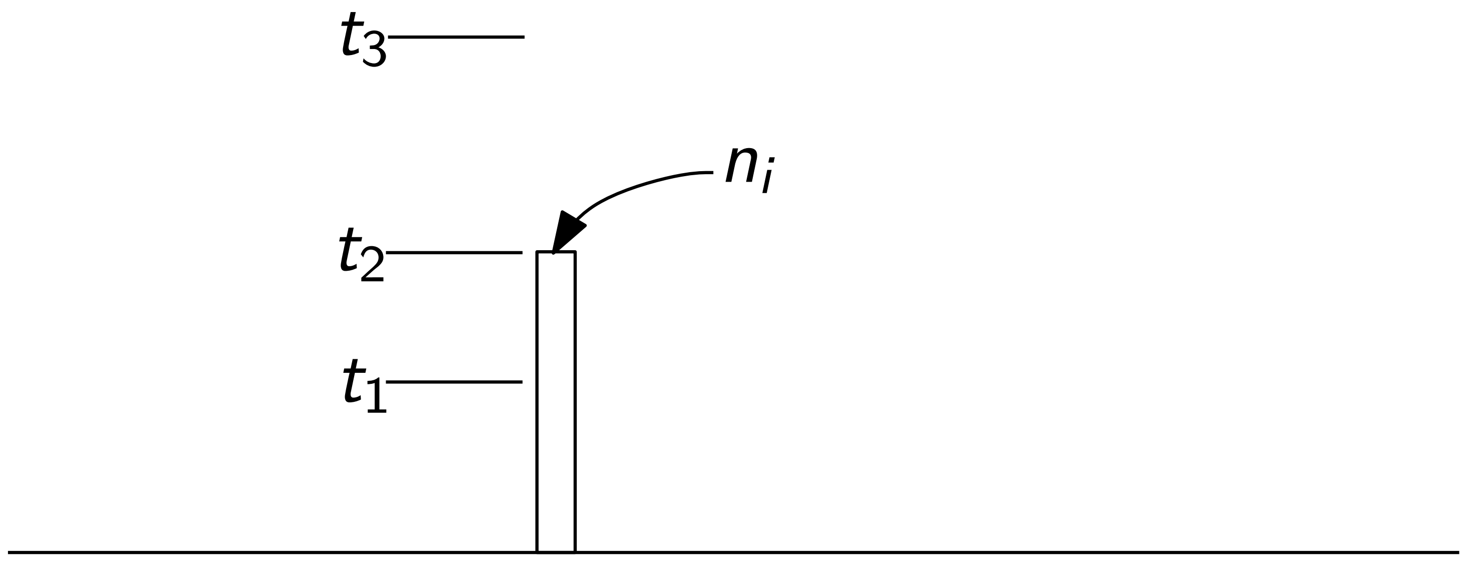
Sends a message when n_i reaches a threshold



Deterministic Algorithm

Every site uses a series of thresholds:
$t_0 = 1, t_1 = 1 + \varepsilon, t_2 = (1 + \varepsilon)^2, \ldots$

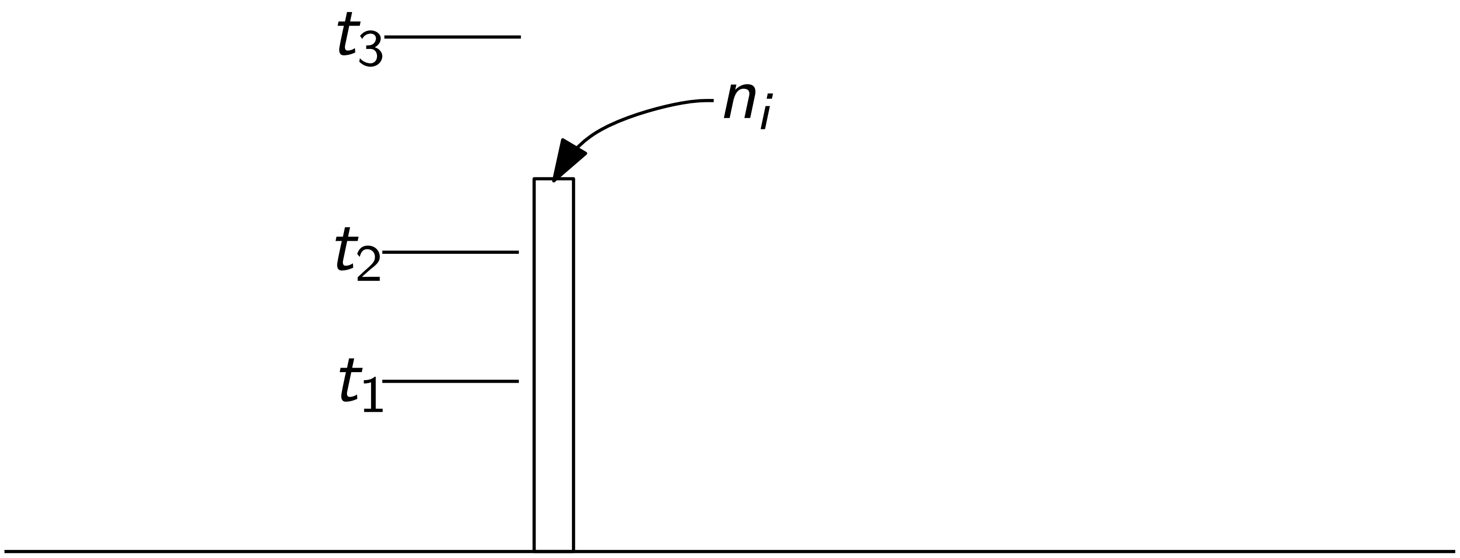
Sends a message when n_i reaches a threshold



Deterministic Algorithm

Every site uses a series of thresholds:
$t_0 = 1, t_1 = 1 + \varepsilon, t_2 = (1 + \varepsilon)^2, \ldots$

Sends a message when n_i reaches a threshold

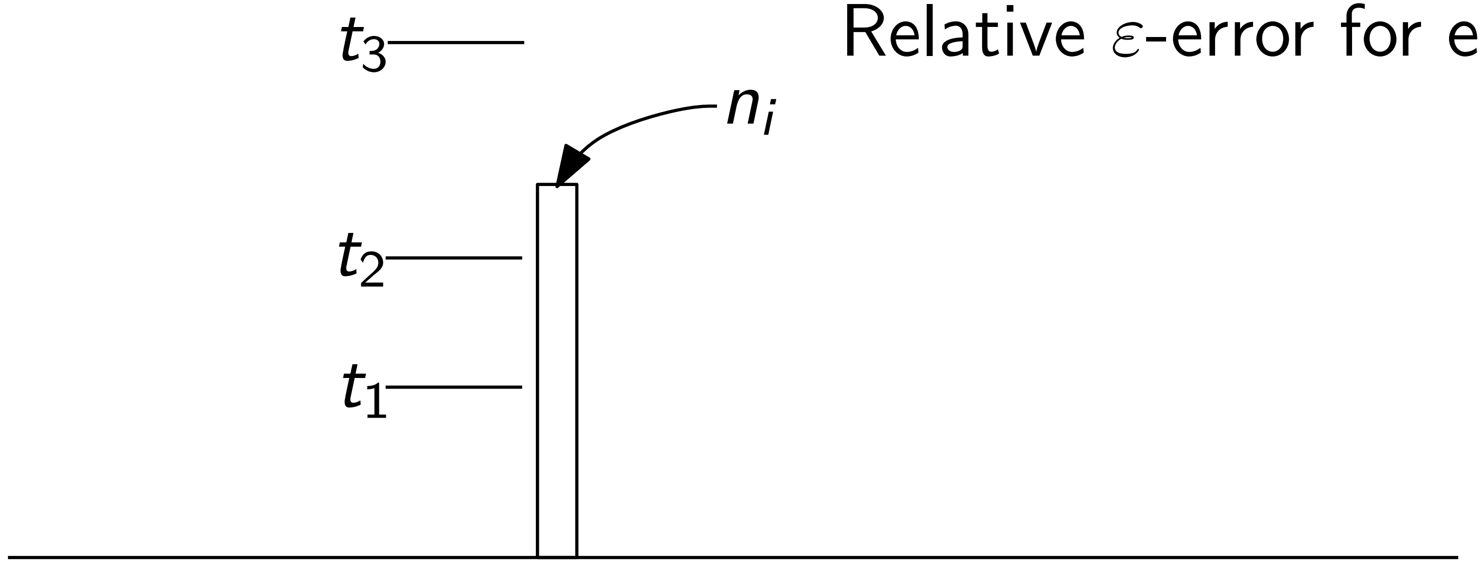


Deterministic Algorithm

Every site uses a series of thresholds:
$t_0 = 1, t_1 = 1 + \varepsilon, t_2 = (1 + \varepsilon)^2, \ldots$

Sends a message when n_i reaches a threshold

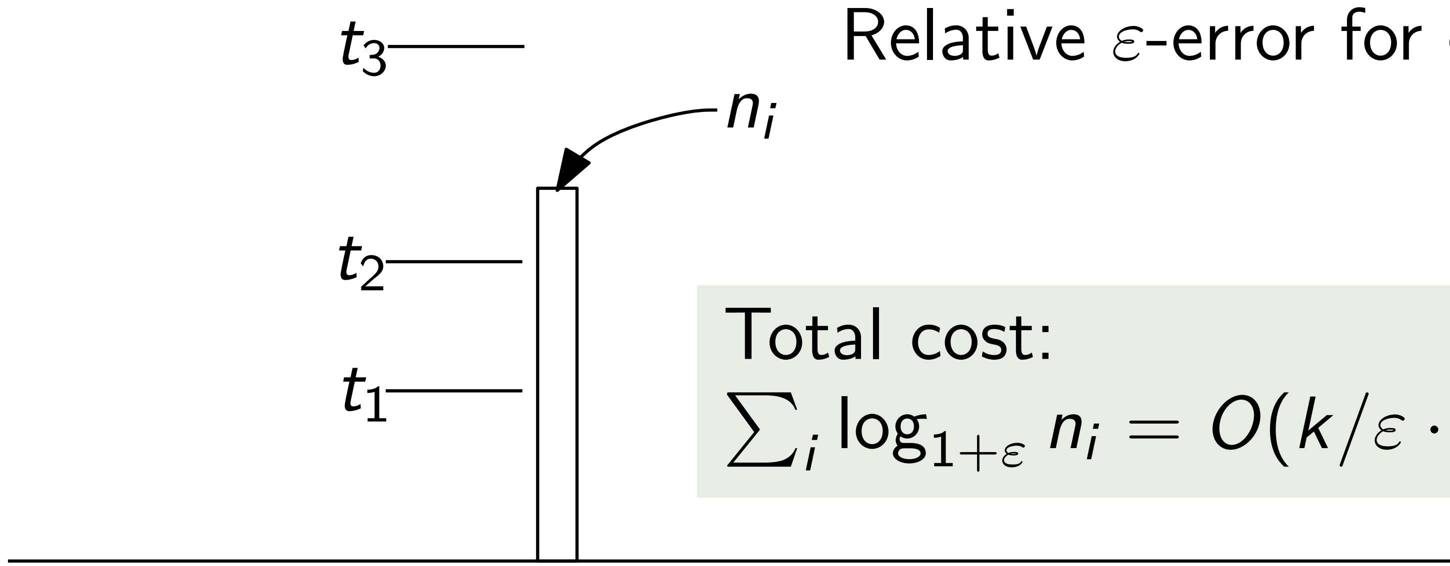
Relative ε-error for each n_i



Deterministic Algorithm

Every site uses a series of thresholds:
$t_0 = 1, t_1 = 1 + \varepsilon, t_2 = (1+\varepsilon)^2, \ldots$

Sends a message when n_i reaches a threshold



Relative ε-error for each n_i

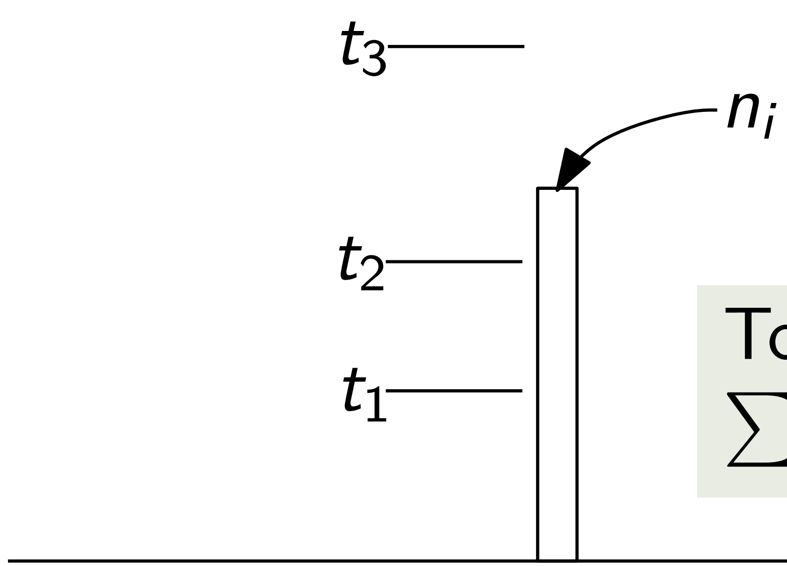
Total cost:
$\sum_i \log_{1+\varepsilon} n_i = O(k/\varepsilon \cdot \log n)$

Deterministic Algorithm

Every site uses a series of thresholds:
$t_0 = 1, t_1 = 1 + \varepsilon, t_2 = (1 + \varepsilon)^2, \ldots$

Sends a message when n_i reaches a threshold

Relative ε-error for each n_i



Total cost:
$\sum_i \log_{1+\varepsilon} n_i = O(k/\varepsilon \cdot \log n)$

Communication is one-way

Deterministic Lower Bound

Theorem

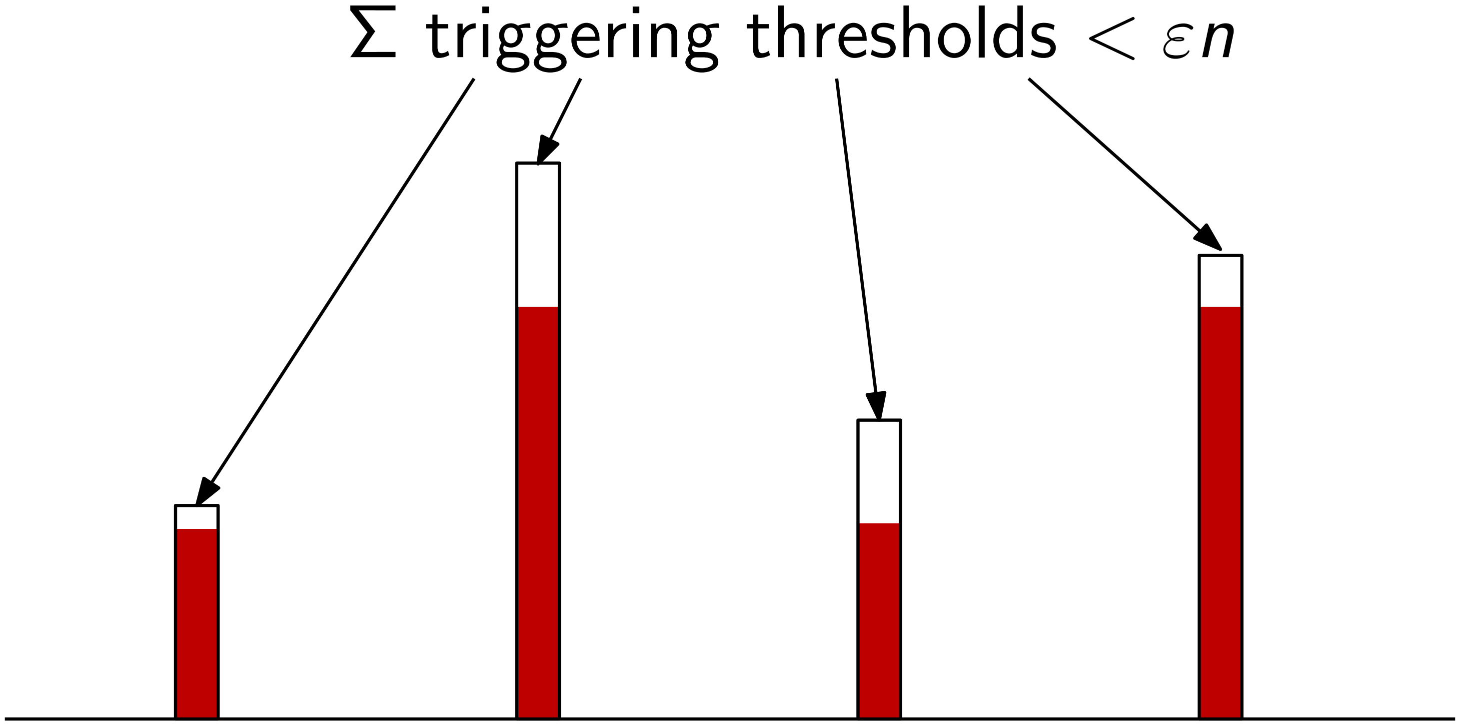
Any deterministic protocol that solves the count-tracking problem must communicate $\Omega(k/\varepsilon \cdot \log n)$ messages, even with two-way commucation.

[Yi, Zhang, PODS'09]

Deterministic Lower Bound

Theorem

Any deterministic protocol that solves the count-tracking problem must communicate $\Omega(k/\varepsilon \cdot \log n)$ messages, even with two-way commucation.

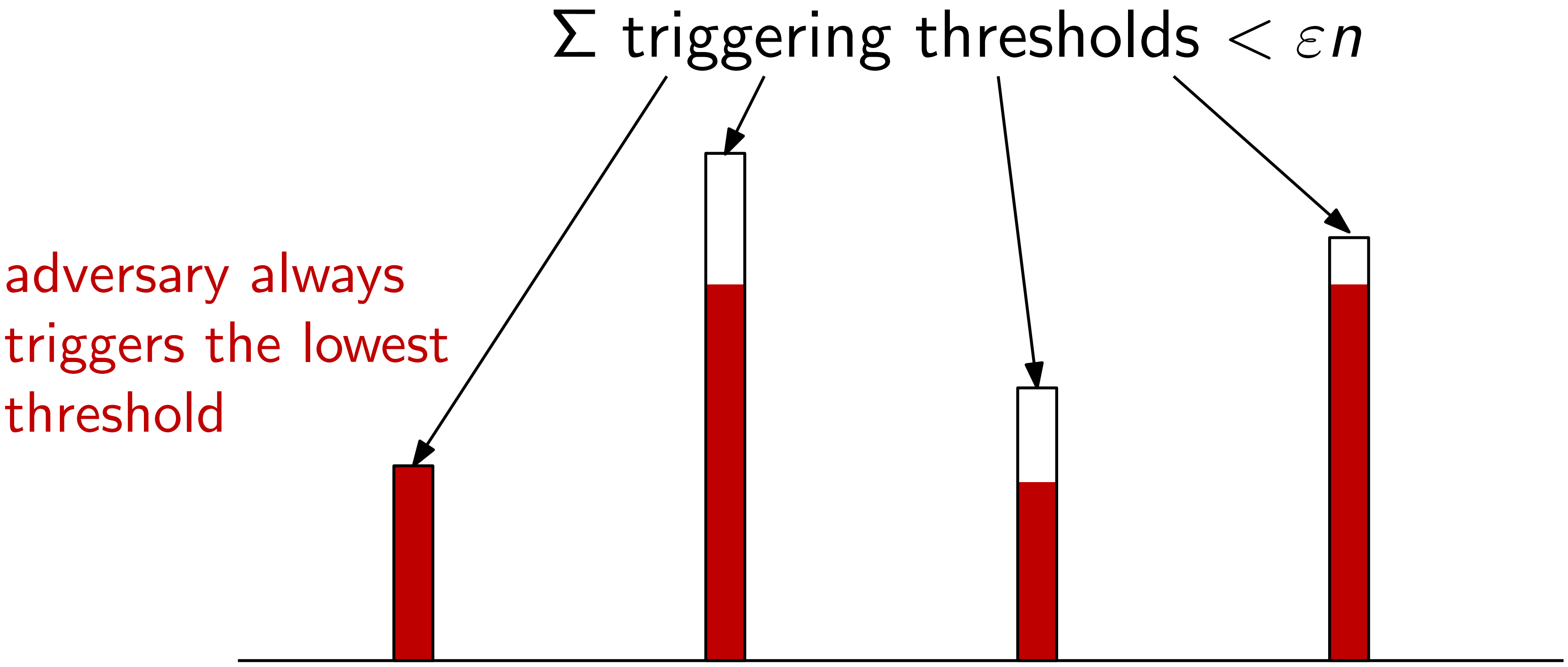


[Yi, Zhang, PODS'09]

Deterministic Lower Bound

Theorem

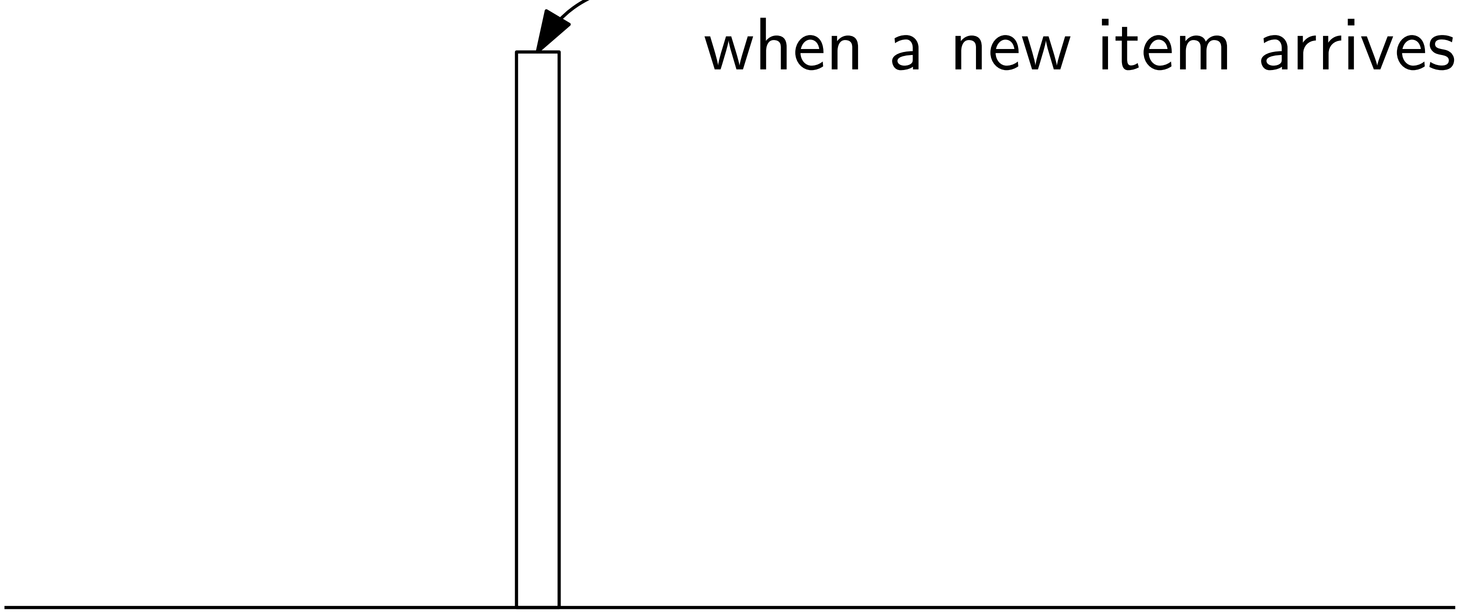
Any deterministic protocol that solves the count-tracking problem must communicate $\Omega(k/\varepsilon \cdot \log n)$ messages, even with two-way commucation.



[Yi, Zhang, PODS'09]

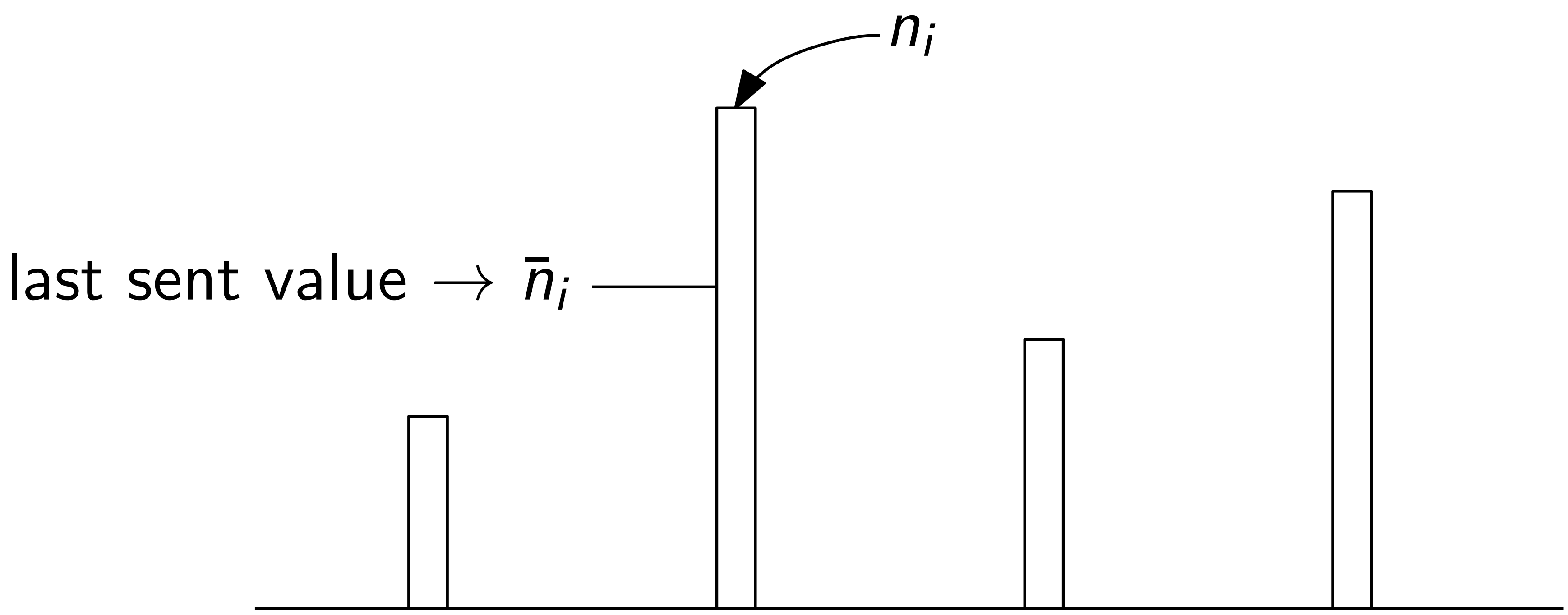
Randomized Algorithm

Sends n_i with probability p
when a new item arrives



Analysis

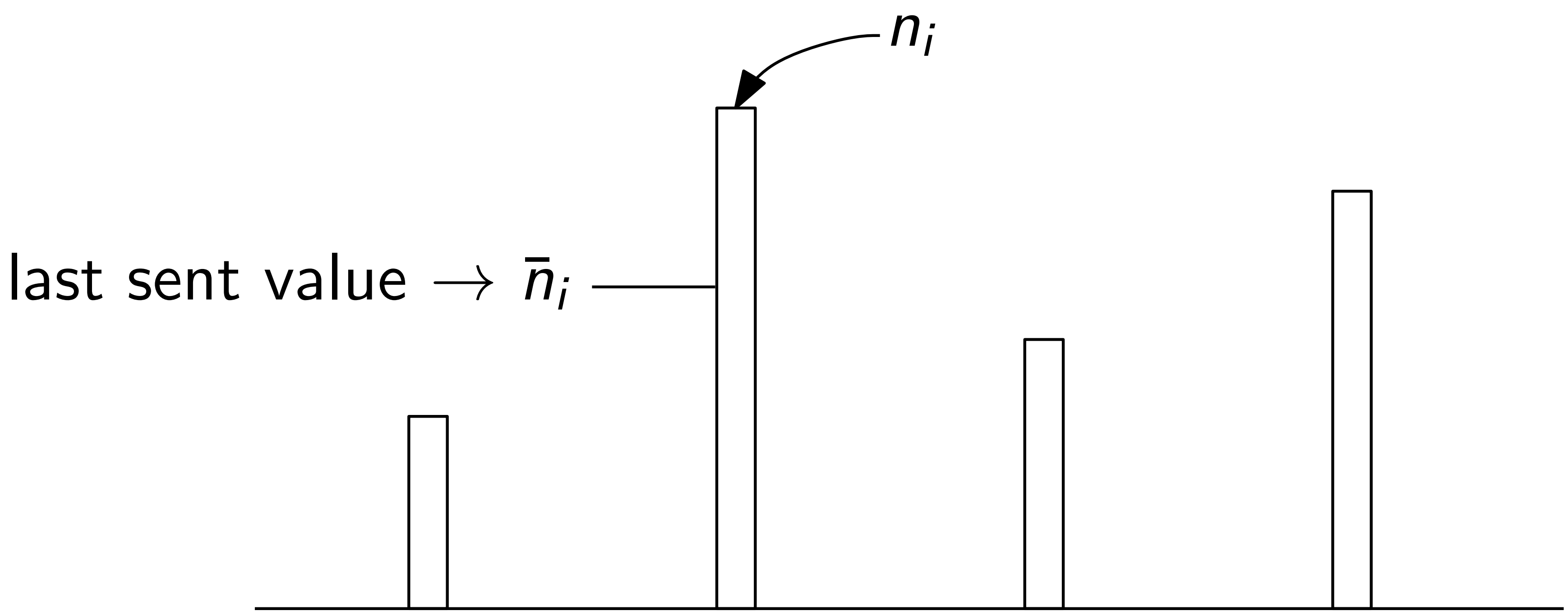
$n_i - \bar{n}_i$ is a random variable



Analysis

$n_i - \bar{n}_i$ is a random variable

$$\hat{n}_i = \begin{cases} \bar{n}_i - 1 + 1/p, & \text{if } \bar{n}_i \text{ exists;} \\ 0, & \text{else.} \end{cases}$$



Analysis

$n_i - \bar{n}_i$ is a random variable

$$\hat{n}_i = \begin{cases} \bar{n}_i - 1 + 1/p, & \text{if } \bar{n}_i \text{ exists;} \\ 0, & \text{else.} \end{cases}$$

$\mathrm{E}[\hat{n}_i] = n_i$, $\mathrm{Var}[\hat{n}_i] = 1/p^2$

Analysis

$n_i - \bar{n}_i$ is a random variable

$$\hat{n}_i = \begin{cases} \bar{n}_i - 1 + 1/p, & \text{if } \bar{n}_i \text{ exists;} \\ 0, & \text{else.} \end{cases}$$

$E[\hat{n}_i] = n_i$, $\mathrm{Var}[\hat{n}_i] = 1/p^2$

$$\hat{n} = \sum \hat{n}_i$$

$E[\hat{n}] = \sum \hat{n}_i = n$, $\mathrm{Var}[\hat{n}] = k/p^2$

Rounds

Chebyshev inequality

SD less than $\varepsilon n \to p = O(\sqrt{k}/\varepsilon n)$
constant probability of success (at any one time instance)

Rounds

Chebyshev inequality

SD less than $\varepsilon n \to p = O(\sqrt{k}/\varepsilon n)$
constant probability of success (at any one time instance)

- Track a 2-approximation $\bar{n}$ of n using the deterministic algorithm
  - Broadcast $\bar{n}$ whenever $\bar{n}$ doubles
  - Set $p = \frac{\sqrt{k}}{2\bar{n}}$
- Divide the tracking period into rounds
  - n changes by at most a constant factor in a round
  - p is fixed in a round

Communication Cost

- Communication cost
  - Tracking a 2-approximation: $O(k \log n)$
  - Number of messages in a round: $O(np) = O(\sqrt{k}/\varepsilon)$
  - Total: $O(k \log n + \sqrt{k}/\varepsilon \cdot \log n)$
    - Can be improved to $O(k \log n / \log(k\varepsilon^2) + \sqrt{k}/\varepsilon \cdot \log n)$

Communication Cost

- Communication cost
  - Tracking a 2-approximation: $O(k \log n)$
  - Number of messages in a round: $O(np) = O(\sqrt{k}/\varepsilon)$
  - Total: $O(k \log n + \sqrt{k}/\varepsilon \cdot \log n)$
    - Can be improved to $O(k \log n / \log(k\varepsilon^2) + \sqrt{k}/\varepsilon \cdot \log n)$
- Lower bounds
  - Only allow one-way communication: $\Omega(k/\varepsilon \cdot \log n)$ (randomization doesn't help)
  - Two-way communication: $\Omega(k + \sqrt{k}/\varepsilon \cdot \log n)$

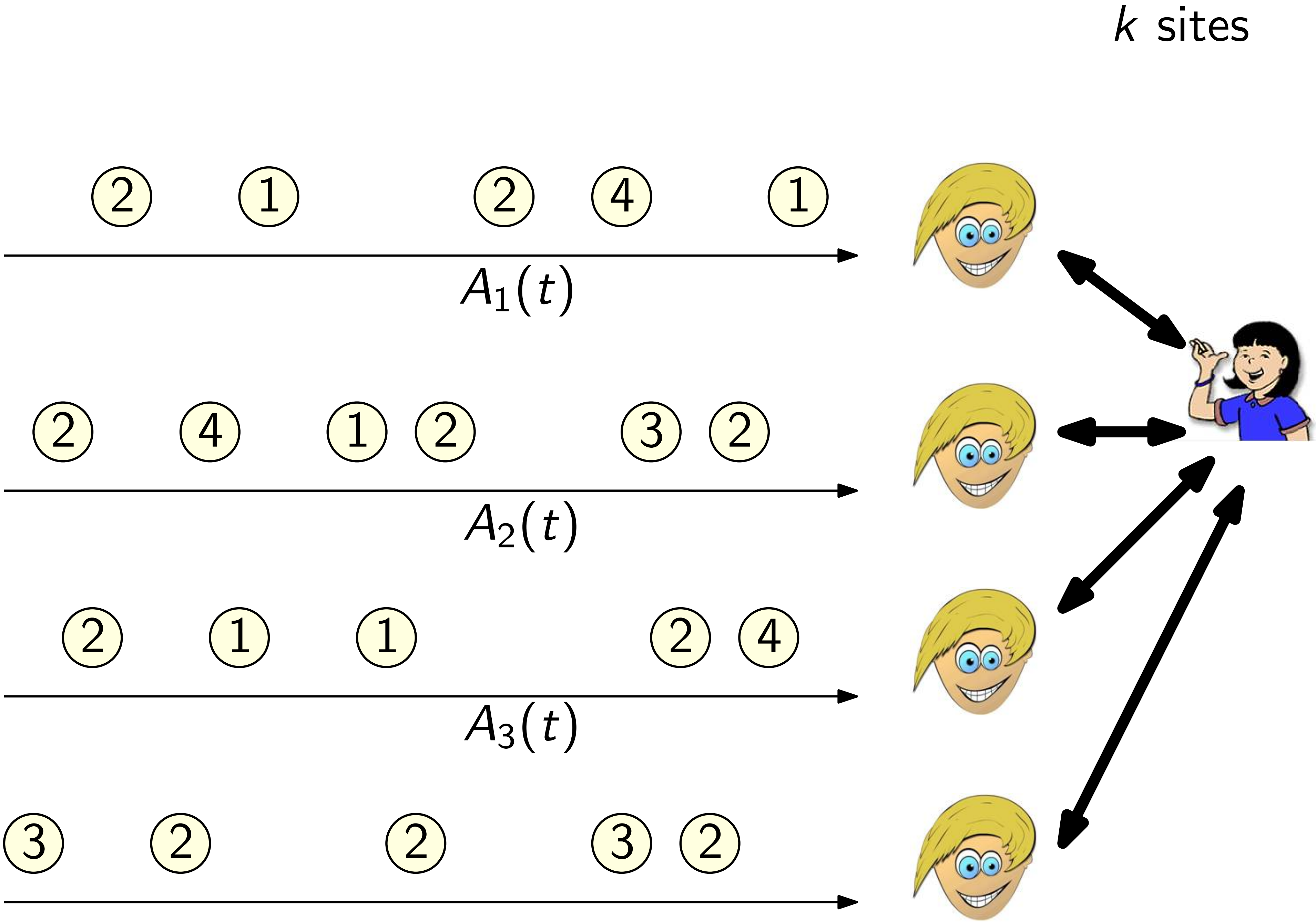
Tight Bounds for Count-Tracking

- Upper bound in words
- Lower bound in number of messages

$k < 1/\varepsilon^2$	$k > 1/\varepsilon^2$
$\Theta(\sqrt{k}/\varepsilon \cdot \log n)$	$\Theta\left(k \frac{\log n}{\log(k\varepsilon^2)}\right)$

[Huang, Yi, Zhang, PODS'12]

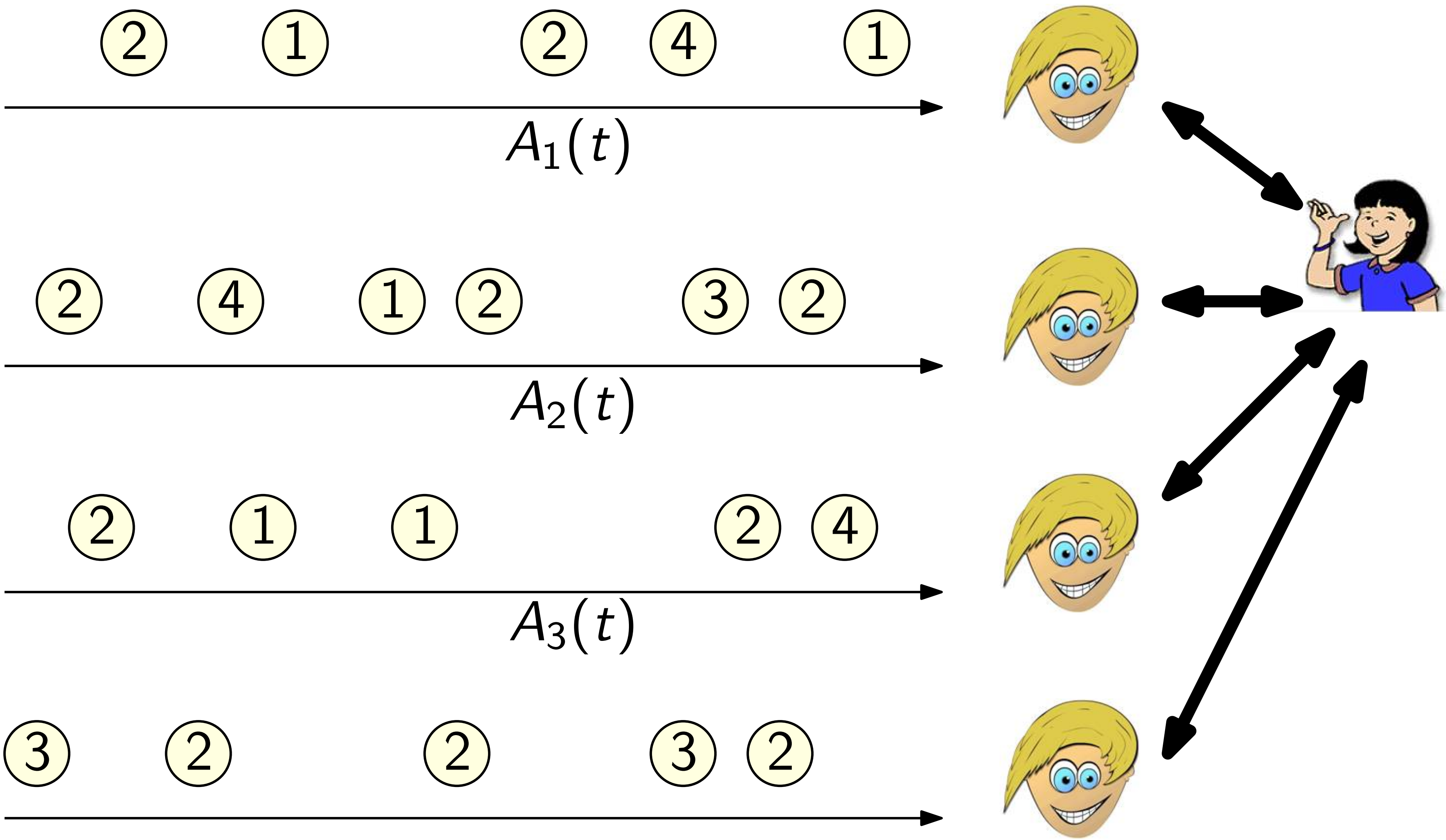
The Distributed Streaming Model



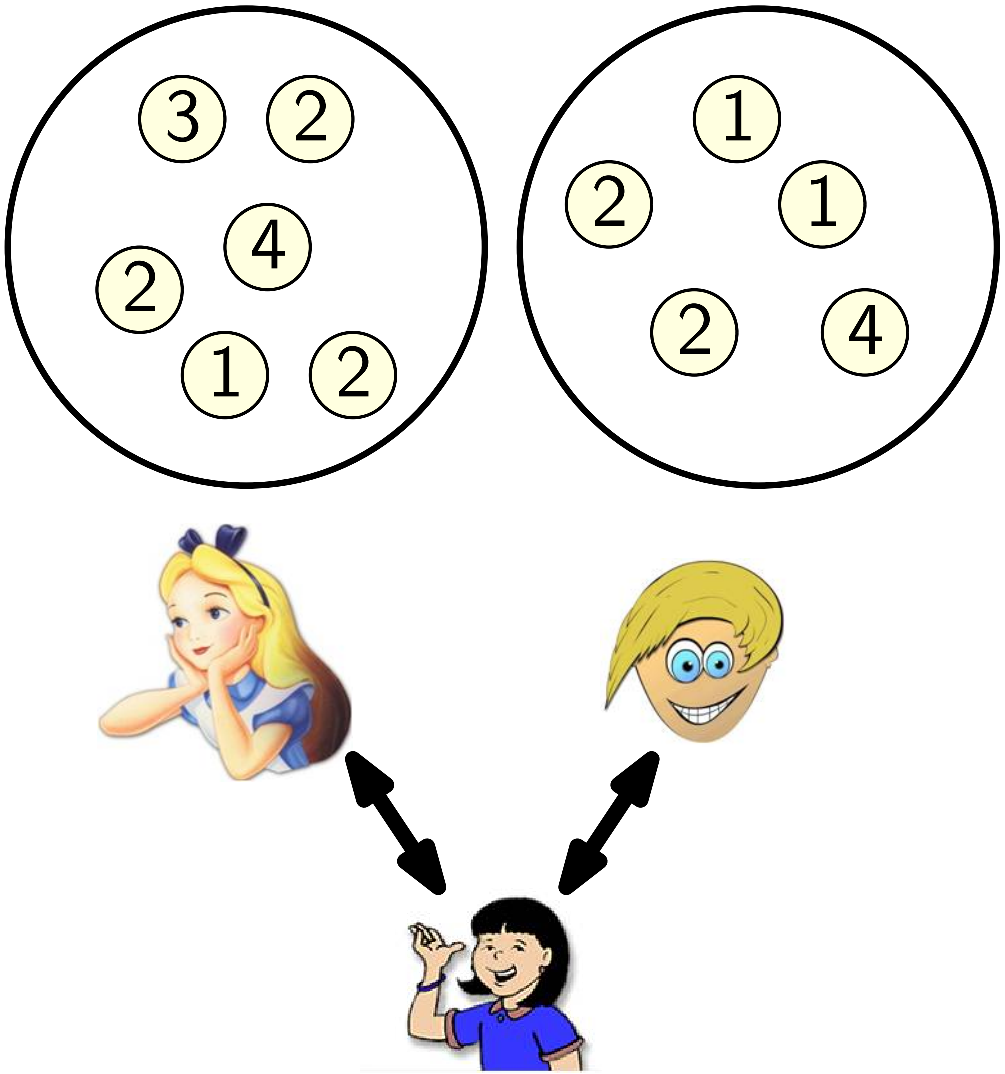
The Distributed Streaming Model

Coordinator tries to compute
$f(A_1(t) \uplus A_2(t) \uplus \cdots \uplus A_k(t))$ for all t

k sites

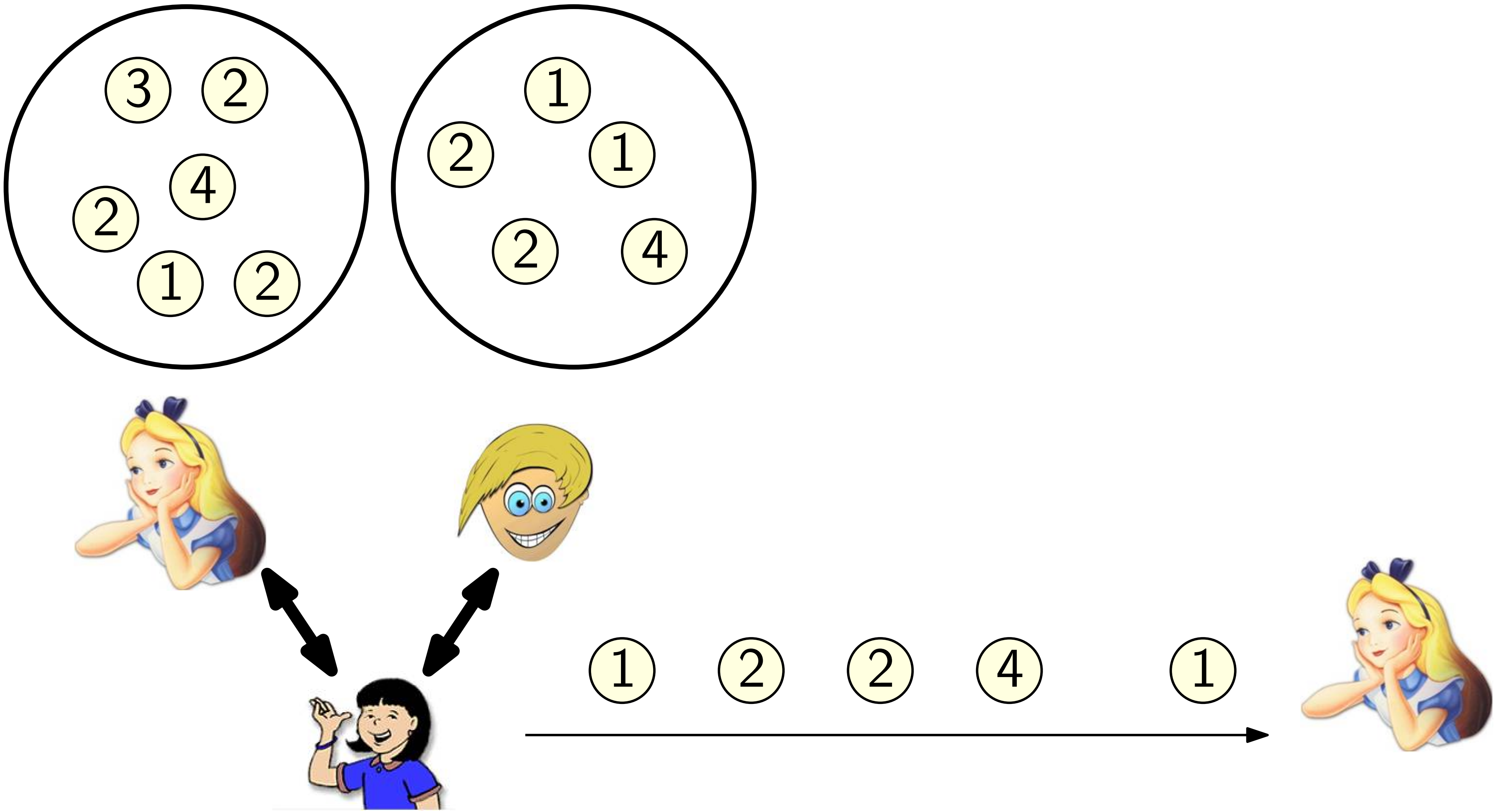


Generalization of Two Models



Communication model
(One-shot model)

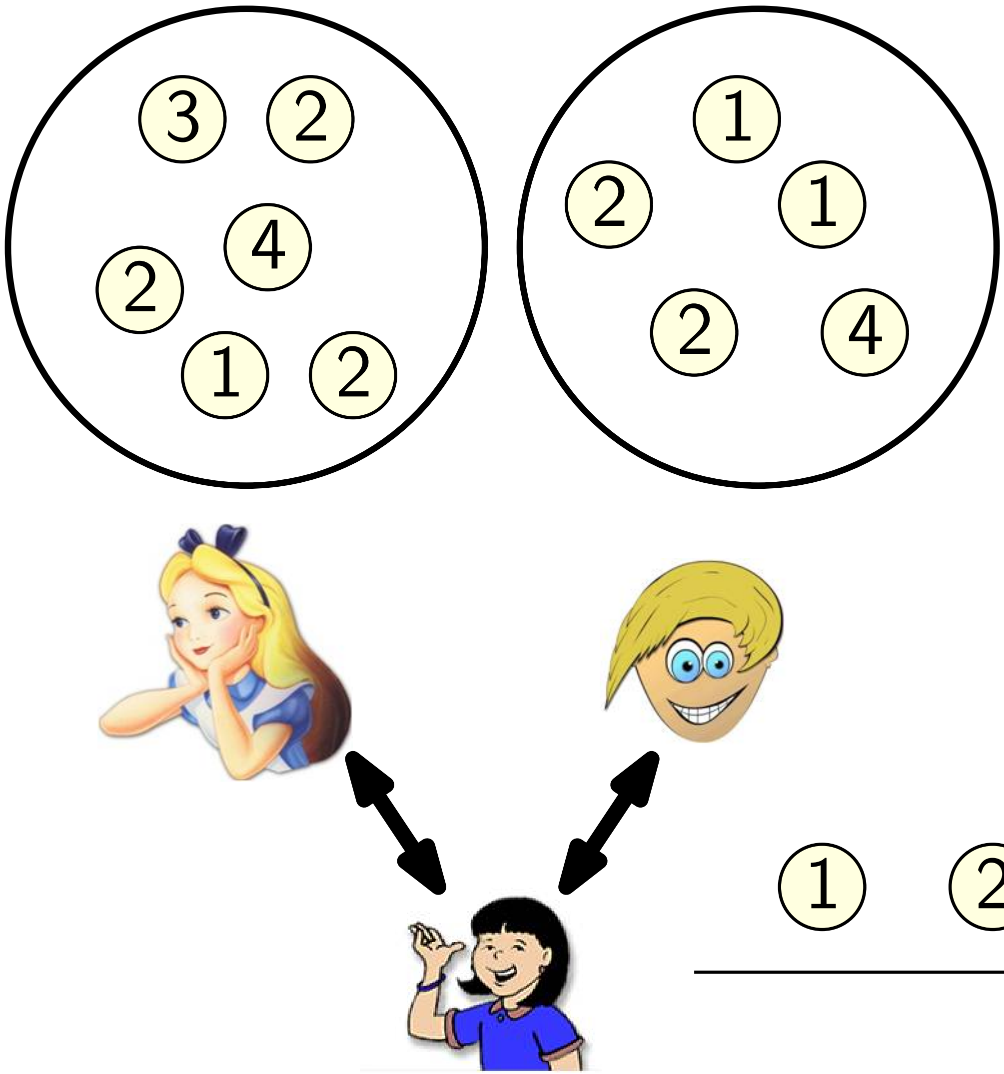
Generalization of Two Models



Communication model
(One-shot model)

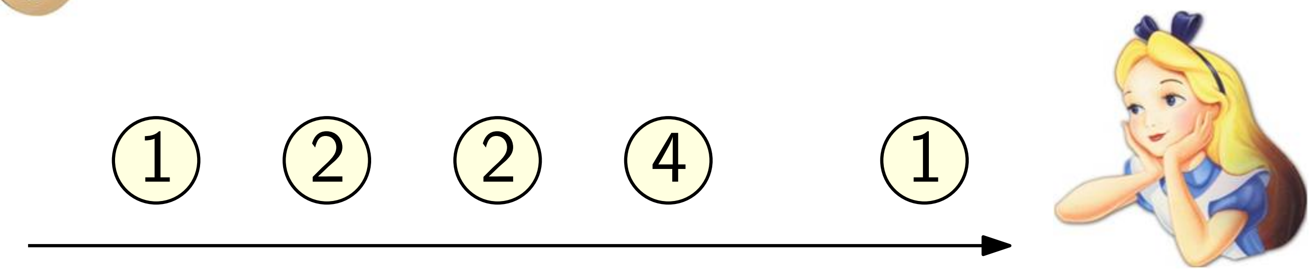
Data stream model

Generalization of Two Models



Goal

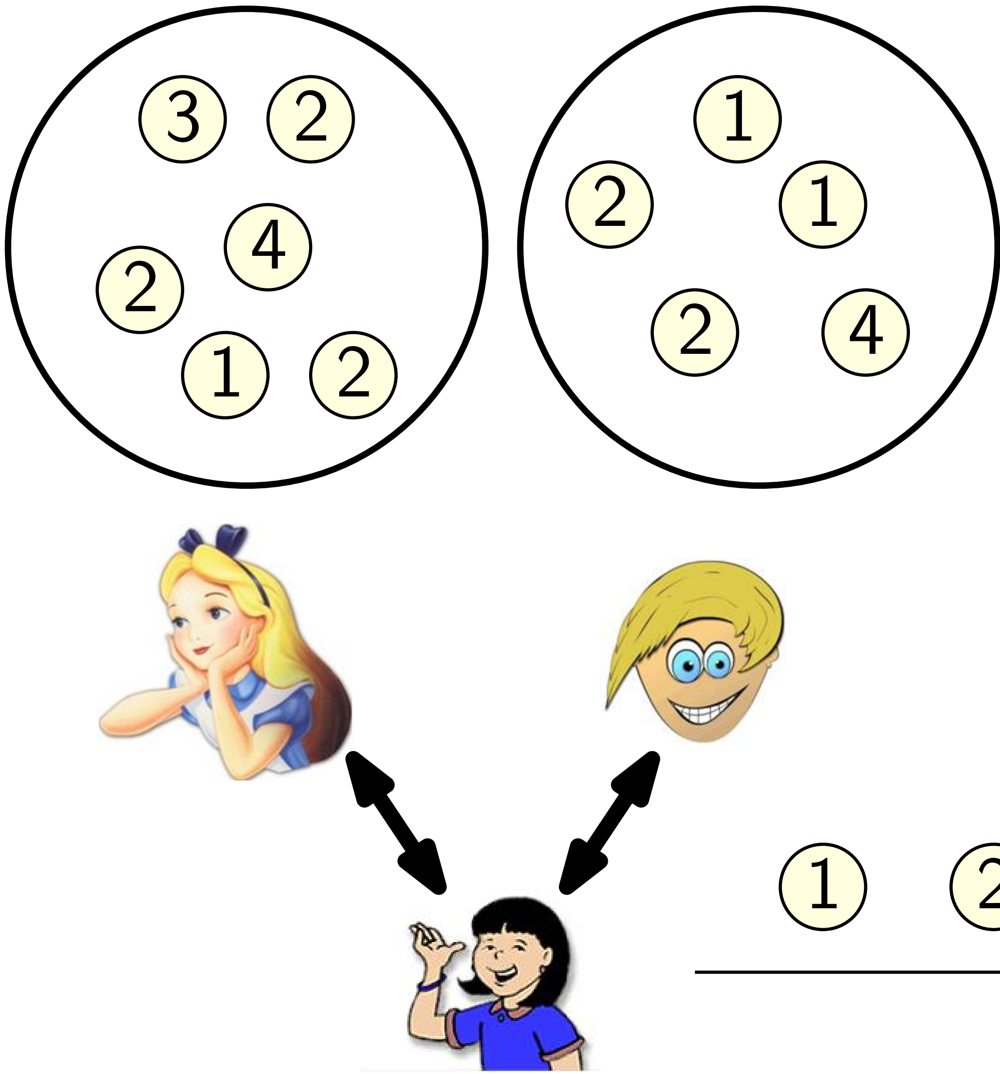
- Communication cost
- Space



Communication model
(One-shot model)

Data stream model

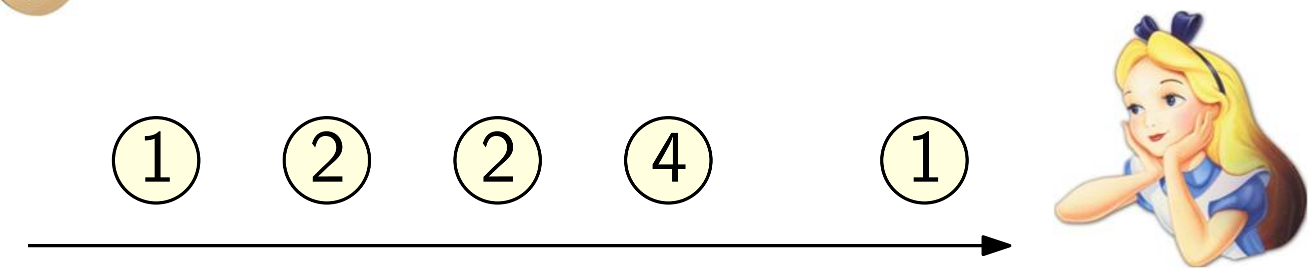
Generalization of Two Models



Trivial problems in these two models could be highly nontrivial in the combined model!

Goal

- Communication cost
- Space



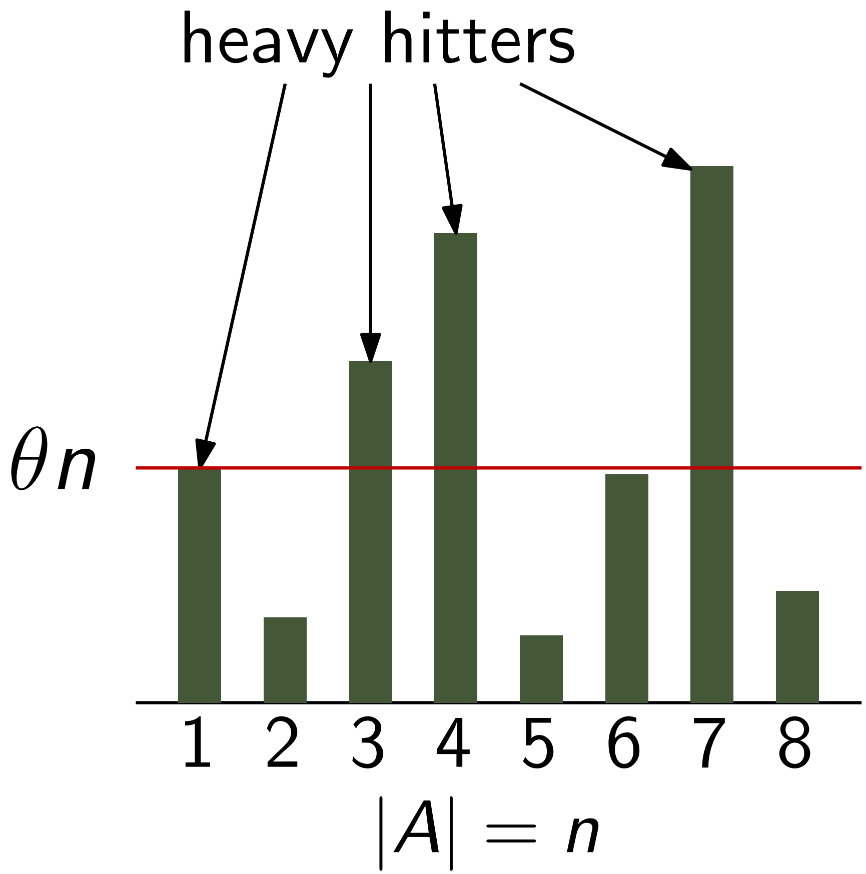
Communication model
(One-shot model)

Data stream model

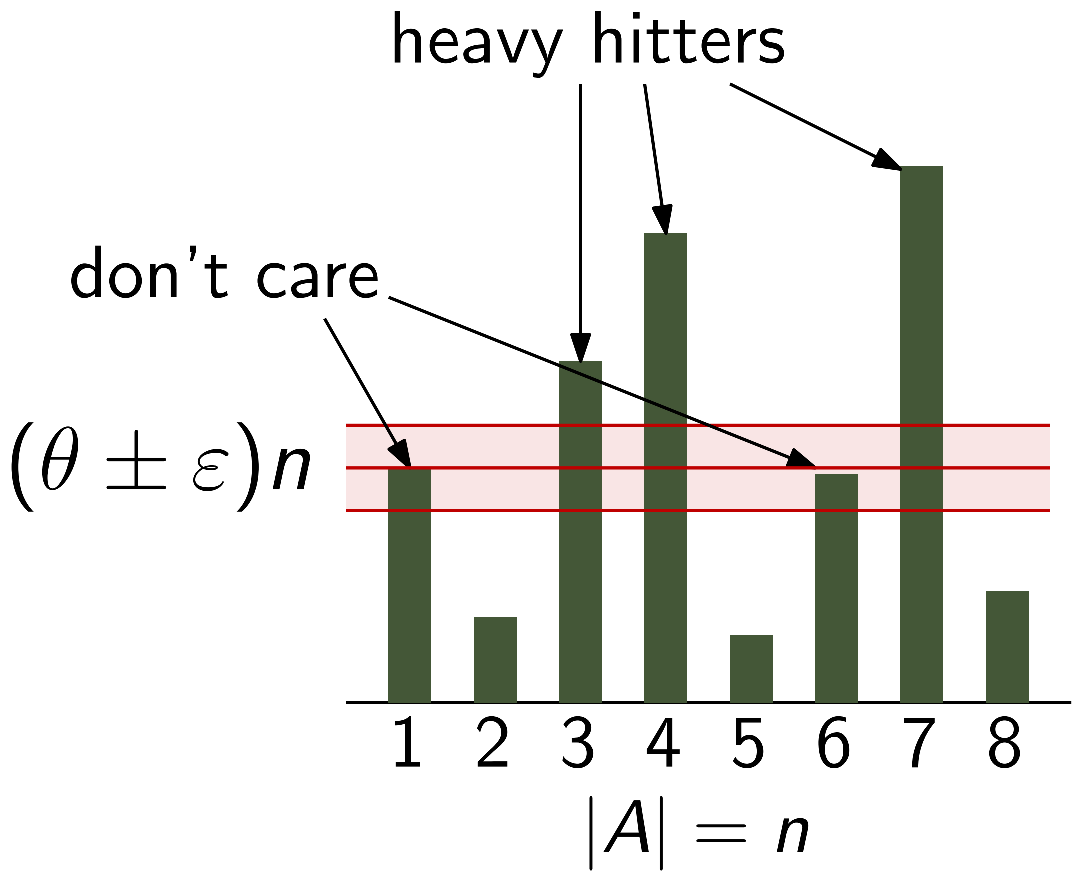
Problems

- The count-down problem
- Count-tracking
- Frequent items (heavy hitters)
- Random sampling
- Other problems

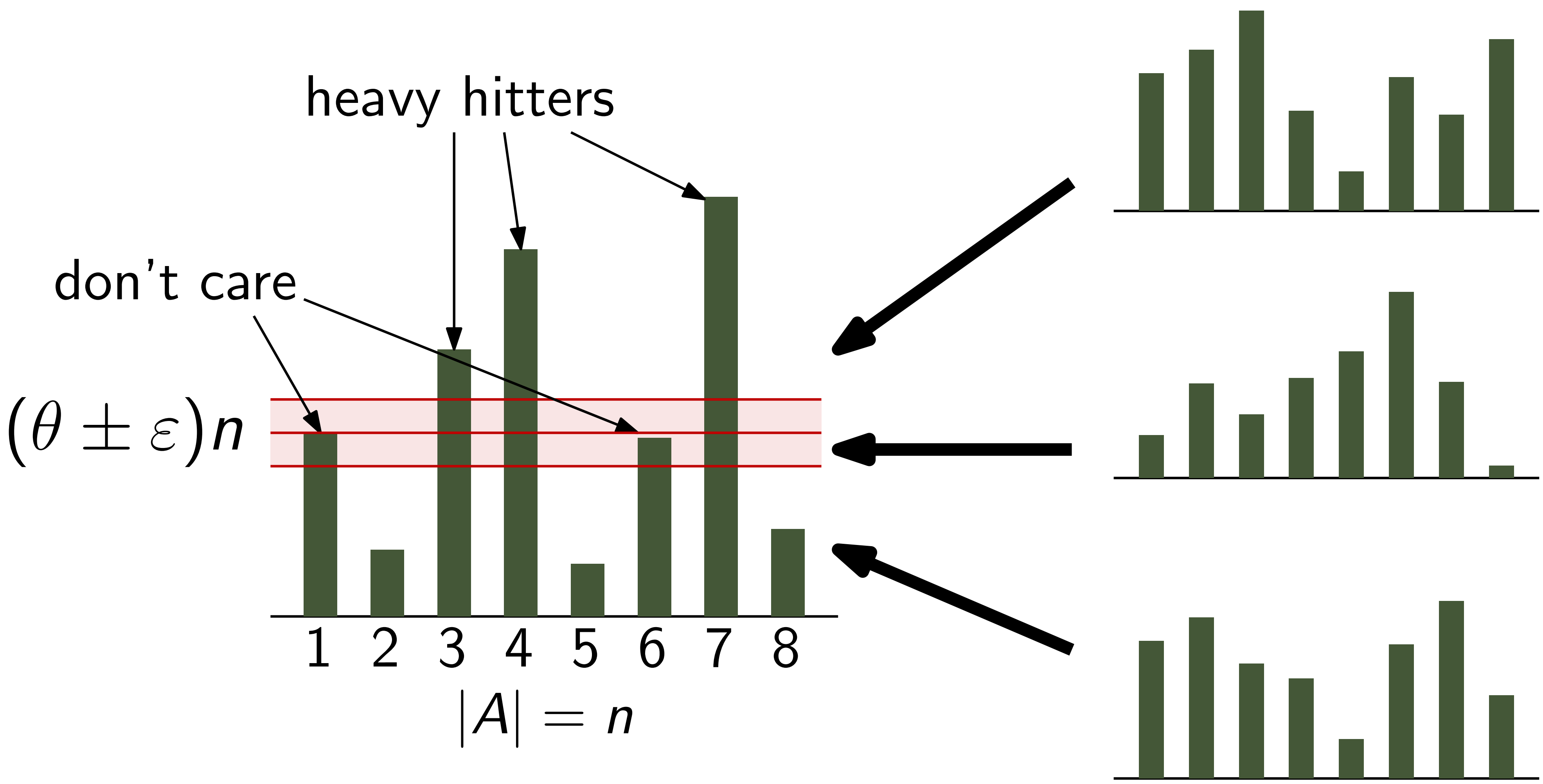
Frequent Items: Definition



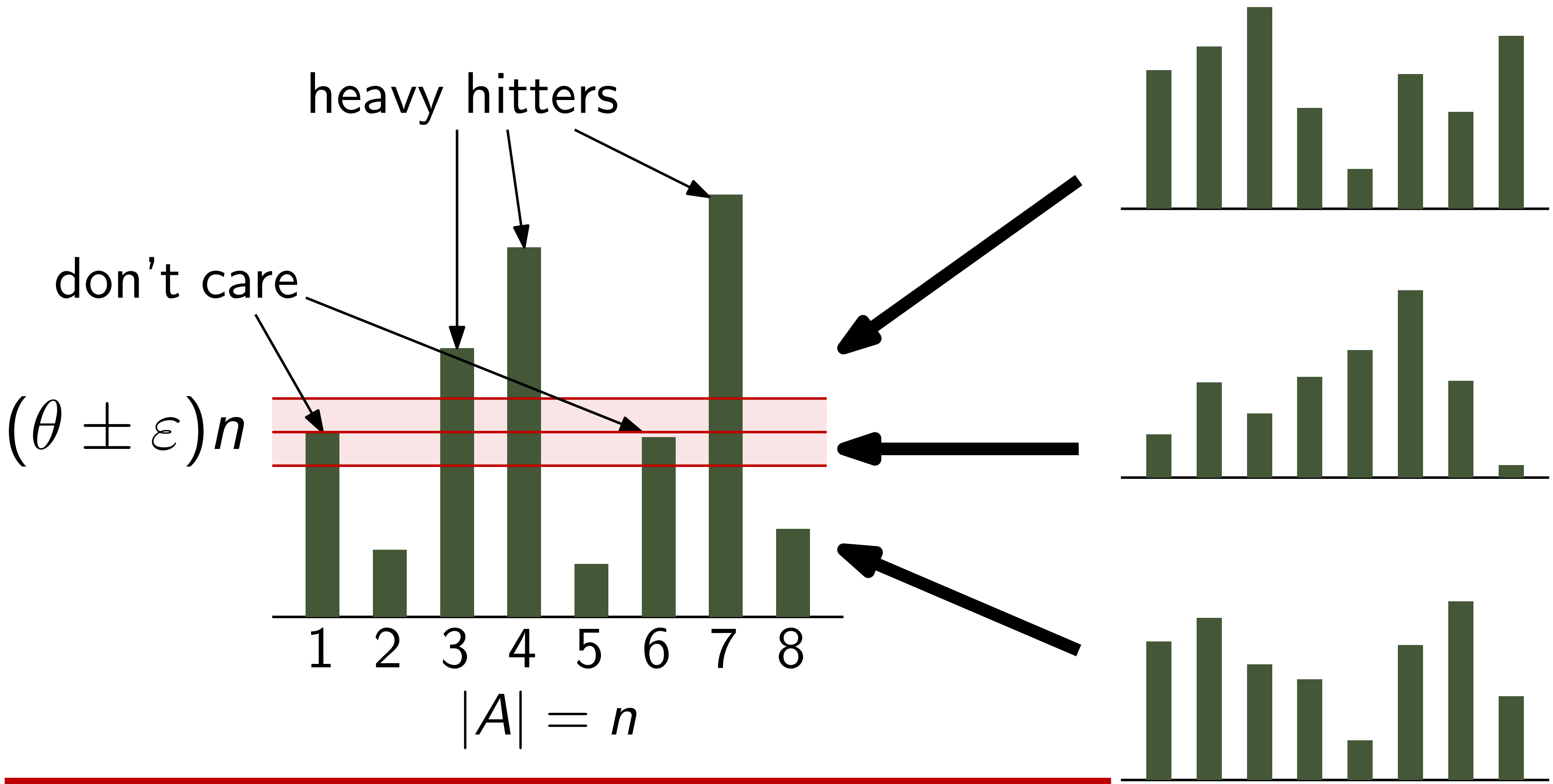
Frequent Items: Definition



Frequent Items: Definition



Frequent Items: Definition



Frequency estimation with F_1 error

Estimate the frequency of every element with additive error εn.

Frequent Items

Use the previous algorithm on each item i

- Maintain a count for each item at each site
- Space

Frequent Items

Use the previous algorithm on each item i

- Maintain a count for each item at each site
- Space

Streaming algorithm (Misra-Gries)

cost per site: $O(1/\varepsilon)$

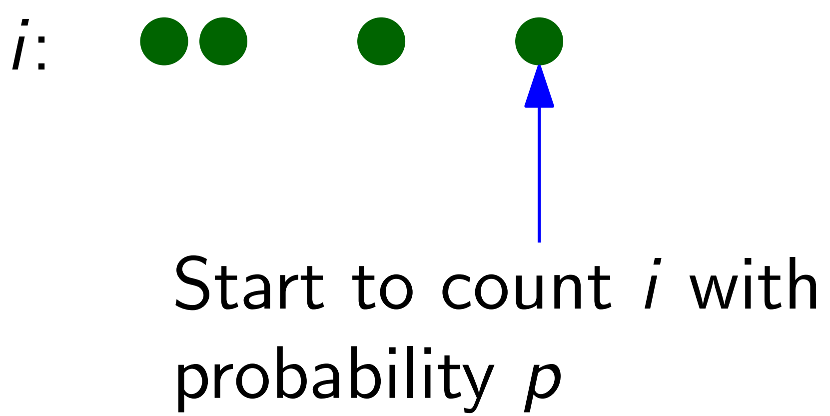
- total: $O(k/\varepsilon)$
- improve to $O(\sqrt{k}/\varepsilon)$

Frequent Items: Algorithm

Idea: maintain only large enough counts

Frequent Items: Algorithm

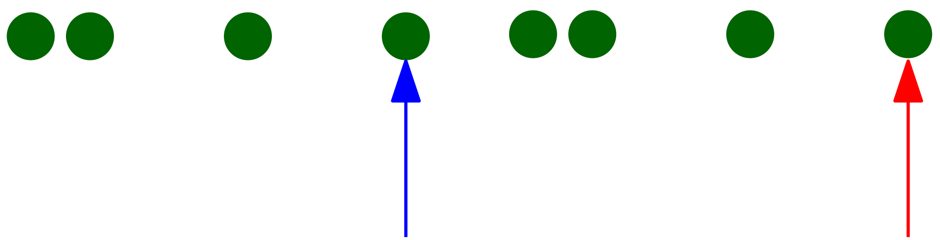
Idea: maintain only large enough counts



Frequent Items: Algorithm

Idea: maintain only large enough counts

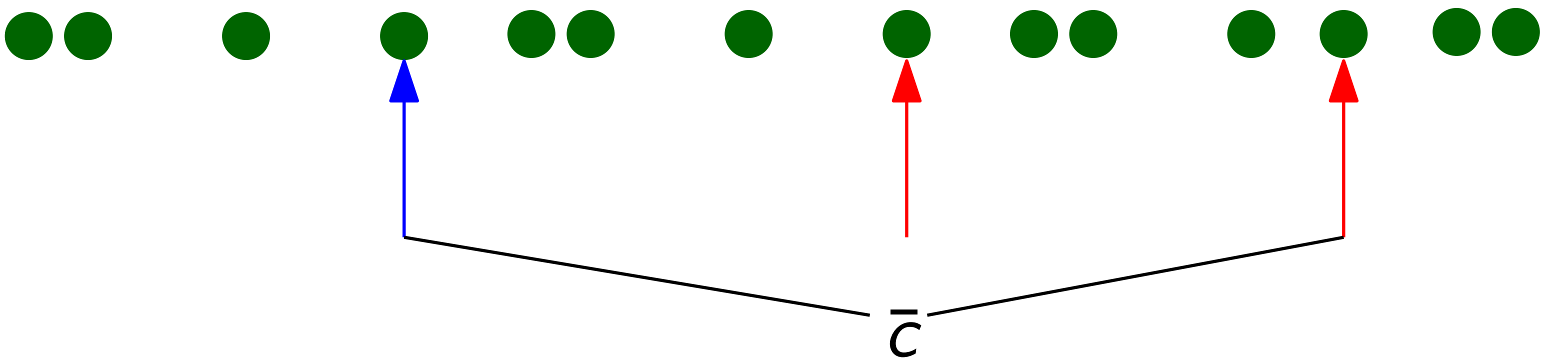
i:



Start to count i with probability p

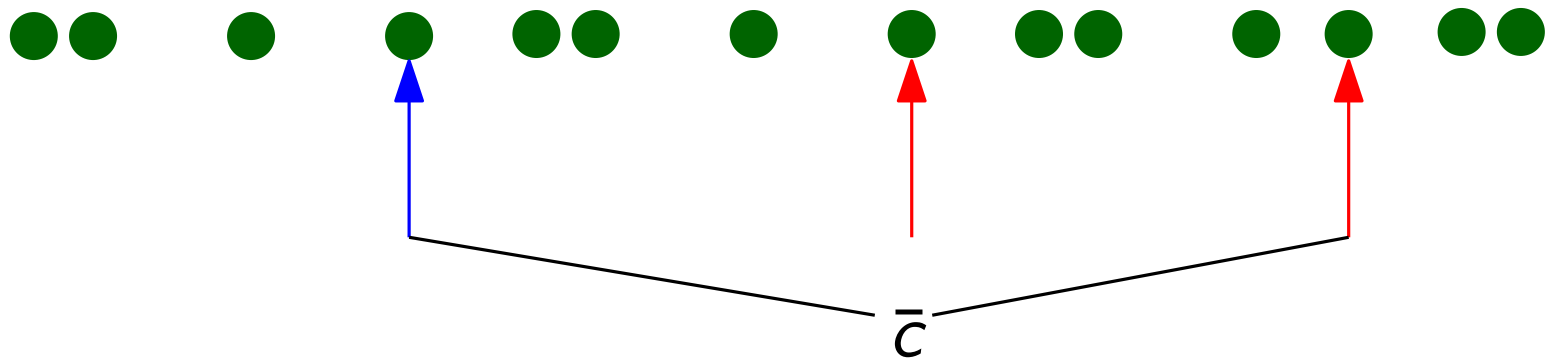
Update the count with probability p

Frequent Items: Analysis



Coordinator only know $\bar{c}$

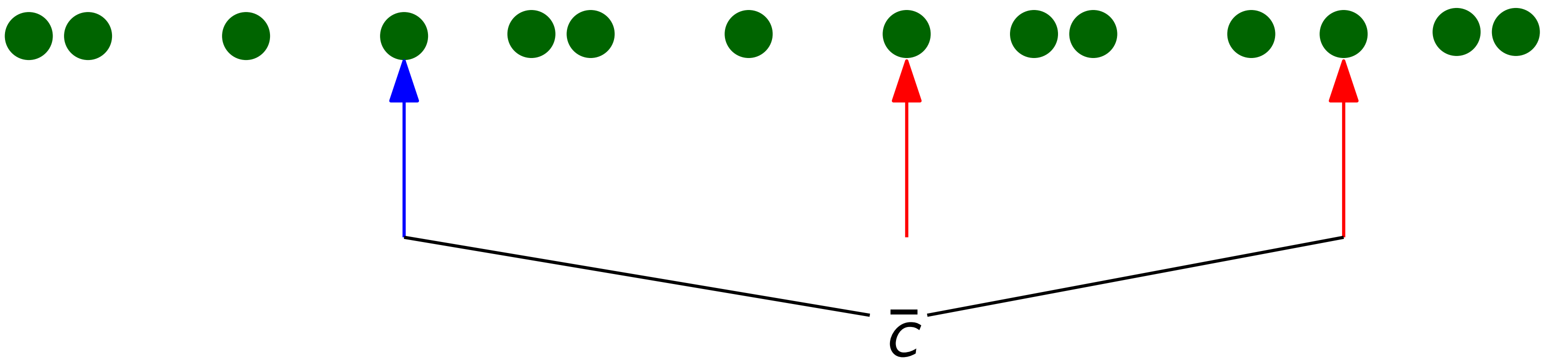
Frequent Items: Analysis



Coordinator only know $\bar{c}$

$$\hat{f}_i = \begin{cases} \bar{c} - 1 + 2/p, & \text{if } \bar{c} > 0; \\ 0, & \text{else.} \end{cases}$$

Frequent Items: Analysis

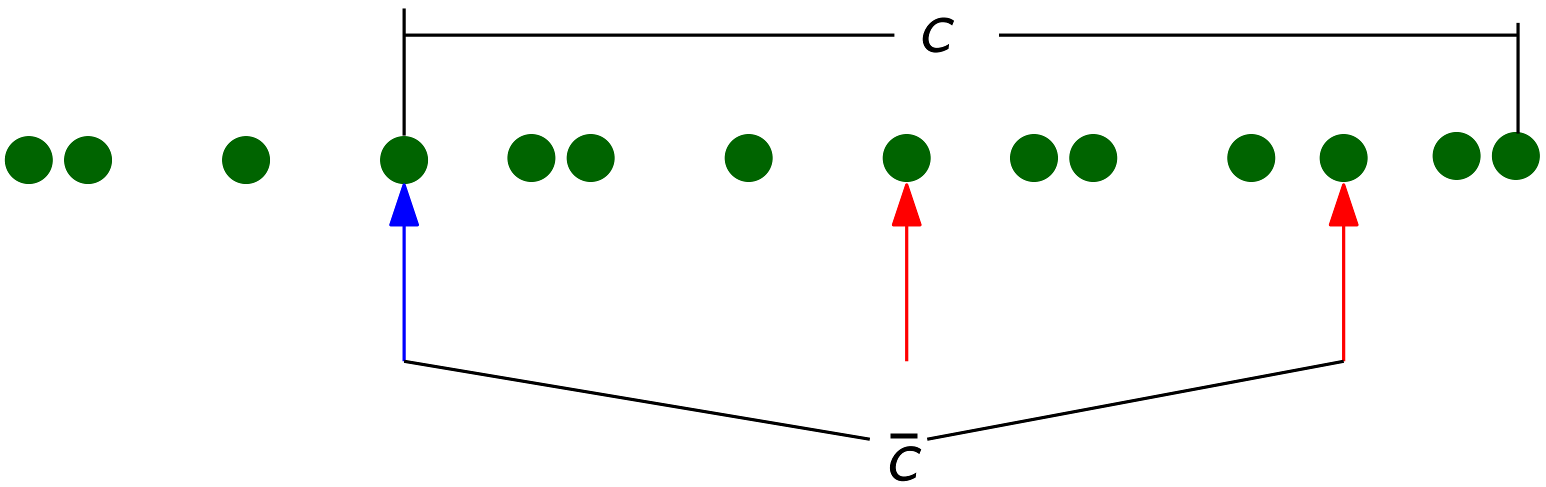


Coordinator only know $\bar{c}$

$$\hat{f}_i = \begin{cases} \bar{c} - 1 + 2/p, & \text{if } \bar{c} > 0; \\ 0, & \text{else.} \end{cases}$$

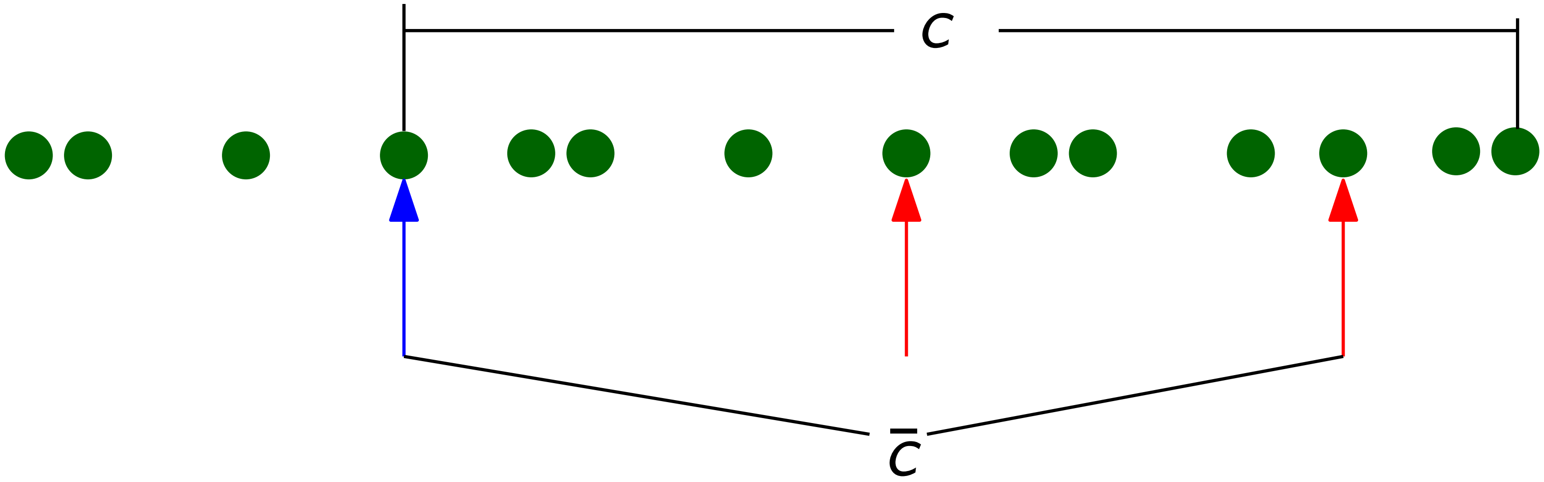
Bias might be as large as $\varepsilon n/\sqrt{k}$

Frequent Items: Analysis



Coordinator only know $\bar{c}$

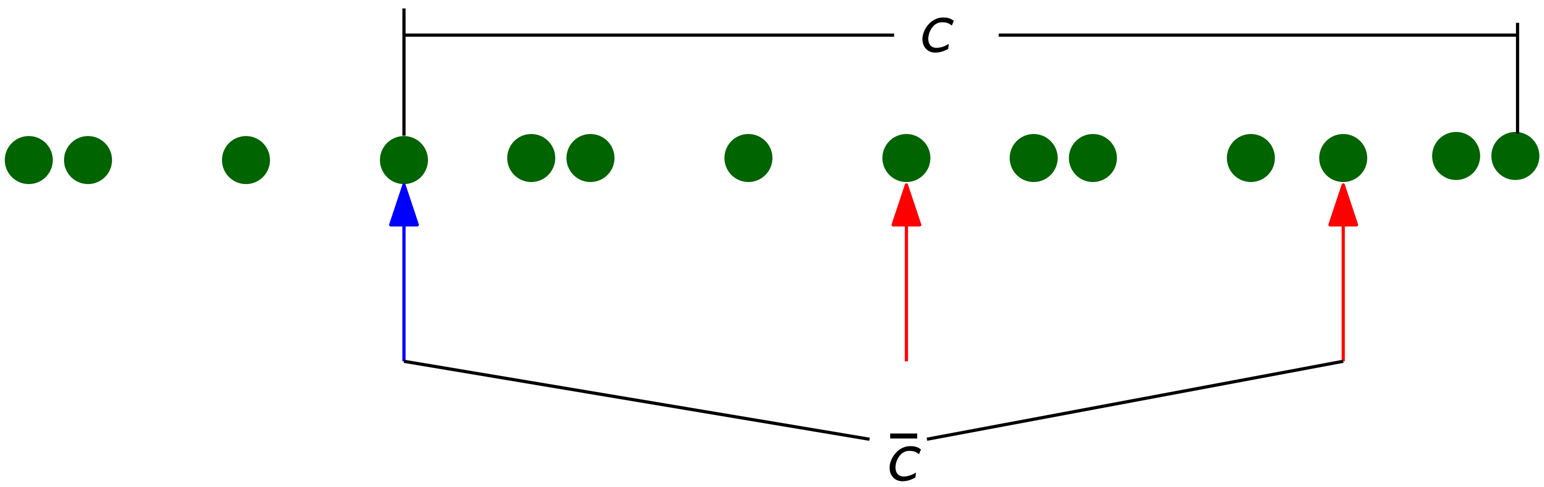
Frequent Items: Analysis



Coordinator only know $\bar{c}$

$$\hat{f}_i = \begin{cases} c - 1 + 1/p, & \text{if } c > 0; \\ 0, & \text{else.} \end{cases}$$

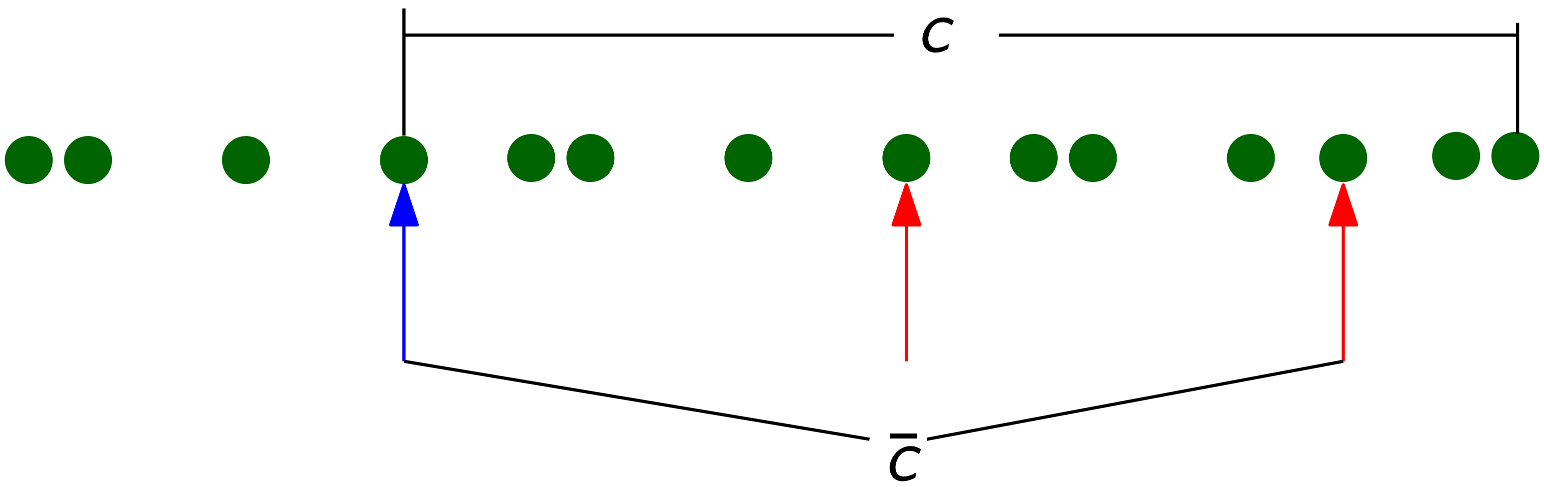
Frequent Items: Analysis



Estimate c by $\bar{c}$

$$\hat{c} = \begin{cases} \bar{c} - 1 + 1/p, & \text{if } \bar{c} > 0; \\ 0, & \text{else.} \end{cases}$$

Frequent Items: Analysis



Combined estimator

$$\hat{f}_i = \begin{cases} \bar{c} - 2 + 2/p, & \text{if } \bar{c} \geq 2; \\ 1/p, & \text{if } \bar{c} = 1; \\ 0, & \text{else.} \end{cases}$$

Frequent Items: Analysis

- $\mathrm{E}[\hat{f}_i] = f_i$
- $\mathrm{Var}[\hat{f}_i] \leq 2/p^2$

Frequent Items: Analysis

- $\mathrm{E}[\hat{f}_i] = f_i$
- $\mathrm{Var}[\hat{f}_i] \leq 2/p^2$

  set $p = O(\frac{\sqrt{k}}{\varepsilon n})$

  space: $O(\sqrt{k}/\varepsilon)$

  space per site: $O(1/(\varepsilon\sqrt{k}))$

  communication: same as before

Frequent Items: Lower Bound

- Communication lower bound still hold
- Space lower bound

Frequent Items: Lower Bound

- Communication lower bound still hold
- Space lower bound
  - Communication-space tradeoff

Communication-Space Tradeoff

Theorem

Any randomized algorithm that solves the frequency tracking problem with communication C bits and uses M bits of space per site, we have $C \cdot M = \Omega(\log n/\varepsilon^2)$.

Communication-Space Tradeoff

Theorem

Any randomized algorithm that solves the frequency tracking problem with communication C bits and uses M bits of space per site, we have $C \cdot M = \Omega(\log n/\varepsilon^2)$.

Communication cost: $O(\sqrt{k}/\varepsilon \cdot \log n)$ bits

Space per site: $\Omega(1/(\varepsilon\sqrt{k}))$ bits

Communication Complexity

Theorem

The k-party communication complexity for the one-shot frequency estimation problem is $\Omega(\sqrt{k}/\varepsilon)$ bits.

[Woodruff, Zhang, STOC'12]

Communication Complexity

Theorem

The k-party communication complexity for the one-shot frequency estimation problem is $\Omega(\sqrt{k}/\varepsilon)$ bits.

Direct-Sum theorem

Solve ℓ instances of the frequency estimation problem simultaneously needs $\Omega(\ell \cdot \sqrt{k}/\varepsilon)$ bits of communication.

[Woodruff, Zhang, STOC'12]

Communication-Space Tradeoff

Proof sketch

Let $\mathcal{A}$ be a k-party tracking algorithm with communication C and space M

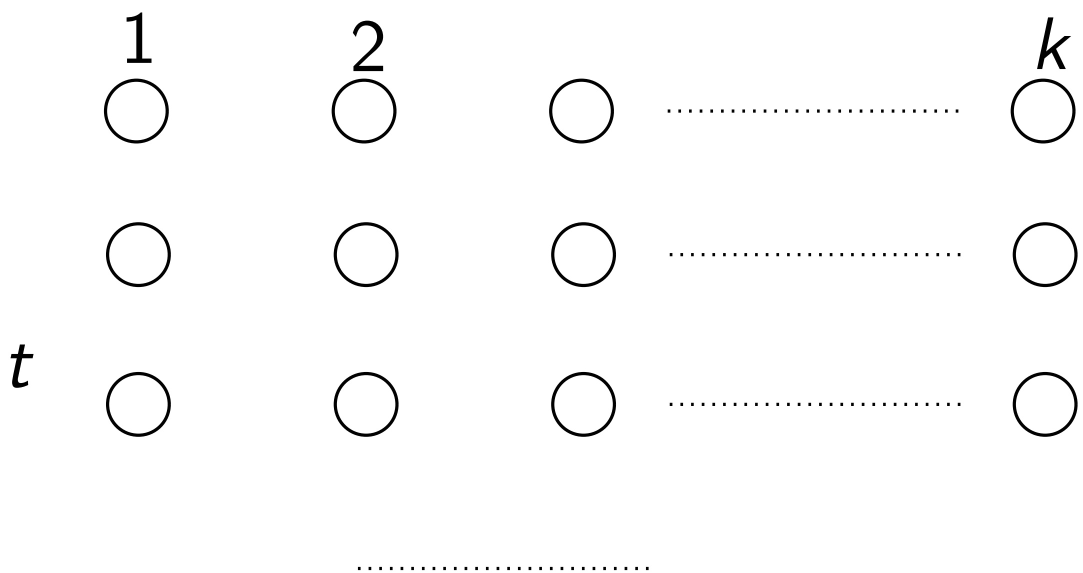
Use $\mathcal{A}$ to solve tk-party one-shot problem.

Communication-Space Tradeoff

Proof sketch

Let $\mathcal{A}$ be a k-party tracking algorithm with communication C and space M

Use $\mathcal{A}$ to solve tk-party one-shot problem.

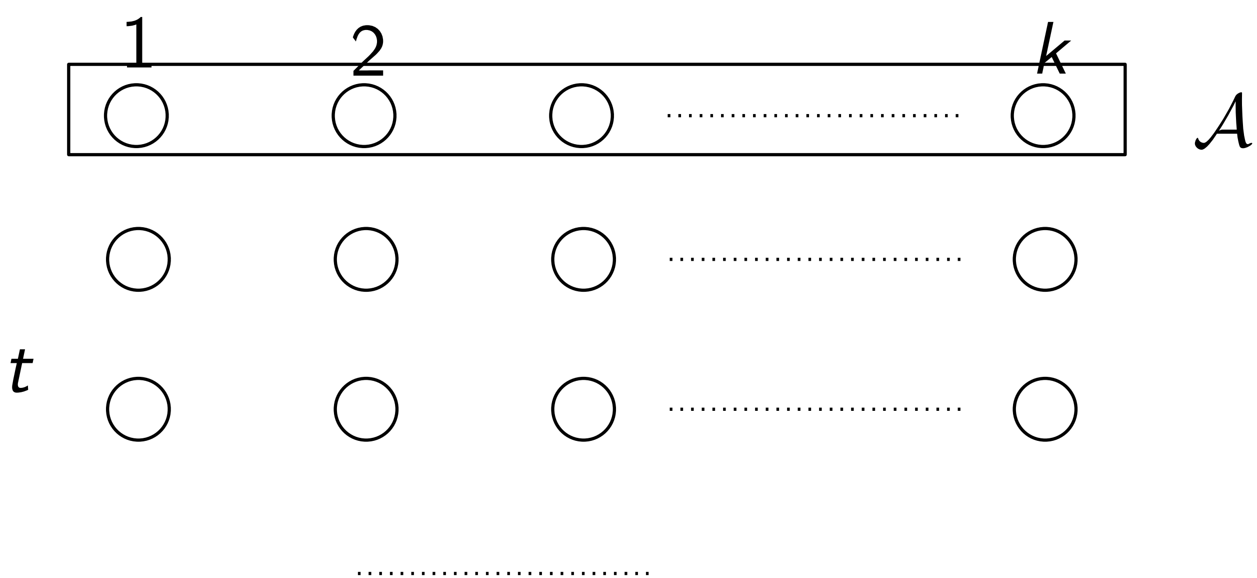


Communication-Space Tradeoff

Proof sketch

Let $\mathcal{A}$ be a k-party tracking algorithm with communication C and space M

Use $\mathcal{A}$ to solve tk-party one-shot problem.

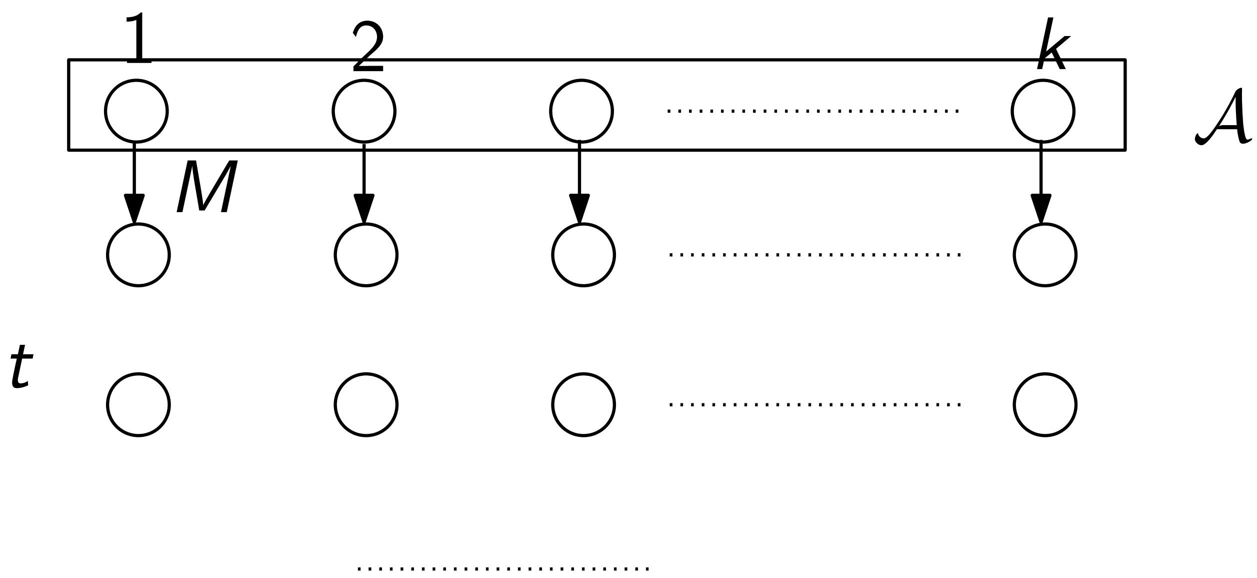


Communication-Space Tradeoff

Proof sketch

Let $\mathcal{A}$ be a k-party tracking algorithm with communication C and space M

Use $\mathcal{A}$ to solve tk-party one-shot problem.

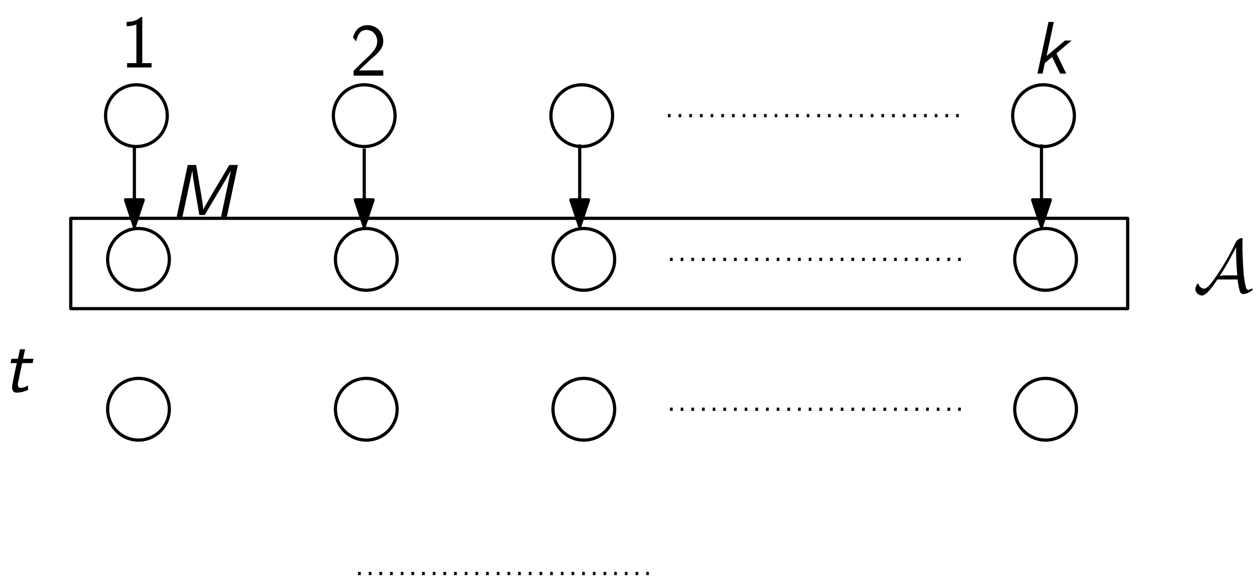


Communication-Space Tradeoff

Proof sketch

Let $\mathcal{A}$ be a k-party tracking algorithm with communication C and space M

Use $\mathcal{A}$ to solve tk-party one-shot problem.

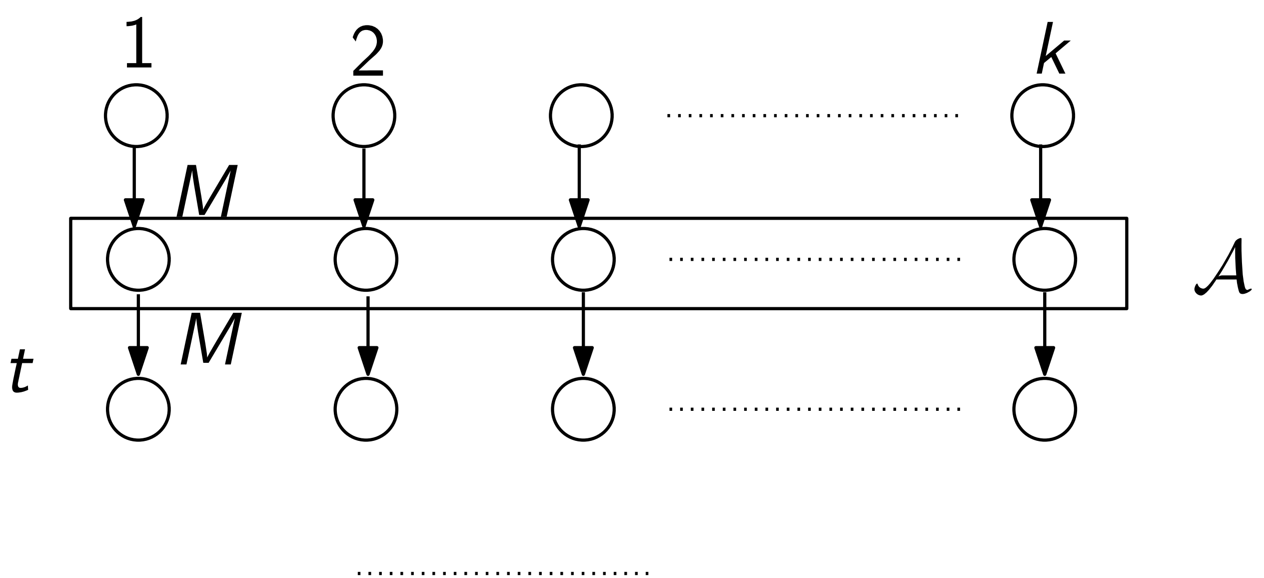


Communication-Space Tradeoff

Proof sketch

Let $\mathcal{A}$ be a k-party tracking algorithm with communication C and space M

Use $\mathcal{A}$ to solve tk-party one-shot problem.

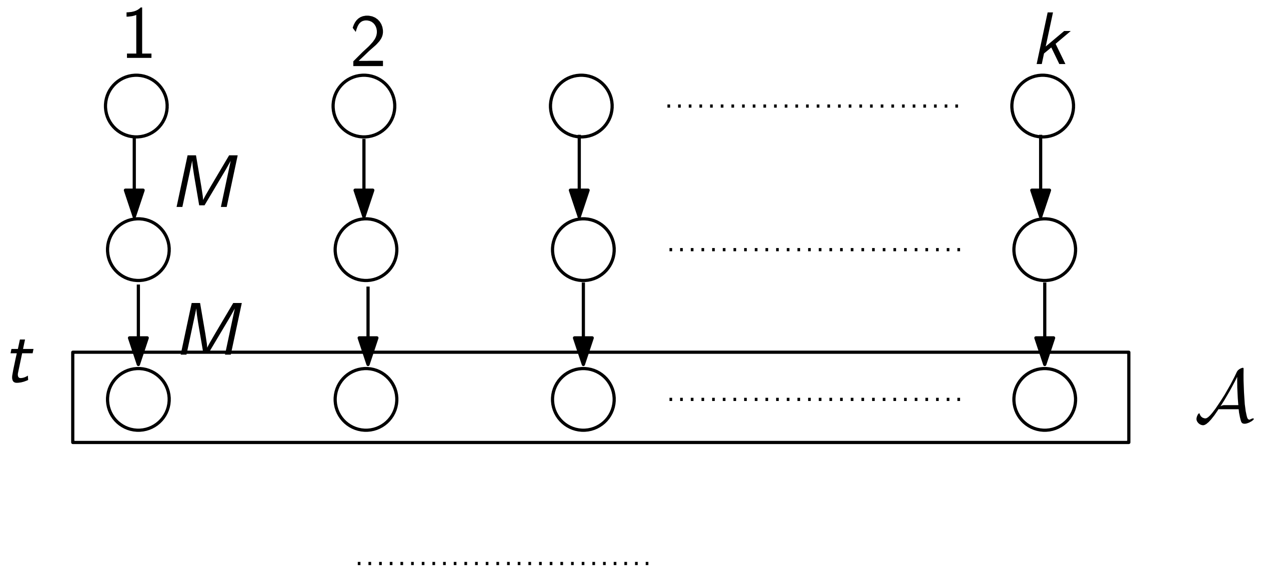


Communication-Space Tradeoff

Proof sketch

Let $\mathcal{A}$ be a k-party tracking algorithm with communication C and space M

Use $\mathcal{A}$ to solve tk-party one-shot problem.

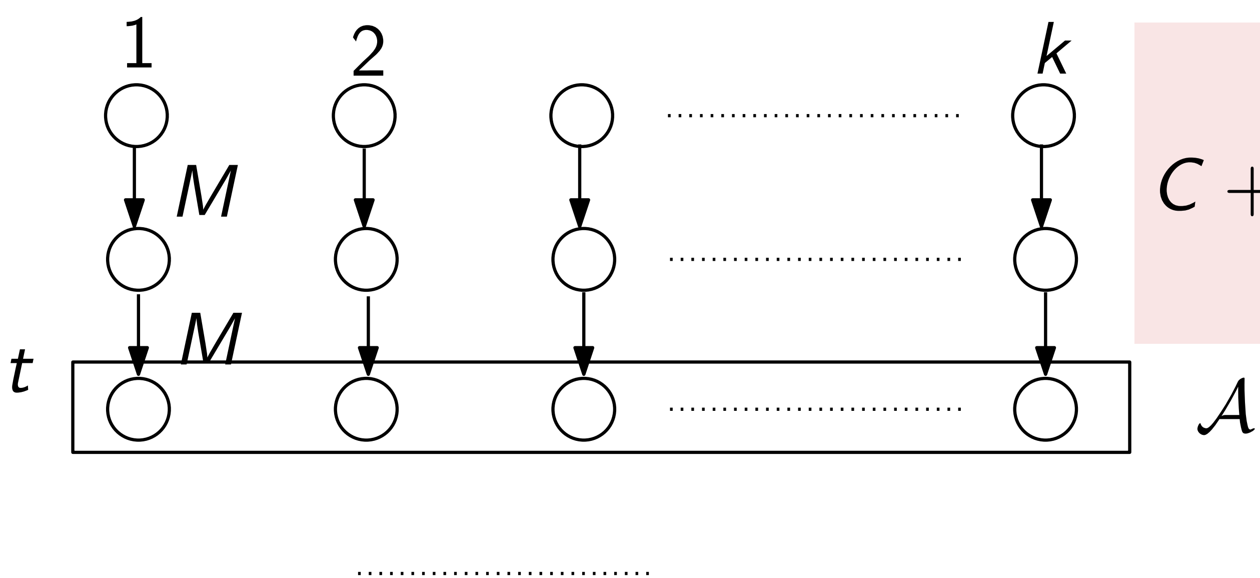


Communication-Space Tradeoff

Proof sketch

Let $\mathcal{A}$ be a k-party tracking algorithm with communication C and space M

Use $\mathcal{A}$ to solve tk-party one-shot problem.



$$C + M \cdot tk \geq \Omega(\frac{\sqrt{kt}}{\varepsilon})$$

Problems

- The count-down problem
- Count-tracking
- Frequent items (heavy hitters)
- Random sampling
- Other problems

Reservoir Sampling [Waterman '??; Vitter '85]

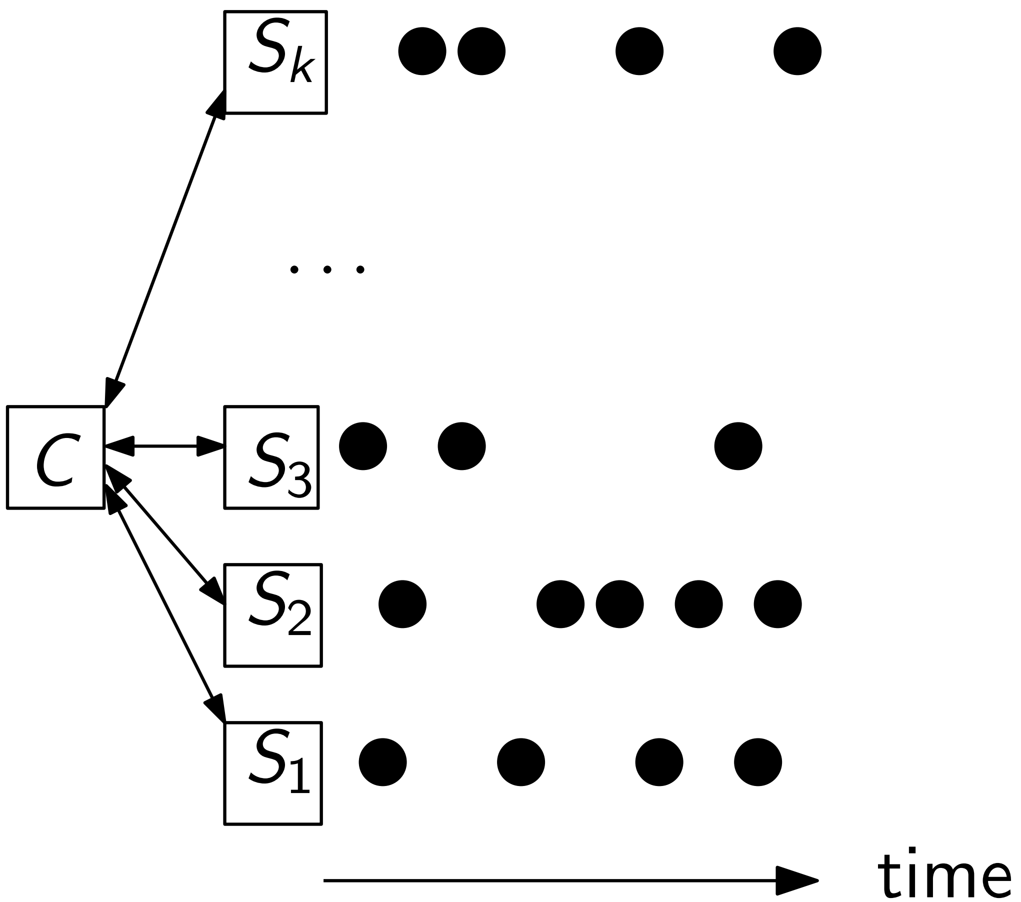
- Maintain a (uniform) sample (w/o replacement) of size s from a stream of n items
  - Every subset of size s has equal probability to be the sample

Reservoir Sampling [Waterman '??; Vitter '85]

- Maintain a (uniform) sample (w/o replacement) of size s from a stream of n items
  - Every subset of size s has equal probability to be the sample
- When the i-th item arrives
  - With probability s/i, use it to replace an item in the current sample chosen uniformly at ranfom
  - With probability $1 - s/i$, throw it away

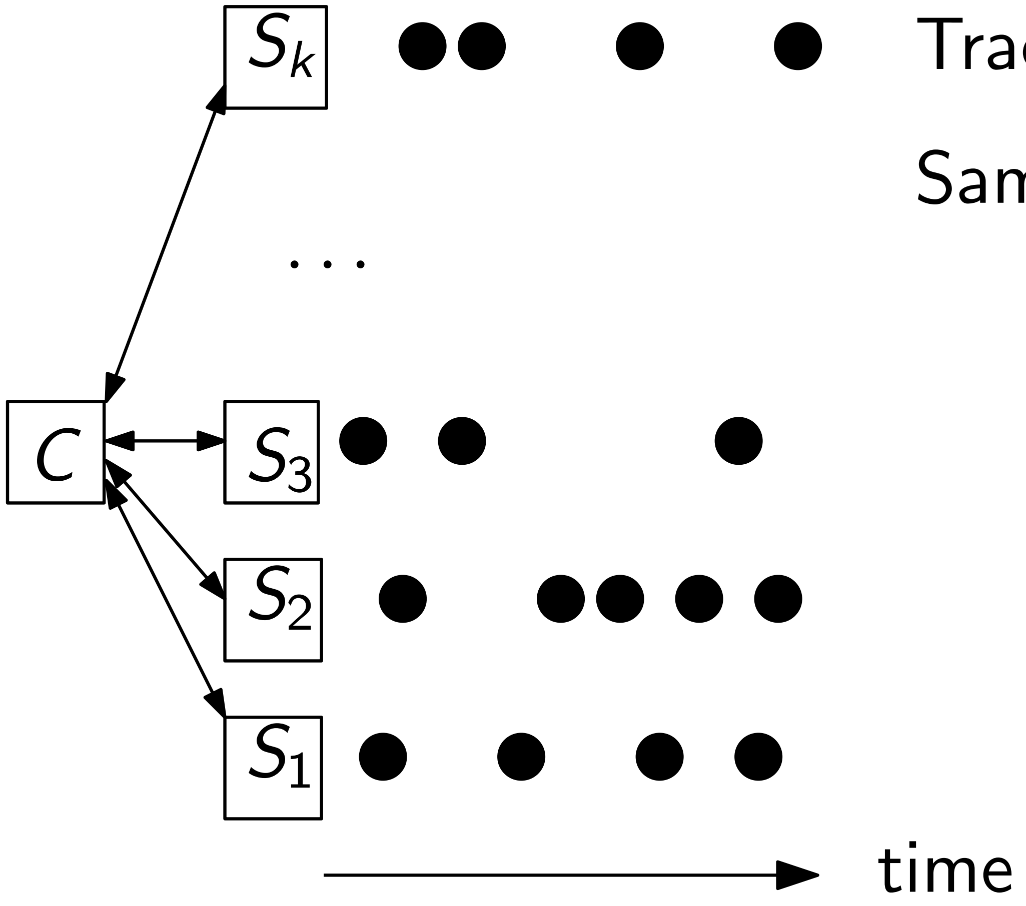
Reservoir Sampling from Distributed Streams

- When $k = 1$, reservoir sampling has cost $\Theta(s \log n)$
- When $k \geq 2$, reservoir sampling has cost $O(n)$ because it's costly to track i



Reservoir Sampling from Distributed Streams

- When $k = 1$, reservoir sampling has cost $\Theta(s \log n)$
- When $k \geq 2$, reservoir sampling has cost $O(n)$ because it's costly to track i

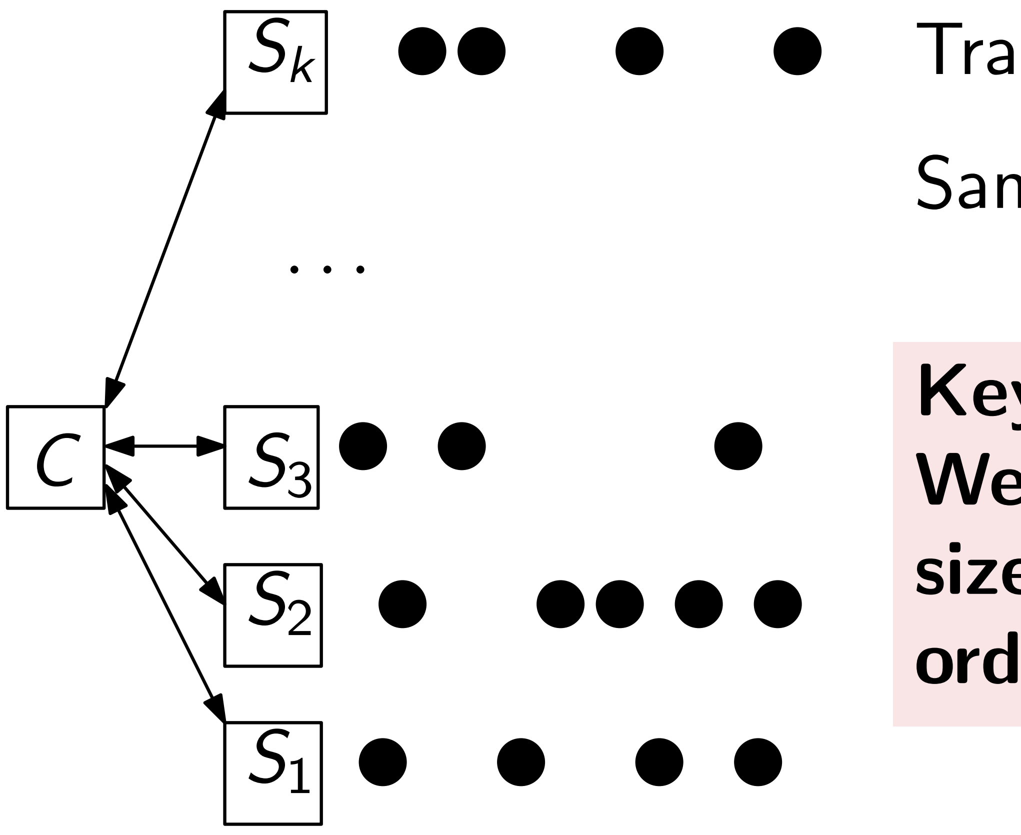


Tracking i approximately?

Sampling won't be uniform

Reservoir Sampling from Distributed Streams

- When $k = 1$, reservoir sampling has cost $\Theta(s \log n)$
- When $k \geq 2$, reservoir sampling has cost $O(n)$ because it's costly to track i



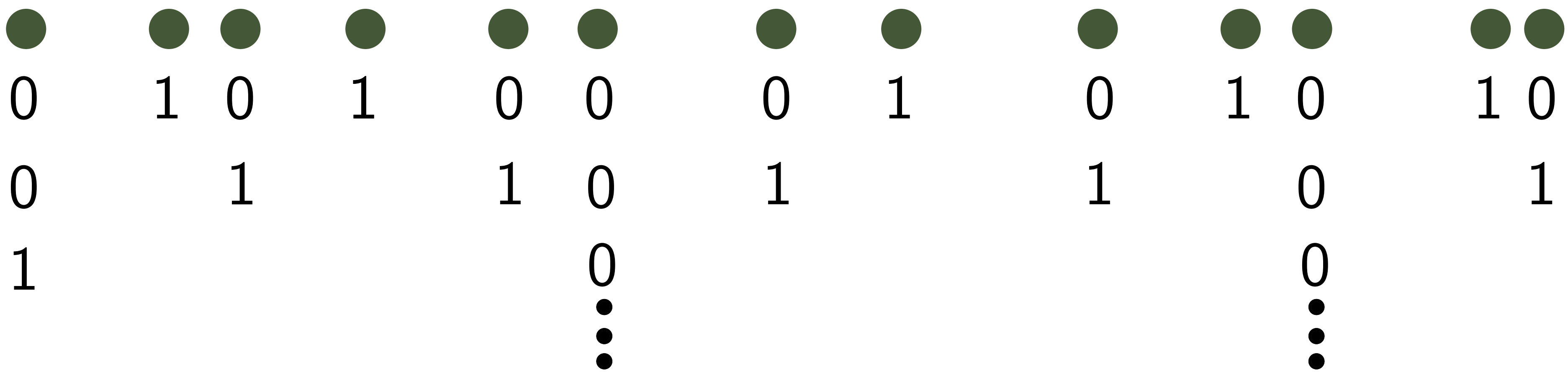
Tracking i approximately?

Sampling won't be uniform

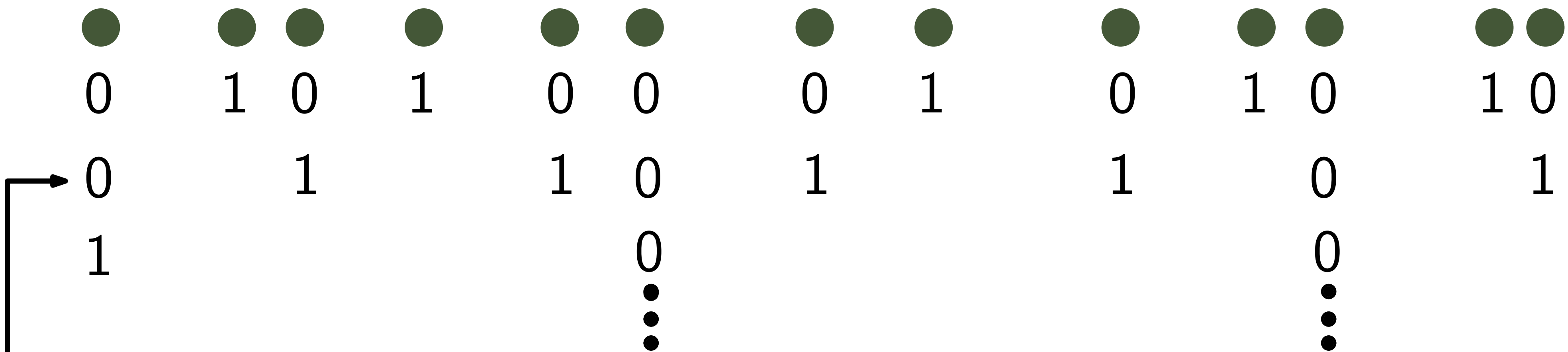
Key observation:
We don't have to know the size of the population in order to sample!

Basic Idea: Binary Bernoulli Sampling

Basic Idea: Binary Bernoulli Sampling

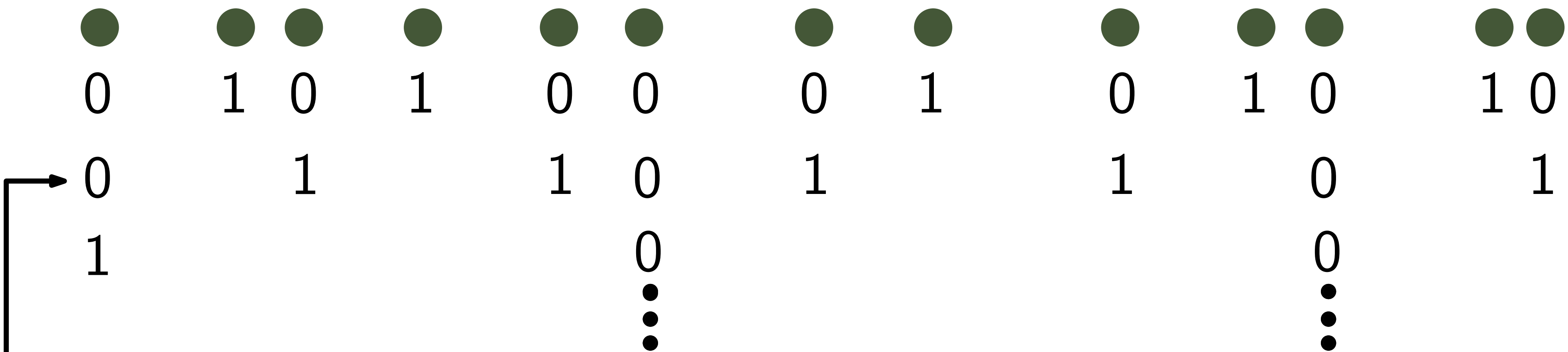


Basic Idea: Binary Bernoulli Sampling



Conditioned upon a row having $\geq s$ active items, we can draw a sample from the active items

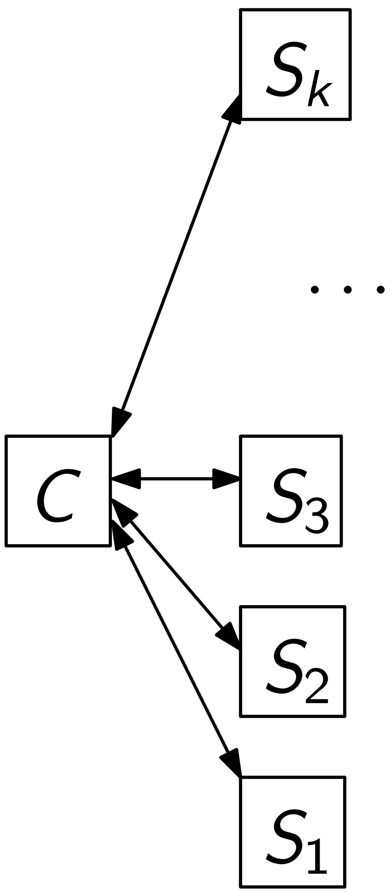
Basic Idea: Binary Bernoulli Sampling



Conditioned upon a row having $\geq s$ active items, we can draw a sample from the active items

The coordinator could maintain a Bernoulli sample of size between s and $O(s)$

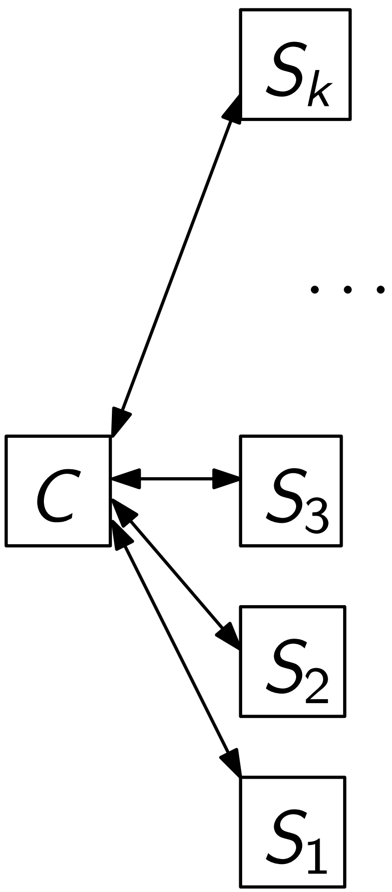
Sampling from Distributed Streams



- Initialize $i = 0$
- In round i:
  - Sites send in every item w.p. 2^{-i}

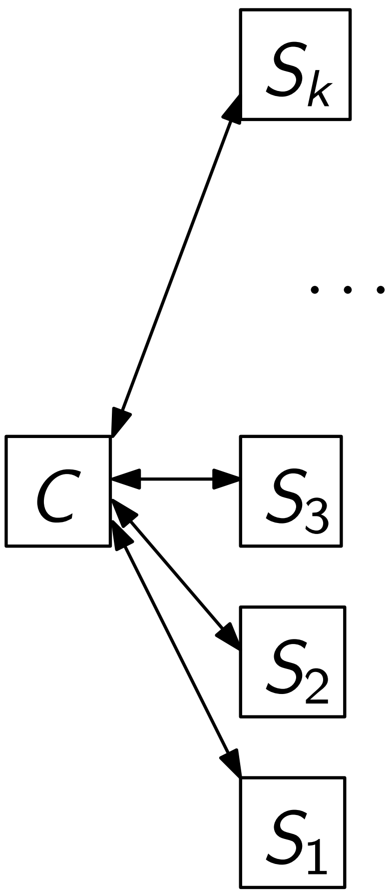
    (This is a Bernoulli sample with prob. 2^{-i})

Sampling from Distributed Streams



- Initialize $i = 0$
- In round i:
  - Sites send in every item w.p. 2^{-i}
    (This is a Bernoulli sample with prob. 2^{-i})
  - Coordinator maintains a lower sample and a higher sample: each received item goes to either with equal prob.
    (The lower sample is a sample with prob. 2^{-i-1})

Sampling from Distributed Streams



- Initialize $i = 0$
- In round i:
  - Sites send in every item w.p. 2^{-i}

    (This is a Bernoulli sample with prob. 2^{-i})
  - Coordinator maintains a lower sample and a higher sample: each received item goes to either with equal prob.

    (The lower sample is a sample with prob. 2^{-i-1})
  - When the lower sample reaches size s, the coordinator broadcasts to advance to round $i \leftarrow i + 1$

    Discard the upper sample

    Split the lower sample into a new lower sample and a higher sample

Sampling from Distributed Streams: Analysis

- Communication cost of round i: $O(k + s)$
  - Expect to receive $O(s)$ sampled items before round ends
  - Broadcast to end round: $O(k)$

[Cormode, Muthukrishnan, Yi, Zhang, PODS'10, JACM'12]

[Woodruff, Tirthapura, DISC'11]

Sampling from Distributed Streams: Analysis

- Communication cost of round i: $O(k + s)$
  - Expect to receive $O(s)$ sampled items before round ends
  - Broadcast to end round: $O(k)$
- Number of rounds: $O(\log(n/s))$
  - In round i, need $\Theta(s)$ items being sampled to end round
  - Each item has prob. 2^{-i} to contribute: need $\Theta(2^i s)$ items

[Cormode, Muthukrishnan, Yi, Zhang, PODS'10, JACM'12]

[Woodruff, Tirthapura, DISC'11]

Sampling from Distributed Streams: Analysis

- Communication cost of round i: $O(k+s)$
  - Expect to receive $O(s)$ sampled items before round ends
  - Broadcast to end round: $O(k)$
- Number of rounds: $O(\log(n/s))$
  - In round i, need $\Theta(s)$ items being sampled to end round
  - Each item has prob. 2^{-i} to contribute: need $\Theta(2^i s)$ items
- Communication: $O((k+s)\log n)$
  - Can be improved to $O(k \log_{k/s} n + s \log n)$
  - A matching lower bound

[Cormode, Muthukrishnan, Yi, Zhang, PODS'10, JACM'12]

[Woodruff, Tirthapura, DISC'11]

Problems

- The count-down problem
- Count-tracking
- Frequent items (heavy hitters)
- Random sampling
- Other problems

Other Results on Distributed Tracking

- Frequency moments
  - F_2: $\tilde{O}(k^2/\varepsilon^2 + k^{1.5}/\varepsilon^4)$ [Cormode, Muthukrishnan, Yi, SODA'08]
  - F_2: $\tilde{O}(k/\text{poly}(\varepsilon))$ [Woodruff, Zhang, STOC'12]

Other Results on Distributed Tracking

- Frequency moments
  - F_2: $\tilde{O}(k^2/\varepsilon^2 + k^{1.5}/\varepsilon^4)$ [Cormode, Muthukrishnan, Yi, SODA'08]
  - F_2: $\tilde{O}(k/\text{poly}(\varepsilon))$ [Woodruff, Zhang, STOC'12]
  - F_2: $\tilde{\Omega}(k/\varepsilon^2)$ [Woodruff, Zhang, STOC'12]

Other Results on Distributed Tracking

- Frequency moments
  - F_2: $\tilde{O}(k^2/\varepsilon^2 + k^{1.5}/\varepsilon^4)$ [Cormode, Muthukrishnan, Yi, SODA'08]
  - F_2: $\tilde{O}(k/\text{poly}(\varepsilon))$ [Woodruff, Zhang, STOC'12]
  - F_2: $\tilde{\Omega}(k/\varepsilon^2)$ [Woodruff, Zhang, STOC'12]
  - $F_p, p > 1$: $\tilde{\Theta}(k^{p-1}/\text{poly}(\varepsilon))$ [Woodruff, Zhang, STOC'12]

Other Results on Distributed Tracking

- Frequency moments
  - F_2: $\tilde{O}(k^2/\varepsilon^2 + k^{1.5}/\varepsilon^4)$ [Cormode, Muthukrishnan, Yi, SODA'08]
  - F_2: $\tilde{O}(k/\text{poly}(\varepsilon))$ [Woodruff, Zhang, STOC'12]
  - F_2: $\tilde{\Omega}(k/\varepsilon^2)$ [Woodruff, Zhang, STOC'12]
  - $F_p, p > 1$: $\tilde{\Theta}(k^{p-1}/\text{poly}(\varepsilon))$ [Woodruff, Zhang, STOC'12]
  - F_0 (distinct count): $\tilde{\Theta}(k/\varepsilon^2)$ [Woodruff, Zhang, STOC'12]

Other Results on Distributed Tracking

- Entropy [Arackaparambil, Brody, Chakrabarti, ICALP'08]
- Heavy hitters and quantiles [Yi, Zhang, PODS'09]
  [Huang, Yi, Zhang, PODS'12]
- Sliding windows [Chan, Lam, Lee, Ting, STACS'10]
  [Cormode, Yi, SSDBM'12]

Open Problems

- Any streaming problem
  - Histograms, clustering, graph problems, geometric problems, ...

Open Problems

- Any streaming problem
  - Histograms, clustering, graph problems, geometric problems, ...
- Does it have to be streaming?
  - If we don't care about space ...

Open Problems

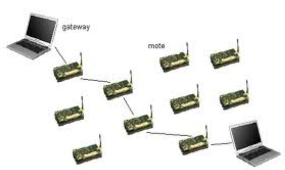
- Any streaming problem
  - Histograms, clustering, graph problems, geometric problems, ...
- Does it have to be streaming?
  - If we don't care about space ...
  - Even if we care about space... streaming lower bounds do not apply!

Open Problems

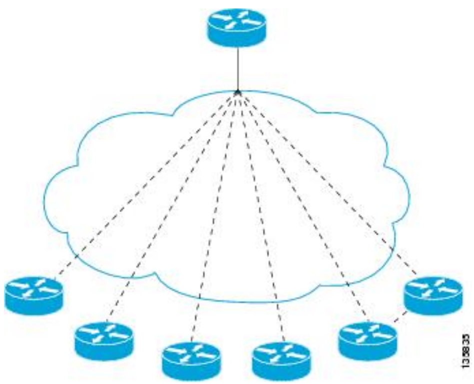
- Any streaming problem
  - Histograms, clustering, graph problems, geometric problems, ...
- Does it have to be streaming?
  - If we don't care about space ...
  - Even if we care about space... streaming lower bounds do not apply!
- How to model deletions?
  - Competitive analysis? [Yi, Zhang, SODA'09]

The Greater Picture: Distributed Tracking/Monitoring

- Motivated by database/networking applications
  - Adaptive filters [Olston, Jiang, Widom, SIGMOD'03]
  - A generic geometric approach [Scharfman et al. SIGMOD'06]
  - Prediction models [Cormode, Garofalakis, Muthukrishnan, Rastogi, SIGMOD'05]



sensor networks



network monitoring

environment monitoring

cloud computing

Thank you!