Optimal Sampling from Distributed Streams

Ke Yi

Hong Kong University of Science and Technology

Joint work with Graham Cormode (AT&T)
S. Muthukrishnan (Rutgers)
Qin Zhang (HKUST)

To appear in PODS'10

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 [Waterman '??; Vitter '85]

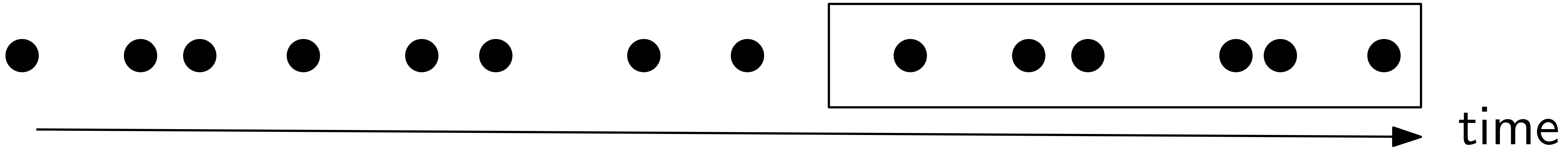
- Maintain a (uniform) sample (w/o replacement) of size s from a stream of n items
  - Every subset of size s has equal probability to be the sample
- When the i-th item arrives
  - With probability s/i, use it to replace an item in the current sample chosen uniformly at ranfom
  - With probability $1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

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
- Correctness: intuitive
- Space: $O(s)$, time $O(1)$

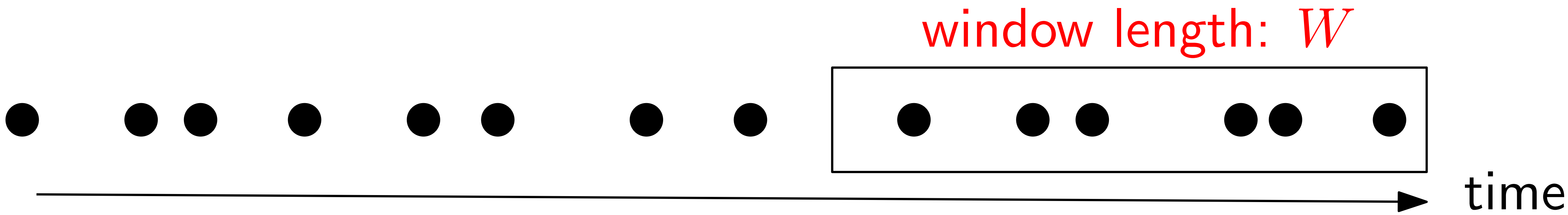
Sampling from a sliding window

[Babcock, Datar, Motwani, SODA'02; Gemulla, Lehner, SIGMOD'08; Braverman, Ostrovsky, Zaniolo, PODS'09]



Sampling from a sliding window

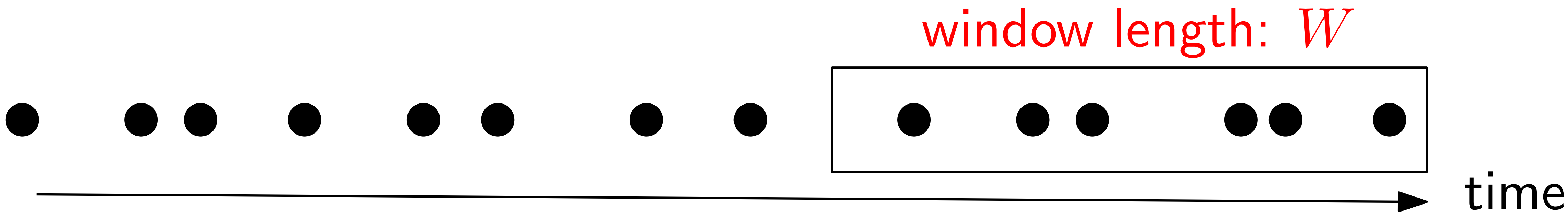
[Babcock, Datar, Motwani, SODA'02; Gemulla, Lehner, SIGMOD'08; Braverman, Ostrovsky, Zaniolo, PODS'09]



Time based window and sequence based window

Sampling from a sliding window

[Babcock, Datar, Motwani, SODA'02; Gemulla, Lehner, SIGMOD'08; Braverman, Ostrovsky, Zaniolo, PODS'09]

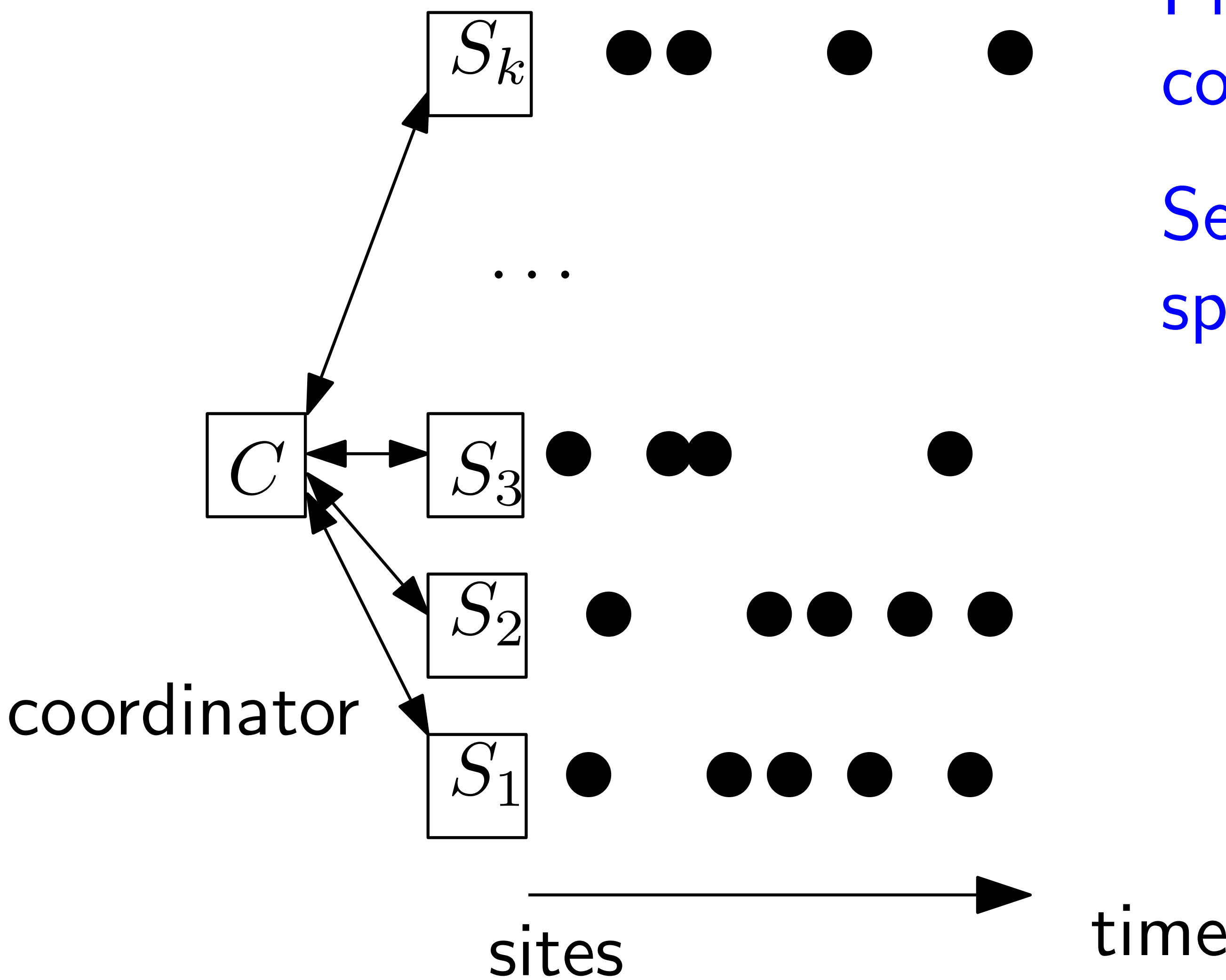


Time based window and sequence based window

- Space: $\Theta(s \log w)$
  - w: number of items in the sliding window
- Time: $\Theta(\log w)$

Sampling from distributed streams

- Maintain a (uniform) sample (w/o replacement) of size s from k streams of a total of n items

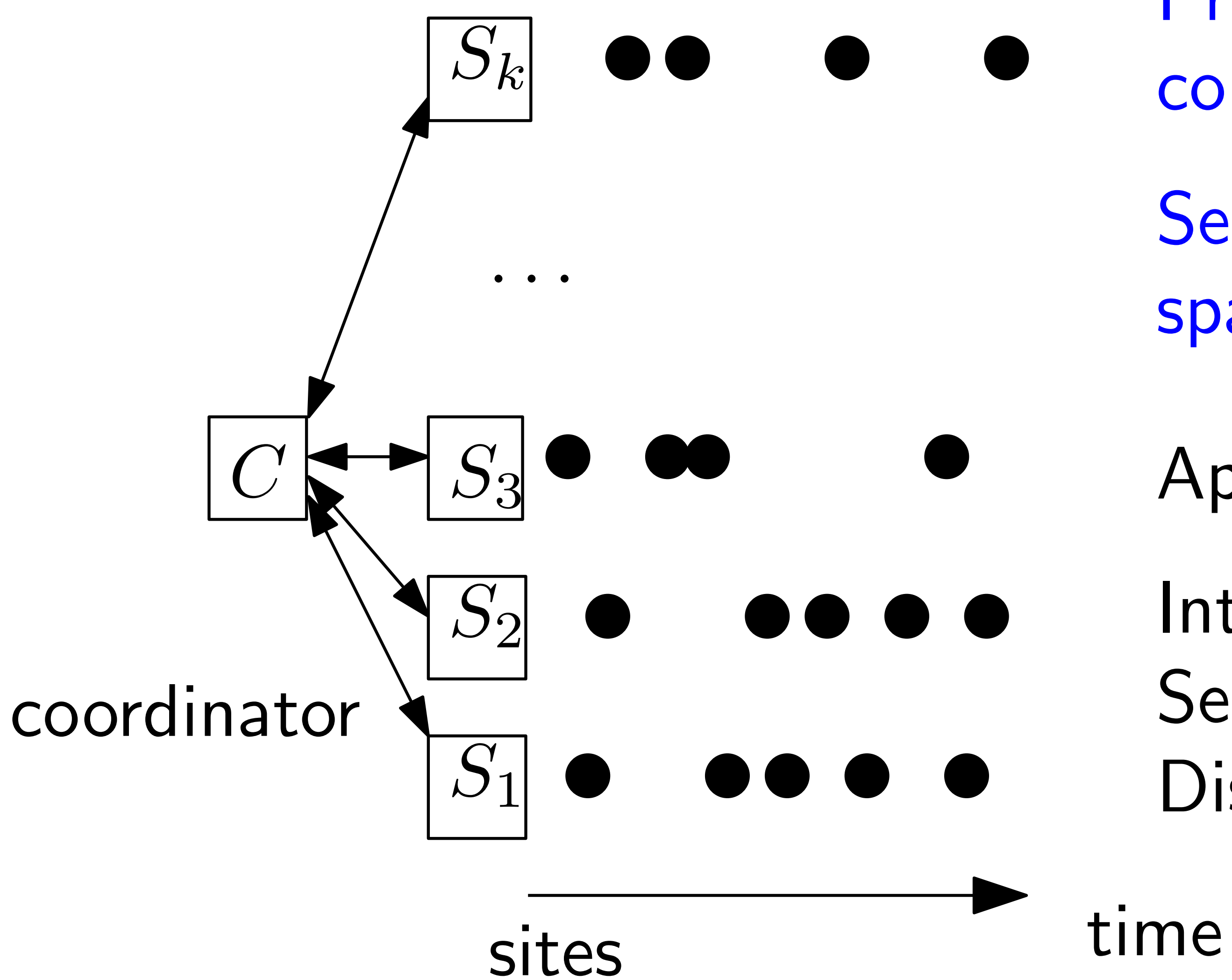


Primary goal:
communication

Secondary goal:
space/time at coordinator/site

Sampling from distributed streams

- Maintain a (uniform) sample (w/o replacement) of size s from k streams of a total of n items



Primary goal:
communication

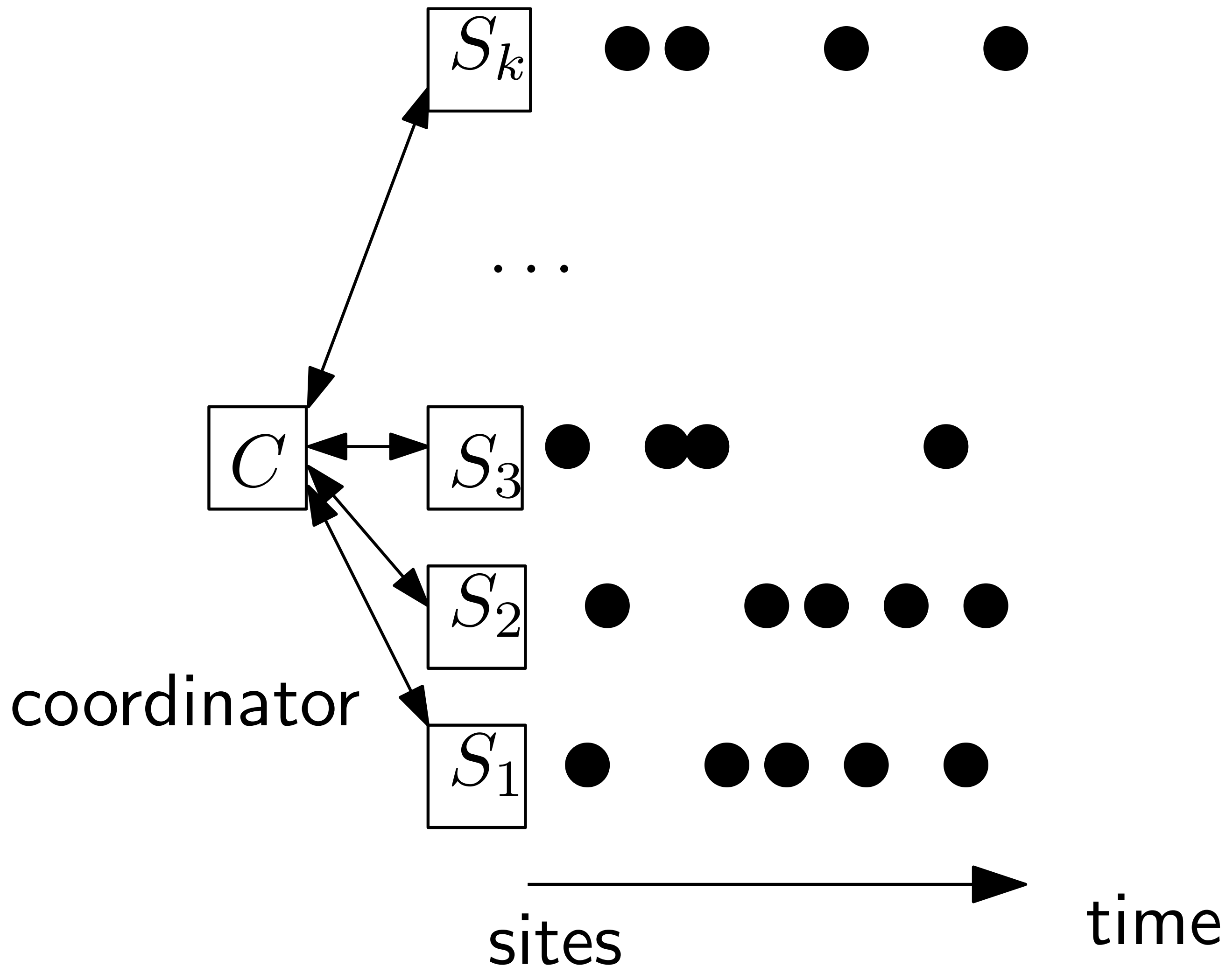
Secondary goal:
space/time at coordinator/site

Applications:

Internet routers
Sensor networks
Distributed computing

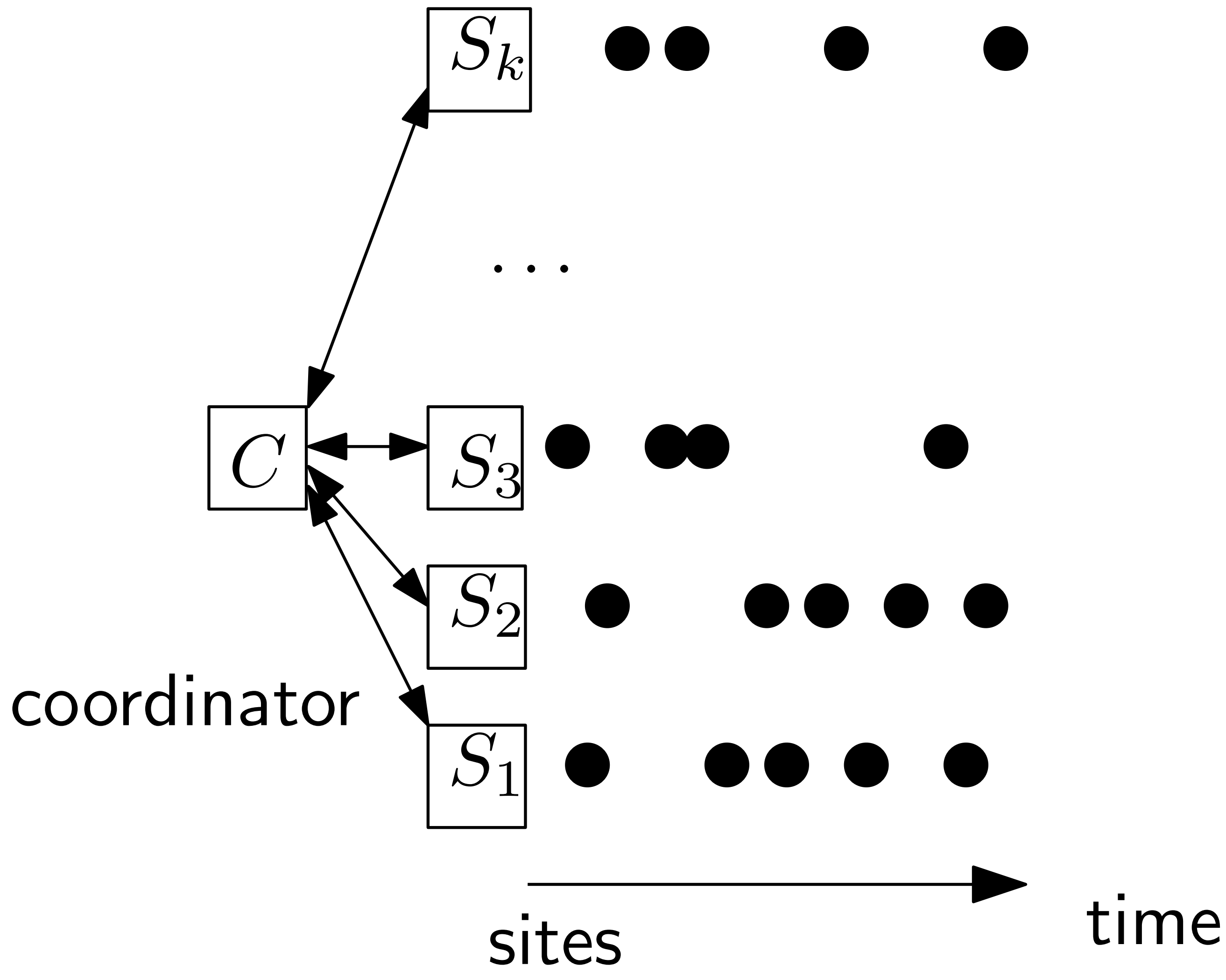
Why existing solutions don't work

- When $k = 1$, reservoir sampling has communication $\Theta(s \log n)$



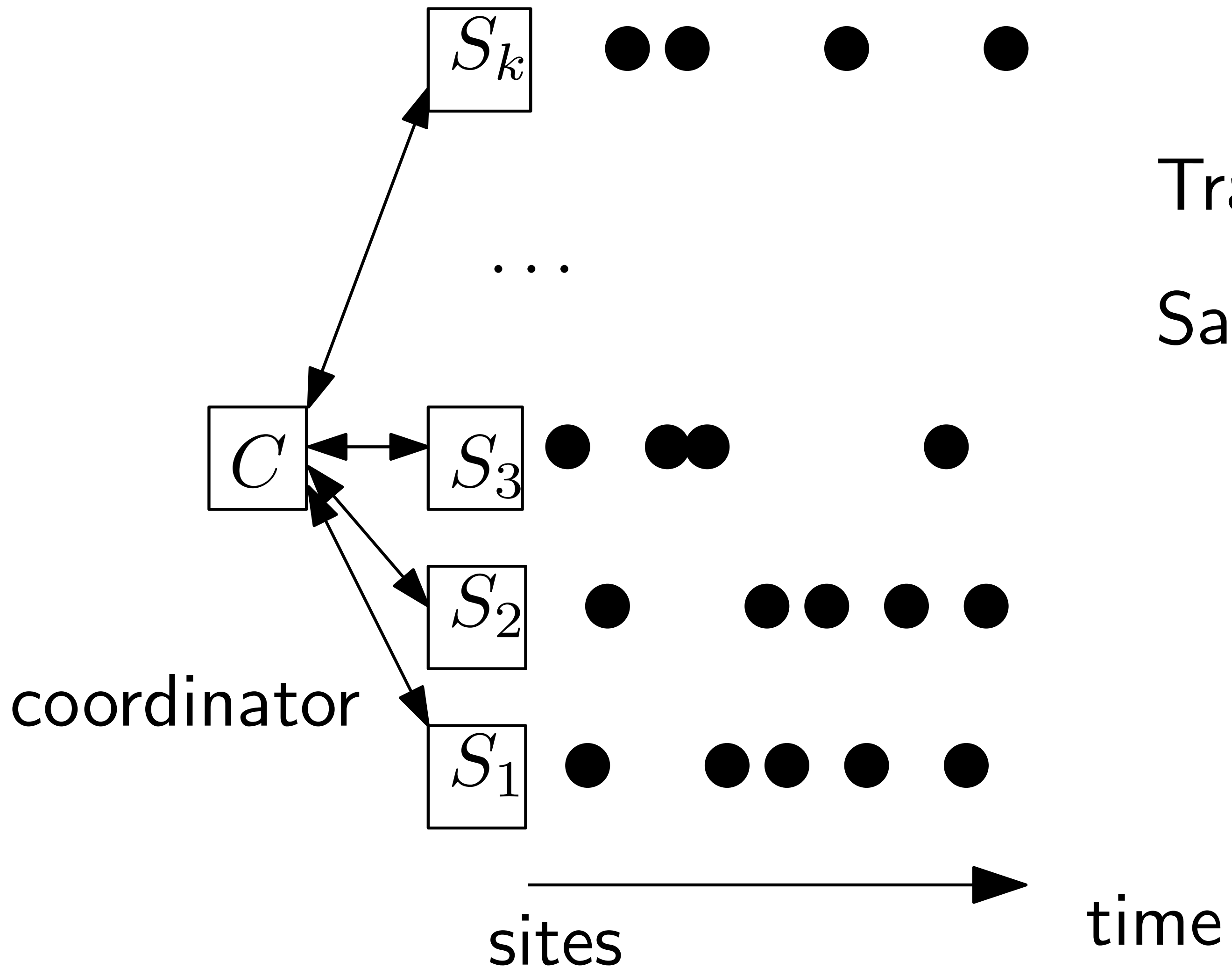
Why existing solutions don't work

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



Why existing solutions don't work

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

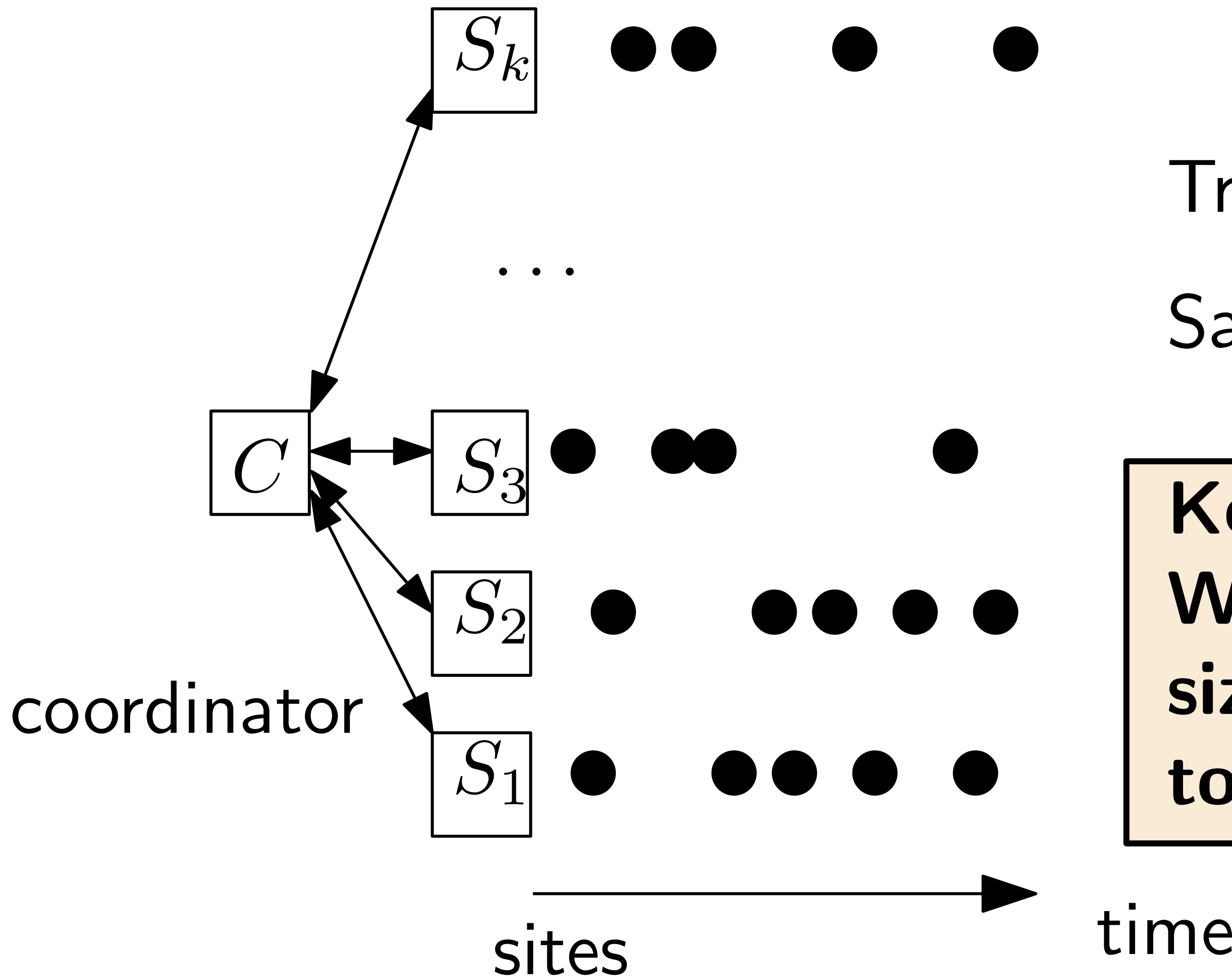


Tracking i approximately?

Sampling won't be uniform

Why existing solutions don't work

- When $k = 1$, reservoir sampling has communication $\Theta(s \log n)$
- 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!

Previous results on distributed streaming

- A lot of heuristics in the database/networking literature
  - But random sampling has not been studied, even heuristically

Previous results on distributed streaming

- A lot of heuristics in the database/networking literature
  - But random sampling has not been studied, even heuristically
- Threshold monitoring, frequency moments [Cormode, Muthukrishnan, Yi, SODA'08]
- Entropy [Arackaparambil, Brody, Chakrabarti, ICALP'08]
- Heavy hitters and quantiles [Yi, Zhang, PODS'09]
- Basic counting, heavy hitters, quantiles in sliding windows [Chan, Lam, Lee, Ting, STACS'10]

Previous results on distributed streaming

- A lot of heuristics in the database/networking literature
  - But random sampling has not been studied, even heuristically
- Threshold monitoring, frequency moments [Cormode, Muthukrishnan, Yi, SODA'08]
- Entropy [Arackaparambil, Brody, Chakrabarti, ICALP'08]
- Heavy hitters and quantiles [Yi, Zhang, PODS'09]
- Basic counting, heavy hitters, quantiles in sliding windows [Chan, Lam, Lee, Ting, STACS'10]

- All of them are deterministic algorithms, or use randomized sketches as black boxes

Our results on random sampling

window	upper bounds	lower bounds
infinite	$O((k+s)\log n)$	$\Omega(k+s\log n)$
sequence-based	$O(ks\log(w/s))$	$\Omega(ks\log(w/ks))$
time-based	$O((k+s)\log w)$	$\Omega(k+s\log w)$

(per window)

Our results on random sampling

window	upper bounds	lower bounds
infinite	$O((k+s)\log n)$	$\Omega(k+s\log n)$
sequence-based	$O(ks\log(w/s))$	$\Omega(ks\log(w/ks))$
time-based	$O((k+s)\log w)$	$\Omega(k+s\log w)$
	(per window)	

- Applications
  - Heavy hitters and quantiles can be tracked in $\tilde{O}(k+1/\epsilon^2)$
    Beats deterministic bound $\tilde{\Theta}(k/\epsilon)$ for $k \gg 1/\epsilon$
  - Also for sliding windows

Our results on random sampling

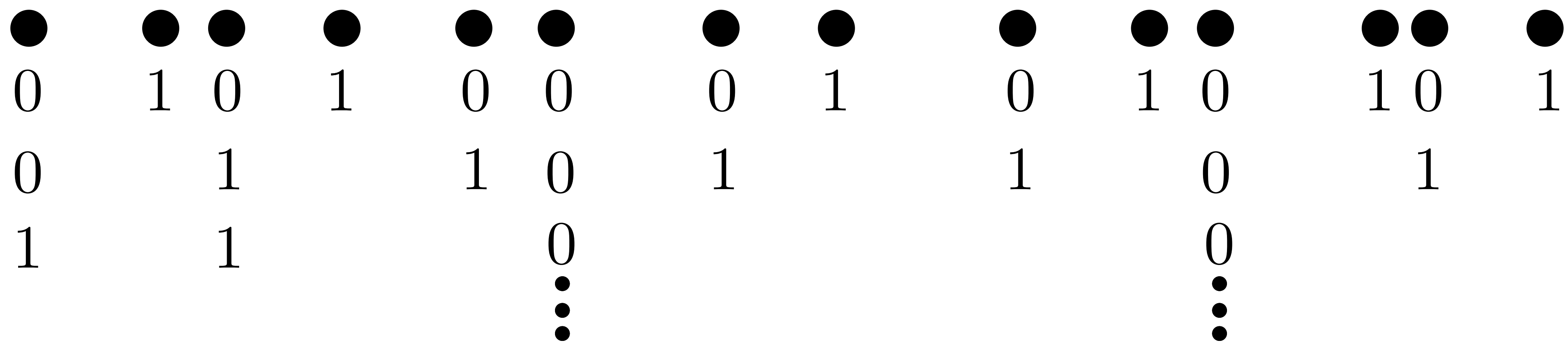
window	upper bounds	lower bounds
infinite	$O((k+s)\log n)$	$\Omega(k+s\log n)$
sequence-based	$O(ks\log(w/s))$	$\Omega(ks\log(w/ks))$
time-based	$O((k+s)\log w)$	$\Omega(k+s\log w)$

(per window)

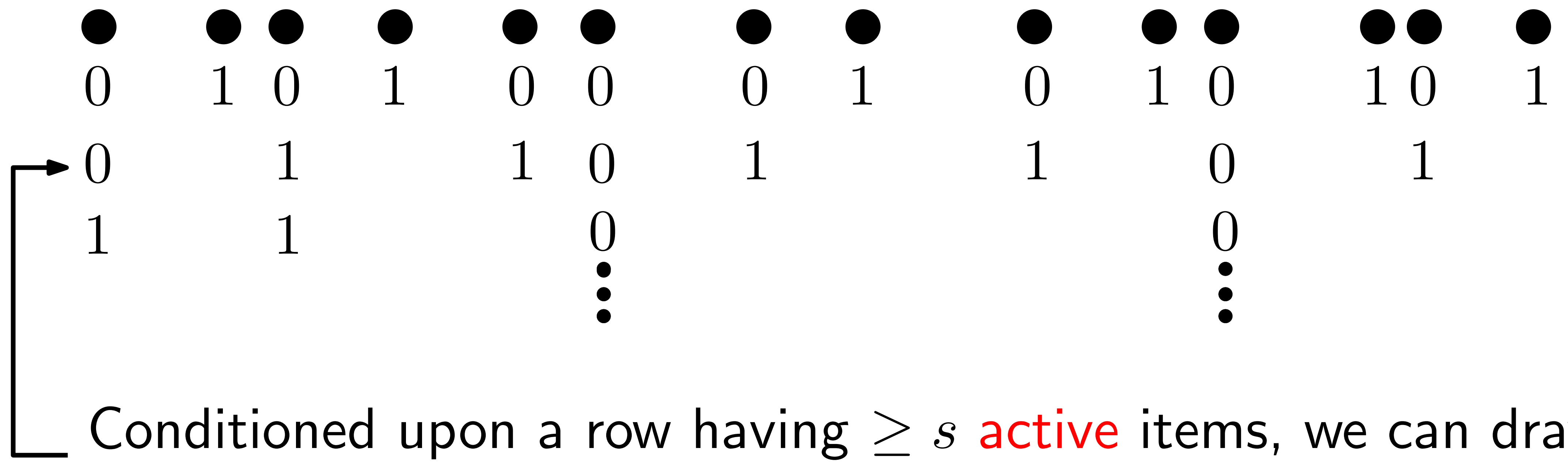
- Applications
  - Heavy hitters and quantiles can be tracked in $\tilde{O}(k+1/\epsilon^2)$
    Beats deterministic bound $\tilde{\Theta}(k/\epsilon)$ for $k \gg 1/\epsilon$
  - Also for sliding windows
  - ϵ-approximations in bounded VC dimensions: $\tilde{O}(k+1/\epsilon^2)$
  - ϵ-nets: $\tilde{O}(k+1/\epsilon)$
  - $\cdots$

The basic idea: Binary Bernoulli sampling

The basic idea: Binary Bernoulli sampling

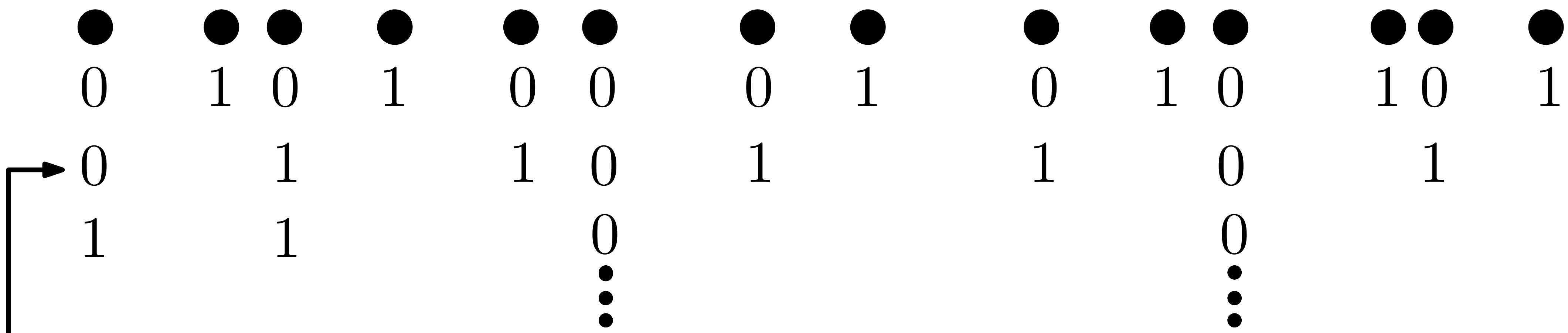


The basic idea: Binary Bernoulli sampling



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

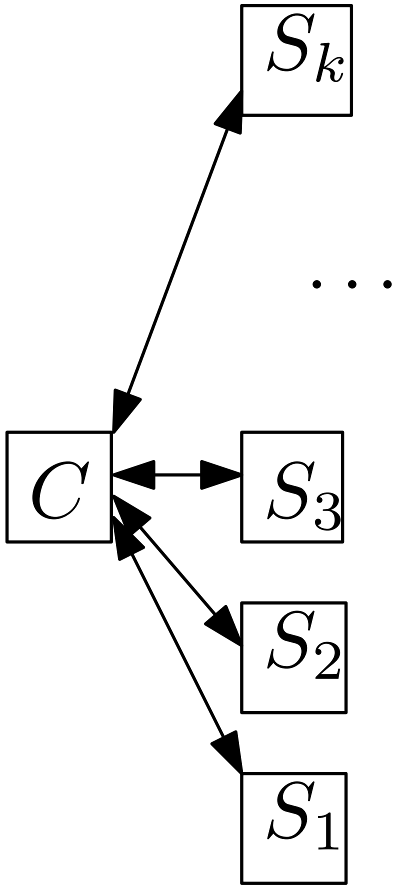
The basic idea: Binary Bernoulli sampling



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 $O(s)$

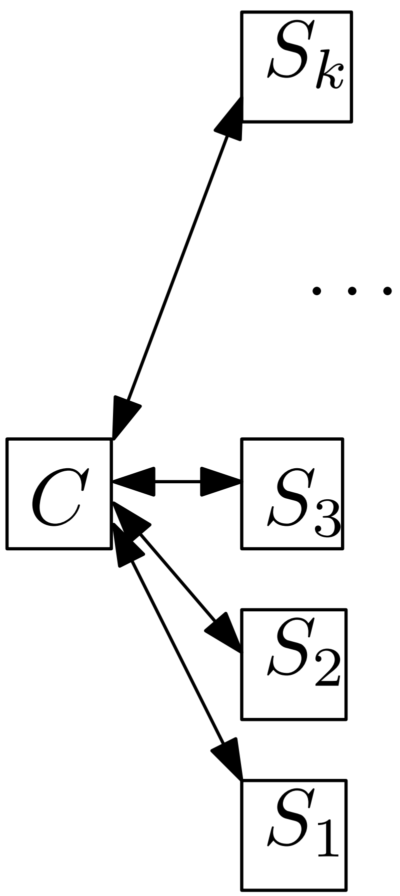
Sampling from an infinite window



- 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 an infinite window

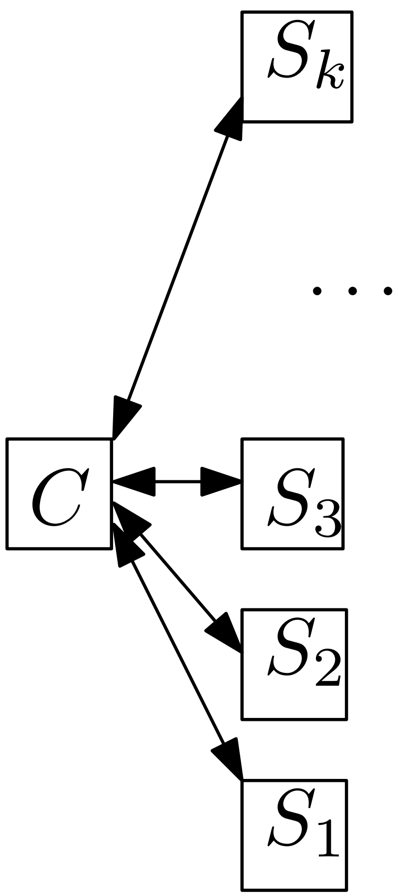


- Initialize $i = 0$
- In round i:
  - Sites send in every item w.p. 2^{-i}

    (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 Bernoulli sample with prob. 2^{-i-1})

Sampling from an infinite window



- 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 Bernoulli 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 an infinite window: Analysis

- Communication cost of round i: $O(k + s)$
  - Coordinator maintains a lower sample and a higher sample: each received item goes to either with equal prob.

    Expect to receive $O(s)$ sampled items before round ends

Sampling from an infinite window: Analysis

- Communication cost of round i: $O(k+s)$
  - Coordinator maintains a lower sample and a higher sample: each received item goes to either with equal prob.

    Expect to receive $O(s)$ sampled items before round ends
  - Broadcast to end round: $O(k)$

Sampling from an infinite window: Analysis

- Communication cost of round i: $O(k+s)$
  - Coordinator maintains a lower sample and a higher sample: each received item goes to either with equal prob.

    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

Sampling from an infinite window: Analysis

- Communication cost of round i: $O(k+s)$
  - Coordinator maintains a lower sample and a higher sample: each received item goes to either with equal prob.

    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: $O((k+s)\log n)$
  - Lower bound: $\Omega(k + s\log n)$

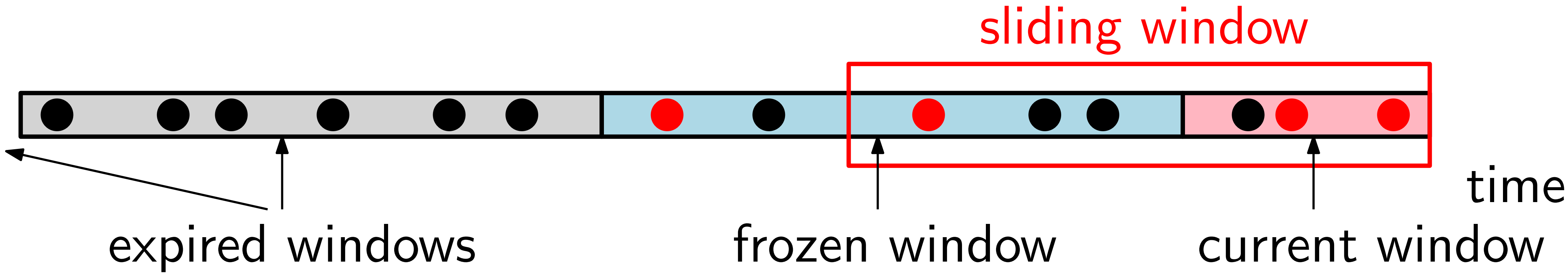
Sampling from an infinite window: Analysis

- Communication cost of round i: $O(k+s)$
  - Coordinator maintains a lower sample and a higher sample: each received item goes to either with equal prob.

    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: $O((k+s)\log n)$
  - Lower bound: $\Omega(k + s\log n)$
- Site space: $O(1)$, time: $O(1)$

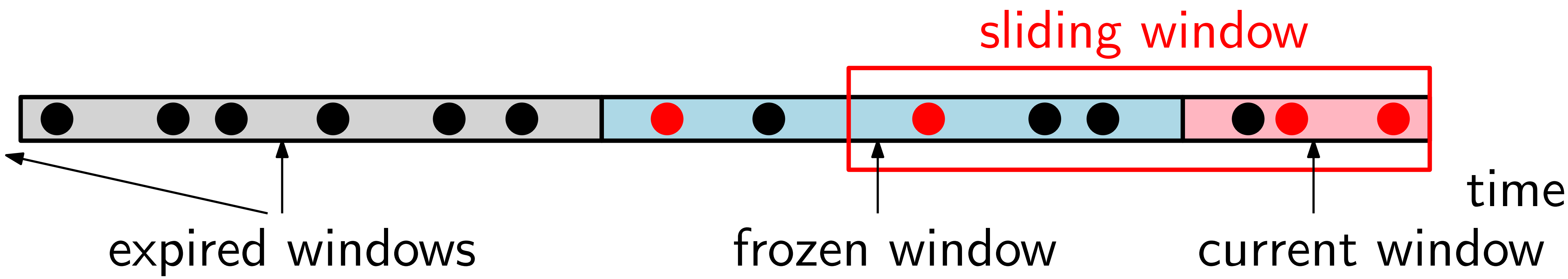
  Coordinator space: $O(s)$, total time: $O((k+s)\log n)$

Sampling from a sliding window: Idea



Sample for sliding window =
a subsample of the (unexpired) sample of frozen window +
a subsample of the sample of current window

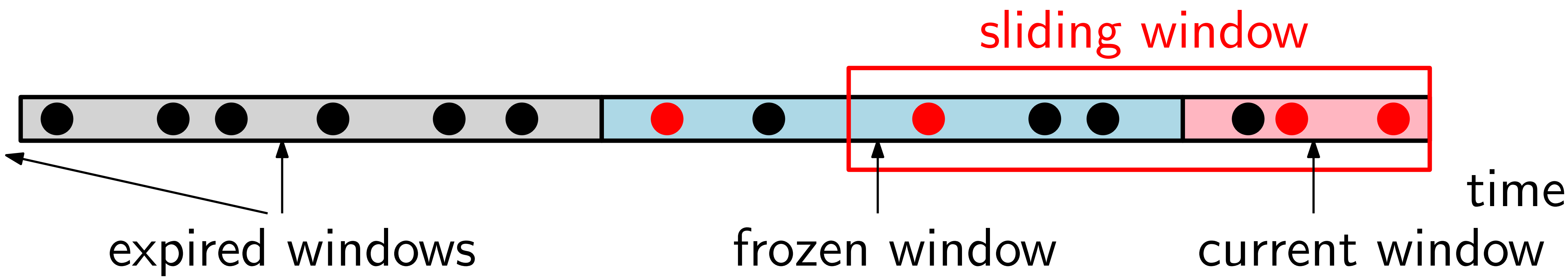
Sampling from a sliding window: Idea



Sample for sliding window =
a subsample of the (unexpired) sample of frozen window +
a subsample of the sample of current window

Key: As long as either Bernoulli sample has size $\geq s$, we can subsample the sample with the larger probability to match up their probabilities

Sampling from a sliding window: Idea

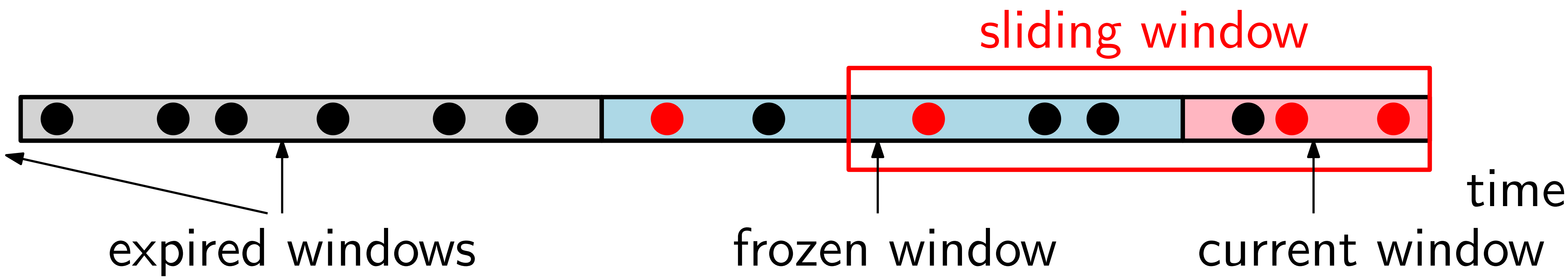


Sample for sliding window $=$
a subsample of the (unexpired) sample of frozen window $+$
a subsample of the sample of current window

Key: As long as either Bernoulli sample has size $\geq s$, we can subsample the sample with the larger probability to match up their probabilities

- Current window: Run our infinite-window algorithm
  - A Bernoulli sample with prob. 2^{-i} such that size $\geq s$

Sampling from a sliding window: Idea

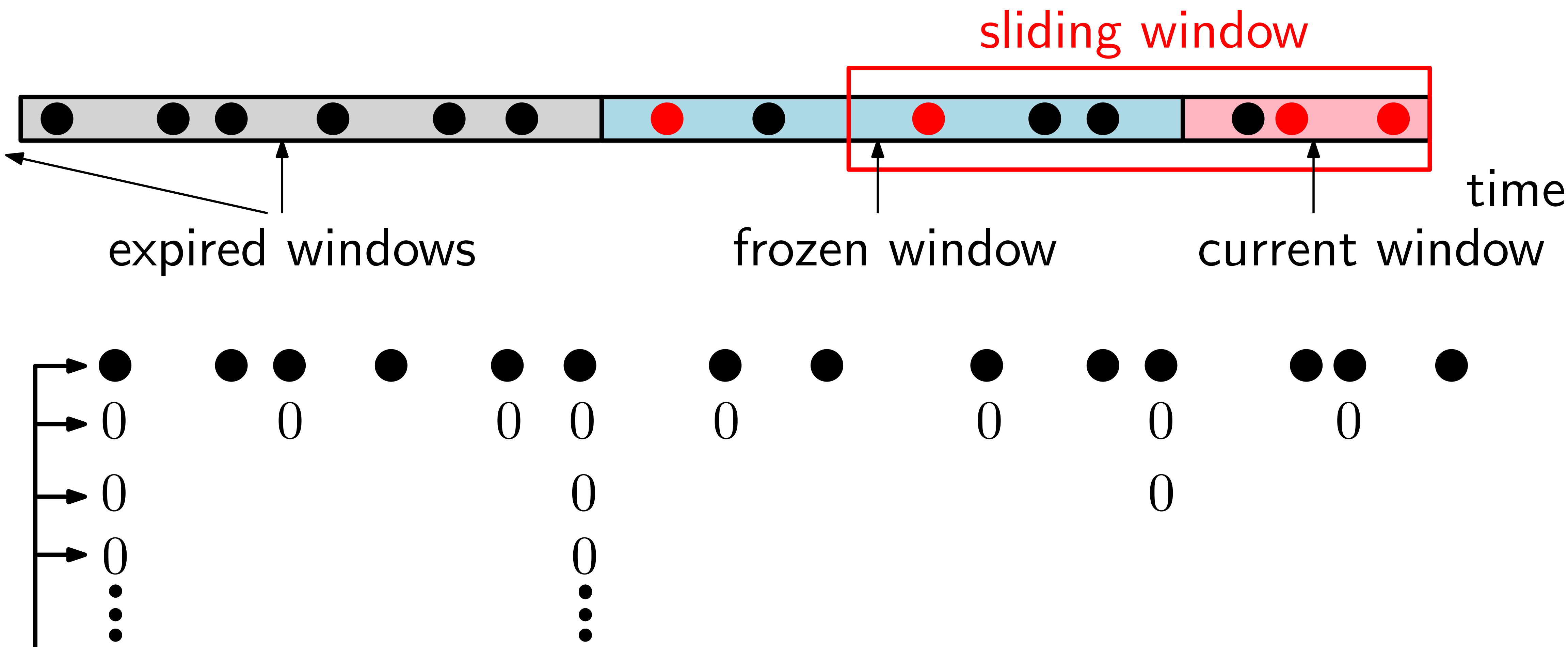


Sample for sliding window $=$
a subsample of the (unexpired) sample of frozen window $+$
a subsample of the sample of current window

Key: As long as either Bernoulli sample has size $\geq s$, we can subsample the sample with the larger probability to match up their probabilities

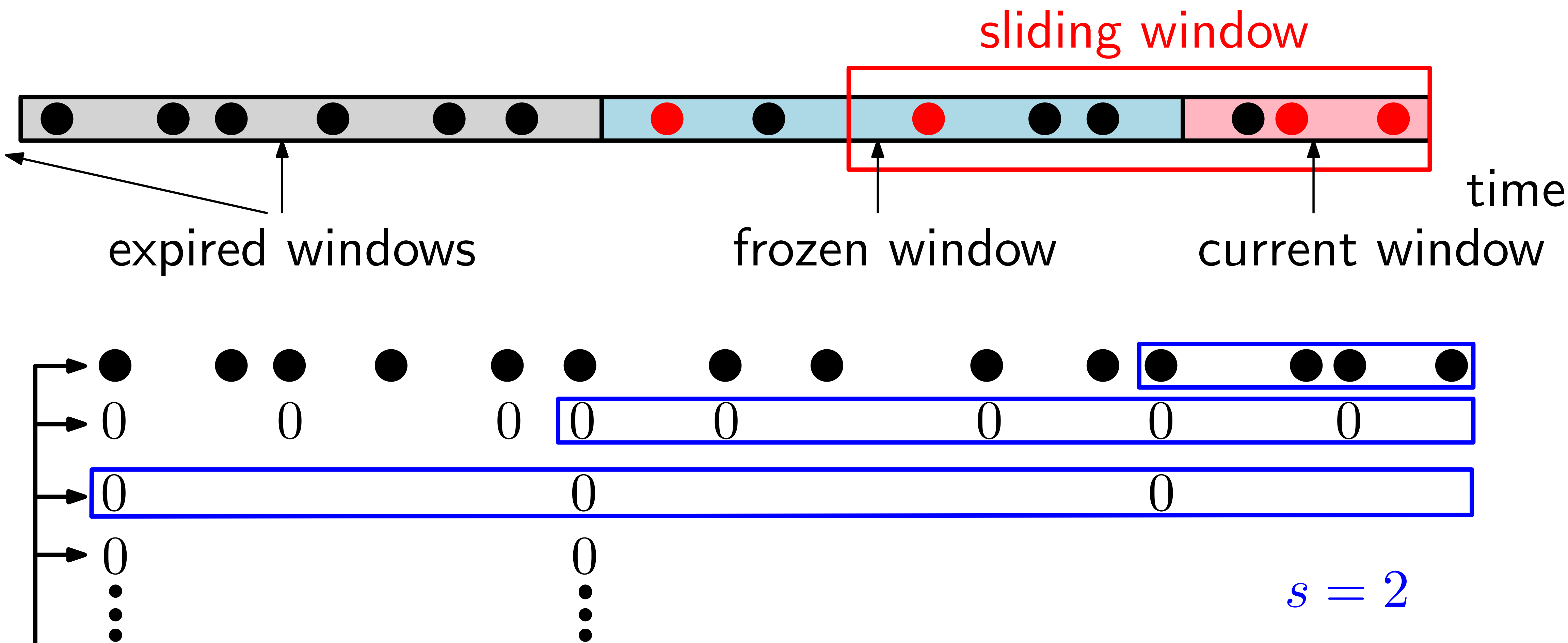
- Current window: Run our infinite-window algorithm
  - A Bernoulli sample with prob. 2^{-i} such that size $\geq s$
- Frozen window: Need to have the same

Dealing with the frozen window



Keep all the levels? Need $O(w)$ communication

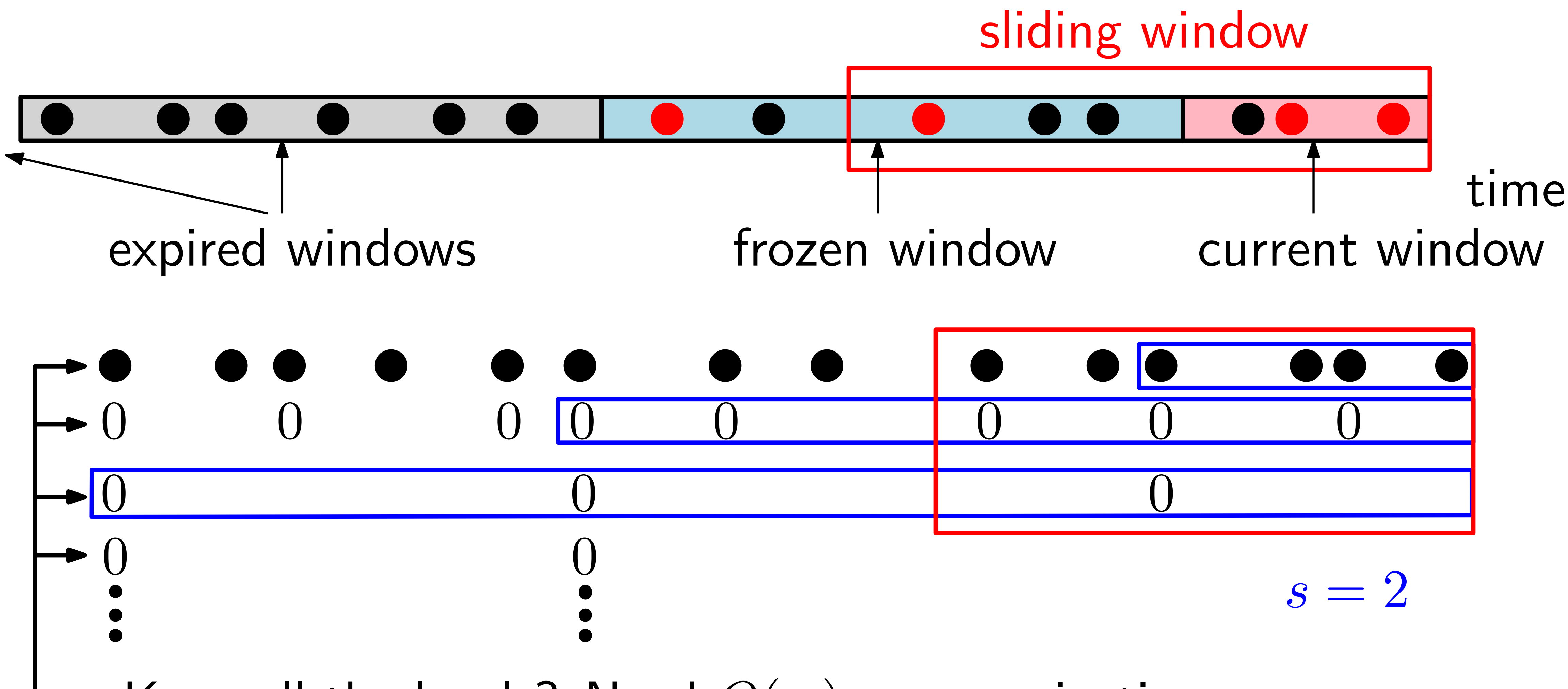
Dealing with the frozen window



Keep all the levels? Need $O(w)$ communication

Keep most recent sampled items in a level until s of them are also sampled at the next level. Total size: $O(s \log w)$

Dealing with the frozen window

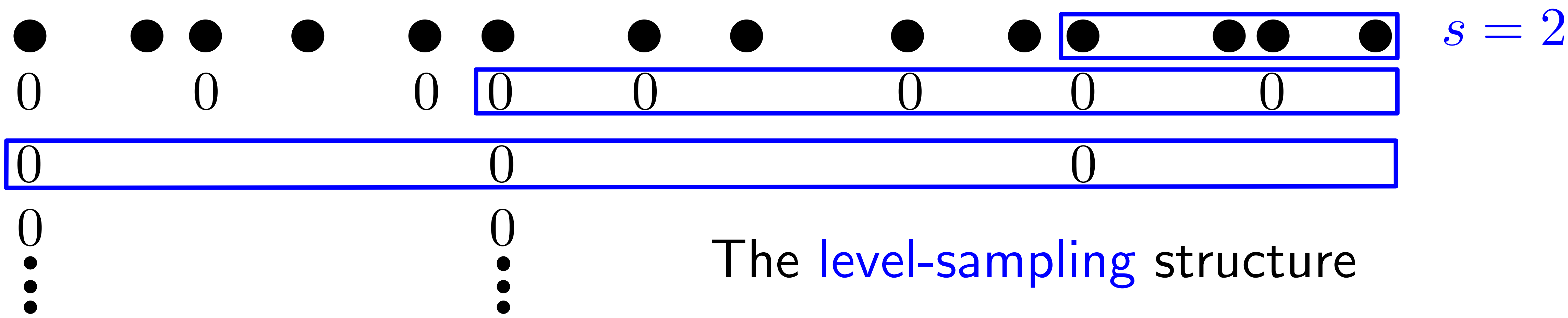


Keep all the levels? Need $O(w)$ communication

Keep most recent sampled items in a level until s of them are also sampled at the next level. Total size: $O(s \log w)$

Guaranteed: There is a blue window with $\geq s$ sampled items that covers the unexpired portion of the frozen window

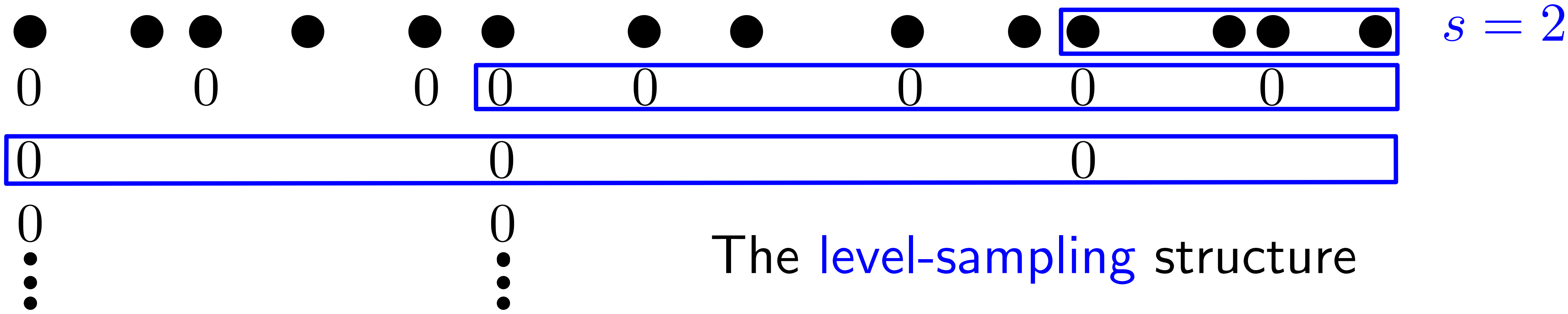
Dealing with the frozen window: The algorithm



The level-sampling structure

- Each site builds its own level-sampling structure for the current window until it freezes
  - Needs $O(s \log w)$ space and $O(1)$ time per item
  - Necessary unless communication is $\Omega(w)$

Dealing with the frozen window: The algorithm



The level-sampling structure

- Each site builds its own level-sampling structure for the current window until it freezes
  - Needs $O(s \log w)$ space and $O(1)$ time per item
  - Necessary unless communication is $\Omega(w)$
- When the current window freezes
  - For each level, do a k-way merge to build the level of the global structure at the coordinator
    Total communication $O((k + s) \log w)$

Future directions

- Applications
  - Heavy hitters and quantiles can be tracked in $\tilde{O}(k + 1/\epsilon^2)$
    Beats deterministic bound $\tilde{\Theta}(k/\epsilon)$ for $k \gg 1/\epsilon$
  - Also for sliding windows
  - ϵ-approximations in bounded VC dimensions: $\tilde{O}(k + 1/\epsilon^2)$
  - ϵ-nets: $\tilde{O}(k + 1/\epsilon)$
  - $\cdots$

Future directions

- Applications
  - Heavy hitters and quantiles can be tracked in $\tilde{O}(k + 1/\epsilon^2)$
    Beats deterministic bound $\tilde{\Theta}(k/\epsilon)$ for $k \gg 1/\epsilon$
  - Also for sliding windows
  - ϵ-approximations in bounded VC dimensions: $\tilde{O}(k + 1/\epsilon^2)$
  - ϵ-nets: $\tilde{O}(k + 1/\epsilon)$
  - $\cdots$
- Is random sampling the best way to solve these problems?

Future directions

- Applications
  - Heavy hitters and quantiles can be tracked in $\tilde{O}(k + 1/\epsilon^2)$
    Beats deterministic bound $\tilde{\Theta}(k/\epsilon)$ for $k \gg 1/\epsilon$
  - Also for sliding windows
  - ϵ-approximations in bounded VC dimensions: $\tilde{O}(k + 1/\epsilon^2)$
  - ϵ-nets: $\tilde{O}(k + 1/\epsilon)$
  - $\cdots$
- Is random sampling the best way to solve these problems?
  - New result: Heavy hitters and quantiles can be tracked in $\tilde{O}(k + \sqrt{k}/\epsilon)$, using a different sampling method

The End

THANK YOU

Q and A