Balanced Allocations with Incomplete Information: The Power of Two Queries

Dimitrios Los[1], Thomas Sauerwald[1]

[1]University of Cambridge, UK

Balanced allocations: Background

Balanced allocations setting

Allocate m tasks (balls) sequentially into n machines (bins).

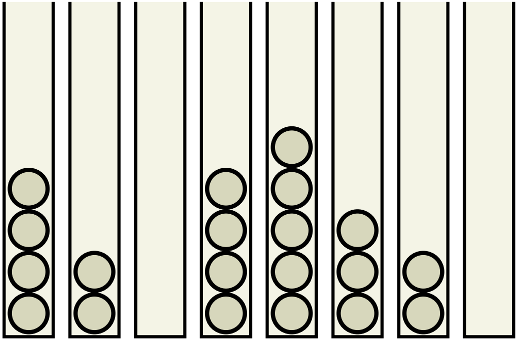
Balanced allocations setting

Allocate m tasks (balls) sequentially into n machines (bins).
Goal: minimise the **maximum load** $\max_{i \in [n]} x_i^m$, where x^t is the load vector after ball t.

Balanced allocations setting

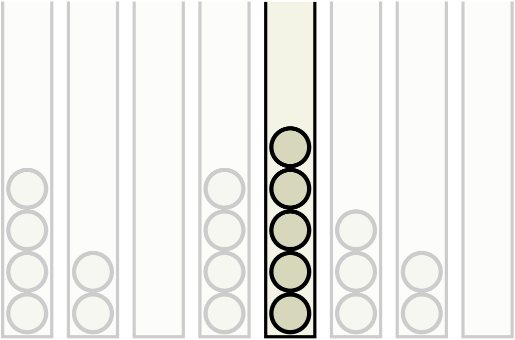
Allocate m tasks (balls) sequentially into n machines (bins).

<u>**Goal:**</u> minimise the **maximum load** $\max_{i \in [n]} x_i^m$, where x^t is the load vector after ball t.



Balanced allocations setting

Allocate m tasks (balls) sequentially into n machines (bins).
Goal: minimise the **maximum load** $\max_{i \in [n]} x_i^m$, where x^t is the load vector after ball t.

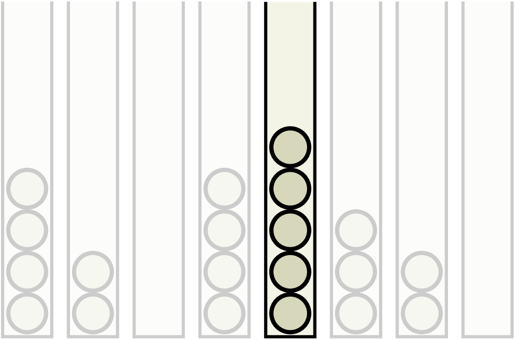


Balanced allocations setting

Allocate m tasks (balls) sequentially into n machines (bins).

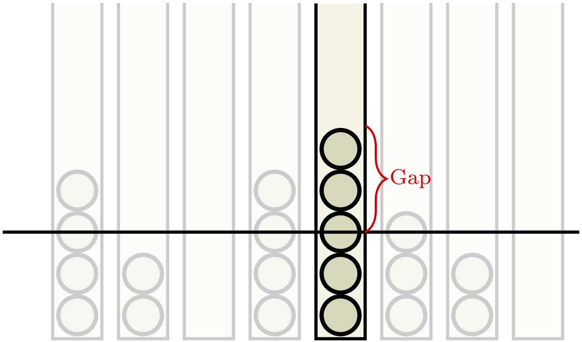
Goal: minimise the **maximum load** $\max_{i \in [n]} x_i^m$, where x^t is the load vector after ball t.

$\Leftrightarrow$ minimise the **gap**, where $\text{Gap}(m) = \max_{i \in [n]}(x_i^m - m/n)$.



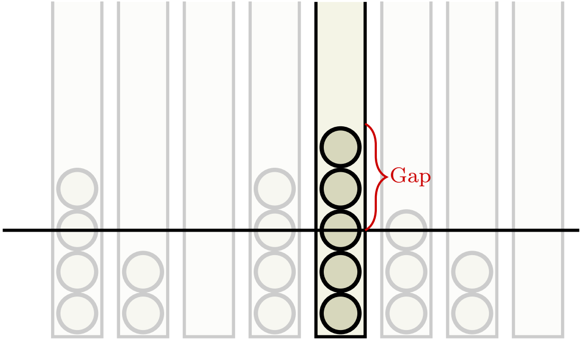
Balanced allocations setting

Allocate m tasks (balls) sequentially into n machines (bins).
Goal: minimise the **maximum load** $\max_{i \in [n]} x_i^m$, where x^t is the load vector after ball t.
$\Leftrightarrow$ minimise the **gap**, where $\text{Gap}(m) = \max_{i \in [n]}(x_i^m - m/n)$.



Balanced allocations setting

Allocate m tasks (balls) sequentially into n machines (bins).
Goal: minimise the **maximum load** $\max_{i \in [n]} x_i^m$, where x^t is the load vector after ball t.
$\Leftrightarrow$ minimise the **gap**, where $\mathrm{Gap}(m) = \max_{i \in [n]}(x_i^m - m/n)$.



- Applications in hashing, load balancing and routing.

One-Choice and Two-Choice processes

One-Choice Process:
Iteration: For each $t \geq 0$, sample **one** bin uniformly at random (u.a.r.) and allocate the ball there.

One-Choice and Two-Choice processes

> <u>One-Choice Process:</u>
> Iteration: For each $t \geq 0$, sample **one** bin uniformly at random (u.a.r.) and allocate the ball there.

- In the lightly-loaded case ($m = n$), w.h.p. $\text{Gap}(n) = \Theta\left(\frac{\log n}{\log \log n}\right)$ [Gon81].

ONE-CHOICE and TWO-CHOICE processes

<u>ONE-CHOICE PROCESS:</u>
Iteration: For each $t \geq 0$, sample **one** bin uniformly at random (u.a.r.) and allocate the ball there.

- In the lightly-loaded case ($m = n$), w.h.p. $\text{Gap}(n) = \Theta\left(\frac{\log n}{\log \log n}\right)$ [Gon81].

Meaning with probability at least $1 - n^{-c}$ for constant $c > 0$.

One-Choice and Two-Choice processes

<u>One-Choice Process:</u>
Iteration: For each $t \geq 0$, sample **one** bin uniformly at random (u.a.r.) and allocate the ball there.

- In the lightly-loaded case ($m = n$), w.h.p. $\mathrm{Gap}(n) = \Theta\left(\frac{\log n}{\log \log n}\right)$ [Gon81].
- In the heavily-loaded case ($m \gg n$), w.h.p. $\mathrm{Gap}(m) = \Theta\left(\sqrt{\frac{m}{n} \cdot \log n}\right)$ (e.g. [RS98]).

One-Choice and Two-Choice processes

<u>One-Choice Process:</u>
Iteration: For each $t \geq 0$, sample **one** bin uniformly at random (u.a.r.) and allocate the ball there.

- In the lightly-loaded case ($m = n$), w.h.p. $\mathrm{Gap}(n) = \Theta\left(\frac{\log n}{\log\log n}\right)$ [Gon81].
- In the heavily-loaded case ($m \gg n$), w.h.p. $\mathrm{Gap}(m) = \Theta\left(\sqrt{\frac{m}{n} \cdot \log n}\right)$ (e.g. [RS98]).

<u>Two-Choice Process:</u>
Iteration: For each $t \geq 0$, sample **two** bins independently u.a.r. and allocate the ball to the least loaded of the two.

One-Choice and Two-Choice processes

<u>One-Choice Process:</u>
Iteration: For each $t \geq 0$, sample **one** bin uniformly at random (u.a.r.) and allocate the ball there.

- In the lightly-loaded case ($m = n$), w.h.p. $\mathrm{Gap}(n) = \Theta\left(\frac{\log n}{\log \log n}\right)$ [Gon81].
- In the heavily-loaded case ($m \gg n$), w.h.p. $\mathrm{Gap}(m) = \Theta\left(\sqrt{\frac{m}{n} \cdot \log n}\right)$ (e.g. [RS98]).

<u>Two-Choice Process:</u>
Iteration: For each $t \geq 0$, sample **two** bins independently u.a.r. and allocate the ball to the least loaded of the two.

- In the lightly-loaded case ($m = n$), w.h.p. $\mathrm{Gap}(n) = \log_2 \log n + \Theta(1)$ [KLMadH96, ABKU99].

One-Choice and Two-Choice processes

<u>One-Choice Process:</u>
Iteration: For each $t \geq 0$, sample **one** bin uniformly at random (u.a.r.) and allocate the ball there.

- In the lightly-loaded case ($m = n$), w.h.p. $\text{Gap}(n) = \Theta\left(\frac{\log n}{\log \log n}\right)$ [Gon81].
- In the heavily-loaded case ($m \gg n$), w.h.p. $\text{Gap}(m) = \Theta\left(\sqrt{\frac{m}{n} \cdot \log n}\right)$ (e.g. [RS98]).

<u>Two-Choice Process:</u>
Iteration: For each $t \geq 0$, sample **two** bins independently u.a.r. and allocate the ball to the least loaded of the two.

- In the lightly-loaded case ($m = n$), w.h.p. $\text{Gap}(n) = \log_2 \log n + \Theta(1)$ [KLMadH96, ABKU99].

One-Choice and Two-Choice processes

<u>One-Choice Process:</u>
Iteration: For each $t \geq 0$, sample **one** bin uniformly at random (u.a.r.) and allocate the ball there.

- In the lightly-loaded case ($m = n$), w.h.p. $\mathrm{Gap}(n) = \Theta\left(\frac{\log n}{\log\log n}\right)$ [Gon81].
- In the heavily-loaded case ($m \gg n$), w.h.p. $\mathrm{Gap}(m) = \Theta\left(\sqrt{\frac{m}{n} \cdot \log n}\right)$ (e.g. [RS98]).

<u>Two-Choice Process:</u>
Iteration: For each $t \geq 0$, sample **two** bins independently u.a.r. and allocate the ball to the least loaded of the two.

- In the lightly-loaded case ($m = n$), w.h.p. $\mathrm{Gap}(n) = \log_2 \log n + \Theta(1)$ [KLMadH96, ABKU99].
- In the heavily-loaded case ($m \gg n$), w.h.p. $\mathrm{Gap}(m) = \log_2 \log n + \Theta(1)$ [BCSV06].

One-Choice and Two-Choice processes

> **One-Choice Process:**
> Iteration: For each $t \geq 0$, sample **one** bin uniformly at random (u.a.r.) and allocate the ball there.

- In the lightly-loaded case ($m = n$), w.h.p. $\mathrm{Gap}(n) = \Theta\left(\frac{\log n}{\log\log n}\right)$ [Gon81].
- In the heavily-loaded case ($m \gg n$), w.h.p. $\mathrm{Gap}(m) = \Theta\left(\sqrt{\frac{m}{n} \cdot \log n}\right)$ (e.g. [RS98]).

> **Two-Choice Process:**
> Iteration: For each $t \geq 0$, sample **two** bins independently u.a.r. and allocate the ball to the least loaded of the two.

- In the lightly-loaded case ($m = n$), w.h.p. $\mathrm{Gap}(n) = \log_2 \log n + \Theta(1)$ [KLMadH96, ABKU99].
- In the heavily-loaded case ($m \gg n$), w.h.p. $\mathrm{Gap}(m) = \log_2 \log n + \Theta(1)$ [BCSV06].

$(1+\beta)$-Process: Definition

$(1+\beta)$-Process:
Parameter: A *mixing factor* $\beta \in (0, 1]$.
Iteration: For each $t \geq 0$, with probability β allocate one ball via the Two-Choice process, otherwise allocate one ball via the One-Choice process.

$(1+\beta)$-Process: Definition

$(1+\beta)$-Process:
Parameter: A *mixing factor* $\beta \in (0,1]$.
Iteration: For each $t \geq 0$, with probability β allocate one ball via the Two-Choice process, otherwise allocate one ball via the One-Choice process.

- [Mit99] interpreted $(1-\beta)/2$ as the probability of making an *erroneous comparison*.

$(1+\beta)$-Process: Definition

$(1+\beta)$-Process:
Parameter: A *mixing factor* $\beta \in (0,1]$.
Iteration: For each $t \geq 0$, with probability β allocate one ball via the Two-Choice process, otherwise allocate one ball via the One-Choice process.

- [Mit99] interpreted $(1-\beta)/2$ as the probability of making an *erroneous comparison*.
- In the heavily-loaded case, [PTW15] proved that w.h.p. $\text{Gap}(m) = \Theta(\log n/\beta)$ for $1/n \leq \beta < 1 - \epsilon$ for any constant $\epsilon > 0$.

k-Threshold and k-Quantile

Adaptive 1-Threshold

Adaptive Threshold(f) Process:

Parameter: A *threshold function* $f(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^t + 1 & \text{if } x_{i_1}^t < f(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^t + 1 & \text{otherwise.} \end{cases}$$

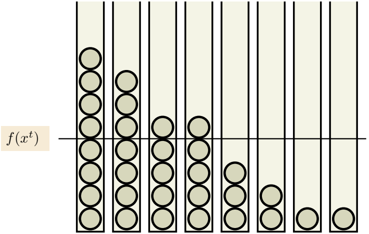
Adaptive 1-Threshold

Adaptive Threshold(f) Process:

Parameter: A *threshold function* $f(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^t + 1 & \text{if } x_{i_1}^t < f(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^t + 1 & \text{otherwise.} \end{cases}$$



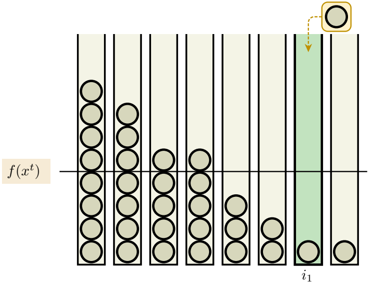
Adaptive 1-Threshold

Adaptive Threshold(f) Process:

Parameter: A *threshold function* $f(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^t + 1 & \text{if } x_{i_1}^t < f(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^t + 1 & \text{otherwise.} \end{cases}$$



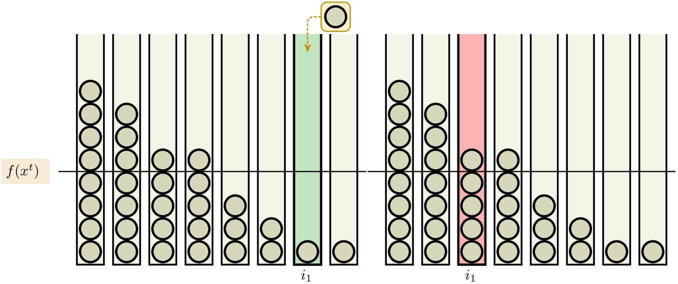
Adaptive 1-Threshold

Adaptive Threshold(f) Process:

Parameter: A *threshold function* $f(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^t + 1 & \text{if } x_{i_1}^t < f(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^t + 1 & \text{otherwise.} \end{cases}$$



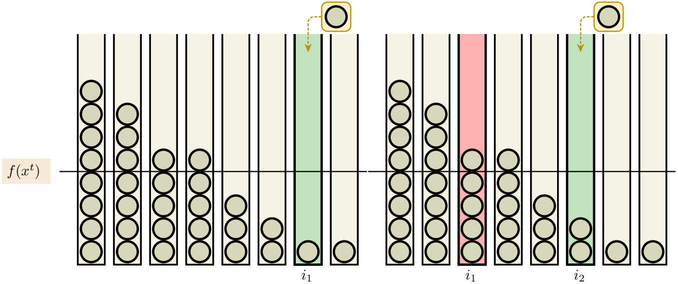
Adaptive 1-THRESHOLD

Adaptive THRESHOLD(f) Process:

Parameter: A *threshold function* $f(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^t + 1 & \text{if } x_{i_1}^t < f(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^t + 1 & \text{otherwise.} \end{cases}$$



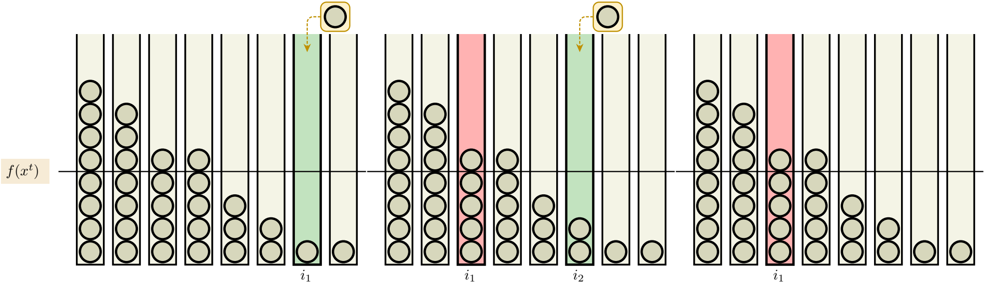
Adaptive 1-Threshold

Adaptive Threshold(f) Process:

Parameter: A *threshold function* $f(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^t + 1 & \text{if } x_{i_1}^t < f(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^t + 1 & \text{otherwise.} \end{cases}$$



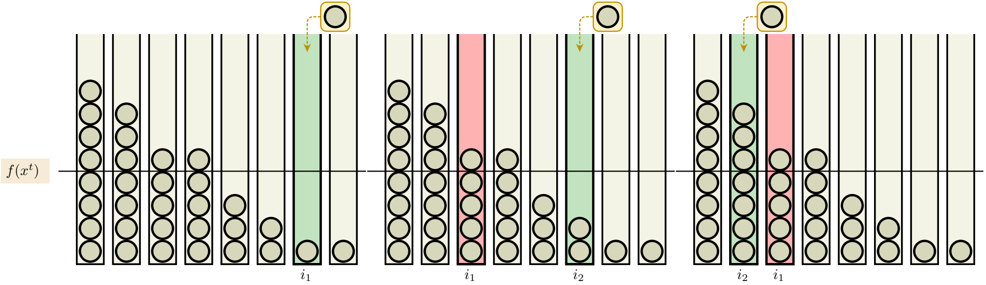
Adaptive 1-Threshold

Adaptive Threshold(f) Process:

Parameter: A *threshold function* $f(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^t + 1 & \text{if } x_{i_1}^t < f(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^t + 1 & \text{otherwise.} \end{cases}$$



Adaptive 1-Threshold

Adaptive Threshold(f) Process:

Parameter: A *threshold function* $f(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^t + 1 & \text{if } x_{i_1}^t < f(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^t + 1 & \text{otherwise.} \end{cases}$$

- For the lightly-loaded case, [FGG21] determined the *optimal threshold*, achieving w.h.p. $\text{Gap}(n) = \mathcal{O}\Big(\sqrt{\frac{\log n}{\log\log n}}\Big)$.

Adaptive 1-Threshold

Adaptive Threshold(f) Process:

Parameter: A *threshold function* $f(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^t + 1 & \text{if } x_{i_1}^t < f(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^t + 1 & \text{otherwise.} \end{cases}$$

- For the lightly-loaded case, [FGG21] determined the *optimal threshold*, achieving w.h.p. $\text{Gap}(n) = \mathcal{O}\left(\sqrt{\frac{\log n}{\log \log n}}\right)$.
- For the heavily-loaded case, [LSS22] proved for $f(x^t) = t/n$ that w.h.p. $\text{Gap}(m) = \mathcal{O}(\log n)$.

Adaptive 1-Threshold

Adaptive Threshold(f) Process:

Parameter: A *threshold function* $f(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^t + 1 & \text{if } x_{i_1}^t < f(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^t + 1 & \text{otherwise.} \end{cases}$$

- For the lightly-loaded case, [FGG21] determined the *optimal threshold*, achieving w.h.p. $\mathrm{Gap}(n) = \mathcal{O}\left(\sqrt{\frac{\log n}{\log \log n}}\right)$.
- For the heavily-loaded case, [LSS22] proved for $f(x^t) = t/n$ that w.h.p. $\mathrm{Gap}(m) = \mathcal{O}(\log n)$.

Threshold: Open in Visualiser.

Adaptive 1-Quantile

Adaptive Quantile(δ) Process:

Parameter: A *quantile function* $\delta(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^{t} + 1 & \text{if } \mathrm{Rank}^t(i_1) > n \cdot \delta(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^{t} + 1 & \text{otherwise.} \end{cases}$$

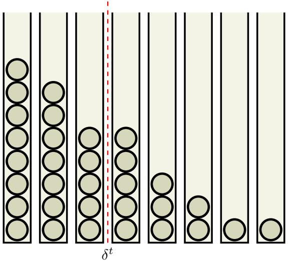
Adaptive 1-Quantile

Adaptive Quantile(δ) Process:

Parameter: A *quantile function* $\delta(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^{t} + 1 & \text{if } \mathrm{Rank}^t(i_1) > n \cdot \delta(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^{t} + 1 & \text{otherwise.} \end{cases}$$



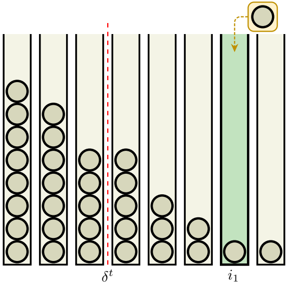
Adaptive 1-Quantile

Adaptive Quantile(δ) Process:

Parameter: A *quantile function* $\delta(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^t + 1 & \text{if } \mathrm{Rank}^t(i_1) > n \cdot \delta(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^t + 1 & \text{otherwise.} \end{cases}$$



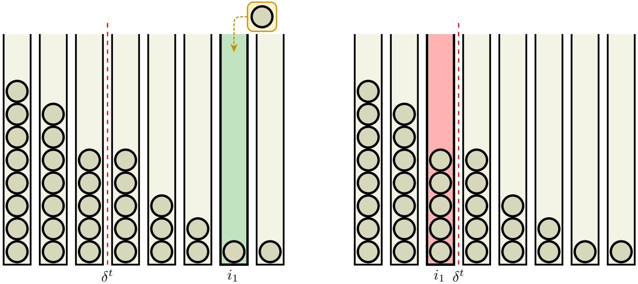
Adaptive 1-Quantile

Adaptive Quantile(δ) Process:

Parameter: A *quantile function* $\delta(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^t + 1 & \text{if } \mathrm{Rank}^t(i_1) > n \cdot \delta(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^t + 1 & \text{otherwise.} \end{cases}$$



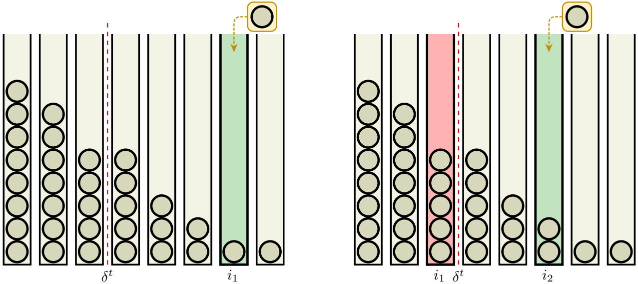
Adaptive 1-Quantile

Adaptive Quantile(δ) Process:

Parameter: A *quantile function* $\delta(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^t + 1 & \text{if } \mathrm{Rank}^t(i_1) > n \cdot \delta(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^t + 1 & \text{otherwise.} \end{cases}$$



Adaptive 1-Quantile

Adaptive Quantile(δ) Process:

Parameter: A *quantile function* $\delta(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^{t} + 1 & \text{if } \mathrm{Rank}^t(i_1) > n \cdot \delta(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^{t} + 1 & \text{otherwise.} \end{cases}$$

- Adaptive Quantile(δ) processes can simulate any adaptive Threshold(f).

Adaptive 1-Quantile

Adaptive Quantile(δ) Process:

Parameter: A *quantile function* $\delta(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^t + 1 & \text{if } \mathrm{Rank}^t(i_1) > n \cdot \delta(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^t + 1 & \text{otherwise.} \end{cases}$$

- Adaptive Quantile(δ) processes can simulate any adaptive Threshold(f).
- Also, adaptive Threshold(f) process can simulate any adaptive Quantile(δ).

Adaptive 1-Quantile

Adaptive Quantile(δ) Process:

Parameter: A *quantile function* $\delta(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^t + 1 & \text{if } \mathrm{Rank}^t(i_1) > n \cdot \delta(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^t + 1 & \text{otherwise.} \end{cases}$$

- Adaptive Quantile(δ) processes can simulate any adaptive Threshold(f).
- Also, adaptive Threshold(f) process can simulate any adaptive Quantile(δ).
- Both are special cases of Two-Thinning [FGG21].

Adaptive 1-Quantile

Adaptive Quantile(δ) Process:

Parameter: A *quantile function* $\delta(x^t)$.

Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^t + 1 & \text{if } \mathrm{Rank}^t(i_1) > n \cdot \delta(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^t + 1 & \text{otherwise.} \end{cases}$$

- Adaptive Quantile(δ) processes can simulate any adaptive Threshold(f).
- Also, adaptive Threshold(f) process can simulate any adaptive Quantile(δ).
- Both are special cases of Two-Thinning [FGG21].
- [IK05, FL20] analyse d-Thinning in the lightly-loaded case.

Adaptive 1-Quantile

<u>Adaptive Quantile(δ) Process:</u>
Parameter: A *quantile function* $\delta(x^t)$.
Iteration: For $t \geq 0$, sample two bins i_1 and i_2 independently and u.a.r. Then, update:

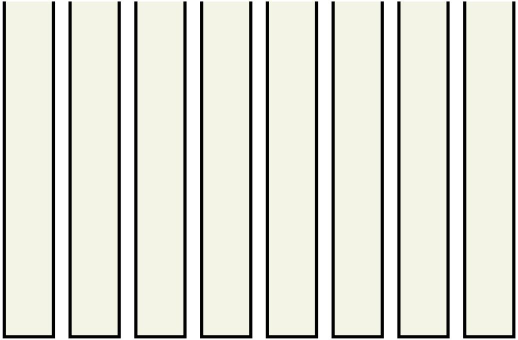
$$\begin{cases} x_{i_1}^{t+1} = x_{i_1}^t + 1 & \text{if } \mathrm{Rank}^t(i_1) > n \cdot \delta(x^t), \\ x_{i_2}^{t+1} = x_{i_2}^t + 1 & \text{otherwise.} \end{cases}$$

- Adaptive Quantile(δ) processes can simulate any adaptive Threshold(f).
- Also, adaptive Threshold(f) process can simulate any adaptive Quantile(δ).
- Both are special cases of Two-Thinning [FGG21].
- [IK05, FL20] analyse d-Thinning in the lightly-loaded case.

Quantile: Open in Visualiser.

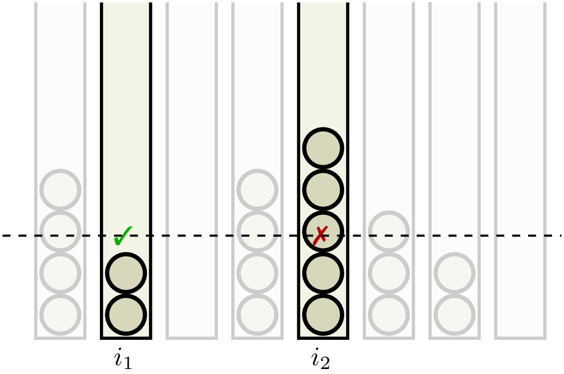
1-Threshold as Two-Choice with incomplete information

We can interpret 1-Threshold as an instance of the Two-Choice process, where we are only able to compare the loads of the two sampled bins if one is *above* the threshold and one is *below*.



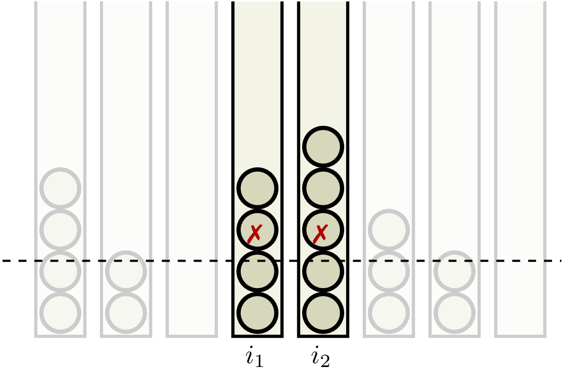
1-Threshold as Two-Choice with incomplete information

We can interpret 1-Threshold as an instance of the Two-Choice process, where we are only able to compare the loads of the two sampled bins if one is *above* the threshold and one is *below*.



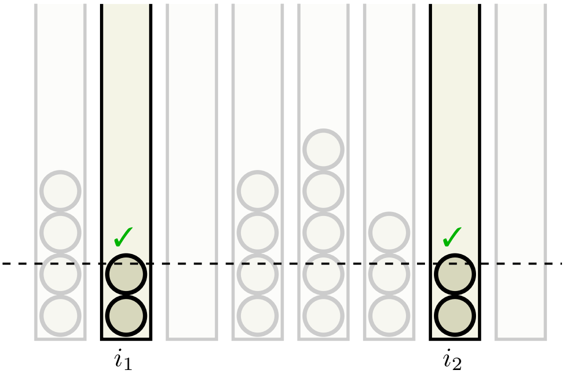
1-Threshold as Two-Choice with incomplete information

We can interpret 1-Threshold as an instance of the Two-Choice process, where we are only able to compare the loads of the two sampled bins if one is *above* the threshold and one is *below*.



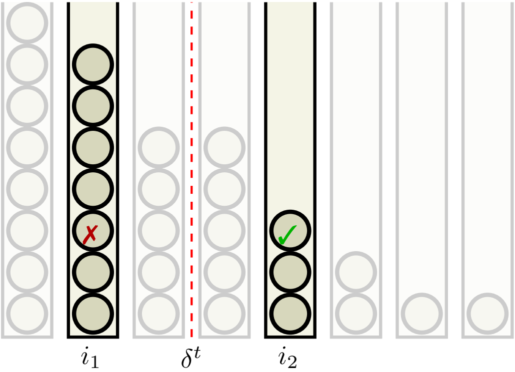
1-Threshold as Two-Choice with incomplete information

We can interpret 1-Threshold as an instance of the Two-Choice process, where we are only able to compare the loads of the two sampled bins if one is *above* the threshold and one is *below*.



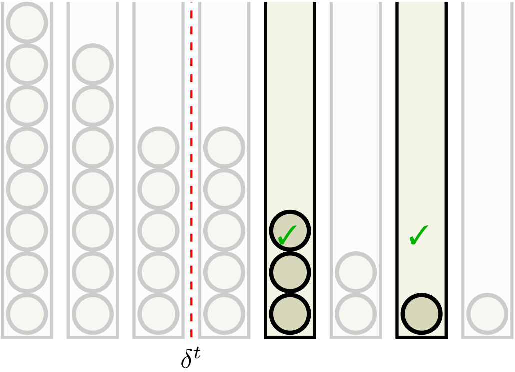
1-Quantile as Two-Choice with incomplete information

Similarly, 1-Quantile is as Two-Choice but we can compare two bins only if these are on *different sides* of the quantile δ^t.



1-Quantile as Two-Choice with incomplete information

Similarly, 1-Quantile is as Two-Choice but we can compare two bins only if these are on *different sides* of the quantile δ^t.

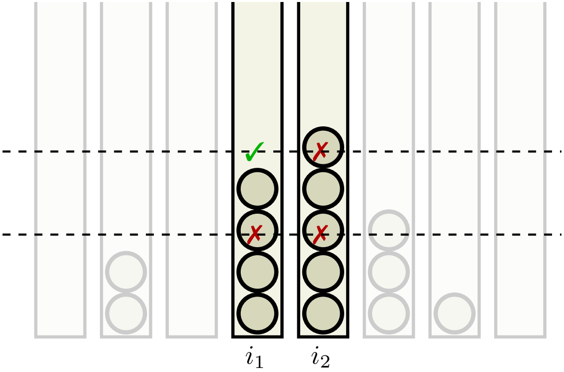


k-Threshold process

- Under this interpretation, we can extend the 1-Threshold process to k thresholds.

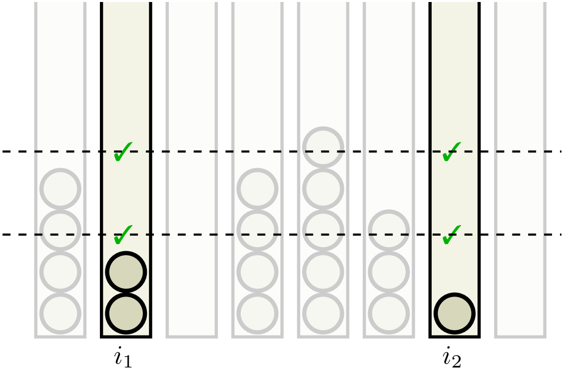
k-Threshold process

- Under this interpretation, we can extend the 1-Threshold process to k thresholds.
- We can only distinguish two bins if they are in *different regions*.



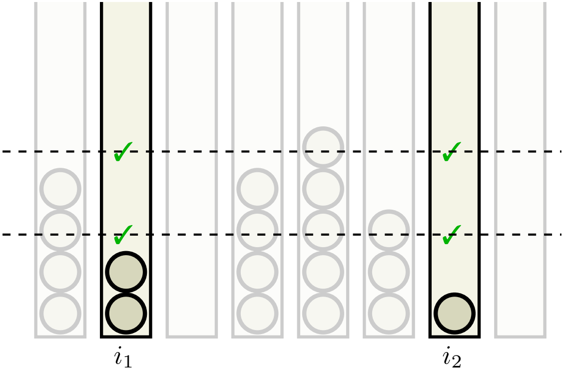
k-Threshold process

- Under this interpretation, we can extend the 1-Threshold process to k thresholds.
- We can only distinguish two bins if they are in *different regions*.



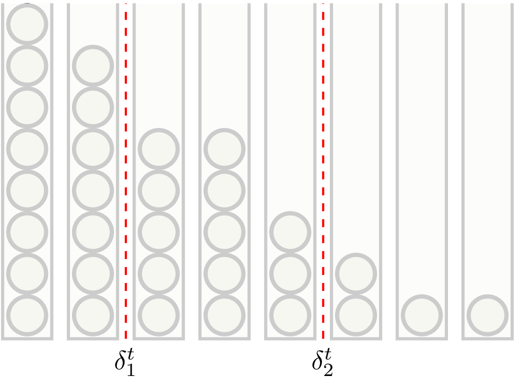
k-Threshold process

- Under this interpretation, we can extend the 1-Threshold process to k thresholds.
- We can only distinguish two bins if they are in *different regions*.
- [IK05] analysed the lightly-loaded case for *equidistant thresholds*.



k-Quantile process

Similarly, we can extend 1-Quantile to obtain the k-Quantile process.



Our results

Our results

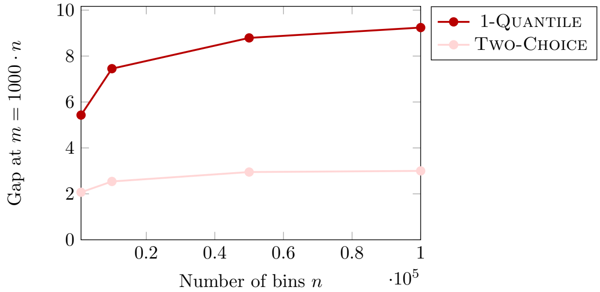
- Any adaptive1-Quantile/1-Threshold process has w.h.p. $\text{Gap}(m) = \Omega(\log n / \log \log n)$ for some $m \in [1, n \log^2 n]$ (disproves [FGG21, Problem 1.3]).

Our results

- Any adaptive1-QUANTILE/1-THRESHOLD process has w.h.p. $\text{Gap}(m) = \Omega(\log n / \log\log n)$ for some $m \in [1, n\log^2 n]$ (disproves [FGG21, Problem 1.3]).
- A k-QUANTILE process with *uniform quantiles* that achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.

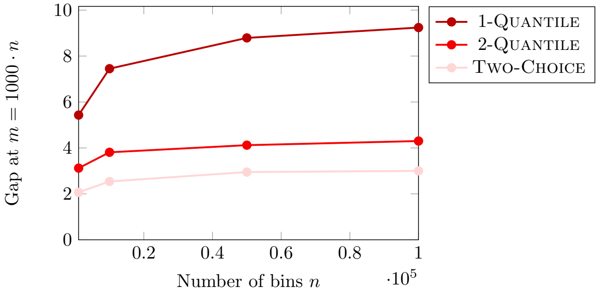
Our results

- Any adaptive 1-Quantile/1-Threshold process has w.h.p. $\text{Gap}(m) = \Omega(\log n / \log\log n)$ for some $m \in [1, n\log^2 n]$ (disproves [FGG21, Problem 1.3]).
- A k-Quantile process with *uniform quantiles* that achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.



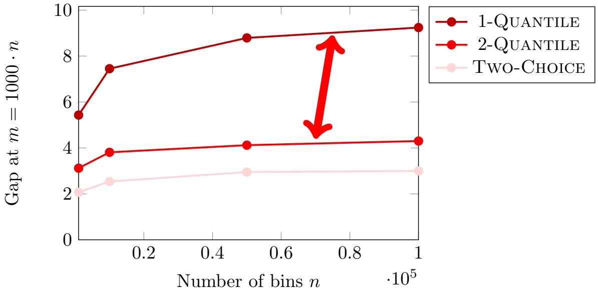
Our results

- Any adaptive 1-Quantile/1-Threshold process has w.h.p. $\mathrm{Gap}(m) = \Omega(\log n / \log\log n)$ for some $m \in [1, n\log^2 n]$ (disproves [FGG21, Problem 1.3]).
- A k-Quantile process with *uniform quantiles* that achieves w.h.p. $\mathrm{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.



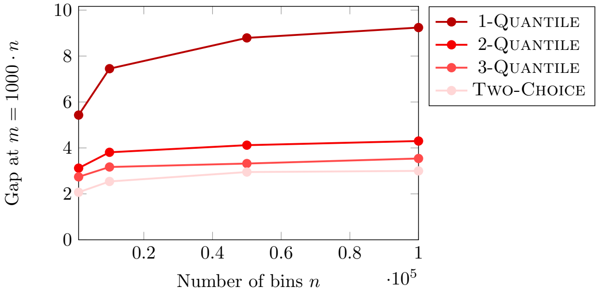
Our results

- Any adaptive1-Quantile/1-Threshold process has w.h.p. $\text{Gap}(m) = \Omega(\log n / \log\log n)$ for some $m \in [1, n\log^2 n]$ (disproves [FGG21, Problem 1.3]).
- A k-Quantile process with *uniform quantiles* that achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.



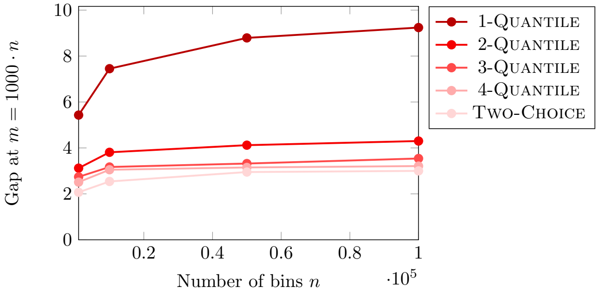
Our results

- Any adaptive 1-Quantile/1-Threshold process has w.h.p. $\text{Gap}(m) = \Omega(\log n / \log\log n)$ for some $m \in [1, n\log^2 n]$ (disproves [FGG21, Problem 1.3]).
- A k-Quantile process with *uniform quantiles* that achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.



Our results

- Any adaptive 1-Quantile/1-Threshold process has w.h.p. $\text{Gap}(m) = \Omega(\log n / \log\log n)$ for some $m \in [1, n \log^2 n]$ (disproves [FGG21, Problem 1.3]).
- A k-Quantile process with *uniform quantiles* that achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.



Our results

- Any adaptive1-Quantile/1-Threshold process has w.h.p. $\text{Gap}(m) = \Omega(\log n / \log \log n)$ for some $m \in [1, n \log^2 n]$ (disproves [FGG21, Problem 1.3]).
- A k-Quantile process with *uniform quantiles* that achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
- Implications:

Our results

- Any adaptive 1-Quantile/1-Threshold process has w.h.p. $\text{Gap}(m) = \Omega(\log n / \log\log n)$ for some $m \in [1, n\log^2 n]$ (disproves [FGG21, Problem 1.3]).
- A k-Quantile process with *uniform quantiles* that achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
- Implications:
  - For $k = \Theta(\log\log n)$, we recover for Two-Choice that $\text{Gap}(m) = \mathcal{O}(\log\log n)$.

Our results

- Any adaptive 1-Quantile/1-Threshold process has w.h.p. $\text{Gap}(m) = \Omega(\log n / \log\log n)$ for some $m \in [1, n \log^2 n]$ (disproves [FGG21, Problem 1.3]).
- A k-Quantile process with *uniform quantiles* that achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
- Implications:
  - For $k = \Theta(\log\log n)$, we recover for Two-Choice that $\text{Gap}(m) = \mathcal{O}(\log\log n)$.
  - For $(1+\beta)$ with $\beta = 1 - 2^{-0.5(\log n)^{(k-1)/k}}$, w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.

Our results

- Any adaptive1-Quantile/1-Threshold process has w.h.p. $\text{Gap}(m) = \Omega(\log n / \log\log n)$ for some $m \in [1, n \log^2 n]$ (disproves [FGG21, Problem 1.3]).
- A k-Quantile process with *uniform quantiles* that achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
- Implications:
  - For $k = \Theta(\log\log n)$, we recover for Two-Choice that $\text{Gap}(m) = \mathcal{O}(\log\log n)$.
  - For $(1+\beta)$ with $\beta = 1 - 2^{-0.5(\log n)^{(k-1)/k}}$, w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
  - For d-Thinning, w.h.p. $\text{Gap}(m) = \mathcal{O}(d \cdot (\log n)^{2/d})$.

Our results

- Any adaptive1-Quantile/1-Threshold process has w.h.p. $\text{Gap}(m) = \Omega(\log n / \log\log n)$ for some $m \in [1, n \log^2 n]$ (disproves [FGG21, Problem 1.3]).
- A k-Quantile process with *uniform quantiles* that achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
- Implications:
  - For $k = \Theta(\log\log n)$, we recover for Two-Choice that $\text{Gap}(m) = \mathcal{O}(\log\log n)$.
  - For $(1+\beta)$ with $\beta = 1 - 2^{-0.5(\log n)^{(k-1)/k}}$, w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
  - For d-Thinning, w.h.p. $\text{Gap}(m) = \mathcal{O}(d \cdot (\log n)^{2/d})$.
  - For **graphical allocations** in *dense expanders*, w.h.p. $\text{Gap}(m) = \mathcal{O}(\log\log n)$ (progress in [PTW15, Open Question 2]).

Our results

- Any adaptive 1-QUANTILE/1-THRESHOLD process has w.h.p. $\text{Gap}(m) = \Omega(\log n / \log \log n)$ for some $m \in [1, n \log^2 n]$ (disproves [FGG21, Problem 1.3]).
- A k-QUANTILE process with *uniform quantiles* that achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
- Implications:
  - For $k = \Theta(\log \log n)$, we recover for TWO-CHOICE that $\text{Gap}(m) = \mathcal{O}(\log \log n)$.
  - For $(1+\beta)$ with $\beta = 1 - 2^{-0.5(\log n)^{(k-1)/k}}$, w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
  - For d-THINNING, w.h.p. $\text{Gap}(m) = \mathcal{O}(d \cdot (\log n)^{2/d})$.
  - For **graphical allocations** in *dense expanders*, w.h.p. $\text{Gap}(m) = \mathcal{O}(\log \log n)$ (progress in [PTW15, Open Question 2]).
- Use *layered induction* over *super-exponential potential* functions.

Our results

- Any adaptive1-Quantile/1-Threshold process has w.h.p. $\text{Gap}(m) = \Omega(\log n / \log\log n)$ for some $m \in [1, n\log^2 n]$ (disproves [FGG21, Problem 1.3]).
- A k-Quantile process with *uniform quantiles* that achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
- Implications:
  - For $k = \Theta(\log\log n)$, we recover for Two-Choice that $\text{Gap}(m) = \mathcal{O}(\log\log n)$.
  - For $(1+\beta)$ with $\beta = 1 - 2^{-0.5(\log n)^{(k-1)/k}}$, w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
  - For d-Thinning, w.h.p. $\text{Gap}(m) = \mathcal{O}(d \cdot (\log n)^{2/d})$.
  - For **graphical allocations** in *dense expanders*, w.h.p. $\text{Gap}(m) = \mathcal{O}(\log\log n)$ (progress in [PTW15, Open Question 2]).
- Use *layered induction* over *super-exponential potential* functions.

  $\rightsquigarrow$ Might be helpful in analyzing other processes.

Lower bound: Proof Outline

Lower bound proof (I)

Theorem

For any adaptive $\textsc{Quantile}(\delta)$ (or $\textsc{Threshold}(f)$) process $\mathcal{P}$,

$$\mathbf{Pr}\left[\max_{t \in [0, n \log^2 n]} \mathrm{Gap}(t) \geq \frac{1}{8} \cdot \frac{\log n}{\log\log n}\right] \geq 1 - o(n^{-2}).$$

Lower bound proof (I)

Theorem

For any adaptive QUANTILE(δ) (or THRESHOLD(f)) process $\mathcal{P}$,

$$\mathbf{Pr}\left[\max_{t\in[0,n\log^2 n]} \mathrm{Gap}(t) \geq \frac{1}{8}\cdot\frac{\log n}{\log\log n}\right] \geq 1 - o(n^{-2}).$$

Proof. We consider two cases:

Lower bound proof (I)

Theorem

For any adaptive $\textsc{Quantile}(\delta)$ (or $\textsc{Threshold}(f)$) process $\mathcal{P}$,

$$\mathbf{Pr}\left[\max_{t \in [0, n \log^2 n]} \mathrm{Gap}(t) \geq \frac{1}{8} \cdot \frac{\log n}{\log\log n}\right] \geq 1 - o(n^{-2}).$$

Proof. We consider two cases:

Case A: $\mathcal{P}$ uses *at most* n quantiles with $\delta^t \geq \frac{1}{\log^2 n}$.

Lower bound proof (I)

Theorem

For any adaptive QUANTILE(δ) (or THRESHOLD(f)) process $\mathcal{P}$,

$$\mathbf{Pr}\left[\max_{t\in[0,n\log^2 n]} \mathrm{Gap}(t) \geq \frac{1}{8}\cdot\frac{\log n}{\log\log n}\right] \geq 1 - o(n^{-2}).$$

Proof. We consider two cases:

Case A: $\mathcal{P}$ uses *at most* n quantiles with $\delta^t \geq \frac{1}{\log^2 n}$.

- A *small* quantile means that the first sample is used often.

Lower bound proof (I)

Theorem

For any adaptive QUANTILE(δ) (or THRESHOLD(f)) process $\mathcal{P}$,

$$\mathbf{Pr}\left[\max_{t\in[0,n\log^2 n]} \mathrm{Gap}(t) \geq \frac{1}{8}\cdot\frac{\log n}{\log\log n}\right] \geq 1 - o(n^{-2}).$$

Proof. We consider two cases:

Case A: $\mathcal{P}$ uses *at most* n quantiles with $\delta^t \geq \frac{1}{\log^2 n}$.

- A *small* quantile means that the first sample is used often.
- $\mathcal{P}$ *disagrees* with ONE-CHOICE w.h.p. in at most $n + \mathcal{O}(m/\log^2 n) = \mathcal{O}(n)$ allocations.

Lower bound proof (I)

Theorem

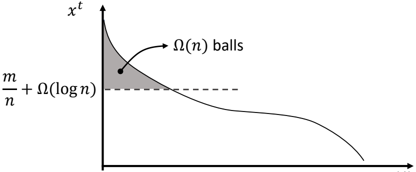
For any adaptive QUANTILE(δ) (or THRESHOLD(f)) process $\mathcal{P}$,

$$\mathbf{Pr}\left[\max_{t \in [0, n \log^2 n]} \mathrm{Gap}(t) \geq \frac{1}{8} \cdot \frac{\log n}{\log \log n}\right] \geq 1 - o(n^{-2}).$$

Proof. We consider two cases:

Case A: $\mathcal{P}$ uses *at most* n quantiles with $\delta^t \geq \frac{1}{\log^2 n}$.

- A *small* quantile means that the first sample is used often.
- $\mathcal{P}$ *disagrees* with ONE-CHOICE w.h.p. in at most $n + \mathcal{O}(m/\log^2 n) = \mathcal{O}(n)$ allocations.
- Using Poissonisation w.h.p. there are $\Omega(n)$ balls above $\frac{m}{n} + \Omega(\log n)$.



Lower bound proof (I)

Theorem

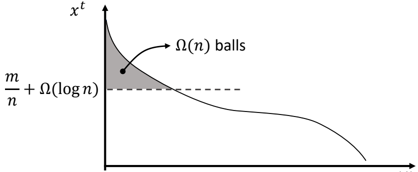
For any adaptive QUANTILE(δ) (or THRESHOLD(f)) process $\mathcal{P}$,

$$\mathbf{Pr}\left[\max_{t\in[0,n\log^2 n]} \mathrm{Gap}(t) \geq \frac{1}{8}\cdot\frac{\log n}{\log\log n}\right] \geq 1 - o(n^{-2}).$$

Proof. We consider two cases:

Case A: $\mathcal{P}$ uses *at most* n quantiles with $\delta^t \geq \frac{1}{\log^2 n}$.

- A *small* quantile means that the first sample is used often.
- $\mathcal{P}$ *disagrees* with ONE-CHOICE w.h.p. in at most $n + \mathcal{O}(m/\log^2 n) = \mathcal{O}(n)$ allocations.
- Using Poissonisation w.h.p. there are $\Omega(n)$ balls above $\frac{m}{n} + \Omega(\log n)$.
- Hence, the $\mathrm{Gap}(m) = \Omega(\log n)$ remains.



Lower bound proof (II)

Theorem

For any adaptive Quantile(δ) (or Threshold(f)) process $\mathcal{P}$,

$$\mathbf{Pr}\left[\max_{t\in[0,n\log^2 n]} \mathrm{Gap}(t) \geq \frac{1}{8}\cdot\frac{\log n}{\log\log n}\right] \geq 1 - o(n^{-2}).$$

Proof (continued). We consider two cases:

Lower bound proof (II)

Theorem

For any adaptive QUANTILE(δ) (or THRESHOLD(f)) process $\mathcal{P}$,

$$\mathbf{Pr}\left[\max_{t\in[0,n\log^2 n]} \text{Gap}(t) \geq \frac{1}{8}\cdot\frac{\log n}{\log\log n}\right] \geq 1 - o(n^{-2}).$$

Proof (continued). We consider two cases:
Case B: $\mathcal{P}$ uses at least n quantiles with $\delta^t \geq \frac{1}{\log^2 n}$.

Lower bound proof (II)

Theorem

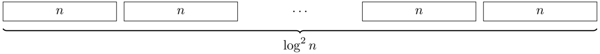
For any adaptive QUANTILE(δ) (or THRESHOLD(f)) process $\mathcal{P}$,

$$\mathbf{Pr}\left[\max_{t \in [0, n \log^2 n]} \mathrm{Gap}(t) \geq \frac{1}{8} \cdot \frac{\log n}{\log \log n} \right] \geq 1 - o(n^{-2}).$$

Proof (continued). We consider two cases:

Case B: $\mathcal{P}$ uses at least n quantiles with $\delta^t \geq \frac{1}{\log^2 n}$.

- Split m into intervals of n allocations:



□

Lower bound proof (II)

Theorem

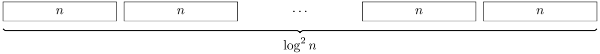
For any adaptive QUANTILE(δ) (or THRESHOLD(f)) process $\mathcal{P}$,

$$\mathbf{Pr}\left[\max_{t\in[0,n\log^2 n]} \mathrm{Gap}(t) \geq \frac{1}{8}\cdot\frac{\log n}{\log\log n}\right] \geq 1 - o(n^{-2}).$$

Proof (continued). We consider two cases:

<u>Case B:</u> $\mathcal{P}$ uses at least n quantiles with $\delta^t \geq \frac{1}{\log^2 n}$.

- Split m into intervals of n allocations:



- One interval $[t, t+n)$ *must have* $\geq n/\log^2 n$ balls allocated with $\delta^s \geq \frac{1}{\log^2 n}$.

□

Lower bound proof (II)

Theorem

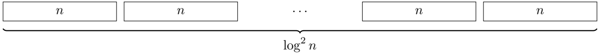
For any adaptive Quantile(δ) (or Threshold(f)) process $\mathcal{P}$,

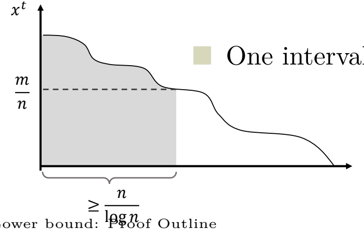
$$\mathbf{Pr}\left[\max_{t \in [0, n \log^2 n]} \mathrm{Gap}(t) \geq \frac{1}{8} \cdot \frac{\log n}{\log \log n} \right] \geq 1 - o(n^{-2}).$$

Proof (continued). We consider two cases:

Case B: $\mathcal{P}$ uses at least n quantiles with $\delta^t \geq \frac{1}{\log^2 n}$.

- Split m into intervals of n allocations:





- One interval $[t, t+n)$ *must have* $\geq n/\log^2 n$ balls allocated with $\delta^s \geq \frac{1}{\log^2 n}$.

□

Lower bound proof (II)

Theorem

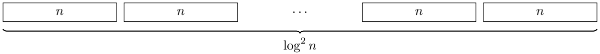
For any adaptive QUANTILE(δ) (or THRESHOLD(f)) process $\mathcal{P}$,

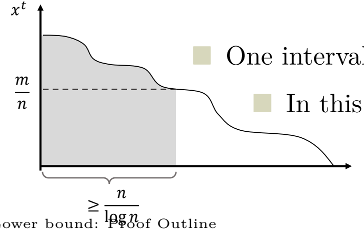
$$\mathbf{Pr}\left[\max_{t\in[0,n\log^2 n]} \mathrm{Gap}(t) \geq \frac{1}{8}\cdot\frac{\log n}{\log\log n}\right] \geq 1 - o(n^{-2}).$$

Proof (continued). We consider two cases:

Case B: $\mathcal{P}$ uses at least n quantiles with $\delta^t \geq \frac{1}{\log^2 n}$.

- Split m into intervals of n allocations:





- One interval $[t, t+n)$ *must have* $\geq n/\log^2 n$ balls allocated with $\delta^s \geq \frac{1}{\log^2 n}$.
  - In this interval, w.h.p. $\Omega(n/\log^4 n)$ balls allocated using ONE-CHOICE.

□

Lower bound proof (II)

Theorem

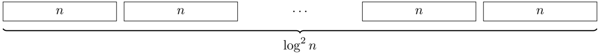
For any adaptive $\textsc{Quantile}(\delta)$ (or $\textsc{Threshold}(f)$) process $\mathcal{P}$,

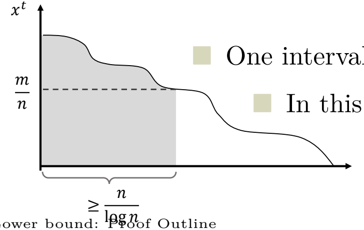
$$\mathbf{Pr}\left[\max_{t \in [0, n \log^2 n]} \mathrm{Gap}(t) \geq \frac{1}{8} \cdot \frac{\log n}{\log \log n}\right] \geq 1 - o(n^{-2}).$$

Proof (continued). We consider two cases:

Case B: $\mathcal{P}$ uses at least n quantiles with $\delta^t \geq \frac{1}{\log^2 n}$.

- Split m into intervals of n allocations:





- One interval $[t, t+n)$ *must have* $\geq n/\log^2 n$ balls allocated with $\delta^s \geq \frac{1}{\log^2 n}$.
- In this interval, w.h.p. $\Omega(n/\log^4 n)$ balls allocated using One-Choice.
- Implies w.h.p. $\mathrm{Gap}(t+n) = \Omega(\log n / \log \log n)$.

□

Upper bound: Proof outline

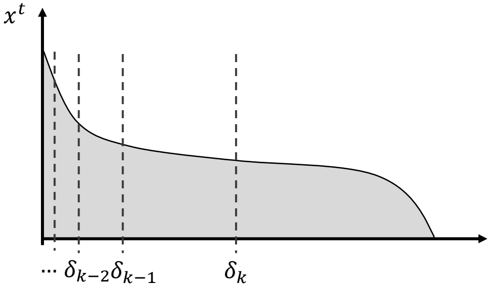
k-QUANTILE process

Theorem

Consider the QUANTILE$(\delta_1, \delta_2, \ldots, \delta_k)$ process with

$$\delta_j := \begin{cases} e^{-\frac{1}{4}(\log n)^{(k-j)/k}} & \text{if } j < k \\ \frac{1}{2} & \text{if } i = k. \end{cases}$$

For any step $m \geq 0$, $\mathbf{Pr}\left[\,\mathrm{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})\,\right] \geq 1 - n^{-3}$.



The hyperbolic cosine potential function

- [PTW15] used the **hyperbolic cosine potential**,

$$\Gamma^t(x^t) := \underbrace{\sum_{i=1}^{n} e^{\gamma(x_i^t - t/n)}}_{\text{Overload potential: } \Phi_0^t} + \underbrace{\sum_{i=1}^{n} e^{-\gamma(x_i^t - t/n)}}_{\text{Underload potential}} .$$

The hyperbolic cosine potential function

- [PTW15] used the **hyperbolic cosine potential**,

$$\Gamma^t(x^t) := \underbrace{\sum_{i=1}^{n} e^{\gamma(x_i^t - t/n)}}_{\text{Overload potential: } \Phi_0^t} + \underbrace{\sum_{i=1}^{n} e^{-\gamma(x_i^t - t/n)}}_{\text{Underload potential}} .$$

- For the $(1+\beta)$-process, $\gamma = \Theta(\beta)$.

The hyperbolic cosine potential function

- [PTW15] used the **hyperbolic cosine potential**,

$$\Gamma^t(x^t) := \underbrace{\sum_{i=1}^{n} e^{\gamma(x_i^t - t/n)}}_{\text{Overload potential: } \Phi_0^t} + \underbrace{\sum_{i=1}^{n} e^{-\gamma(x_i^t - t/n)}}_{\text{Underload potential}} .$$

- For the $(1+\beta)$-process, $\gamma = \Theta(\beta)$.
- [PTW15] showed that $\mathbf{E}\left[\,\Gamma^{t+1} \mid \mathfrak{F}^t\,\right] \leq \Gamma^t \cdot \left(1 - \frac{c_1}{n}\right) + c_2$.

The hyperbolic cosine potential function

- [PTW15] used the **hyperbolic cosine potential**,

$$\Gamma^t(x^t) := \underbrace{\sum_{i=1}^{n} e^{\gamma(x_i^t - t/n)}}_{\text{Overload potential: } \Phi_0^t} + \underbrace{\sum_{i=1}^{n} e^{-\gamma(x_i^t - t/n)}}_{\text{Underload potential}} .$$

- For the $(1+\beta)$-process, $\gamma = \Theta(\beta)$.
- [PTW15] showed that $\mathbf{E}\left[\,\Gamma^{t+1} \mid \mathfrak{F}^t\,\right] \leq \Gamma^t \cdot \left(1 - \frac{c_1}{n}\right) + c_2$.
- This implies $\mathbf{E}\left[\,\Gamma^t\,\right] \leq cn$ for any $t \geq 0$.

The hyperbolic cosine potential function

- [PTW15] used the **hyperbolic cosine potential**,

$$\Gamma^t(x^t) := \underbrace{\sum_{i=1}^{n} e^{\gamma(x_i^t - t/n)}}_{\text{Overload potential: } \Phi_0^t} + \underbrace{\sum_{i=1}^{n} e^{-\gamma(x_i^t - t/n)}}_{\text{Underload potential}} .$$

- For the $(1+\beta)$-process, $\gamma = \Theta(\beta)$.
- [PTW15] showed that $\mathbf{E}\left[\,\Gamma^{t+1} \mid \mathfrak{F}^t\,\right] \leq \Gamma^t \cdot \left(1 - \frac{c_1}{n}\right) + c_2$.
- This implies $\mathbf{E}\left[\,\Gamma^t\,\right] \leq cn$ for any $t \geq 0$.
- By Markov's inequality, we get $\mathbf{Pr}\left[\,\Gamma^m \leq cn^3\,\right] \geq 1 - n^{-2}$ which implies

$$\mathbf{Pr}\left[\mathrm{Gap}(m) \leq \frac{1}{\gamma} \cdot (3\log n + \log c)\right] \geq 1 - n^{-2}.$$

The hyperbolic cosine potential function

- [PTW15] used the **hyperbolic cosine potential**,

$$\Gamma^t(x^t) := \underbrace{\sum_{i=1}^{n} e^{\gamma(x_i^t - t/n)}}_{\text{Overload potential: } \Phi_0^t} + \underbrace{\sum_{i=1}^{n} e^{-\gamma(x_i^t - t/n)}}_{\text{Underload potential}}.$$

- For the $(1+\beta)$-process, $\gamma = \Theta(\beta)$.
- [PTW15] showed that $\mathbf{E}\left[\,\Gamma^{t+1} \mid \mathfrak{F}^t\,\right] \leq \Gamma^t \cdot \left(1 - \frac{c_1}{n}\right) + c_2$.
- This implies $\mathbf{E}\left[\,\Gamma^t\,\right] \leq cn$ for any $t \geq 0$.
- By Markov's inequality, we get $\mathbf{Pr}\left[\,\Gamma^m \leq cn^3\,\right] \geq 1 - n^{-2}$ which implies

$$\mathbf{Pr}\left[\mathrm{Gap}(m) \leq \frac{1}{\gamma} \cdot (3\log n + \log c)\right] \geq 1 - n^{-2}.$$

- In [PTW15], $\gamma = \mathcal{O}(1)$ so the tightest gaps proved were $\mathcal{O}(\log n)$.

The hyperbolic cosine potential function

- [PTW15] used the **hyperbolic cosine potential**,

$$\Gamma^t(x^t) := \underbrace{\sum_{i=1}^{n} e^{\gamma(x_i^t - t/n)}}_{\text{Overload potential: } \Phi_0^t} + \underbrace{\sum_{i=1}^{n} e^{-\gamma(x_i^t - t/n)}}_{\text{Underload potential}}.$$

- For the $(1+\beta)$-process, $\gamma = \Theta(\beta)$.
- [PTW15] showed that $\mathbf{E}\left[\,\Gamma^{t+1} \mid \mathfrak{F}^t\,\right] \leq \Gamma^t \cdot \left(1 - \frac{c_1}{n}\right) + c_2$.
- This implies $\mathbf{E}\left[\,\Gamma^t\,\right] \leq cn$ for any $t \geq 0$.
- By Markov's inequality, we get $\mathbf{Pr}\left[\,\Gamma^m \leq cn^3\,\right] \geq 1 - n^{-2}$ which implies

$$\mathbf{Pr}\left[\operatorname{Gap}(m) \leq \frac{1}{\gamma} \cdot (3 \log n + \log c)\right] \geq 1 - n^{-2}.$$

- In [PTW15], $\gamma = \mathcal{O}(1)$ so the tightest gaps proved were $\mathcal{O}(\log n)$.
- [TW14] used this as a base case for Two-Choice in the heavily-loaded case.

Technique 1: Super-exponential potential functions

Technique 1: Super-exponential potential functions

- We define the following **super-exponential potential functions** for $0 \leq j < k$ and $t \geq 0$:

$$\Phi_j^t := \sum_{i=1}^{n} \exp\left(\gamma \cdot (\log n)^{j/k} \cdot \left(x_i^t - \frac{t}{n} - \frac{2}{\gamma} j (\log n)^{1/k}\right)^+\right),$$

Technique 1: Super-exponential potential functions

- We define the following **super-exponential potential functions** for $0 \leq j < k$ and $t \geq 0$:

$$\Phi_j^t := \sum_{i=1}^{n} \exp\left(\gamma \cdot (\log n)^{j/k} \cdot \left(x_i^t - \frac{t}{n} - \frac{2}{\gamma} j (\log n)^{1/k}\right)^+\right),$$

- We prove that when $y^t_{\delta_{k-j} \cdot n} < \frac{2}{\gamma} j (\log n)^{1/k}$ (*good step* $\mathcal{G}_j^t$), then

$$\mathbf{E}\left[\, \Phi_j^{t+1} \,\middle|\, \mathcal{G}_j^t \,\right] \leq \Phi_j^t \cdot \left(1 - \frac{1}{n}\right) + 2.$$

Technique 1: Super-exponential potential functions

- We define the following **super-exponential potential functions** for $0 \leq j < k$ and $t \geq 0$:

$$\Phi_j^t := \sum_{i=1}^{n} \exp\left(\gamma \cdot (\log n)^{j/k} \cdot \left(x_i^t - \frac{t}{n} - \frac{2}{\gamma} j (\log n)^{1/k}\right)^+\right),$$

- We prove that when $y^t_{\delta_{k-j} \cdot n} < \frac{2}{\gamma} j (\log n)^{1/k}$ (*good step* $\mathcal{G}_j^t$), then

$$\mathbf{E}\left[\, \Phi_j^{t+1} \,\middle|\, \mathcal{G}_j^t \,\right] \leq \Phi_j^t \cdot \left(1 - \frac{1}{n}\right) + 2.$$

- So, after $s = n \cdot \mathrm{polylog}(n)$ steps we get $\mathbf{E}\left[\, \Phi_j^{t+s} \middle| \Phi_0^t = \mathcal{O}(n), \cap_{\tau \in [t, t+s)} \mathcal{G}_j^\tau \,\right] = \mathcal{O}(n)$.

Technique 1: Super-exponential potential functions

- We define the following **super-exponential potential functions** for $0 \leq j < k$ and $t \geq 0$:
$$\Phi_j^t := \sum_{i=1}^{n} \exp\left(\gamma \cdot (\log n)^{j/k} \cdot \left(x_i^t - \frac{t}{n} - \frac{2}{\gamma} j (\log n)^{1/k}\right)^+\right),$$
- We prove that when $y^t_{\delta_{k-j} \cdot n} < \frac{2}{\gamma} j (\log n)^{1/k}$ (*good step* $\mathcal{G}_j^t$), then
$$\mathbf{E}\left[\, \Phi_j^{t+1} \,\middle|\, \mathcal{G}_j^t \,\right] \leq \Phi_j^t \cdot \left(1 - \frac{1}{n}\right) + 2.$$
- So, after $s = n \cdot \text{polylog}(n)$ steps we get $\mathbf{E}\left[\, \Phi_j^{t+s} \middle| \Phi_0^t = \mathcal{O}(n), \cap_{\tau \in [t, t+s)} \mathcal{G}_j^\tau \,\right] = \mathcal{O}(n)$.
- Observe that when $\Phi_0^t = \mathcal{O}(n)$ then at most $\mathcal{O}(n \cdot e^{-\gamma z})$ bins have load $\geq z$.

Technique 1: Super-exponential potential functions

- We define the following **super-exponential potential functions** for $0 \leq j < k$ and $t \geq 0$:

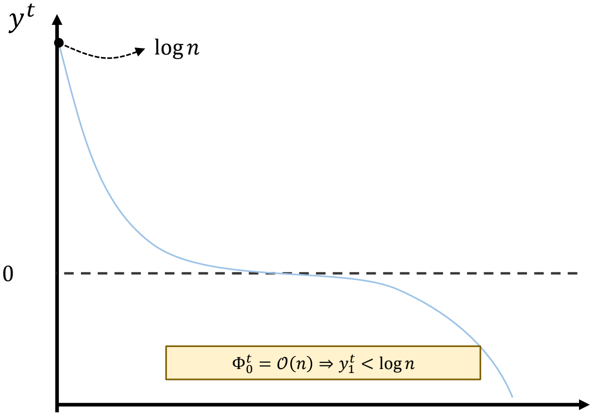
$$\Phi_j^t := \sum_{i=1}^{n} \exp\left(\gamma \cdot (\log n)^{j/k} \cdot \left(x_i^t - \frac{t}{n} - \frac{2}{\gamma} j (\log n)^{1/k}\right)^+\right),$$

- We prove that when $y^t_{\delta_{k-j} \cdot n} < \frac{2}{\gamma} j (\log n)^{1/k}$ (*good step* $\mathcal{G}_j^t$), then

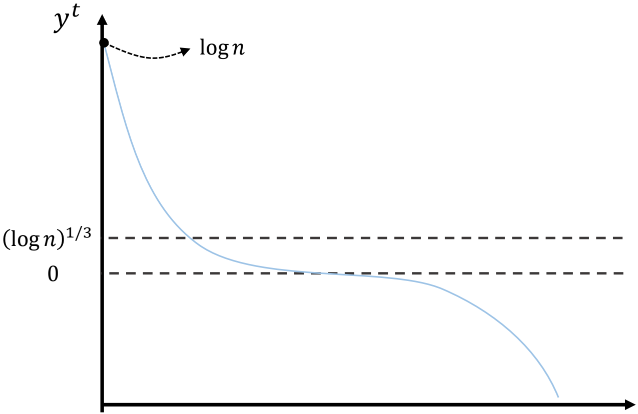
$$\mathbf{E}\left[\, \Phi_j^{t+1} \,\middle|\, \mathcal{G}_j^t \,\right] \leq \Phi_j^t \cdot \left(1 - \frac{1}{n}\right) + 2.$$

- So, after $s = n \cdot \mathrm{polylog}(n)$ steps we get $\mathbf{E}\left[\, \Phi_j^{t+s} \middle| \Phi_0^t = \mathcal{O}(n), \cap_{\tau \in [t, t+s)} \mathcal{G}_j^\tau \,\right] = \mathcal{O}(n)$.
- Observe that when $\Phi_0^t = \mathcal{O}(n)$ then at most $\mathcal{O}(n \cdot e^{-\gamma z})$ bins have load $\geq z$.
- Similarly, when $\Phi_j^t = \mathcal{O}(n)$, then $y_{\delta_{k-j-1} \cdot n} < \frac{2}{\gamma}(j+1)(\log n)^{1/k}$.

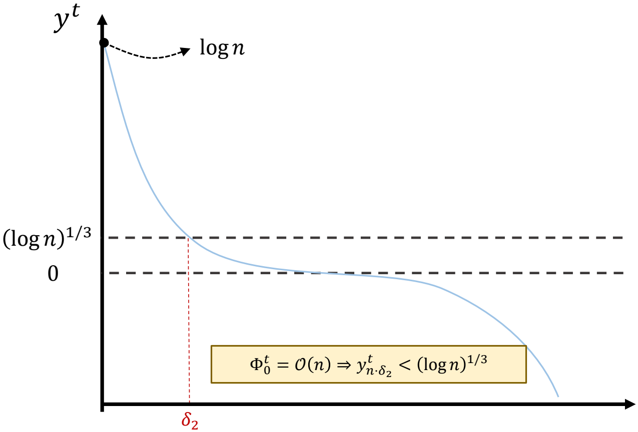
Proving $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$



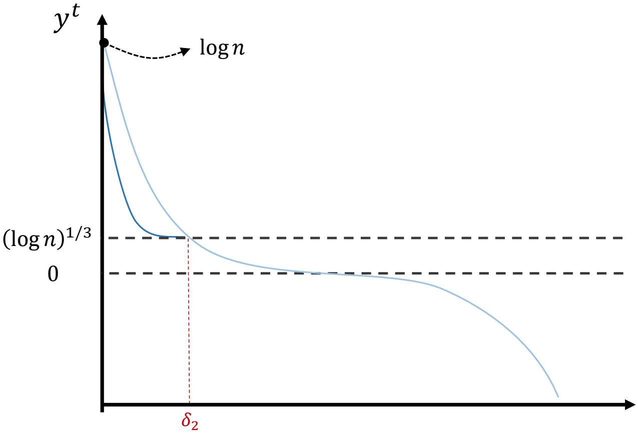
Proving $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$



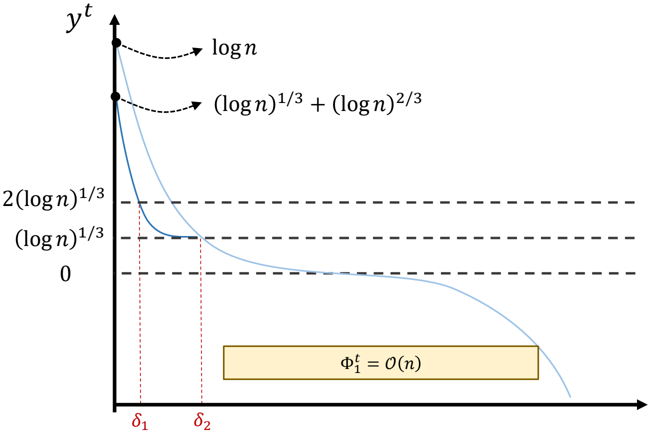
Proving $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$



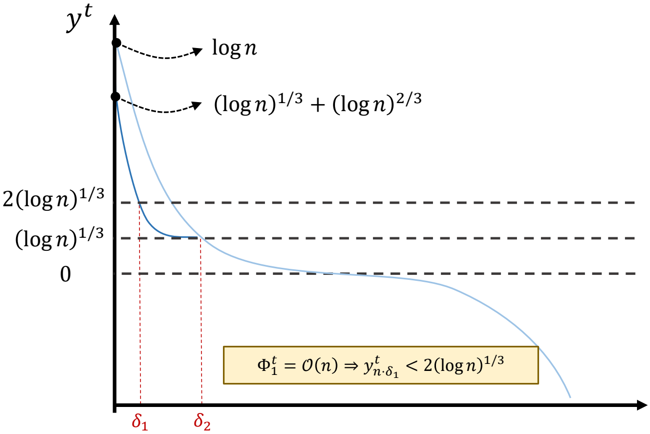
Proving $\mathrm{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$



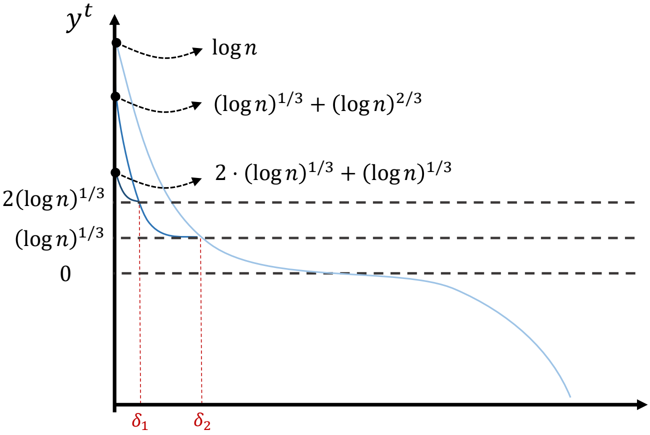
Proving $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$



Proving $\mathrm{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$



Proving $\mathrm{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$



Technique 2: Proving Φ_j^t is linear w.h.p.

- Assume that $\mathbf{E}\left[\,\Phi_j^\tau\,\right] = \mathcal{O}(n)$ and $\mathcal{G}_j^\tau$ for all $\tau \in [t, t + n \cdot \mathrm{polylog}(n))$.

Technique 2: Proving Φ_j^t is linear w.h.p.

- Assume that $\mathbf{E}\left[\, \Phi_j^\tau \,\right] = \mathcal{O}(n)$ and $\mathcal{G}_j^\tau$ for all $\tau \in [t, t + n \cdot \mathrm{polylog}(n))$.
- Using Markov's inequality we get that w.h.p. $\Phi_j^\tau = \mathrm{poly}(n)$.

Technique 2: Proving Φ_j^t is linear w.h.p.

- Assume that $\mathbf{E}\left[\,\Phi_j^\tau\,\right] = \mathcal{O}(n)$ and $\mathcal{G}_j^\tau$ for all $\tau \in [t, t + n \cdot \mathrm{polylog}(n))$.
- Using Markov's inequality we get that w.h.p. $\Phi_j^\tau = \mathrm{poly}(n)$.
- We define Ψ_j^t as Φ_j^t with sufficiently smaller γ.

Technique 2: Proving Φ_j^t is linear w.h.p.

- Assume that $\mathbf{E}\left[\,\Phi_j^\tau\,\right] = \mathcal{O}(n)$ and $\mathcal{G}_j^\tau$ for all $\tau \in [t, t + n \cdot \mathrm{polylog}(n))$.
- Using Markov's inequality we get that w.h.p. $\Phi_j^\tau = \mathrm{poly}(n)$.
- We define Ψ_j^t as Φ_j^t with sufficiently smaller γ.
- When $\Phi_j^\tau = \mathrm{poly}(n)$, then $|\Psi_j^{\tau+1} - \Psi_j^\tau| < n^{1/3}$.

Technique 2: Proving Φ_j^t is linear w.h.p.

- Assume that $\mathbf{E}\left[\,\Phi_j^\tau\,\right] = \mathcal{O}(n)$ and $\mathcal{G}_j^\tau$ for all $\tau \in [t, t + n \cdot \mathrm{polylog}(n))$.
- Using Markov's inequality we get that w.h.p. $\Phi_j^\tau = \mathrm{poly}(n)$.
- We define Ψ_j^t as Φ_j^t with sufficiently smaller γ.
- When $\Phi_j^\tau = \mathrm{poly}(n)$, then $|\Psi_j^{\tau+1} - \Psi_j^\tau| < n^{1/3}$.
- Hence, we apply a *bounded difference inequality* to get that w.h.p. $\Psi_j^\tau = \mathcal{O}(n)$.

Conclusion

Summary of results:

- Introduced a k-Quantile process which achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.

Conclusion

Summary of results:

- Introduced a k-Quantile process which achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
- Proved a lower bound of $\Omega(\log n / \log\log n)$ for any adaptive 1-Threshold and 1-Quantile process (**power of two queries**).

Conclusion

Summary of results:

- Introduced a k-Quantile process which achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
- Proved a lower bound of $\Omega(\log n / \log\log n)$ for any adaptive 1-Threshold and 1-Quantile process (**power of two queries**).
- Implications:

Conclusion

Summary of results:

- Introduced a k-Quantile process which achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
- Proved a lower bound of $\Omega(\log n / \log\log n)$ for any adaptive 1-Threshold and 1-Quantile process (**power of two queries**).
- Implications:
  - For $k = \Theta(\log\log n)$, we recover the $\text{Gap}(m) = \mathcal{O}(\log\log n)$ for Two-Choice (**power of two choices**).

Conclusion

Summary of results:

- Introduced a k-Quantile process which achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
- Proved a lower bound of $\Omega(\log n / \log\log n)$ for any adaptive 1-Threshold and 1-Quantile process (**power of two queries**).
- Implications:
  - For $k = \Theta(\log\log n)$, we recover the $\text{Gap}(m) = \mathcal{O}(\log\log n)$ for Two-Choice (**power of two choices**).
  - Tighter upper bounds for d-Thinning

Conclusion

Summary of results:

- Introduced a k-Quantile process which achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
- Proved a lower bound of $\Omega(\log n / \log\log n)$ for any adaptive 1-Threshold and 1-Quantile process (**power of two queries**).
- Implications:
  - For $k = \Theta(\log\log n)$, we recover the $\text{Gap}(m) = \mathcal{O}(\log\log n)$ for Two-Choice (**power of two choices**).
  - Tighter upper bounds for d-Thinning and $(1+\beta)$ for β close to 1.

Conclusion

Summary of results:

- Introduced a k-QUANTILE process which achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
- Proved a lower bound of $\Omega(\log n / \log\log n)$ for any adaptive 1-THRESHOLD and 1-QUANTILE process (**power of two queries**).
- Implications:
  - For $k = \Theta(\log\log n)$, we recover the $\text{Gap}(m) = \mathcal{O}(\log\log n)$ for TWO-CHOICE (**power of two choices**).
  - Tighter upper bounds for d-THINNING and $(1 + \beta)$ for β close to 1.
  - Graphical allocations on *dense expander graphs* achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(\log\log n)$.

Conclusion

Summary of results:

- Introduced a k-Quantile process which achieves w.h.p. $\mathrm{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
- Proved a lower bound of $\Omega(\log n / \log\log n)$ for any adaptive 1-Threshold and 1-Quantile process (**power of two queries**).
- Implications:
  - For $k = \Theta(\log\log n)$, we recover the $\mathrm{Gap}(m) = \mathcal{O}(\log\log n)$ for Two-Choice (**power of two choices**).
  - Tighter upper bounds for d-Thinning and $(1+\beta)$ for β close to 1.
  - Graphical allocations on *dense expander graphs* achieves w.h.p. $\mathrm{Gap}(m) = \mathcal{O}(\log\log n)$.

Future work:

Conclusion

Summary of results:

- Introduced a k-Quantile process which achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
- Proved a lower bound of $\Omega(\log n / \log\log n)$ for any adaptive 1-Threshold and 1-Quantile process (**power of two queries**).
- Implications:
  - For $k = \Theta(\log\log n)$, we recover the $\text{Gap}(m) = \mathcal{O}(\log\log n)$ for Two-Choice (**power of two choices**).
  - Tighter upper bounds for d-Thinning and $(1+\beta)$ for β close to 1.
  - Graphical allocations on *dense expander graphs* achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(\log\log n)$.

Future work:

- Prove *lower bounds* for adaptive k-Quantile for $k \geq 2$.

Conclusion

Summary of results:

- Introduced a k-QUANTILE process which achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
- Proved a lower bound of $\Omega(\log n / \log\log n)$ for any adaptive 1-THRESHOLD and 1-QUANTILE process (**power of two queries**).
- Implications:
  - For $k = \Theta(\log\log n)$, we recover the $\text{Gap}(m) = \mathcal{O}(\log\log n)$ for TWO-CHOICE (**power of two choices**).
  - Tighter upper bounds for d-THINNING and $(1+\beta)$ for β close to 1.
  - Graphical allocations on *dense expander graphs* achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(\log\log n)$.

Future work:

- Prove *lower bounds* for adaptive k-QUANTILE for $k \geq 2$.
- Prove similar *upper bounds* for k-THRESHOLD.

Conclusion

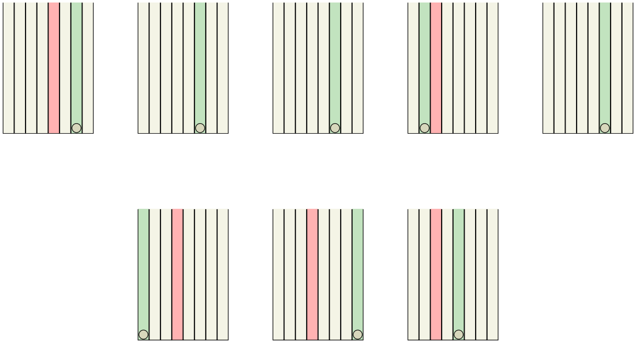
Summary of results:

- Introduced a k-QUANTILE process which achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(k \cdot (\log n)^{1/k})$.
- Proved a lower bound of $\Omega(\log n / \log\log n)$ for any adaptive 1-THRESHOLD and 1-QUANTILE process (**power of two queries**).
- Implications:
  - For $k = \Theta(\log\log n)$, we recover the $\text{Gap}(m) = \mathcal{O}(\log\log n)$ for TWO-CHOICE (**power of two choices**).
  - Tighter upper bounds for d-THINNING and $(1+\beta)$ for β close to 1.
  - Graphical allocations on *dense expander graphs* achieves w.h.p. $\text{Gap}(m) = \mathcal{O}(\log\log n)$.

Future work:

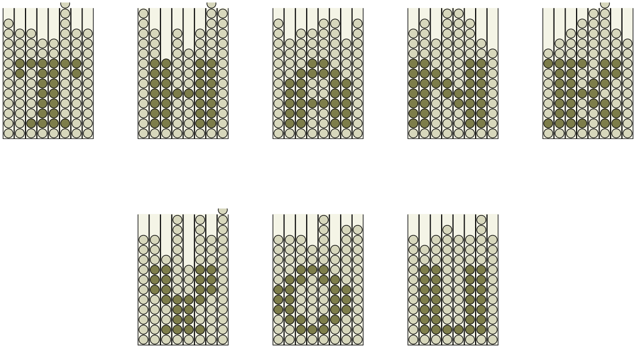
- Prove *lower bounds* for adaptive k-QUANTILE for $k \geq 2$.
- Prove similar *upper bounds* for k-THRESHOLD.
- Analyse TWO-CHOICE with *noise*.

Questions?



More visualisations: `dimitrioslos.com/itcs22`

Questions?



More visualisations: `dimitrioslos.com/itcs22`

Appendix

Appendix A: Detailed experimental results

$(1+\beta)$-process, for $\beta = 1/2$	k-Quantile				Two-Choice
	$k=1$	$k=2$	$k=3$	$k=4$	
20 : 2% **21** : 7% **22** : 9% **23** : 26% **24** : 27% **25** : 14% **26** : 6% **27** : 3% **28** : 4% **29** : 1% **34** : 1%	**8** : 28% **9** : 42% **10** : 18% **11** : 7% **12** : 3% **14** : 1% **15** : 1%	**4** : 72% **5** : 26% **6** : 2%	**3** : 46% **4** : 54%	**3** : 79% **4** : 21%	**3** : 100%

Table: Empirical distribution of the Gap for $n = 10^5$ bins and $m = 1000 \cdot n$ balls.

Appendix B: Random d-regular graphs

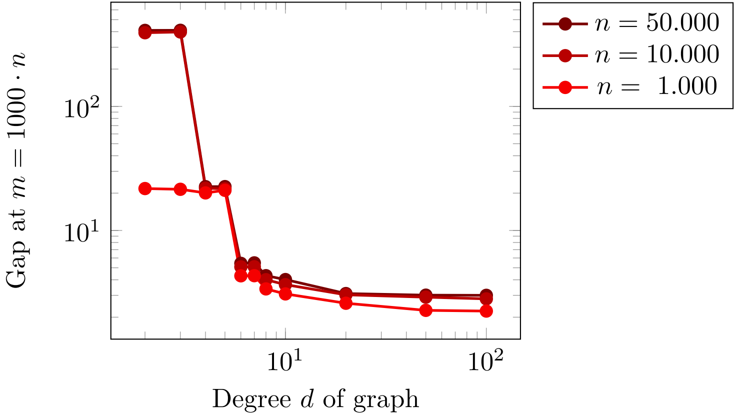


Figure: Average Gap for graphical allocations on d-regular graphs generated using [SW99] for $n \in \{10^3, 10^4, 5 \cdot 10^4\}$ bins and $m = 1000 \cdot n$ balls.

Bibliography I

- Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal, *Balanced allocations*, SIAM J. Comput. **29** (1999), no. 1, 180–200.
- P. Berenbrink, A. Czumaj, A. Steger, and B. Vöcking, *Balanced allocations: the heavily loaded case*, SIAM J. Comput. **35** (2006), no. 6, 1350–1385.
- O. N. Feldheim and O. Gurel-Gurevich, *The power of thinning in balanced allocation*, Electron. Commun. Probab. **26** (2021), Paper No. 34, 8.
- O. N. Feldheim and J. Li, *Load balancing under d-thinning*, Electron. Commun. Probab. **25** (2020), Paper No. 1, 13.
- G. H. Gonnet, *Expected length of the longest probe sequence in hash code searching*, J. Assoc. Comput. Mach. **28** (1981), no. 2, 289–304.
- K. Iwama and A. Kawachi, *Approximated two choices in randomized load balancing*, Algorithms and Computation, Springer Berlin Heidelberg, 2005, pp. 545–557.

Bibliography II

- R. M. Karp, M. Luby, and F. Meyer auf der Heide, *Efficient PRAM simulation on a distributed memory machine*, Algorithmica **16** (1996), no. 4-5, 517–542.
- D. Los, T. Sauerwald, and J. Sylvester, *Balanced allocations: Caching and packing, twinning and thinning*, 33rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'22), SIAM, 2022, pp. 1847–1874.
- M. Mitzenmacher, *On the analysis of randomized load balancing schemes*, Theory Comput. Syst. **32** (1999), no. 3, 361–386.
- Y. Peres, K. Talwar, and U. Wieder, *Graphical balanced allocations and the* $(1+\beta)$*-choice process*, Random Structures Algorithms **47** (2015), no. 4, 760–775.
- M. Raab and A. Steger, *"Balls into bins"—a simple and tight analysis*, 2nd International Workshop on Randomization and Computation (RANDOM'98), vol. 1518, Springer, 1998, pp. 159–170.
- A. Steger and N. C. Wormald, *Generating random regular graphs quickly*, Combinatorics, Probability and Computing **8** (1999), no. 4, 377–396.

Bibliography III

- Kunal Talwar and Udi Wieder, *Balanced allocations: A simple proof for the heavily loaded case*, 41st International Colloquium on Automata, Languages, and Programming (ICALP'14), vol. 8572, Springer, 2014, pp. 979–990.