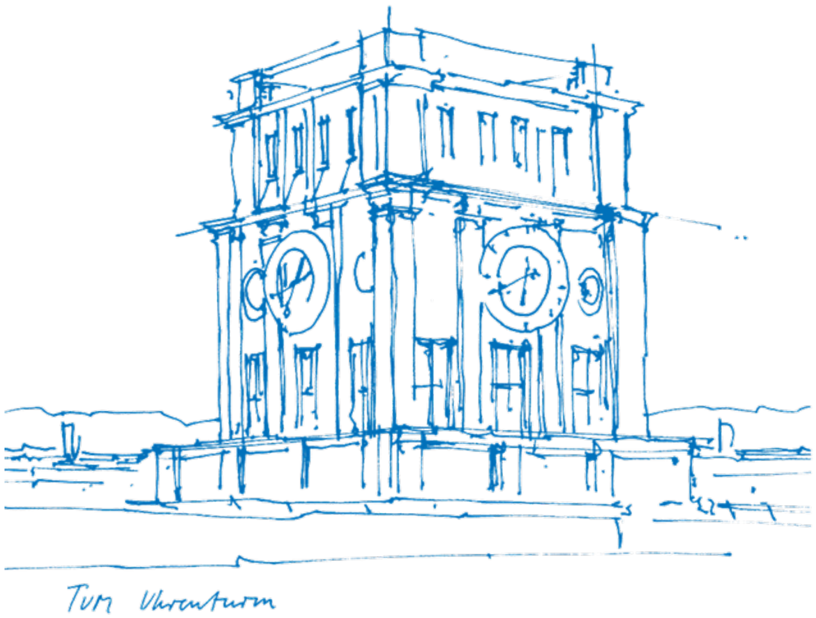
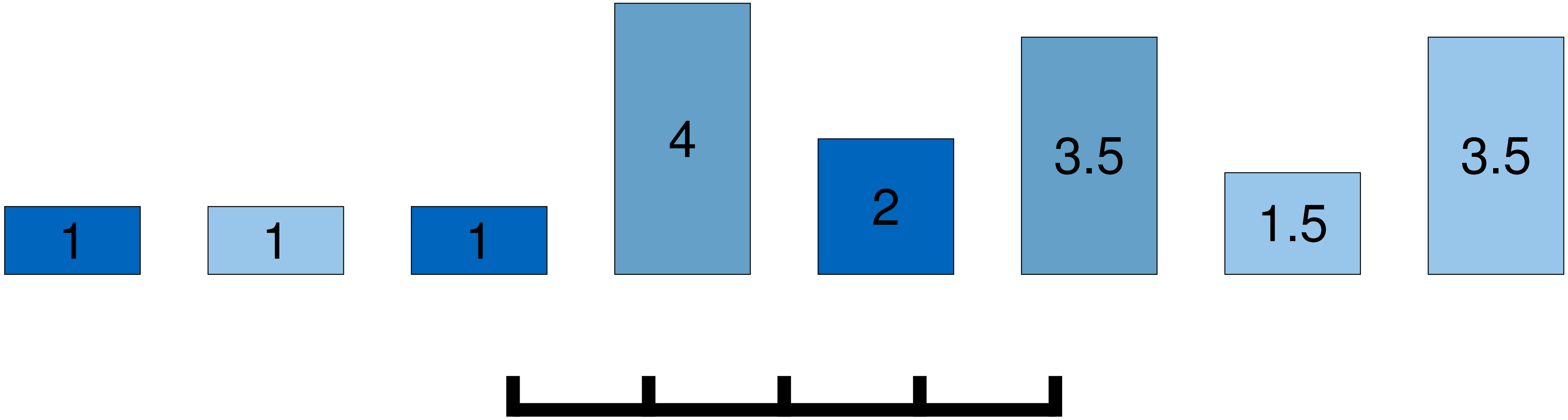
Scheduling in the Random-Order Model

Susanne Albers, Maximilian Janke
Technical University of Munich
Department of Computer Science
Chair of Algorithms and Complexity



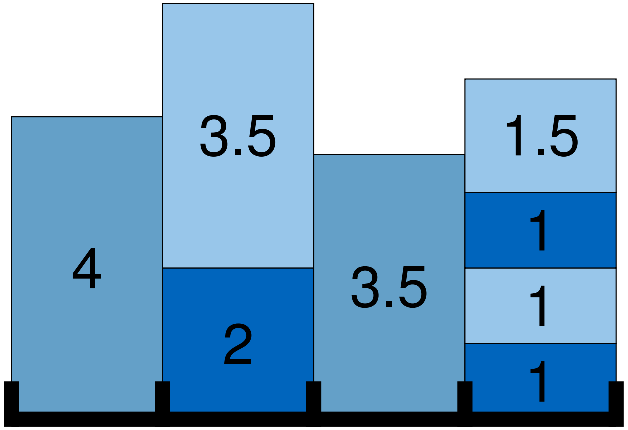
Makespan Minimization

Task: Assign **jobs** to **machines**.



Makespan Minimization

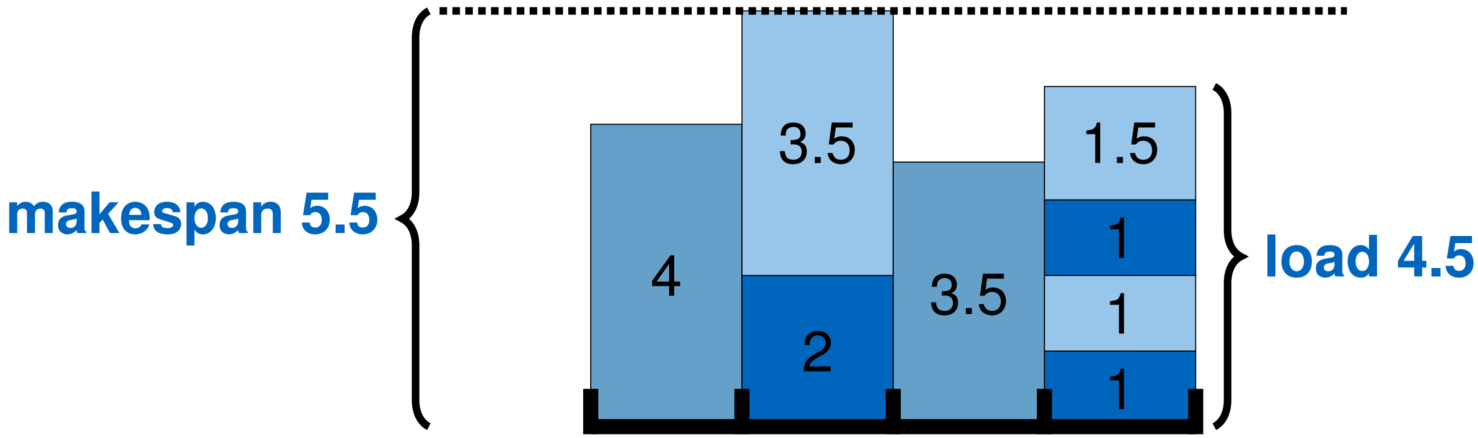
Task: Assign **jobs** to **machines**.



Makespan Minimization

Task: Assign **jobs** to **machines**.

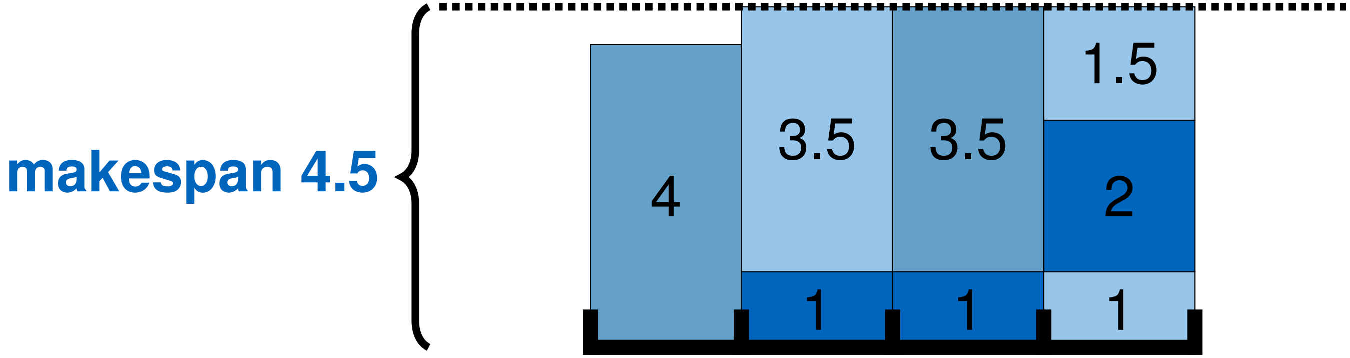
Goal: Minimize the **makespan**.



Makespan Minimization

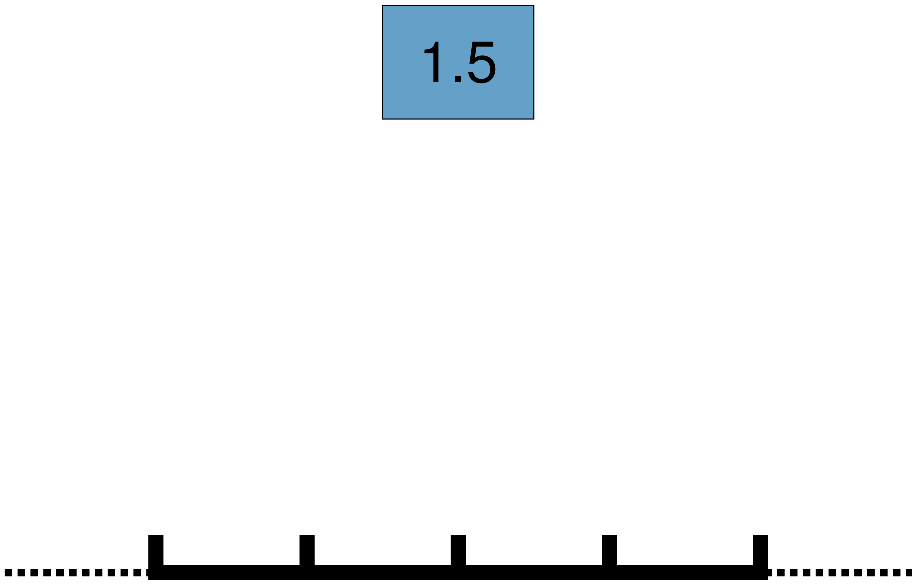
Task: Assign **jobs** to **machines**.

Goal: Minimize the **makespan**.



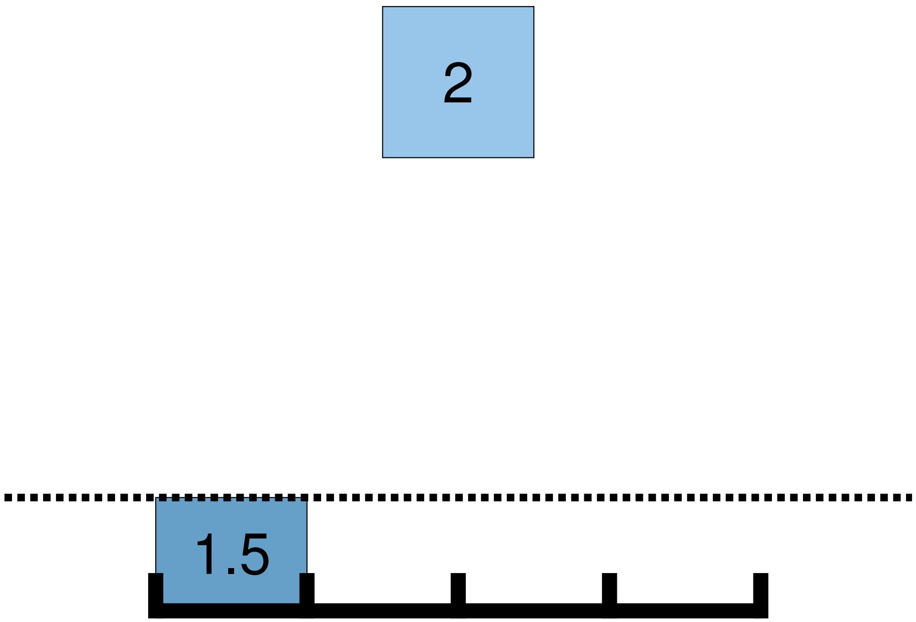
Online Algorithm

Jobs are revealed one by one and assigned immediately.



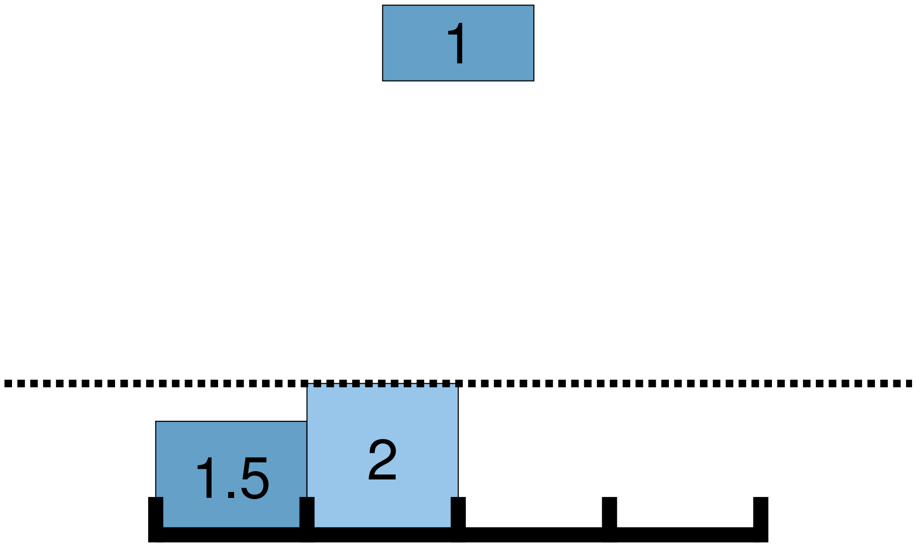
Online Algorithm

Jobs are revealed one by one and assigned immediately.



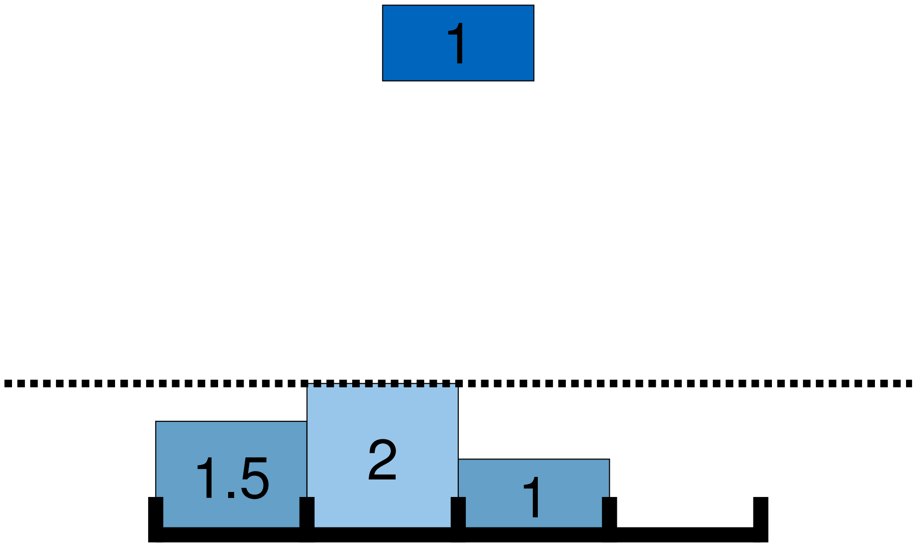
Online Algorithm

Jobs are revealed one by one and assigned immediately.



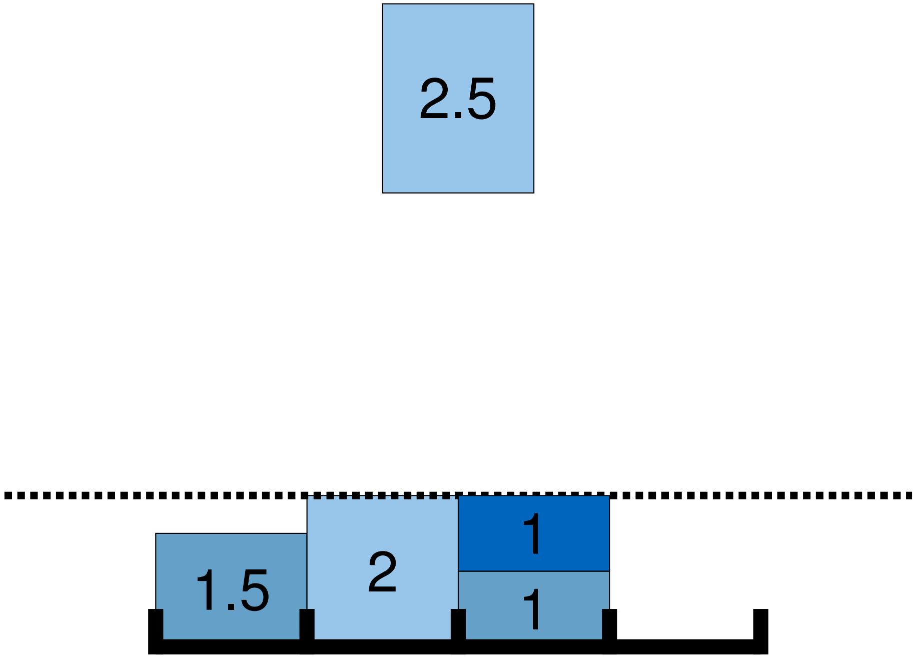
Online Algorithm

Jobs are revealed one by one and assigned immediately.



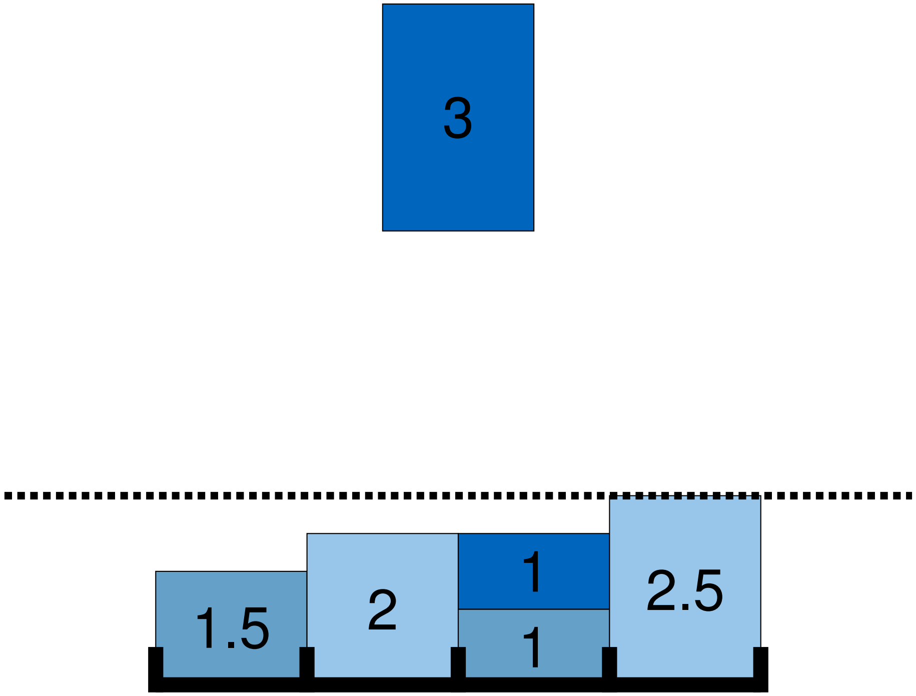
Online Algorithm

Jobs are revealed one by one and assigned immediately.



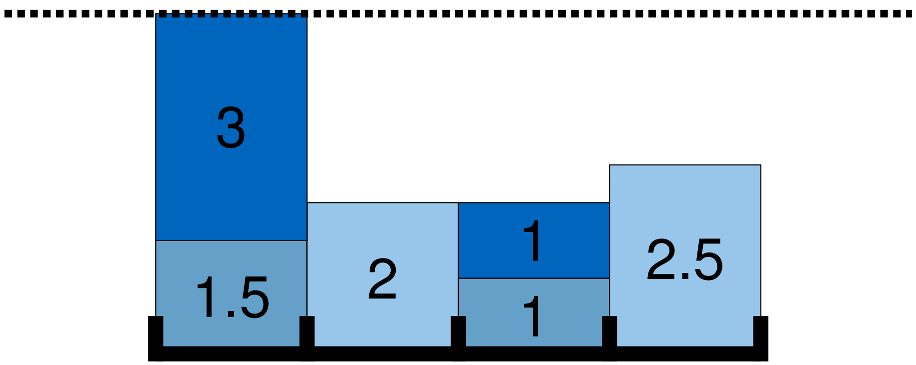
Online Algorithm

Jobs are revealed one by one and assigned immediately.



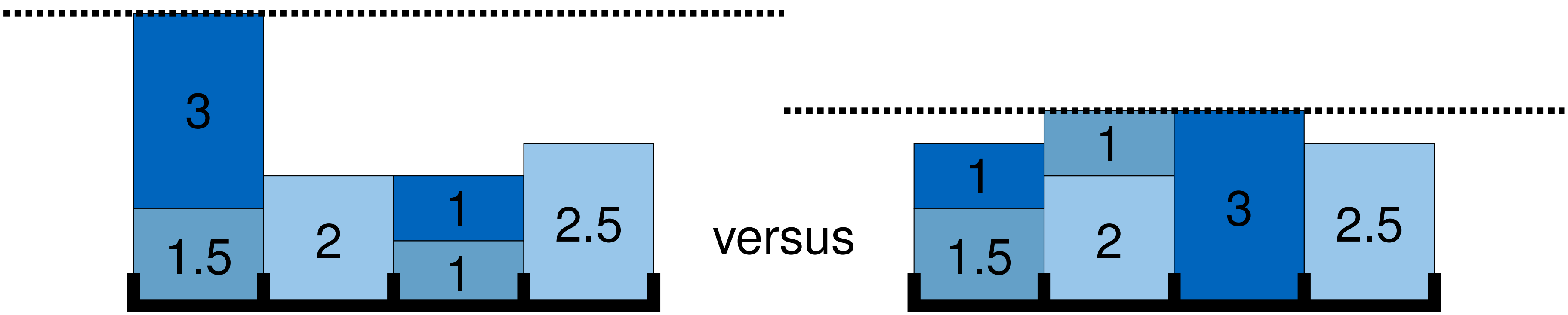
Online Algorithm

Jobs are revealed one by one and assigned immediately.



Competitive ratio

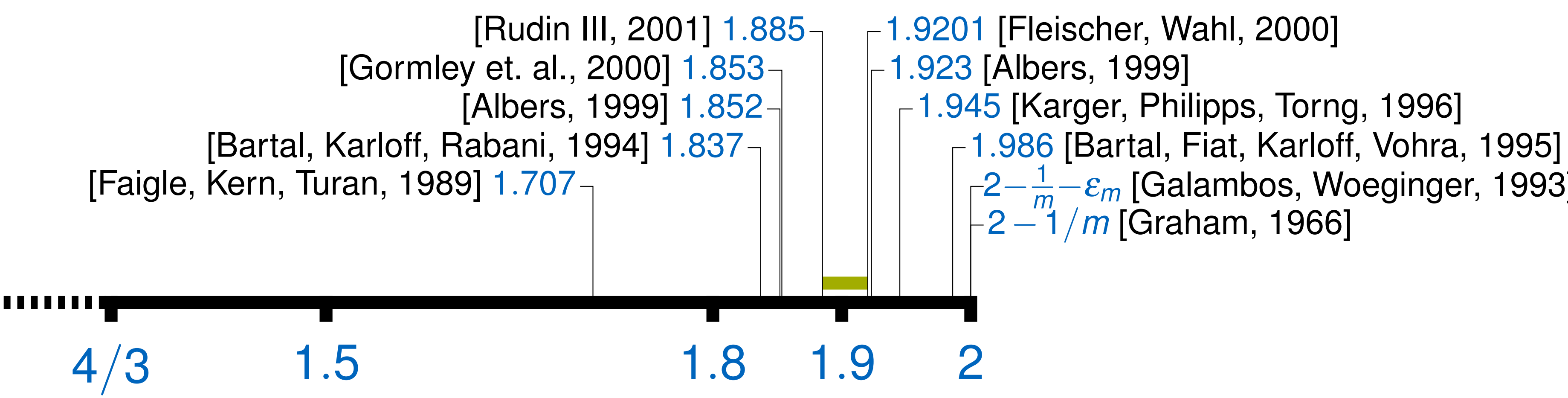
The **worst** ratio an **adversary** can cause.



Literature

Adversarial deterministic algorithms.

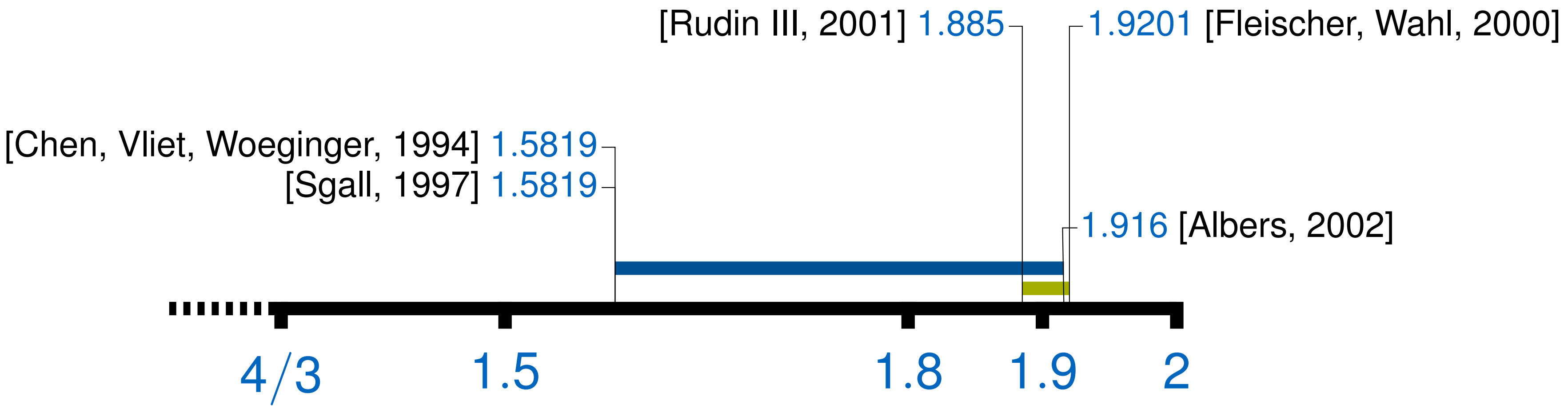
[Günther, Maurer, Megow and Wiese, 2013] and [Chen, Ye, Zhang, 2015] approximate the optimum online algorithm.



Literature

Adversarial deterministic algorithms.

Adversarial randomized algorithms.

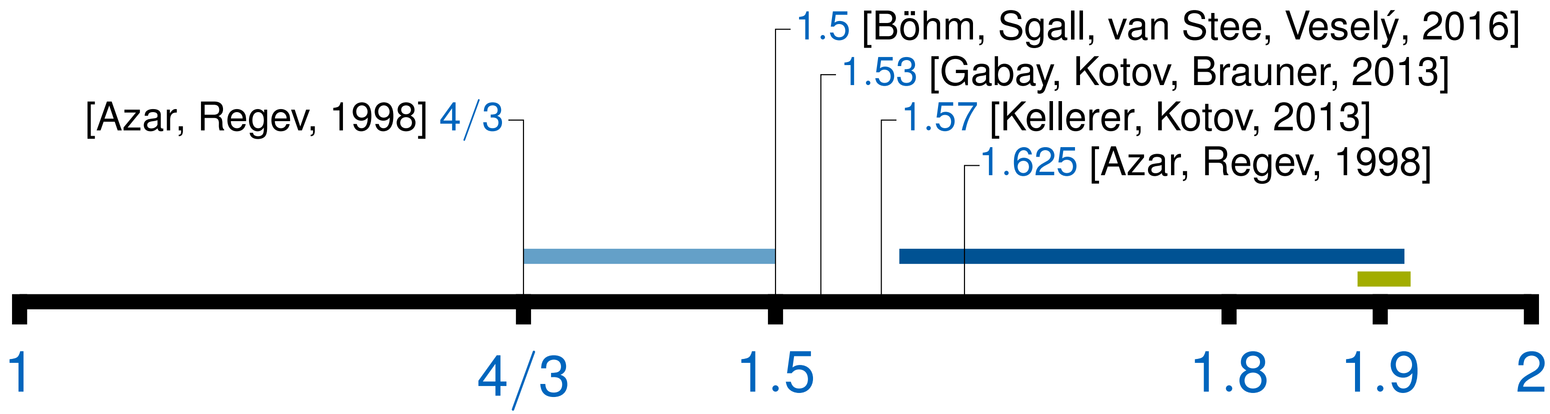


Literature

Adversarial deterministic algorithms.

Adversarial randomized algorithms.

Deterministic algorithms which know OPT. (Bin Stretching)



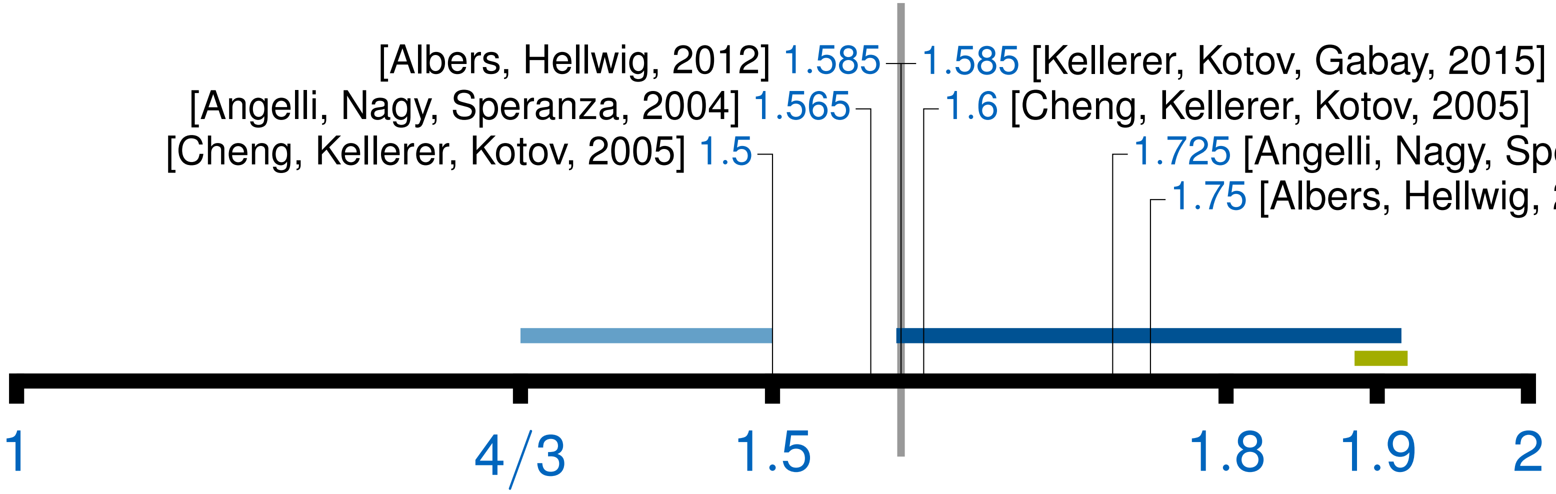
Literature

Adversarial deterministic algorithms.

Adversarial randomized algorithms.

Deterministic algorithms which know OPT. (Bin Stretching)

Deterministic algorithms which know the total processing volume.



Literature

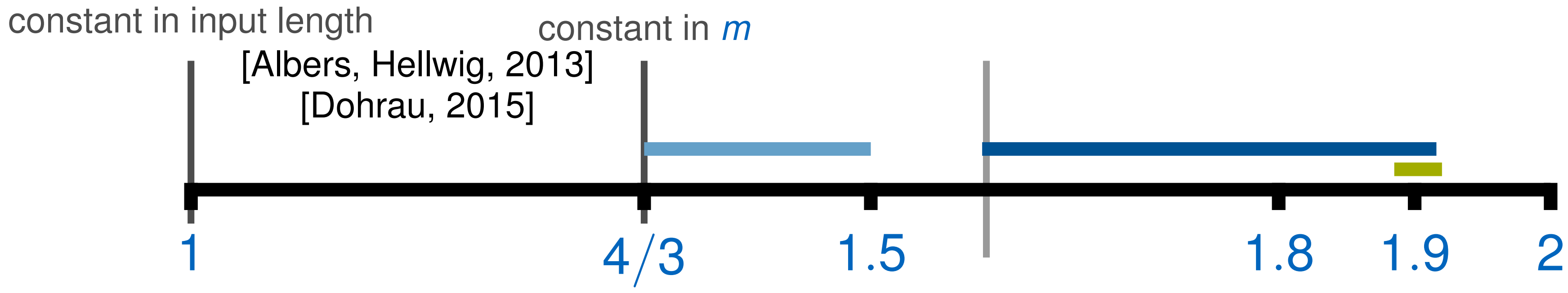
Adversarial deterministic algorithms.

Adversarial randomized algorithms.

Deterministic algorithms which know OPT. (Bin Stretching)

Deterministic algorithms which know the total processing volume.

Deterministic algorithms which advice.



I made some corrections here.

Literature

Adversarial deterministic algorithms.

Adversarial randomized algorithms.

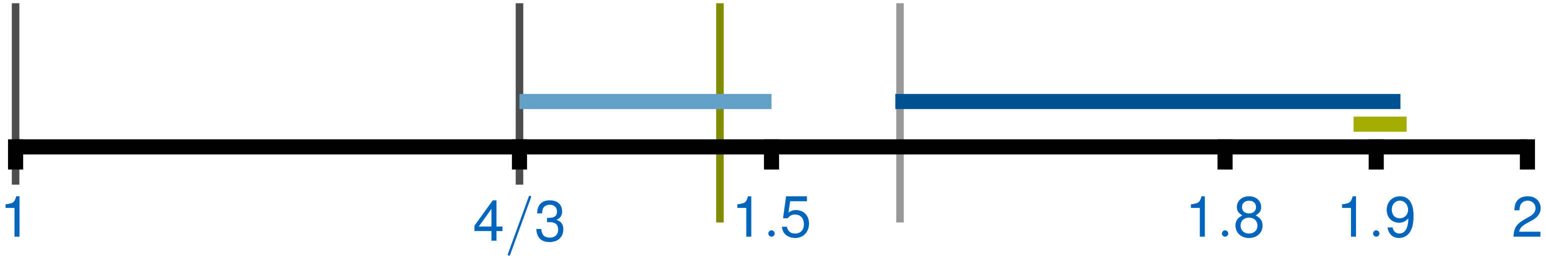
Deterministic algorithms which know OPT. (Bin Stretching)

Deterministic algorithms which know the total processing volume.

Deterministic algorithms which advice.

Buffer Reordering.

1.4659 [Englert, Özmen, Westermann, 2008]



Literature

Adversarial deterministic algorithms.

Adversarial randomized algorithms.

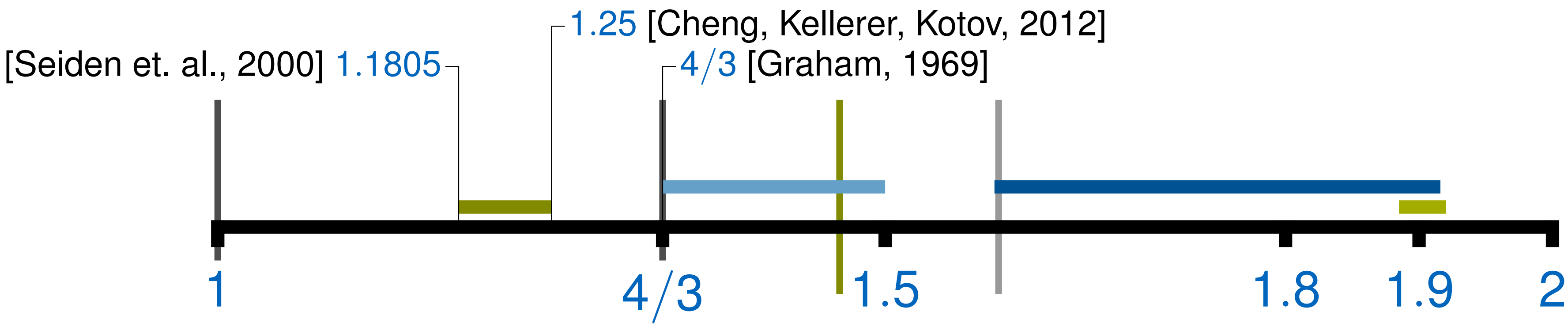
Deterministic algorithms which know OPT. (Bin Stretching)

Deterministic algorithms which know the total processing volume.

Deterministic algorithms which advice.

Buffer Reordering.

Job processing times ordered decreasingly.



Literature

Adversarial deterministic algorithms.

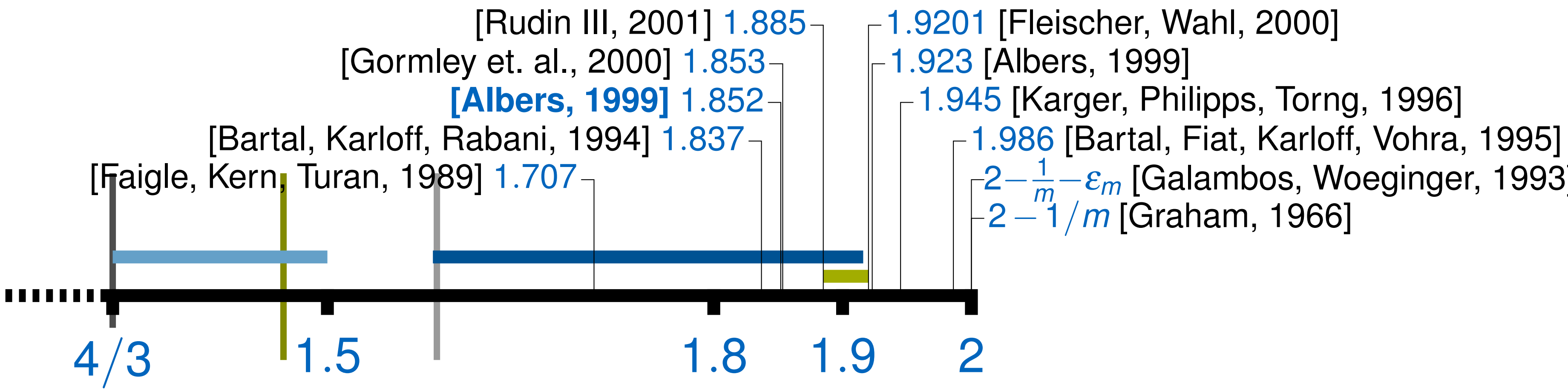
Adversarial randomized algorithms.

Deterministic algorithms which know OPT. (Bin Stretching)

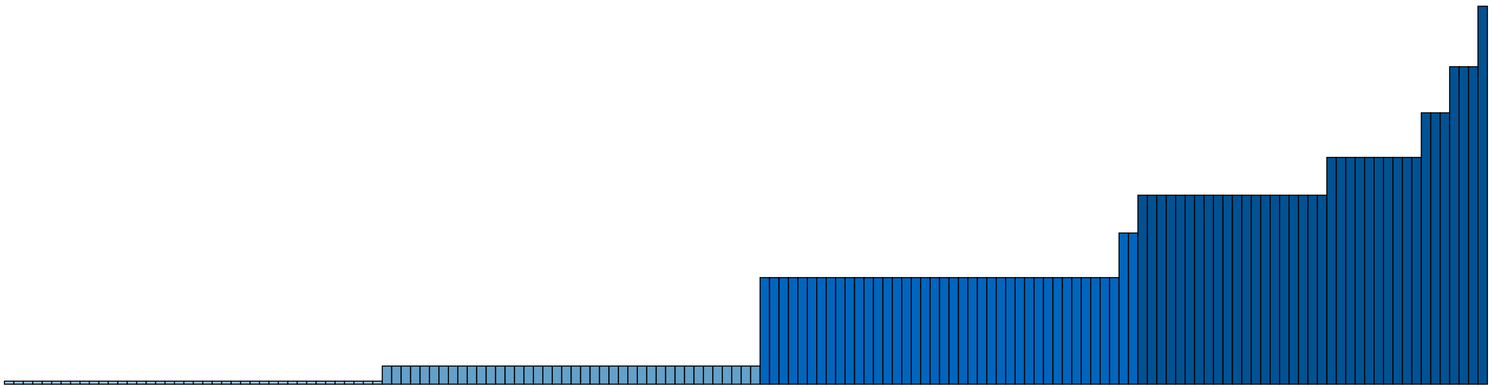
Deterministic algorithms which know the total processing volume.

Deterministic algorithms which advice.

Buffer Reordering.

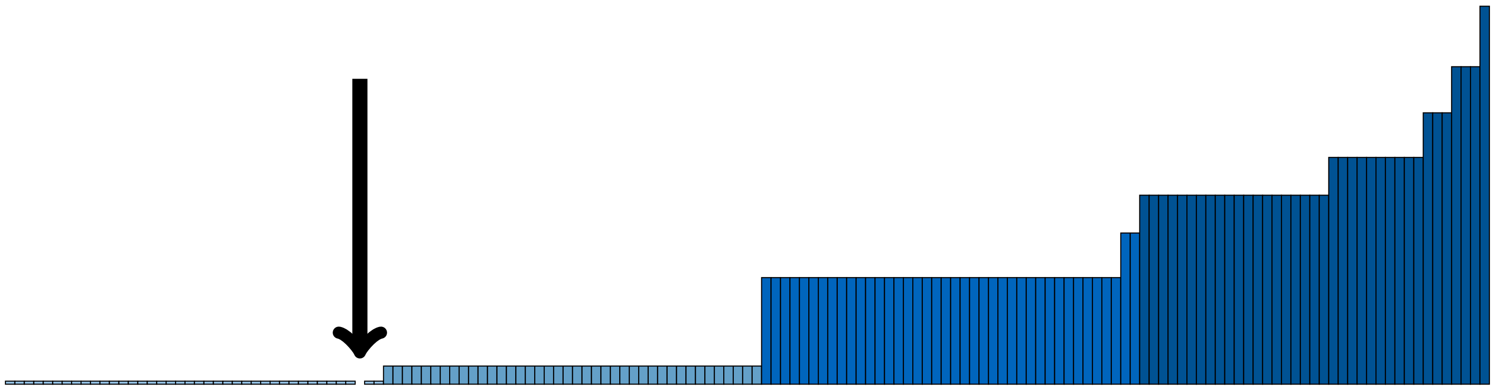


Is **worst-case analysis** too pessimistic?



(Lower bound of [Albers, 1999] for $m = 40$ machines.)

Is **worst-case analysis** too pessimistic?

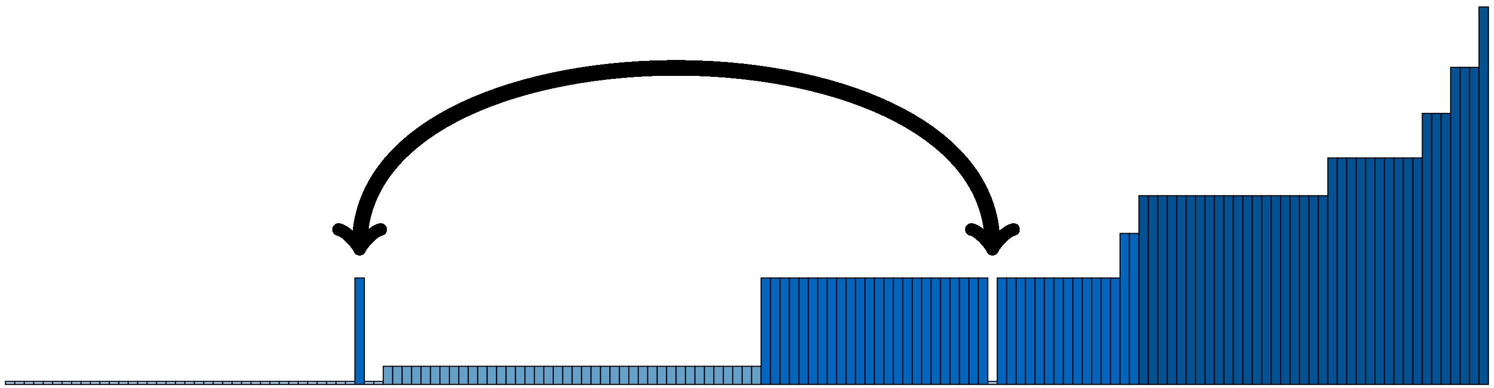


(Lower bound of [Albers, 1999] for $m = 40$ machines.)

The argument of Albers does not hold anymore if we

- Delete any job. (The lower bound would be 1.714)

Is **worst-case analysis** too pessimistic?

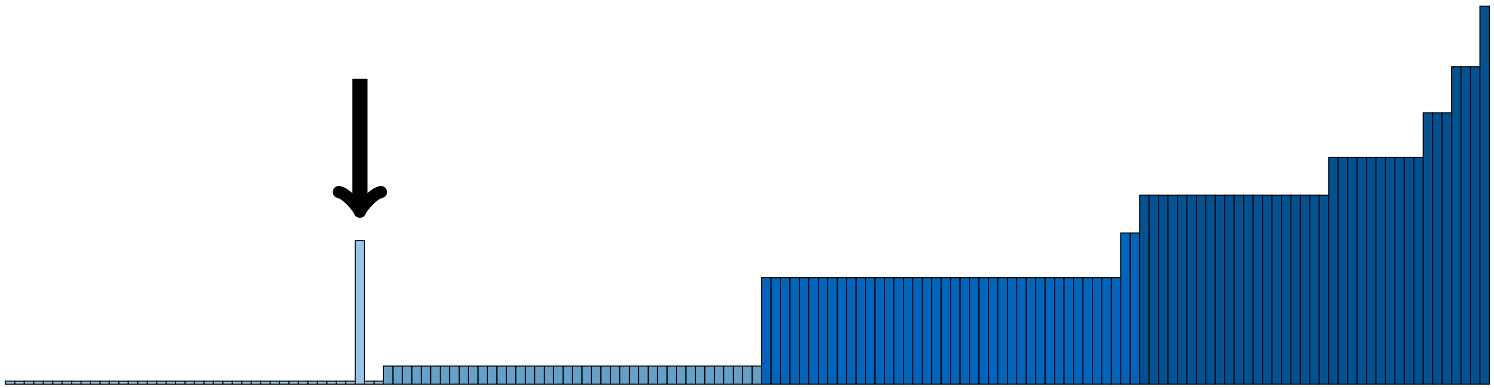


(Lower bound of [Albers, 1999] for $m = 40$ machines.)

The argument of Albers does not hold anymore if we

- Delete any job.
- Swap (non-identical) jobs.

Is **worst-case analysis** too pessimistic?

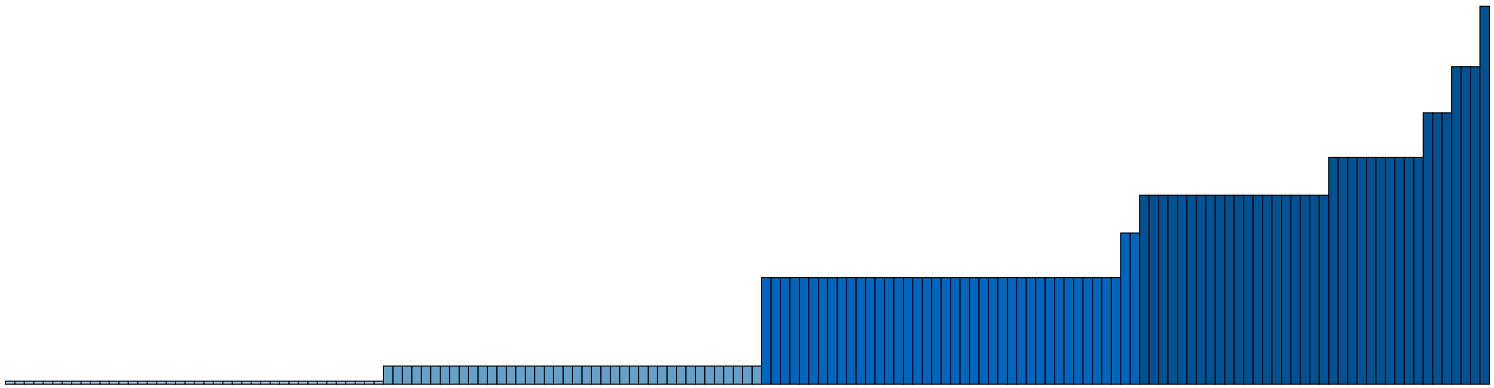


(Lower bound of [Albers, 1999] for $m = 40$ machines.)

The argument of Albers does not hold anymore if we

- Delete any job.
- Swap (non-identical) jobs.
- Change a job size significantly.

Is **worst-case analysis** too pessimistic?

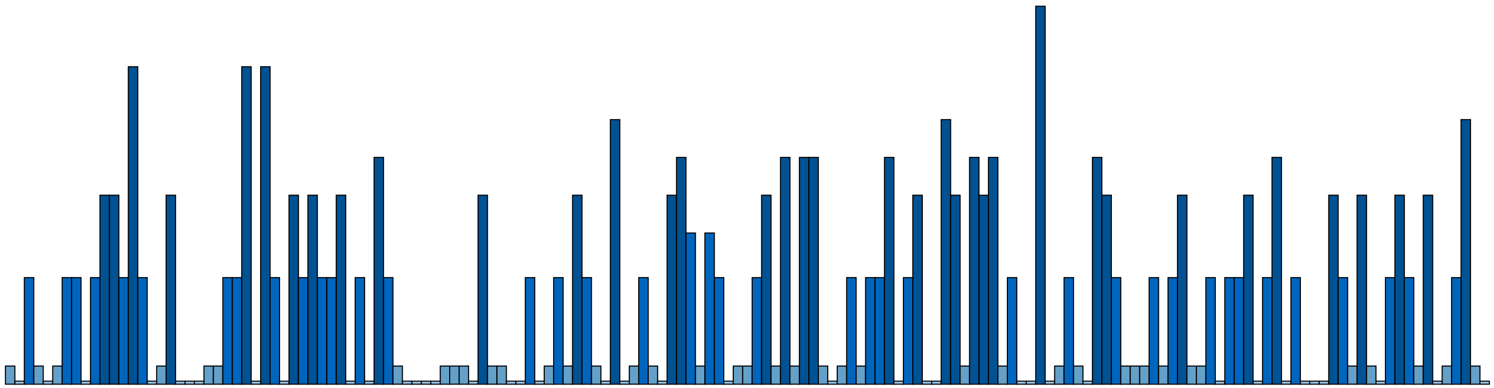


(Lower bound of [Albers, 1999] for $m = 40$ machines.)

The argument of Albers does not hold anymore if we

- Swap (non-identical) jobs. **How important is job order?**

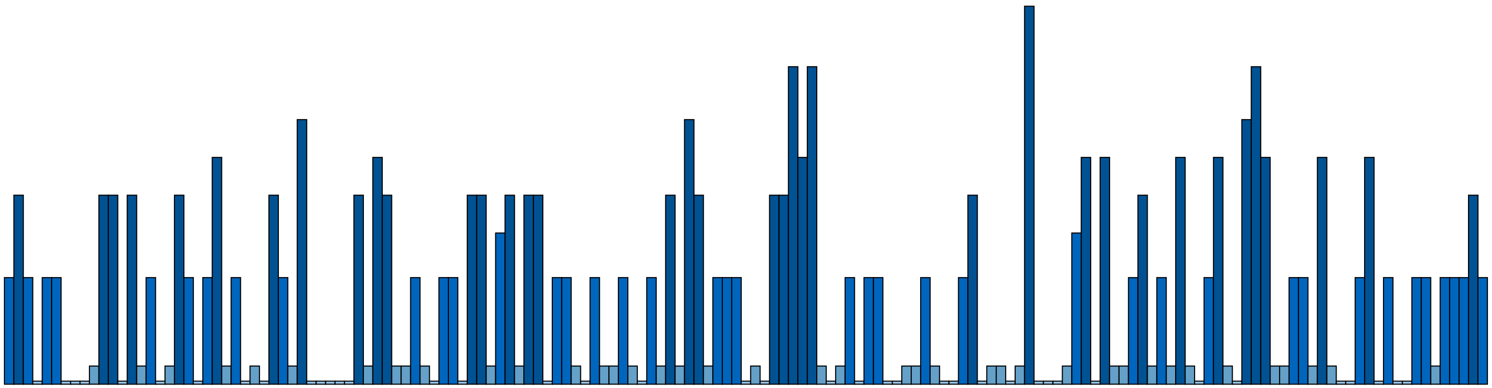
Random-Order Analysis



(Random permutation of [Albers, 1999] for $m = 40$ machines.)

Adversary chooses **job set**, order is **uniformly random**.

Random-Order Analysis

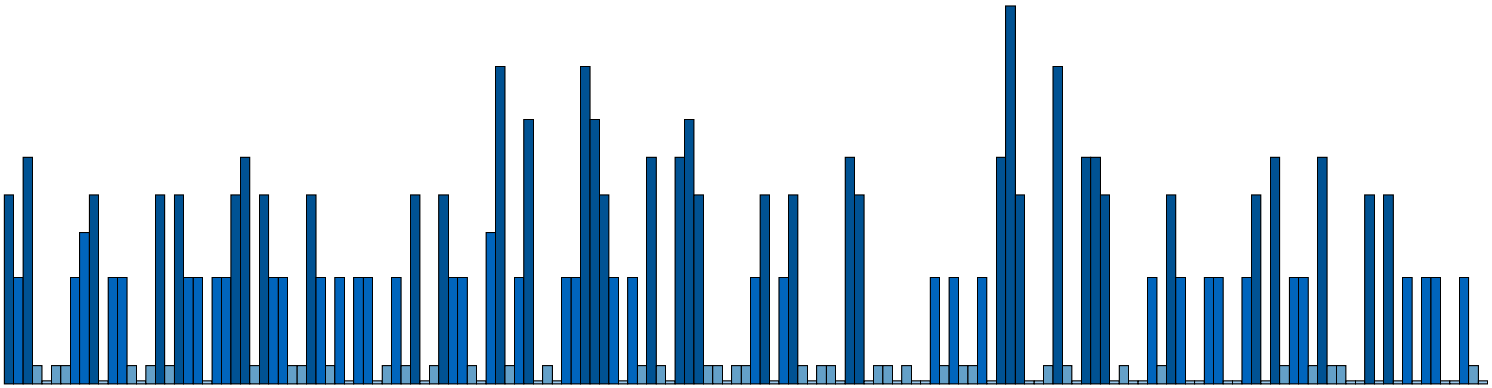


(Random permutation of [Albers, 1999] for $m = 40$ machines.)

Adversary chooses **job set**, order is **uniformly random**.

Expected makespan of A versus **optimum makespan**.

Random-Order Analysis



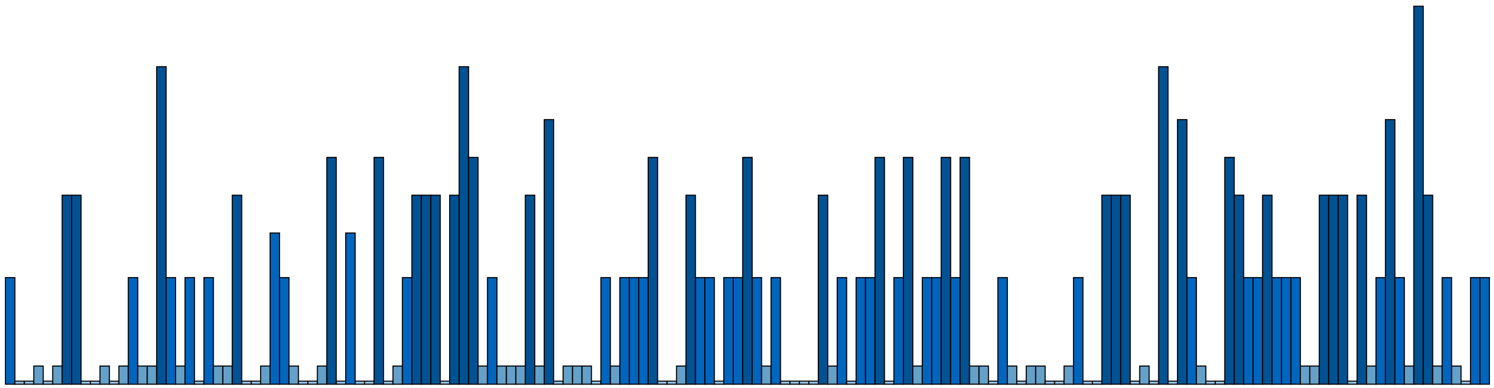
(Random permutation of [Albers, 1999] for $m = 40$ machines.)

Adversary chooses **job set**, order is **uniformly random**.

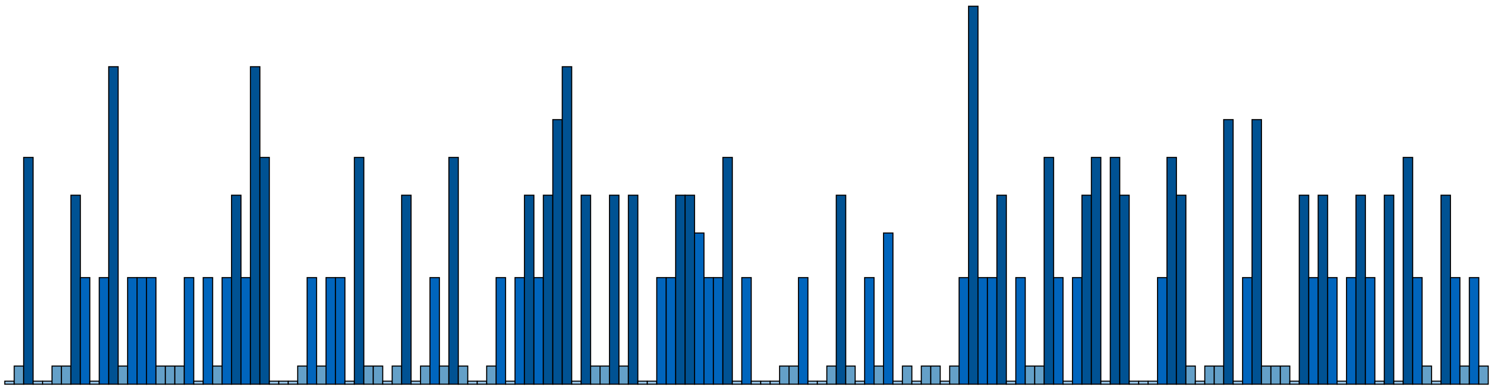
Expected makespan of A versus **optimum makespan**.

Competitive ratio: Worst ratio the adversary may cause.

Is **Random-Order Analysis** too optimistic?

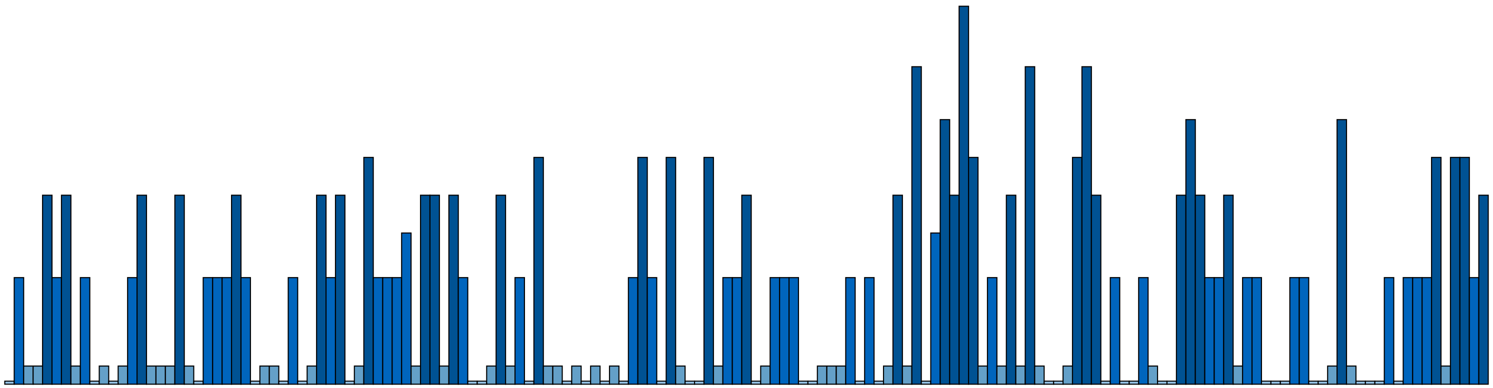


Is **Random-Order Analysis** too optimistic?



We are **nearly c-competitive** iff we are $(c + o_m(1))$-competitive with probability $1 - o_m(1)$ after random permutation and have a constant competitive ratio on worst-case sequences.

Is **Random-Order Analysis** too optimistic?



We are **nearly c-competitive** iff we are $(c + o_m(1))$-competitive with probability $1 - o_m(1)$ after random permutation and have a constant competitive ratio on worst-case sequences.

A **nearly c-competitive** online algorithm is **c-competitive in the random-order model** for $m \to \infty$.

The Random-Order model

Scheduling for the random-order model is not well researched yet:

- [Osborn and Torng, 2008] show Greedy remains 2-competitive.

The Random-Order model

Scheduling for the random-order model is not well researched yet:

- [Osborn and Torng, 2008] show Greedy remains 2-competitive.
- [Göbel, Kesselheim and Tönnis, 2015] and [Molinaro, 2017] analyze different scheduling problems.

Our results

Our new algorithm is **nearly 1.8478-competitive**.

We also provide lower bounds.

Our results

Our new algorithm is **nearly 1.8478-competitive**.

Profit from random-order arrival.

- The **Load Lemma**. Identifying **time measures**.

Profit from random-order arrival.

- The **Load Lemma**. Identifying **time measures**.
- **Large Job Magic**. Large jobs for free.

Profit from random-order arrival.

- The **Load Lemma**. Identifying **time measures**.
- **Large Job Magic**. Large jobs for free.
- **Oracle-like properties**. A taste of semi-online scheduling.

Profit from random-order arrival.

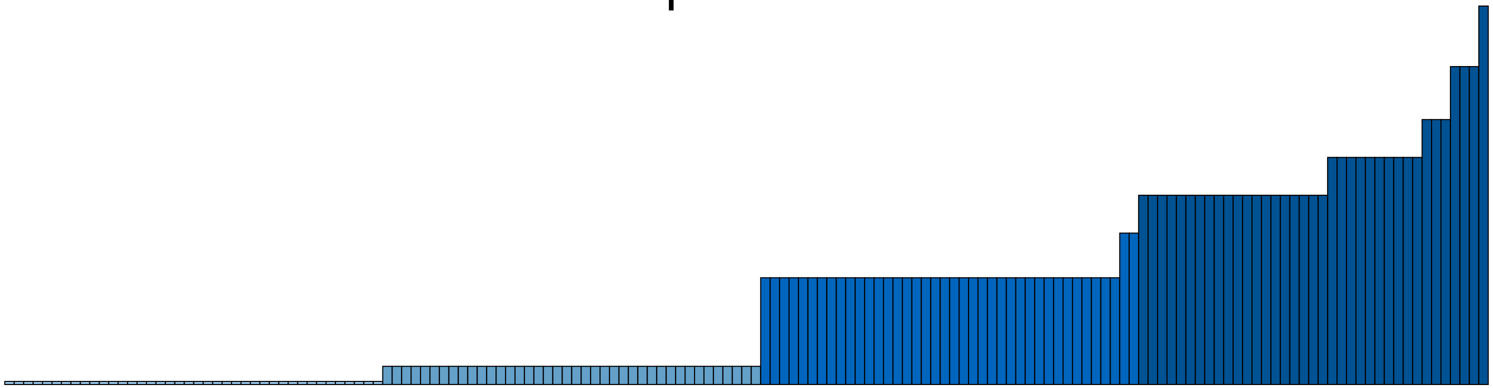
- The **Load Lemma**. Identifying **time measures**.
- **Large Job Magic**. Large jobs for free.
- **Oracle-like properties**. A taste of semi-online scheduling.
- **???**

Profit from random-order arrival.

- **The Load Lemma. Identifying time measures.**
- **Large Job Magic**. Large jobs for free.
- **Oracle-like properties**. A taste of semi-online scheduling.
- **???**
- **Profit**. A nearly 1.8478-competitive algorithm.

Different time measures.

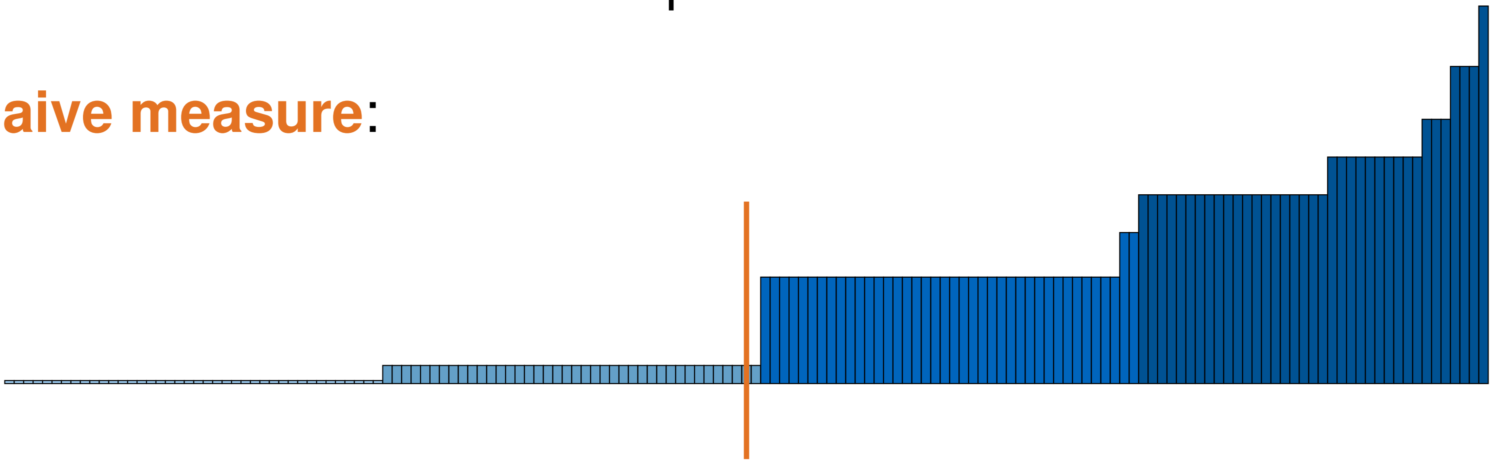
When have we seen half the sequence?



Different time measures.

When have we seen half the sequence?

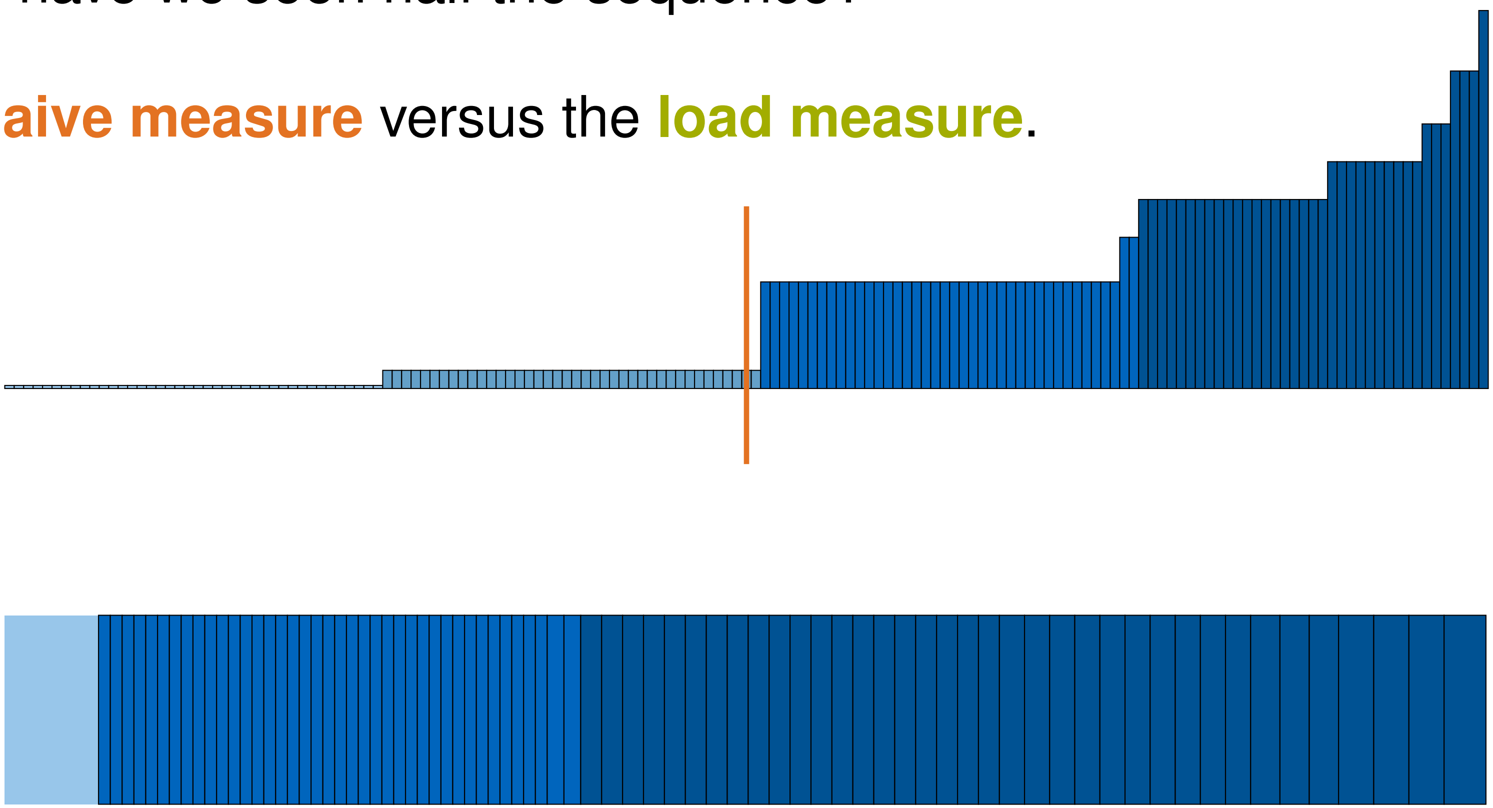
The **naive measure**:



Different time measures.

When have we seen half the sequence?

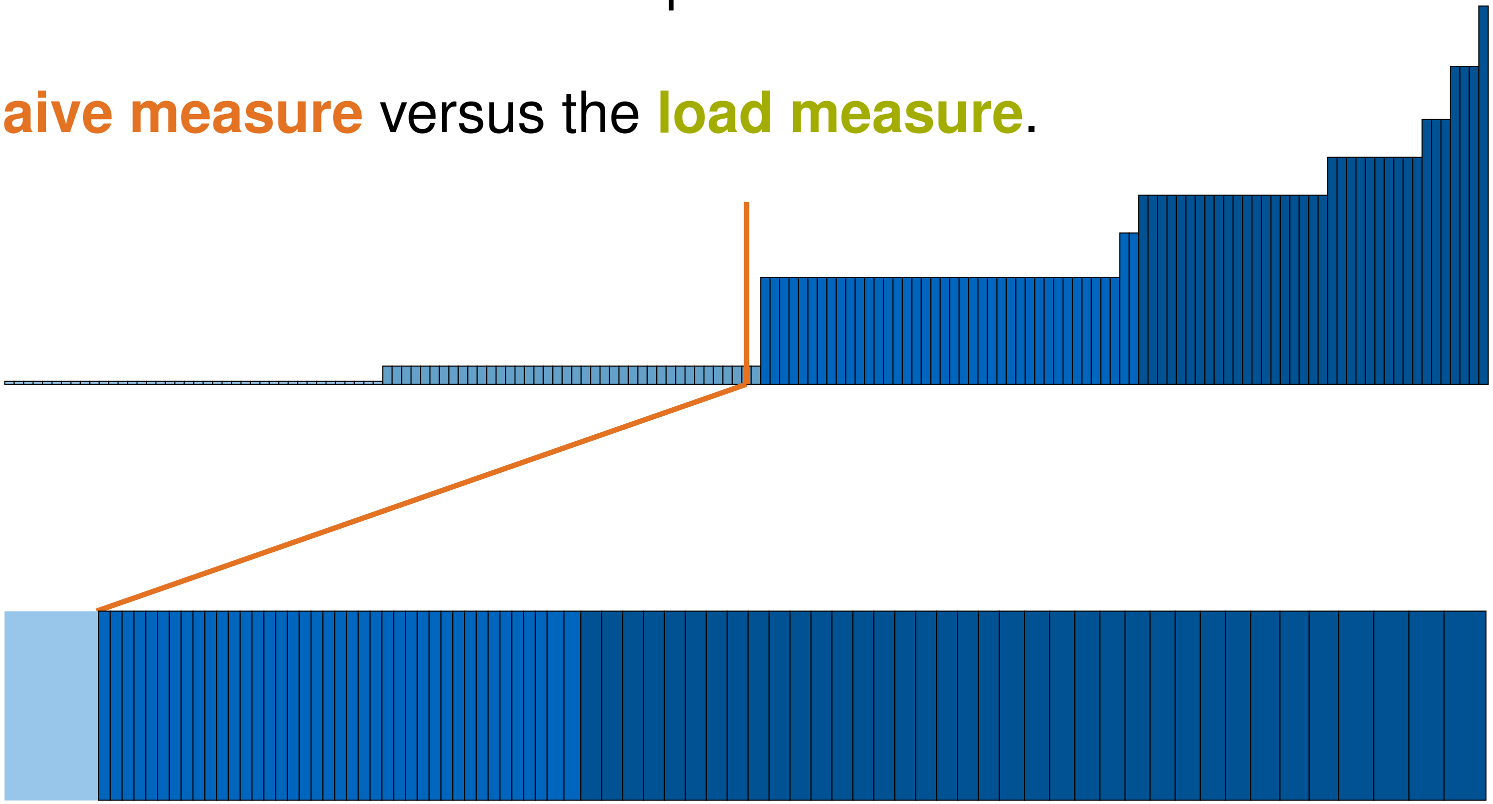
The **naive measure** versus the **load measure**.



Different time measures.

When have we seen half the sequence?

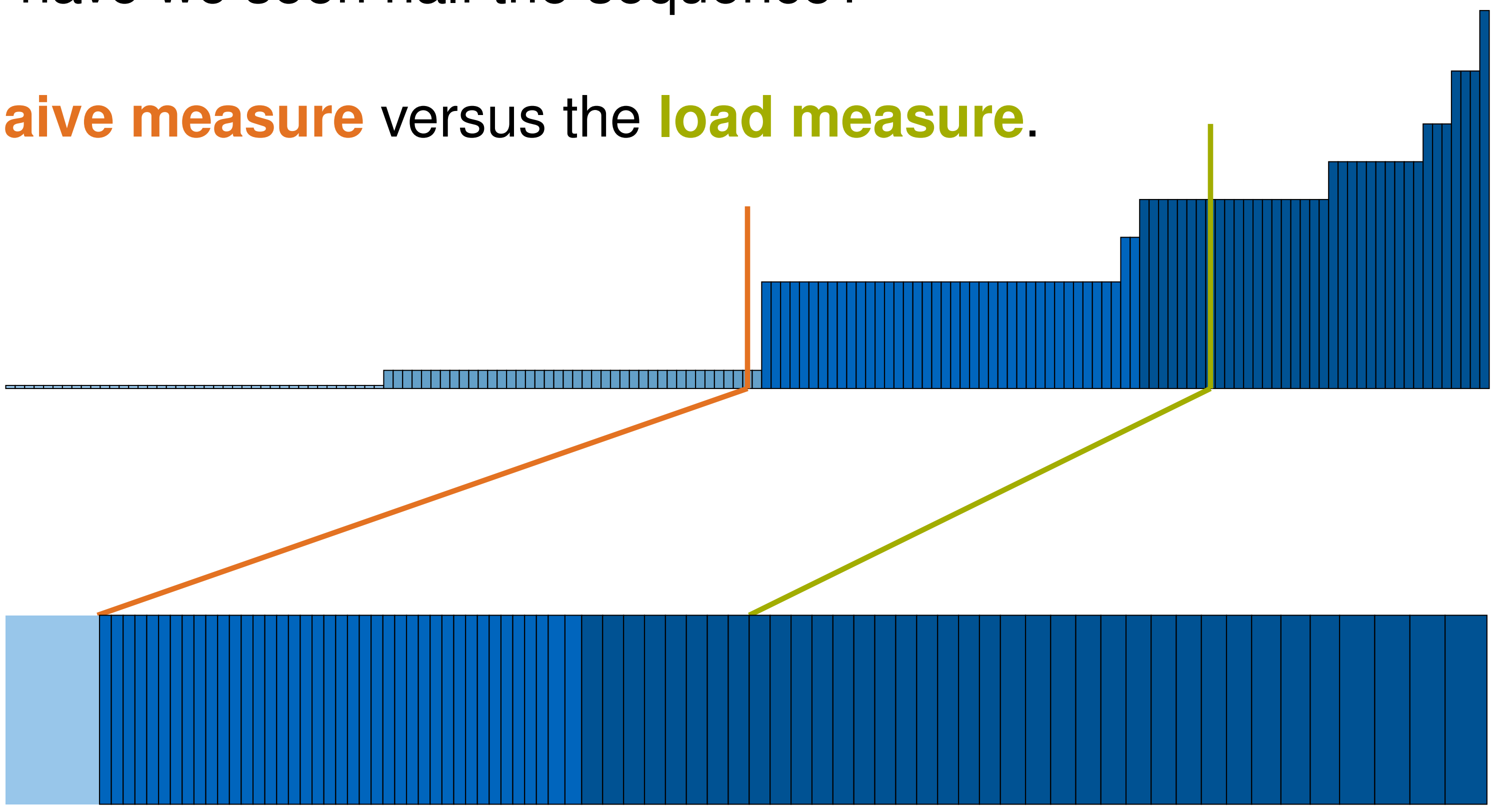
The **naive measure** versus the **load measure**.



Different time measures.

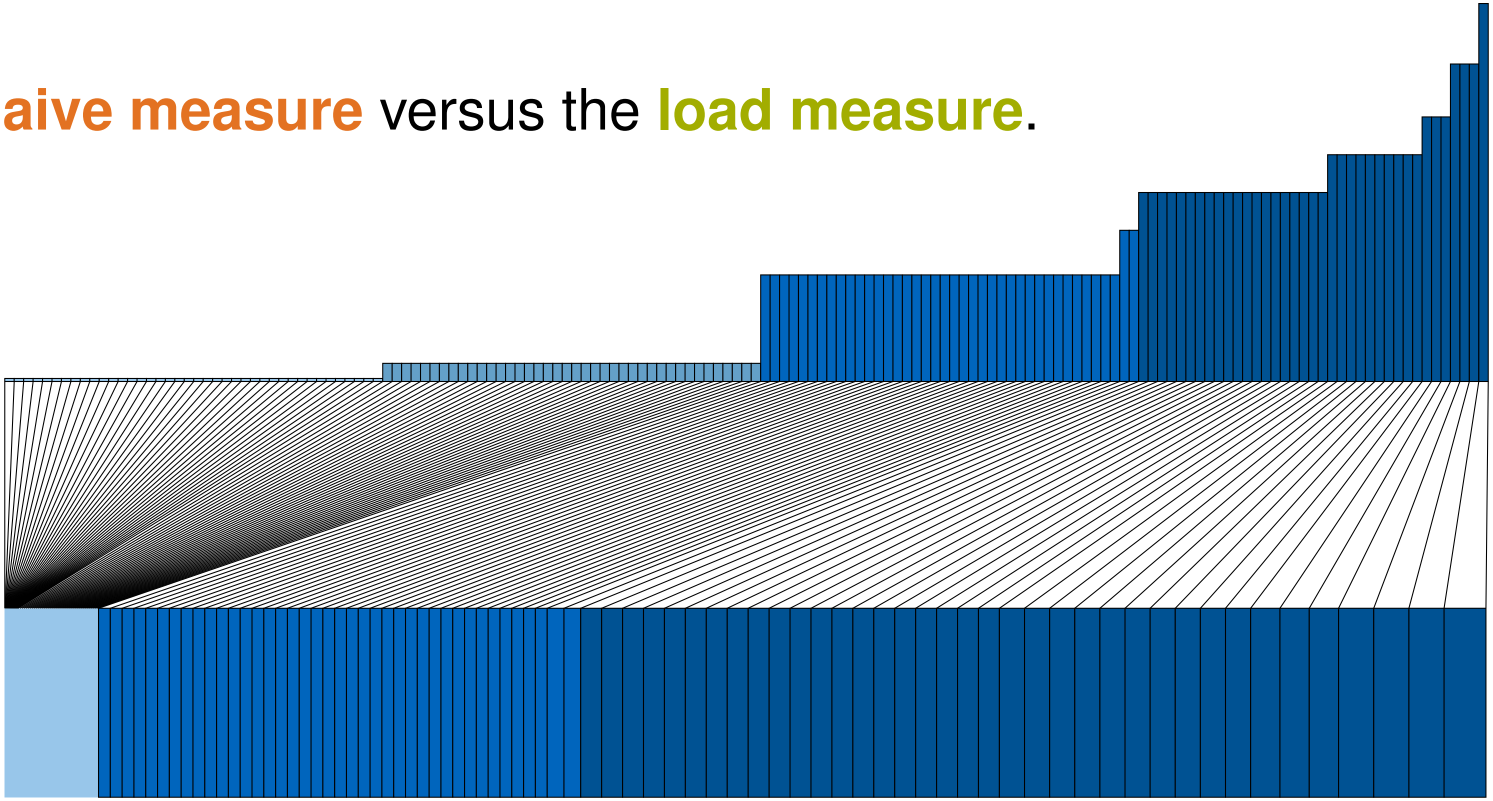
When have we seen half the sequence?

The **naive measure** versus the **load measure**.



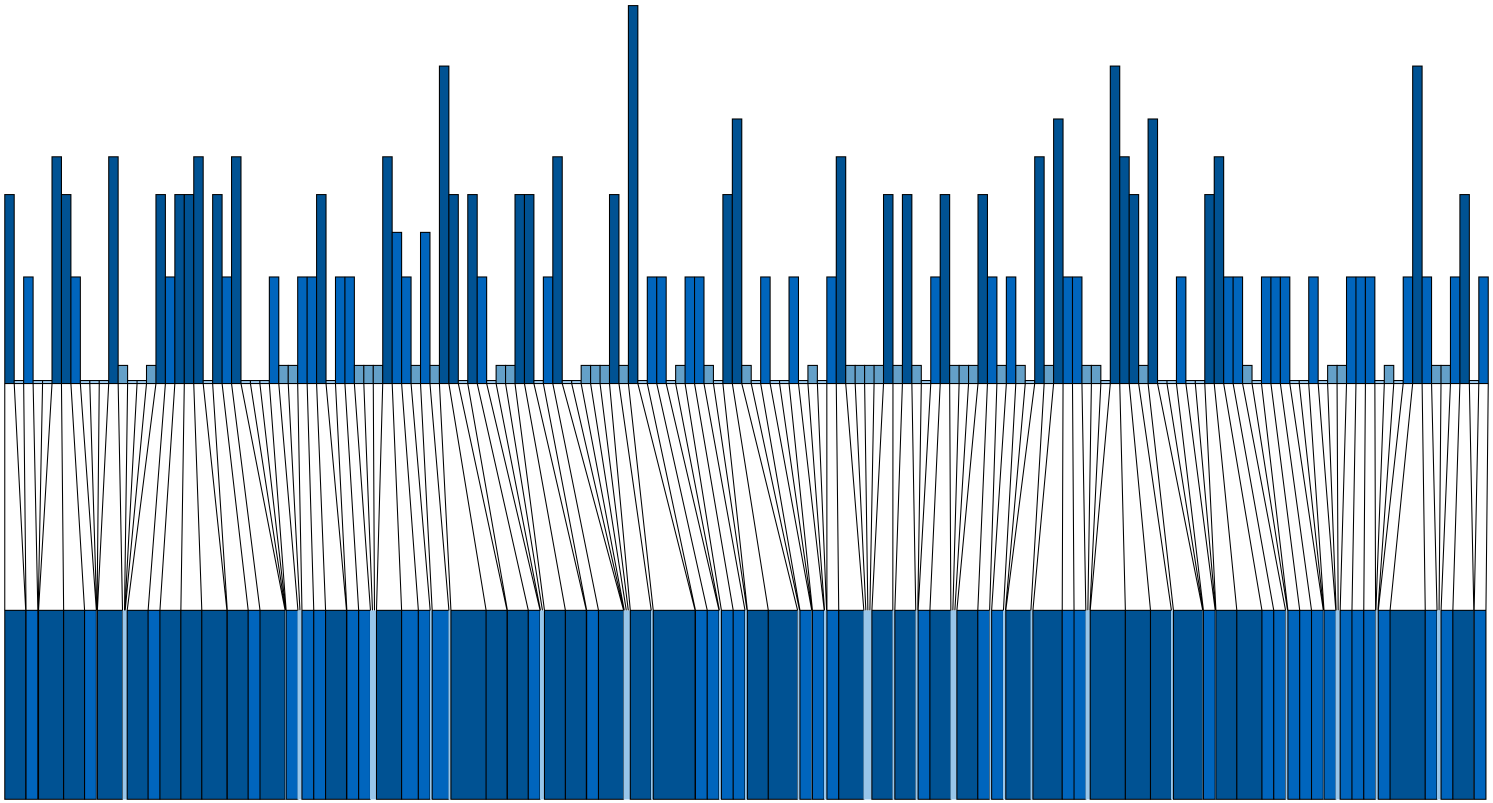
Different time measures.

The **naive measure** versus the **load measure**.

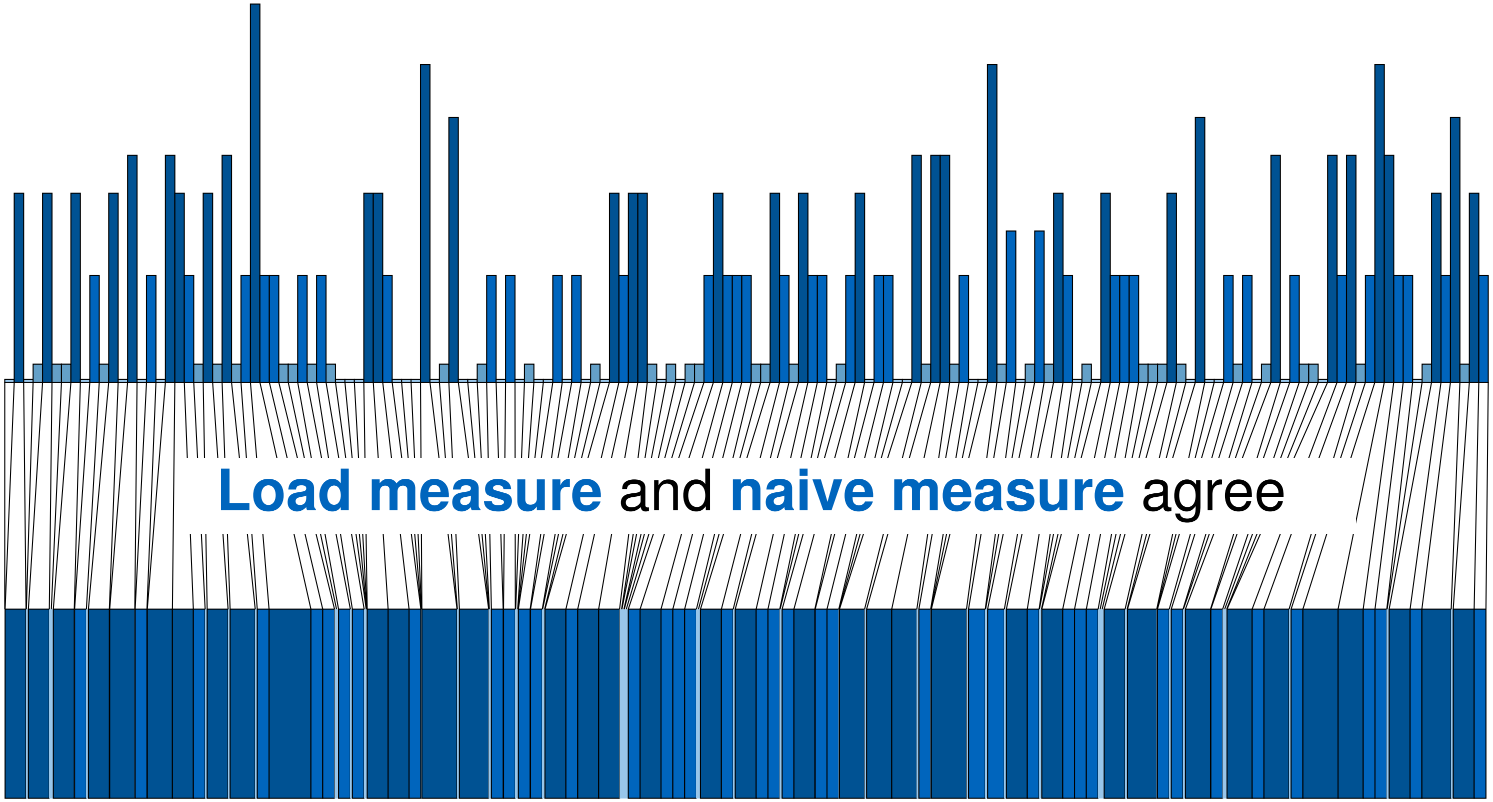


Different time measures.

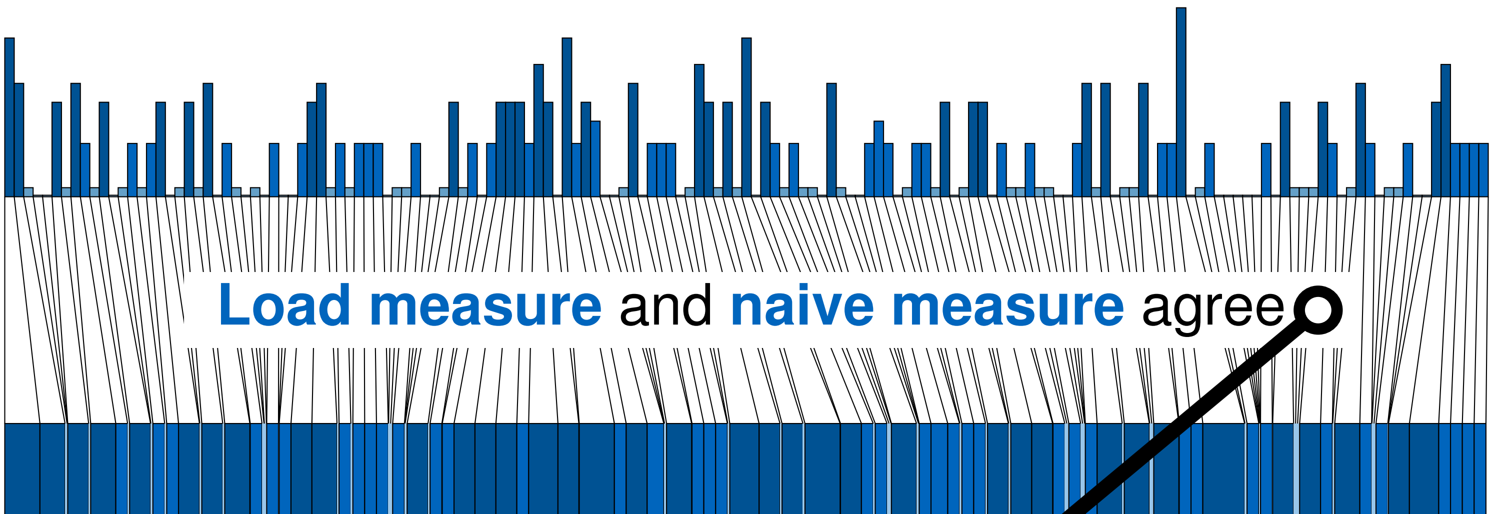
How does random-order arrival impact the measures?



Different time measures.



Different time measures.



Load measure and **naive measure** agree

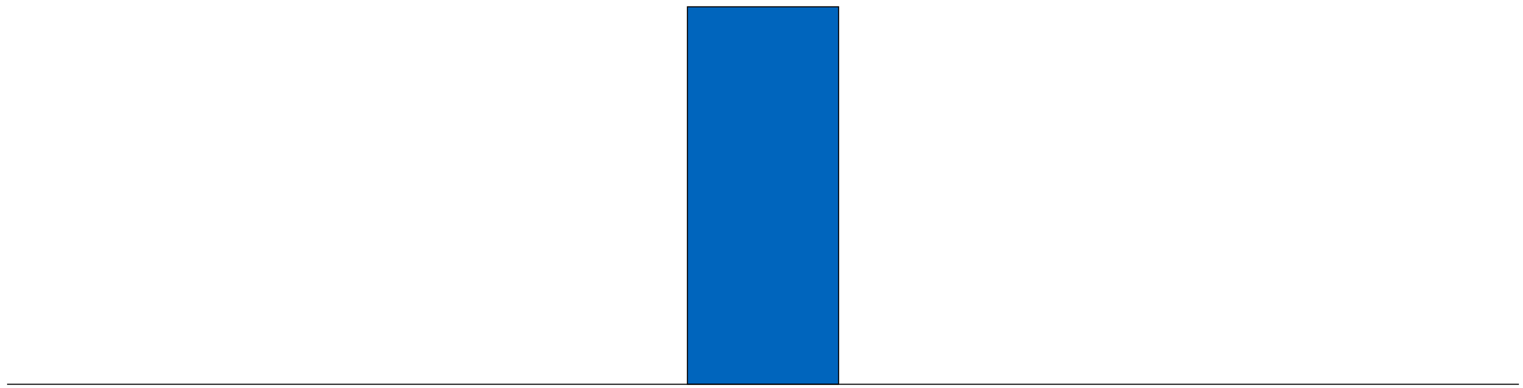
for **m** large

high probability

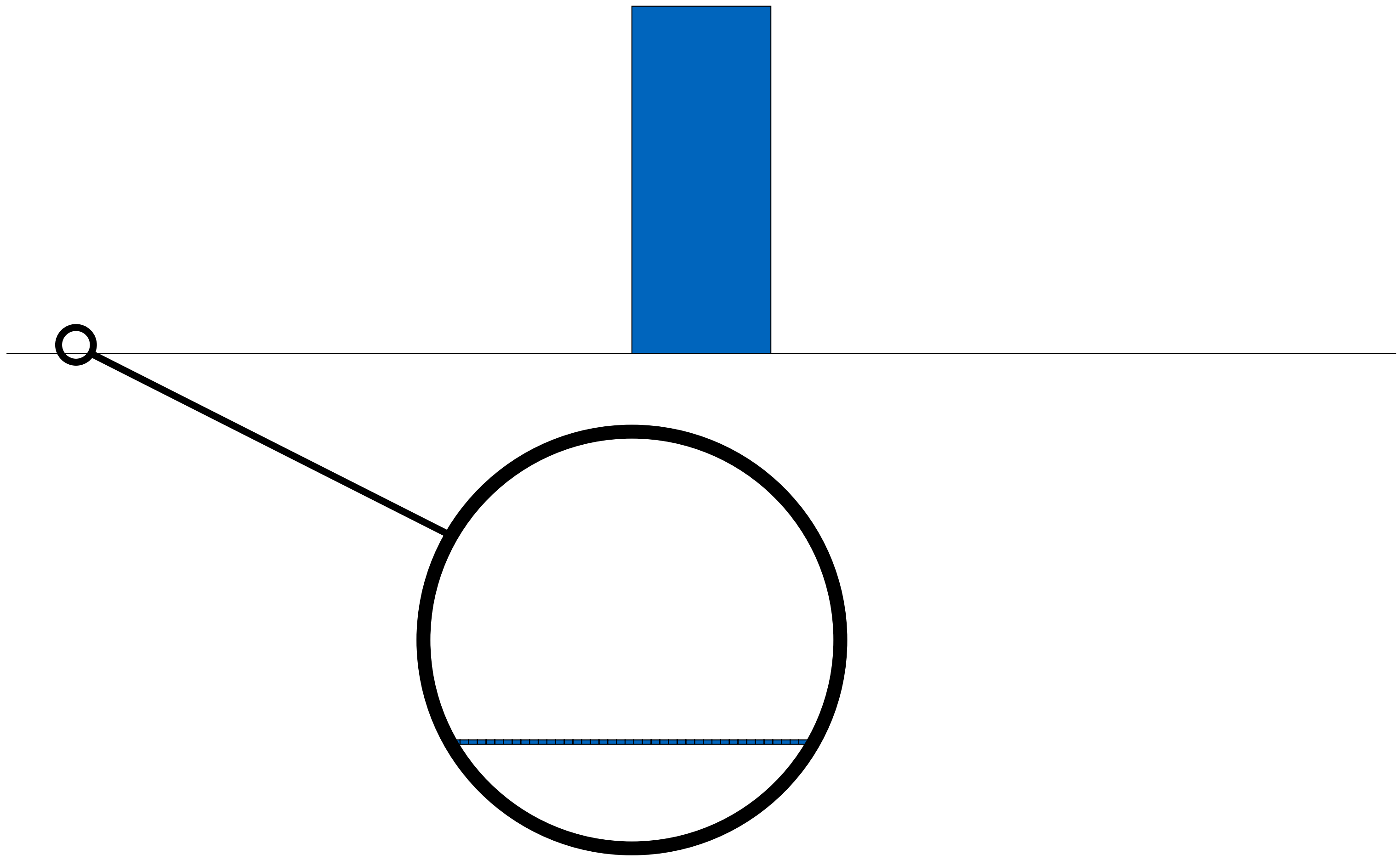
non-simple inputs

margin of error

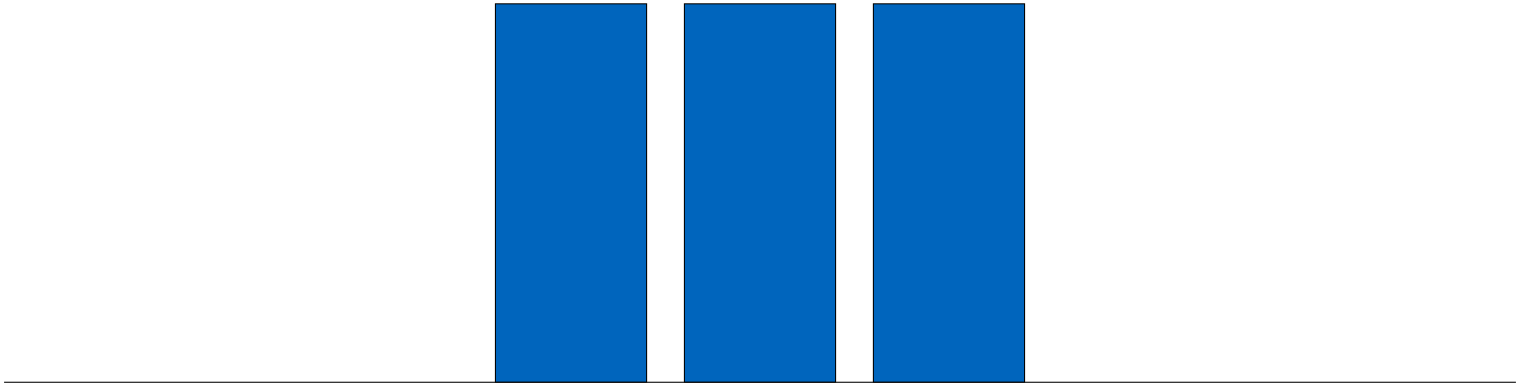
How to foil reordering arguments.



How to foil reordering arguments.

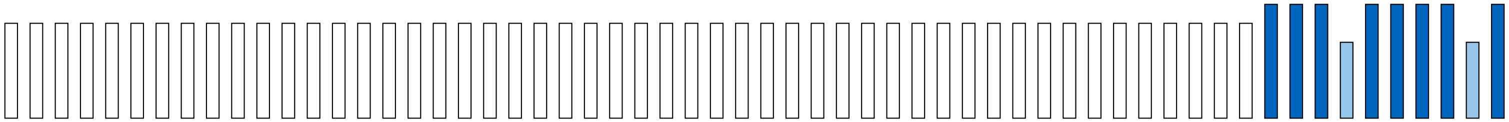


How to foil reordering arguments.



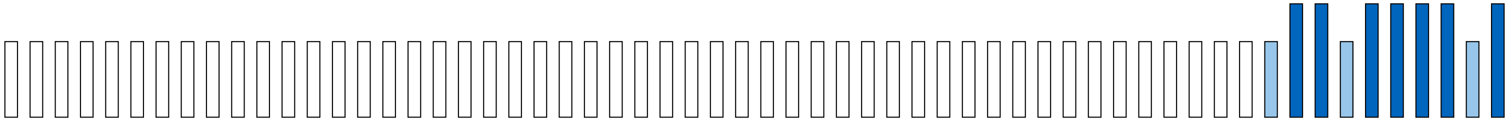
Large Job Magic: Large jobs for free

How to complete the sequence correctly?



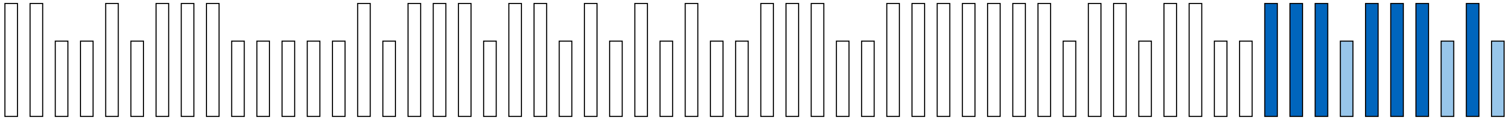
Large Job Magic: Large jobs for free

How to complete the sequence correctly?

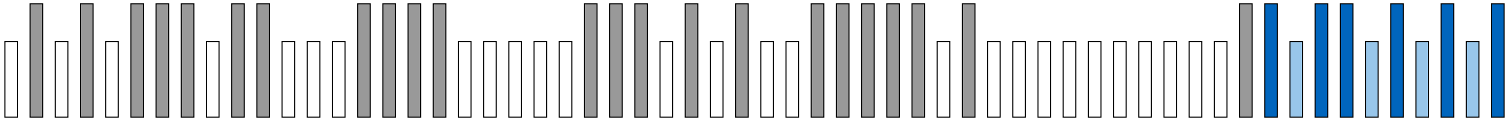


Large Job Magic: Large jobs for free

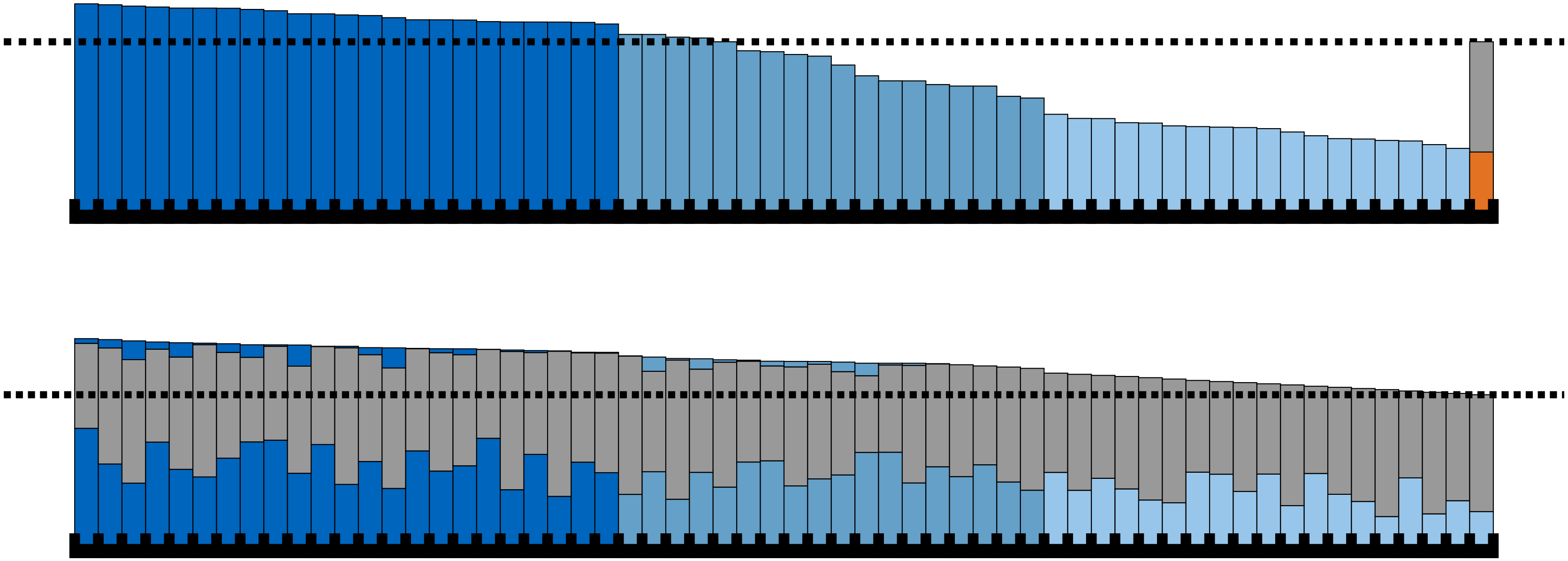
No high concentration of large jobs.



Large Job Magic: Large jobs for free

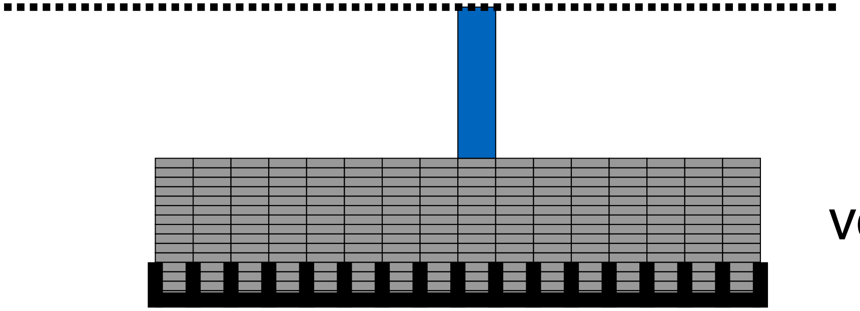


We call it **Large Job Magic**.

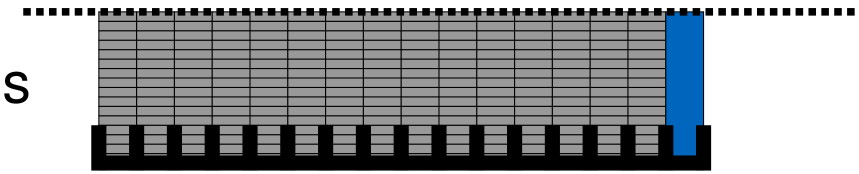


Oracle-like properties

A difficult sequence has **large jobs** close to the **end**.

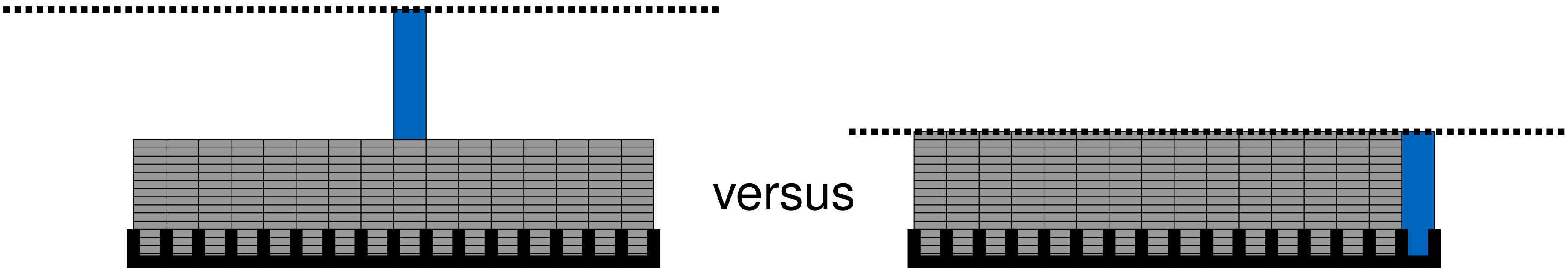


versus



Oracle-like properties

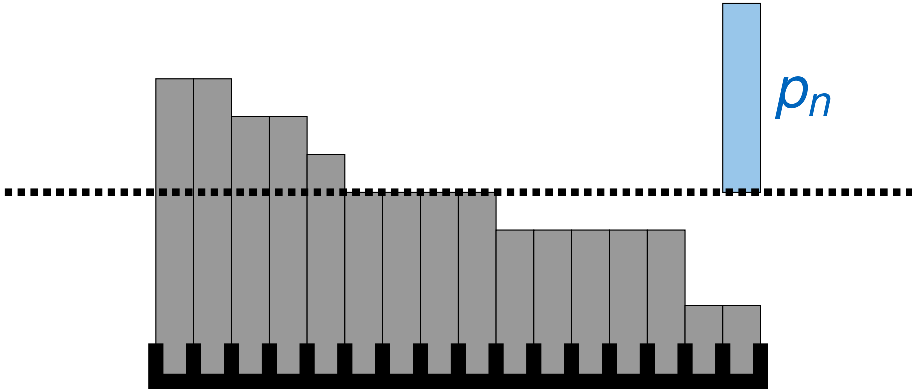
A difficult sequence has **large jobs** close to the **end**.



[Osborn and Torng, 2008] Highly probable if there are **many large jobs**.

Oracle-like properties

A difficult sequence has **large jobs** close to the **end**.

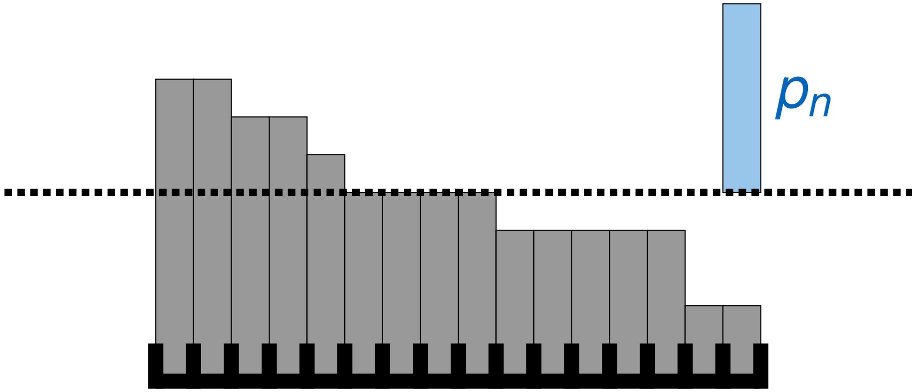


Highly probable if there are **many large jobs**.

Many large jobs work as an oracle for OPT.

Oracle-like properties

A difficult sequence has **large jobs** close to the **end**.



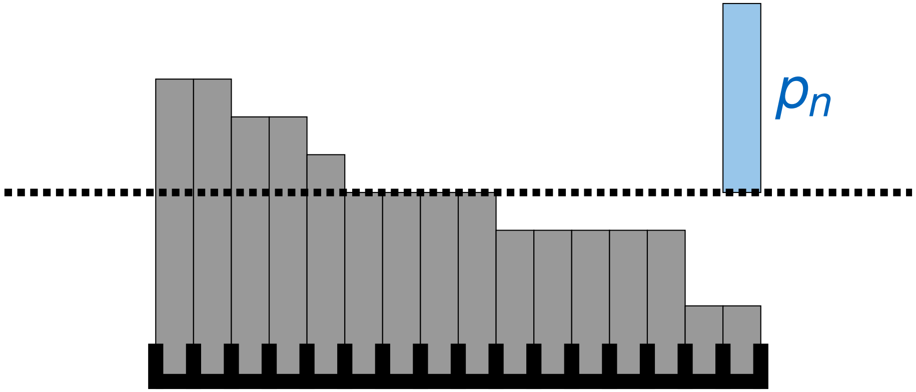
Highly probable if there are **many large jobs**.

Many large jobs work as an oracle for OPT.

A competitive ratio of about 1.89 in expectation.

Oracle-like properties

A difficult sequence has **large jobs** close to the **end**.



Highly probable if there are **many large jobs**.

Many large jobs work as an oracle for OPT.

The difficult part is obtaining such improvement with high probability

Oracle-like properties

Idea: Make the algorithm robust towards few large jobs.

Oracle-like properties

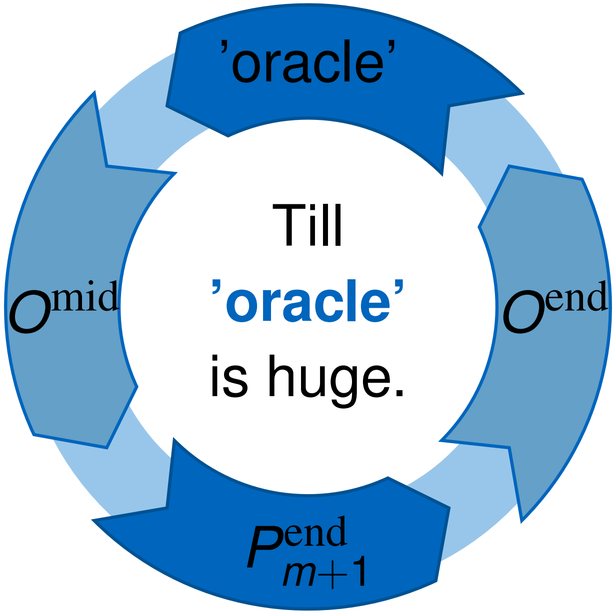
Idea: Make the algorithm robust towards few large jobs.

Problem: This already requires the **’oracle’** .

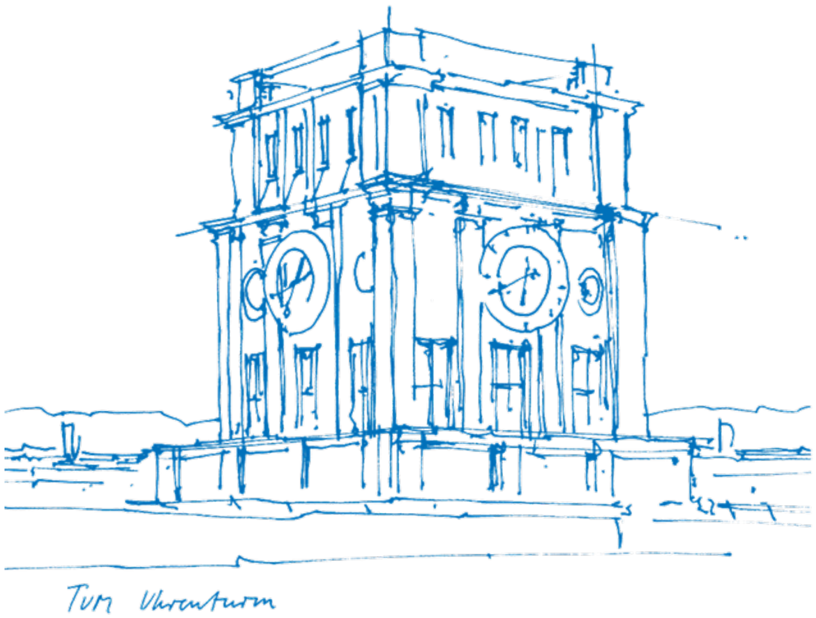
Oracle-like properties

Idea: Make the algorithm robust towards few large jobs.

Problem: This already requires the **’oracle’** .

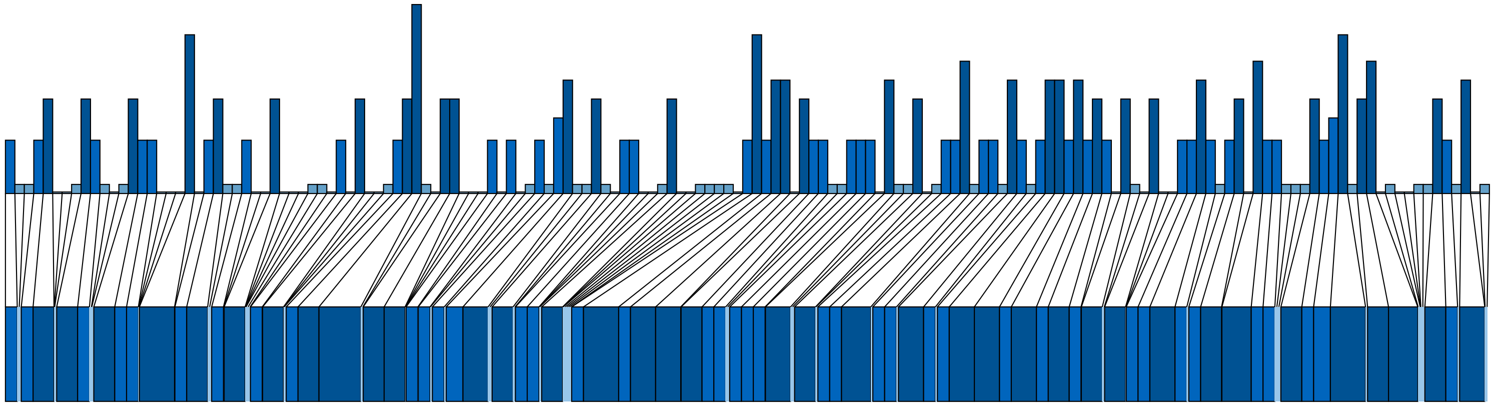


Take-away



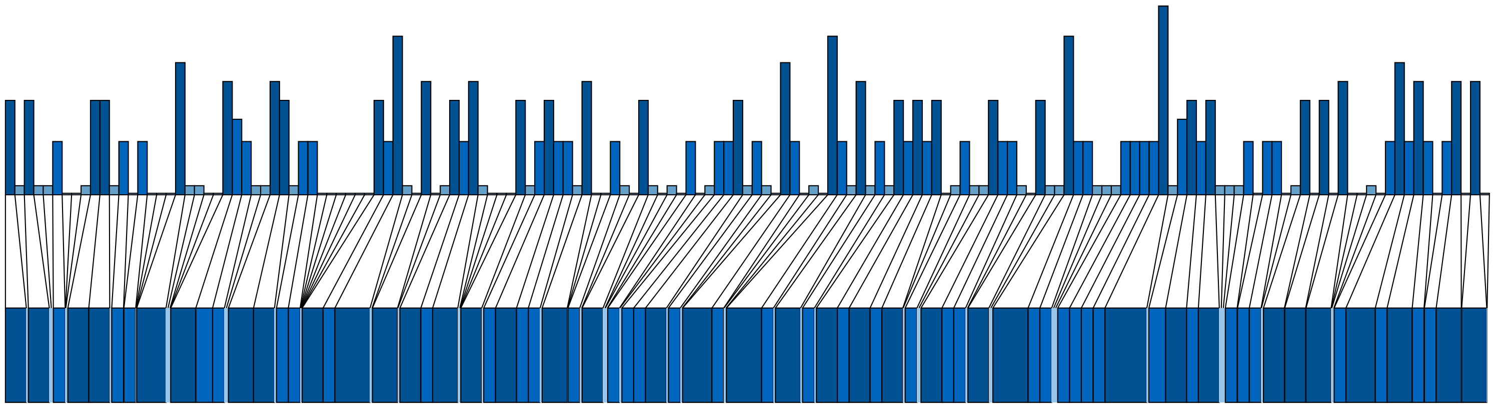
Load Lemma:

Relate algorithmic and probabilistic properties.

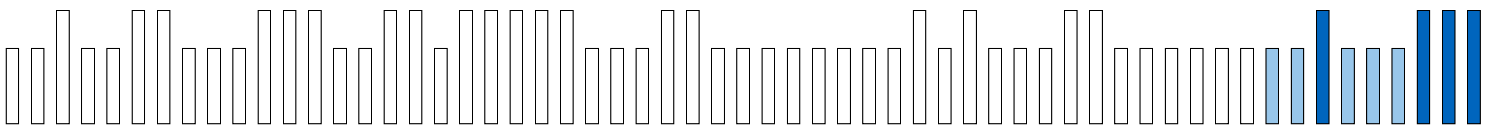


Load Lemma:

Relate algorithmic and probabilistic properties.

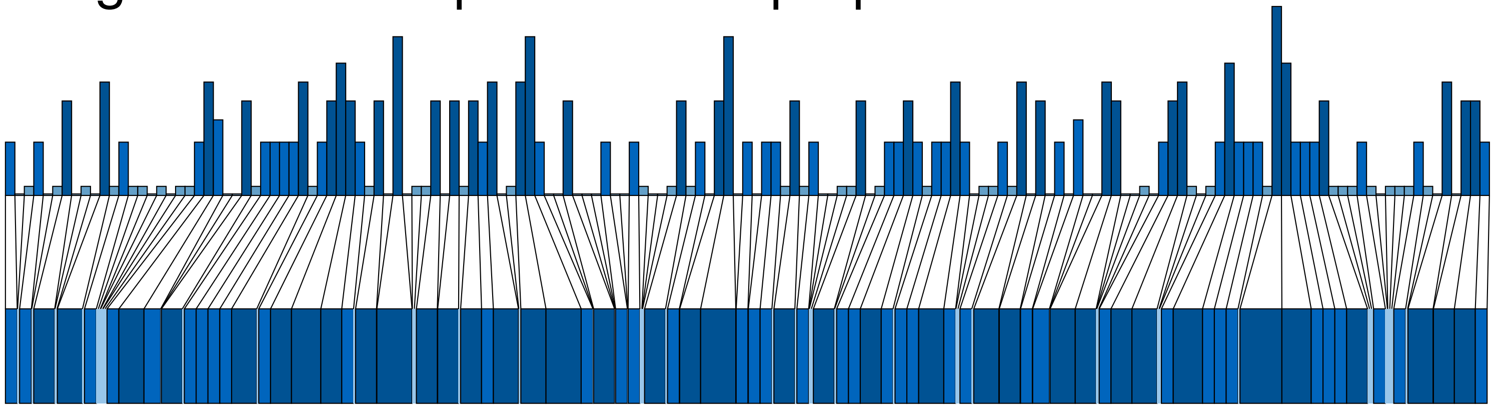


Large Job Magic: Get large jobs for free.

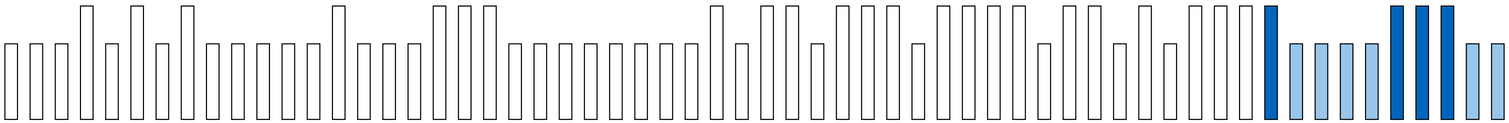


Load Lemma:

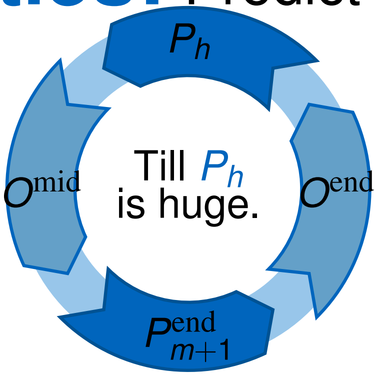
Relate algorithmic and probabilistic properties.



Large Job Magic: Get large jobs for free.



Oracle-like properties: Predict the future.



Thank you!

TUM Uhrenturm