New Tools for

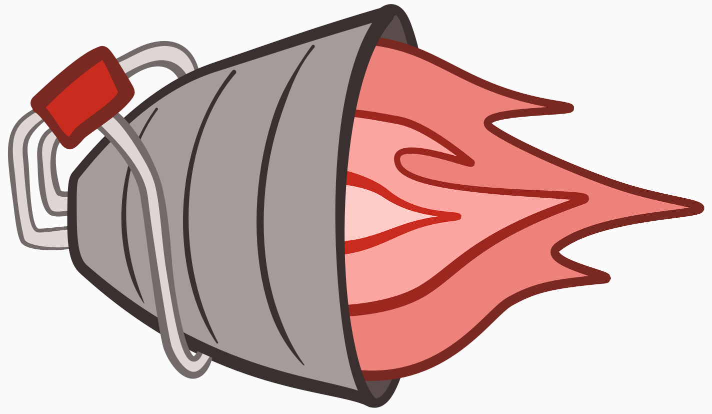
Optimizing Tail Latency in Queues:

Generating Functions, the Gittins Index, and Gurobi

Ziv Scully   Cornell ORIE

Joint work with

George Yu   Cornell ORIE
Amit Harlev   Cornell CAM
Reevu Adakroy   Cornell ORIE



New Tools for

Optimizing Tail Latency in Queues:

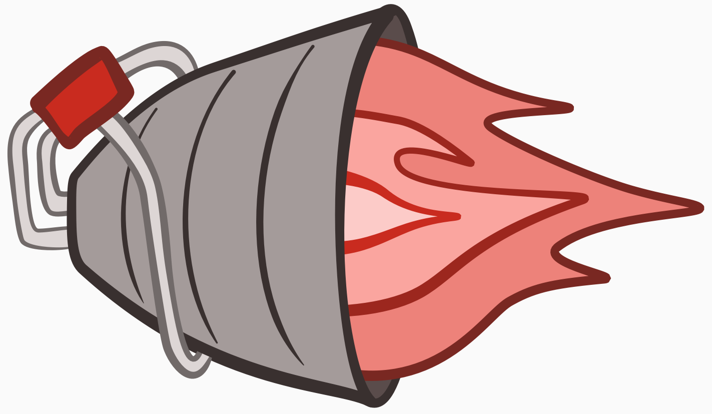
Generating Functions, the Gittins Index, and Gurobi

SIGMETRICS 2024

Ziv Scully Cornell ORIE

Joint work with

George Yu Cornell ORIE
Amit Harlev Cornell CAM
Reevu Adakroy Cornell ORIE



New Tools for

Optimizing Tail Latency in Queues:

Generating Functions, the Gittins Index, and Gurobi

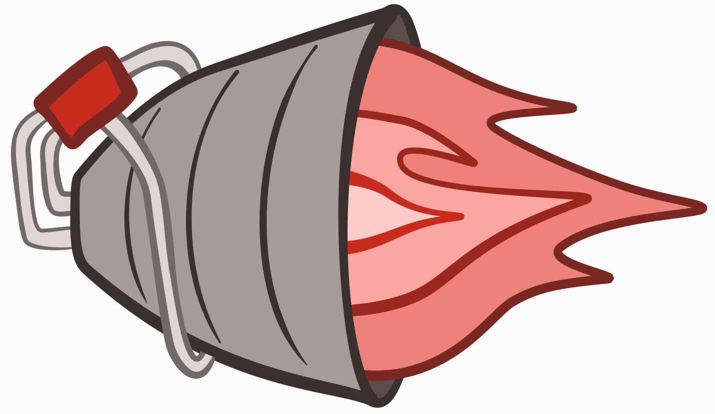
SIGMETRICS 2024

SIGMETRICS 2025

Ziv Scully   Cornell ORIE

Joint work with

George Yu   Cornell ORIE
Amit Harlev   Cornell CAM
Reevu Adakroy   Cornell ORIE



New Tools for

Optimizing Tail Latency in Queues:

Generating Functions, the Gittins Index, and Gurobi

SIGMETRICS 2024

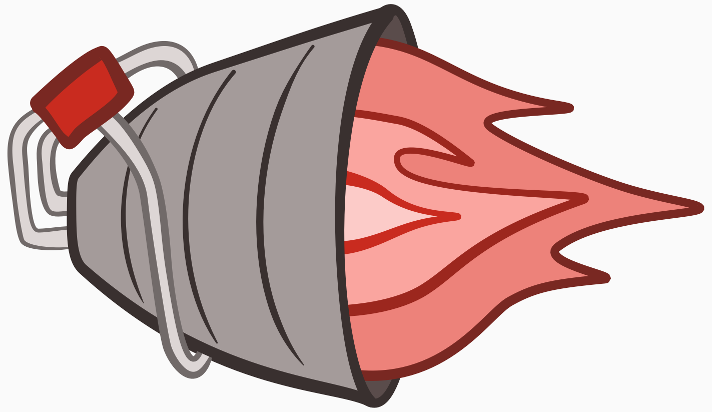
SIGMETRICS 2025

SIGMETRICS 2026

Ziv Scully	Cornell ORIE

Joint work with

George Yu	Cornell ORIE
Amit Harlev	Cornell CAM
Reevu Adakroy	Cornell ORIE



New Tools for

Optimizing Tail Latency in Queues:

Generating Functions, the Gittins Index, and Gurobi

SIGMETRICS 2024

SIGMETRICS 2025

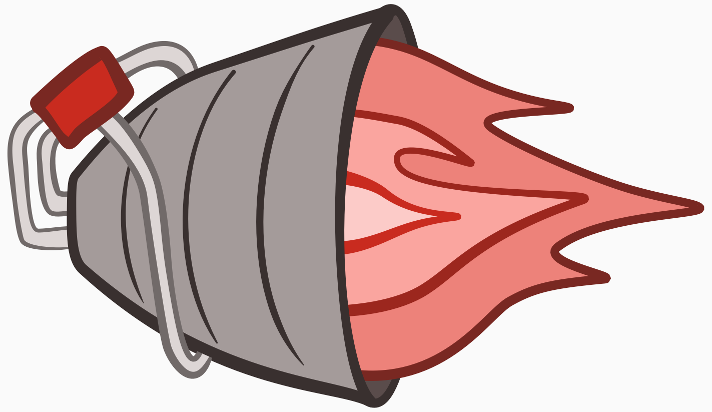
SIGMETRICS 2026

Ziv Scully

SIGMETRICS 2024 best paper,
2024 INFORMS APS student
paper competition finalist

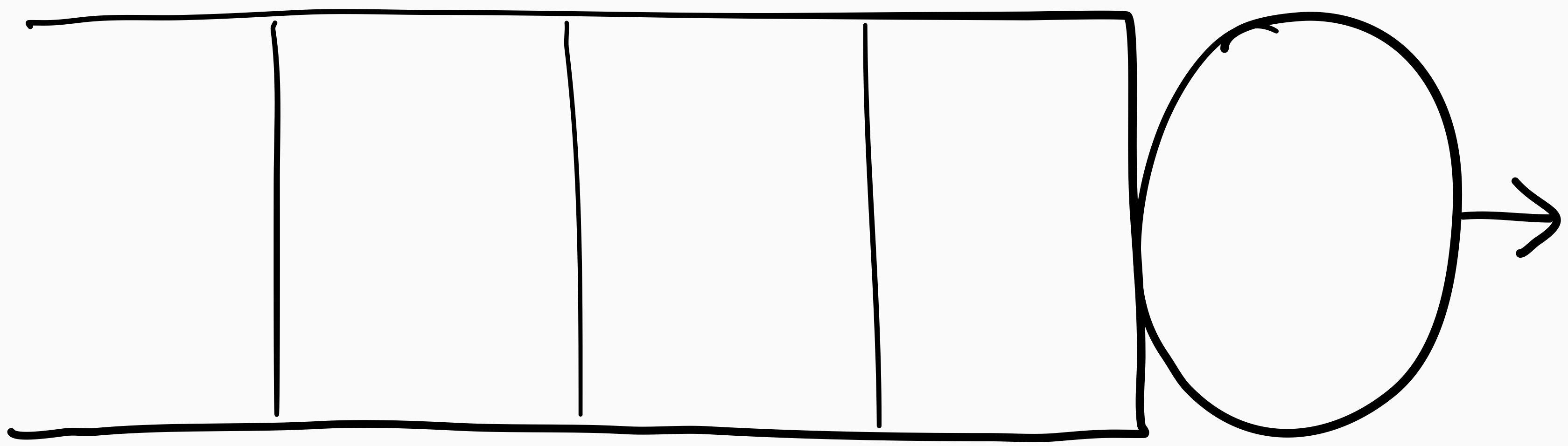
Joint work wit

George Yu Cornell ORIE
Amit Harlev Cornell CAM
Reevu Adakroy Cornell ORIE

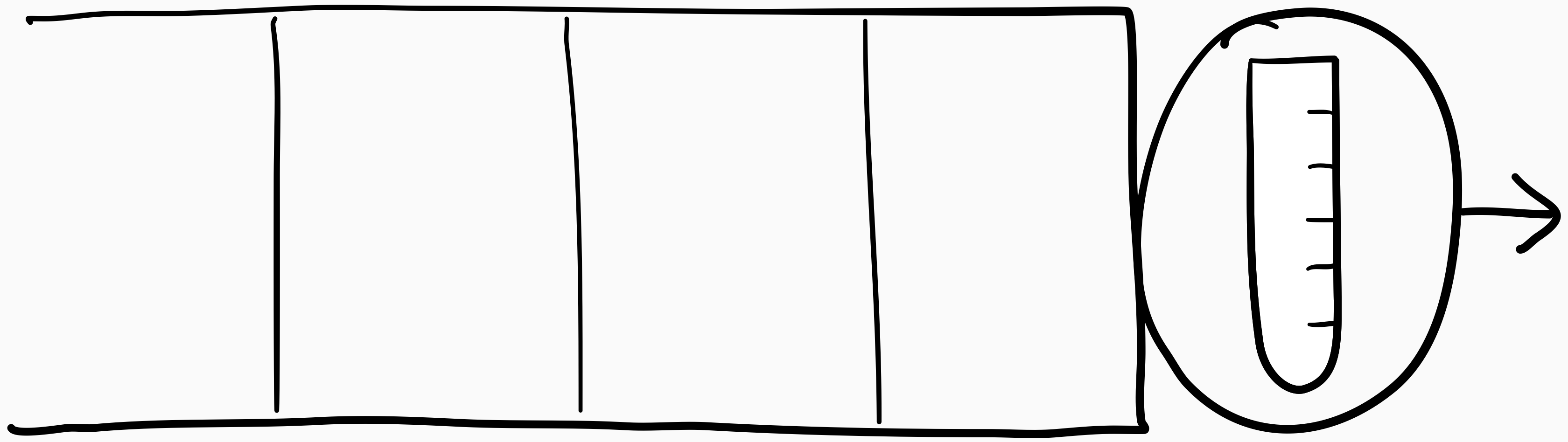


How should we schedule jobs to minimize delay?

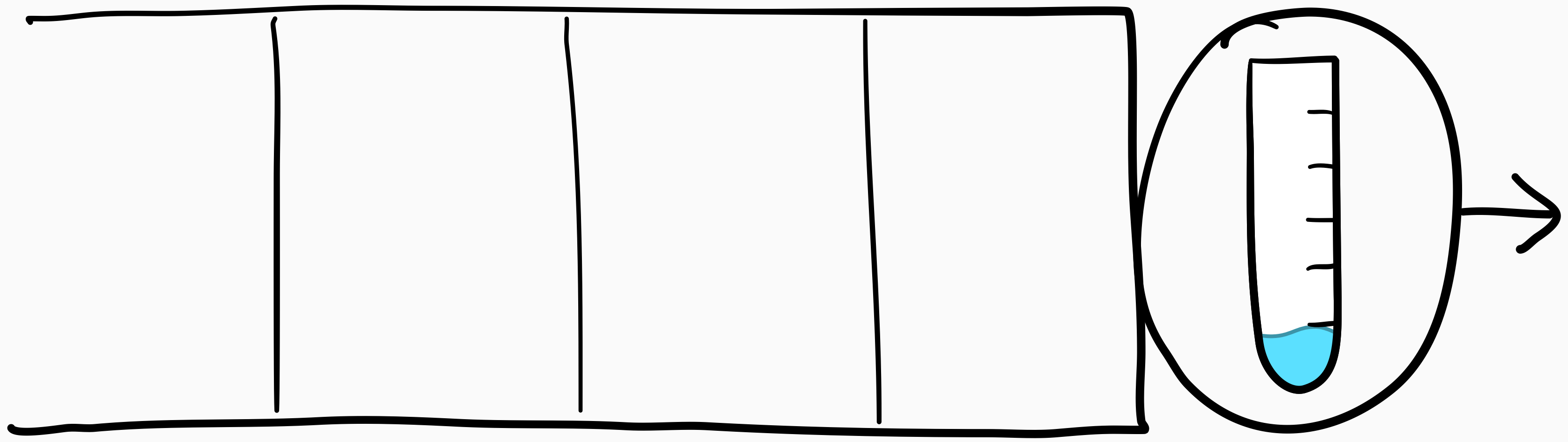
How should we schedule jobs to minimize delay?



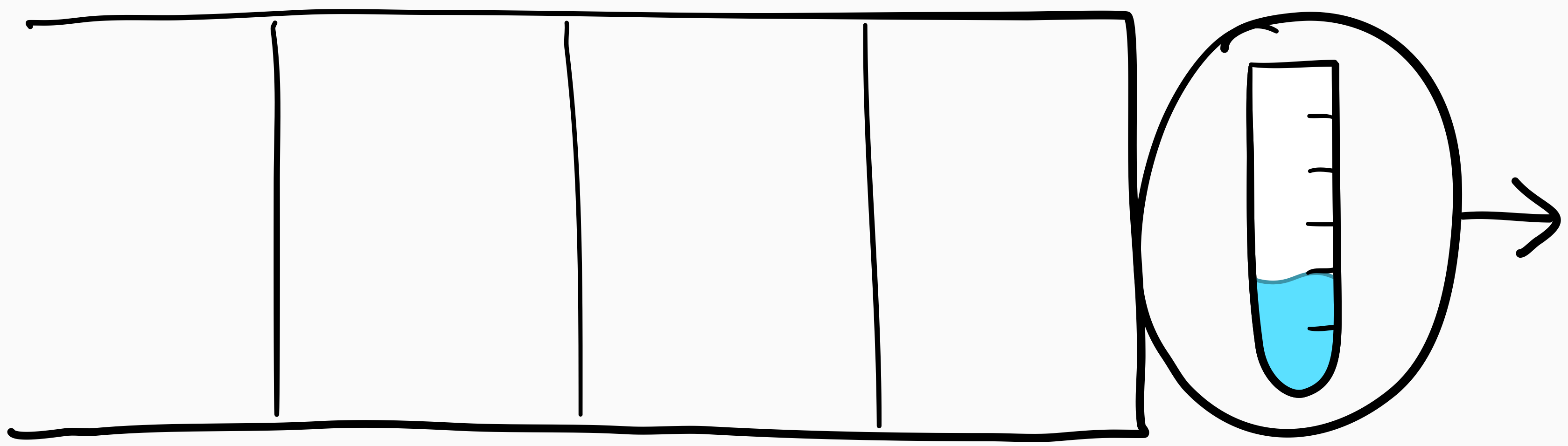
How should we schedule jobs to minimize delay?



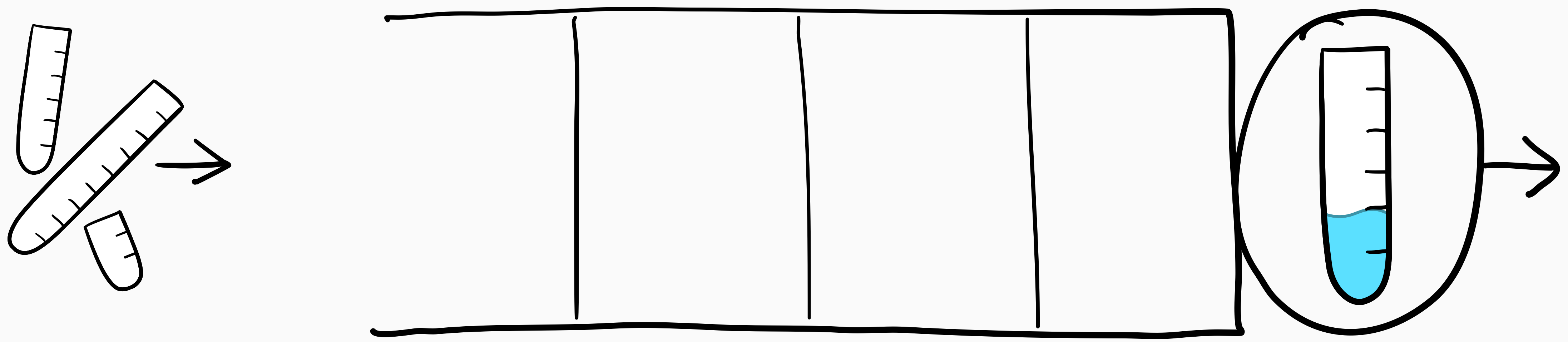
How should we schedule jobs to minimize delay?



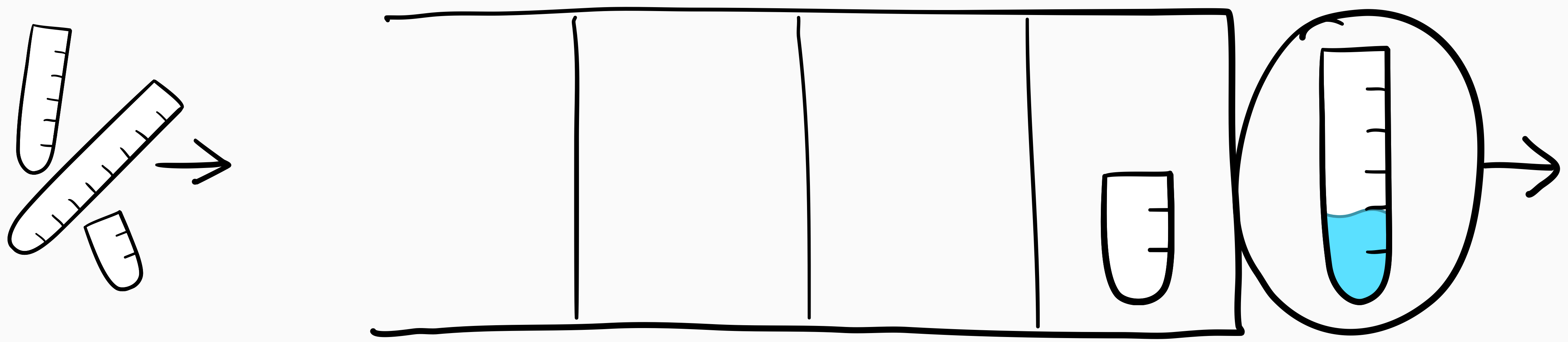
How should we schedule jobs to minimize delay?



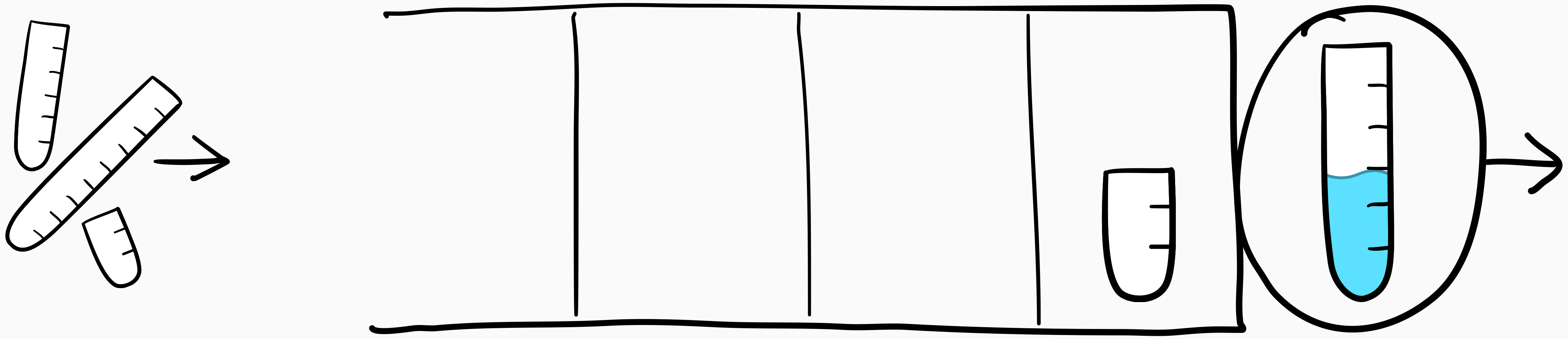
How should we schedule jobs to minimize delay?



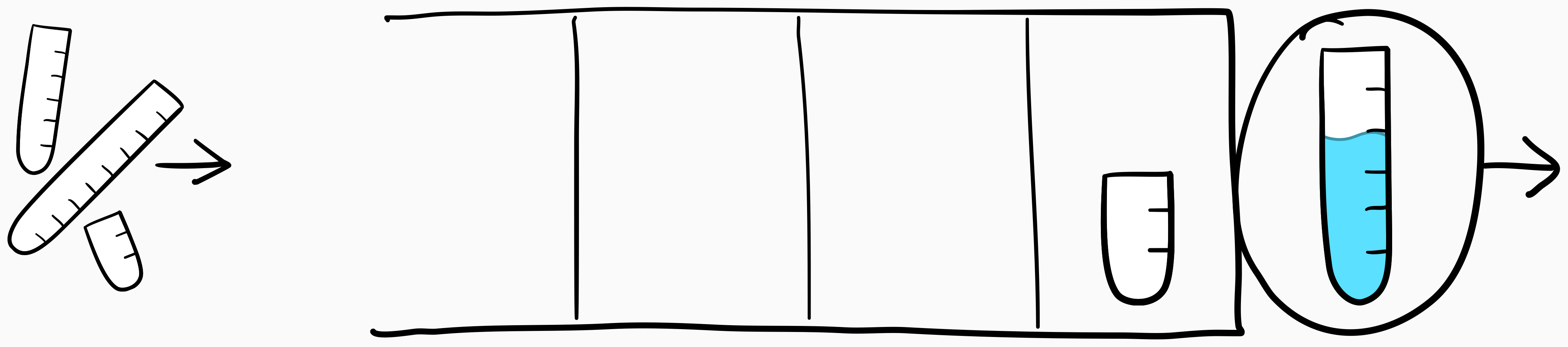
How should we schedule jobs to minimize delay?



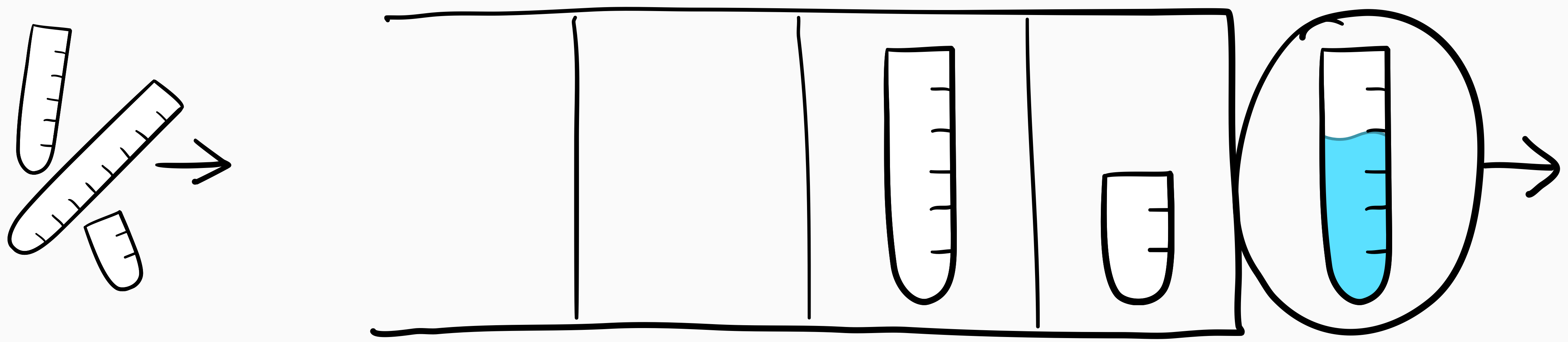
How should we schedule jobs to minimize delay?



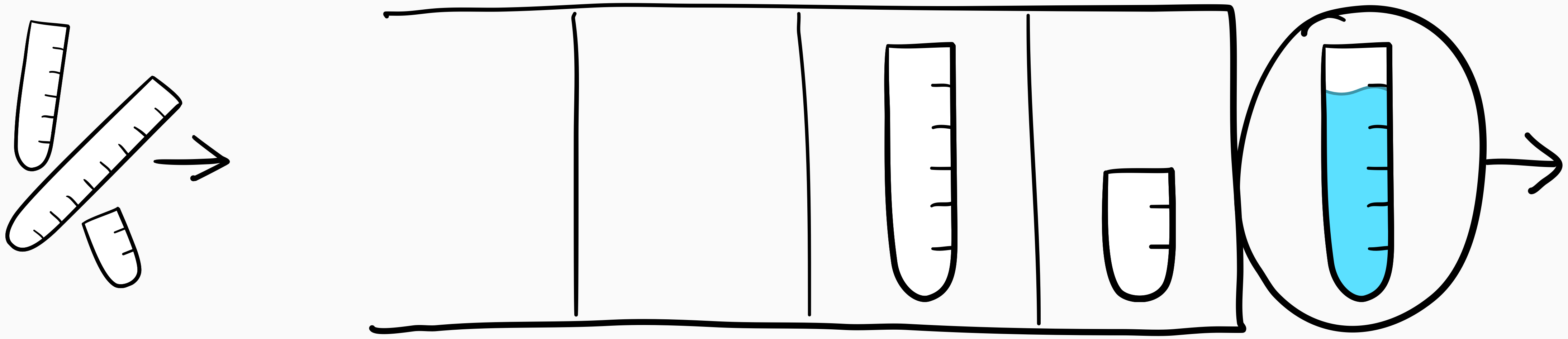
How should we schedule jobs to minimize delay?



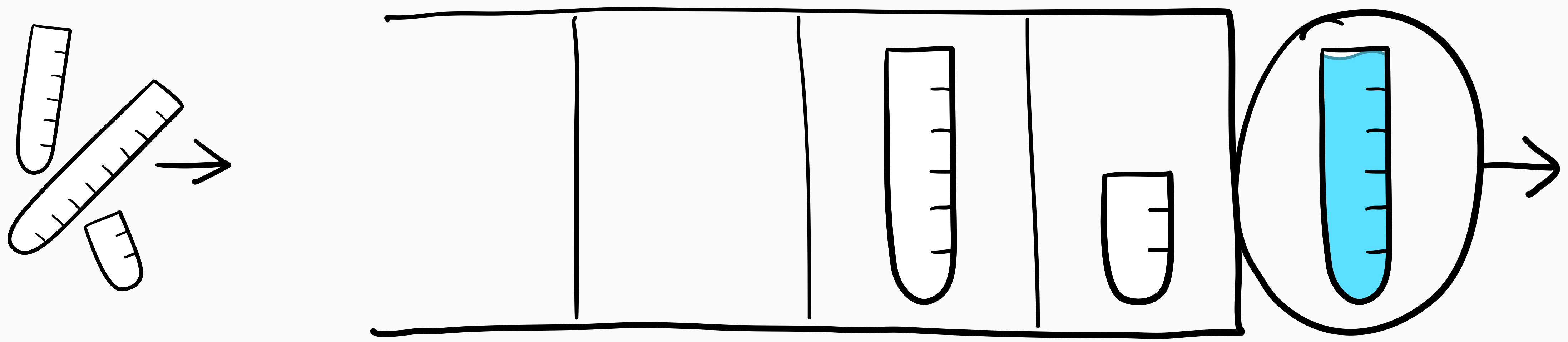
How should we schedule jobs to minimize delay?



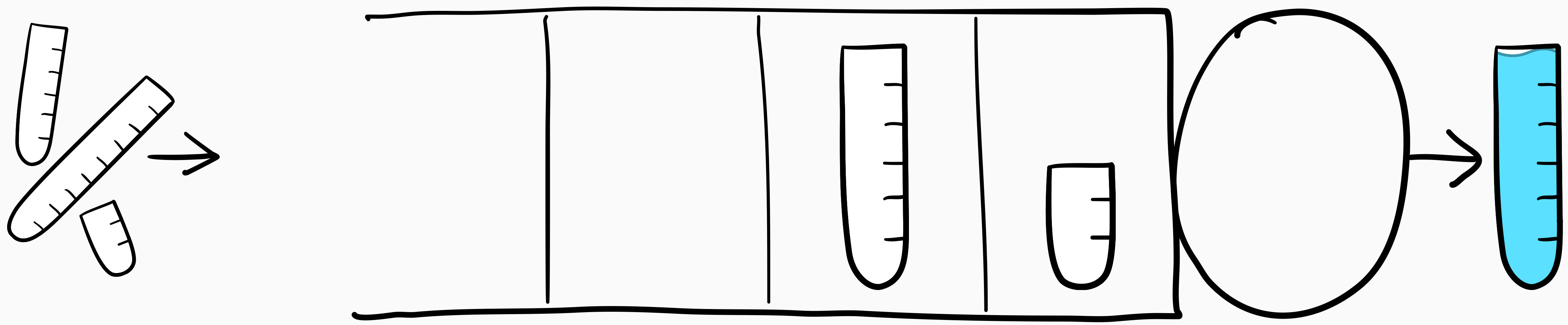
How should we schedule jobs to minimize delay?



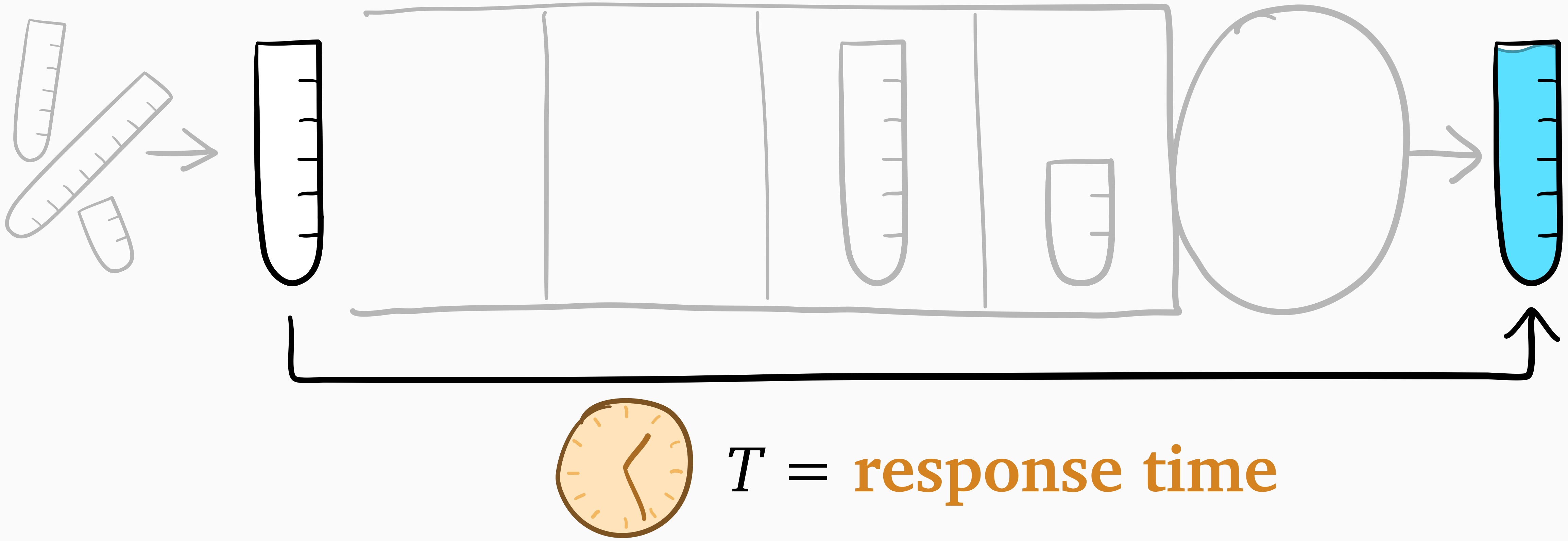
How should we schedule jobs to minimize delay?



How should we schedule jobs to minimize delay?

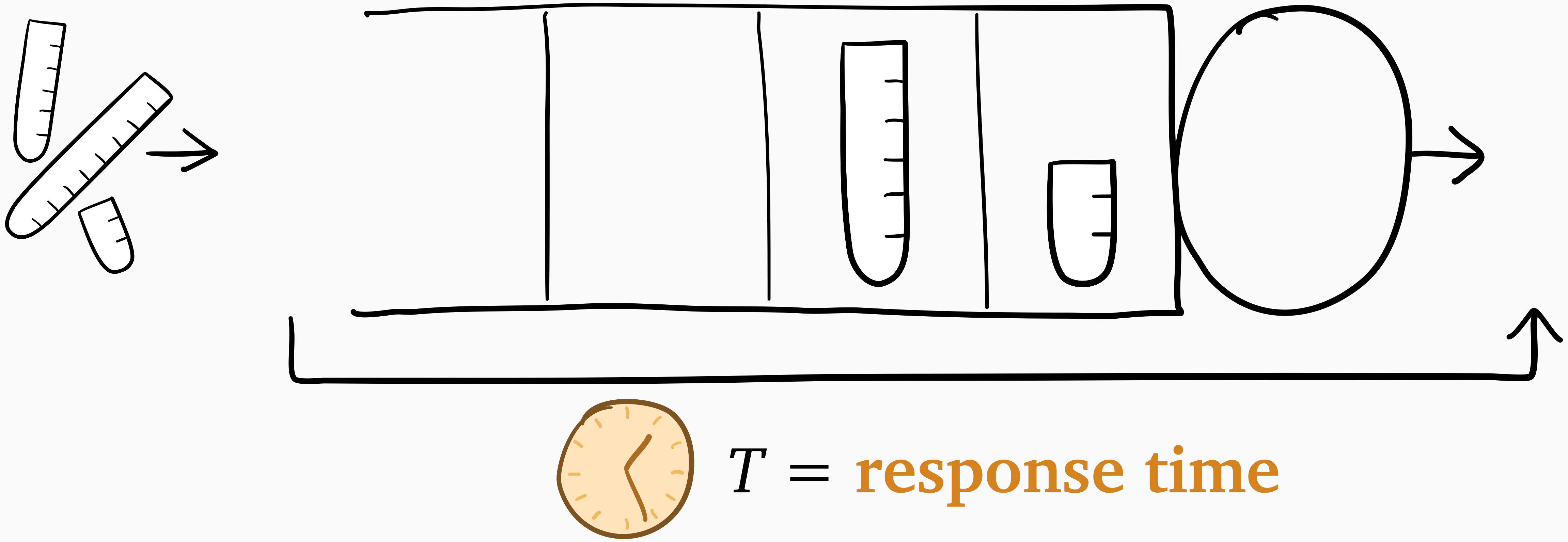


How should we schedule jobs to minimize delay?



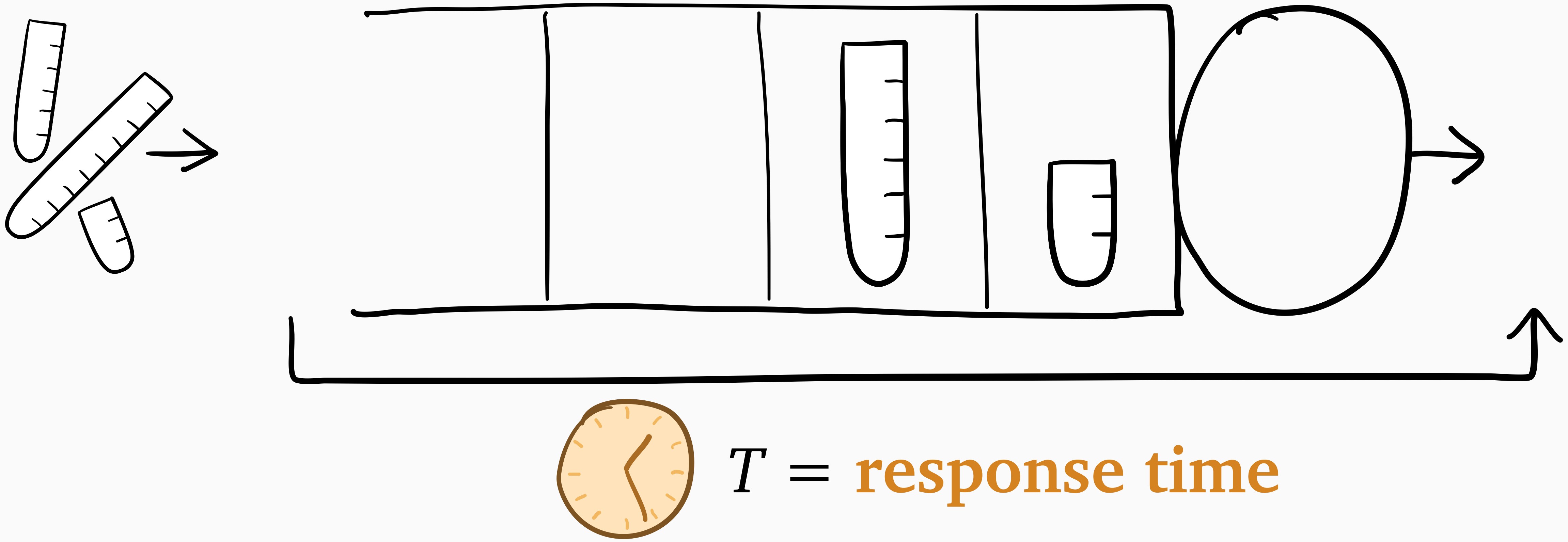
T = **response time**

How should we schedule jobs to minimize delay?



T = **response time**

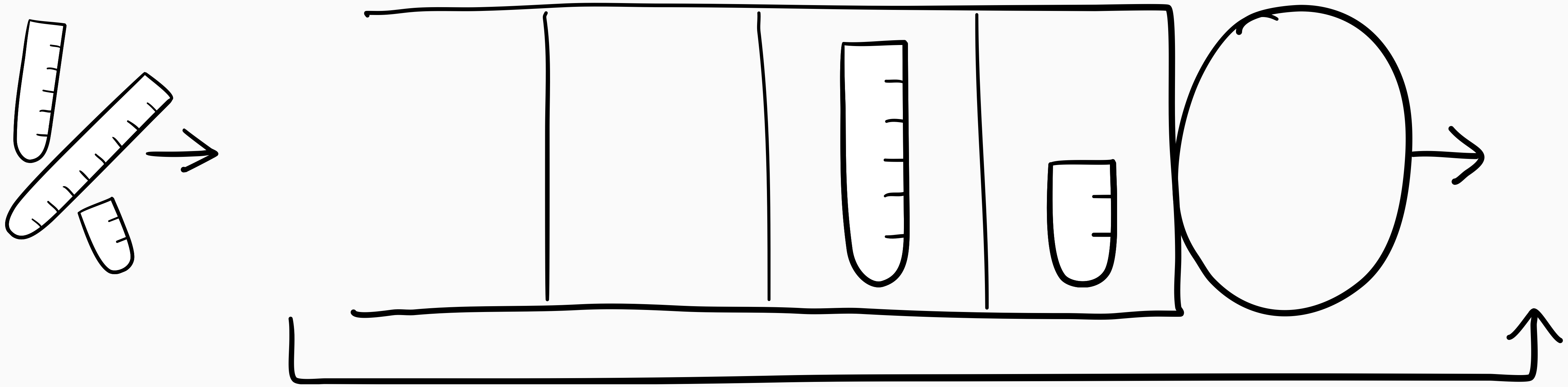
How should we schedule jobs to minimize delay?

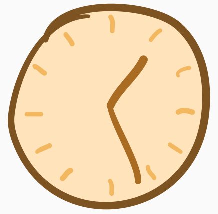


T = **response time**

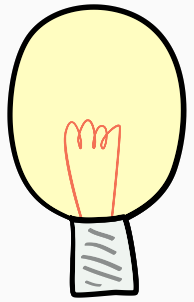
Minimize $\mathbf{E}[T]$?

How should we schedule jobs to minimize delay?

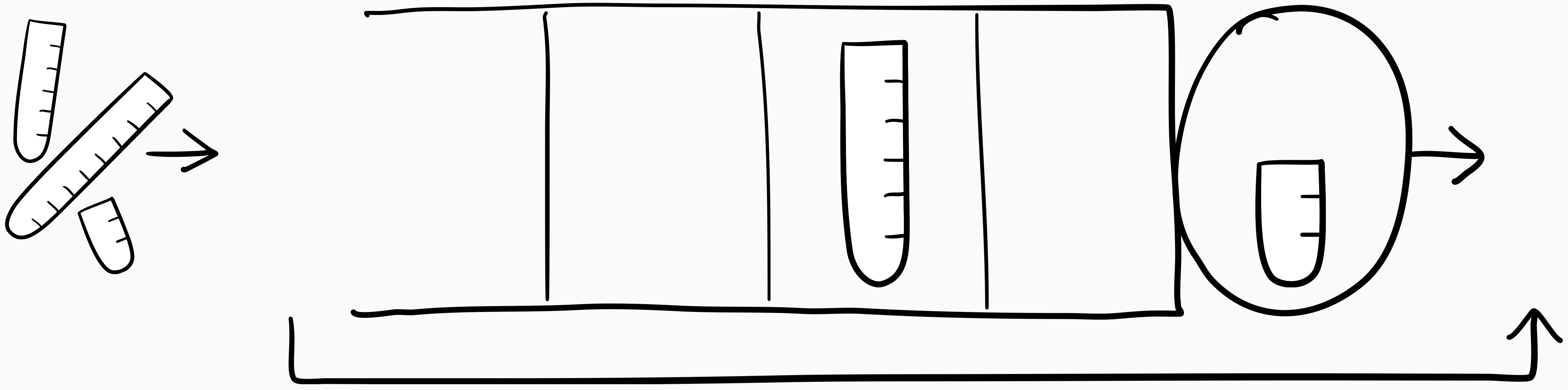


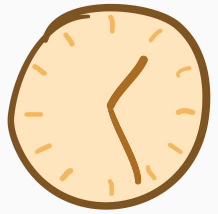
 T = **response time**

 Minimize $\mathbf{E}[T]$?

 Serve short jobs
before long jobs

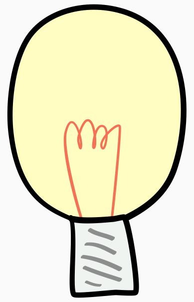
How should we schedule jobs to minimize delay?





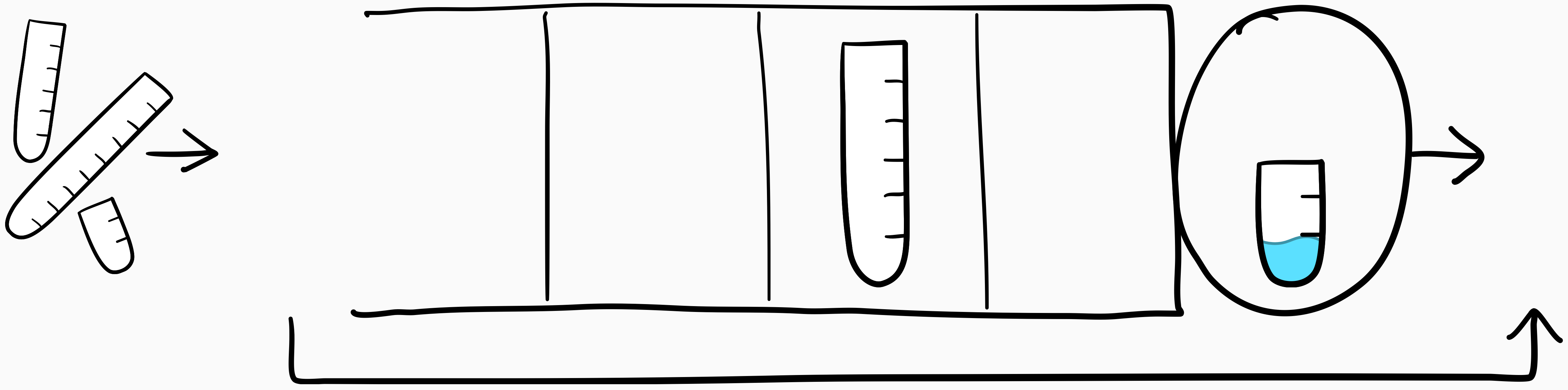
T = **response time**

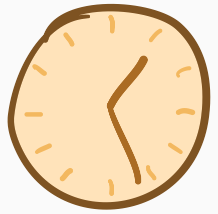
Minimize $\mathbf{E}[T]$?



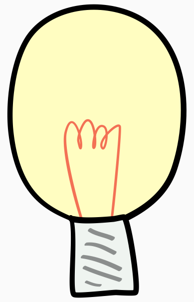
Serve short jobs
before long jobs

How should we schedule jobs to minimize delay?

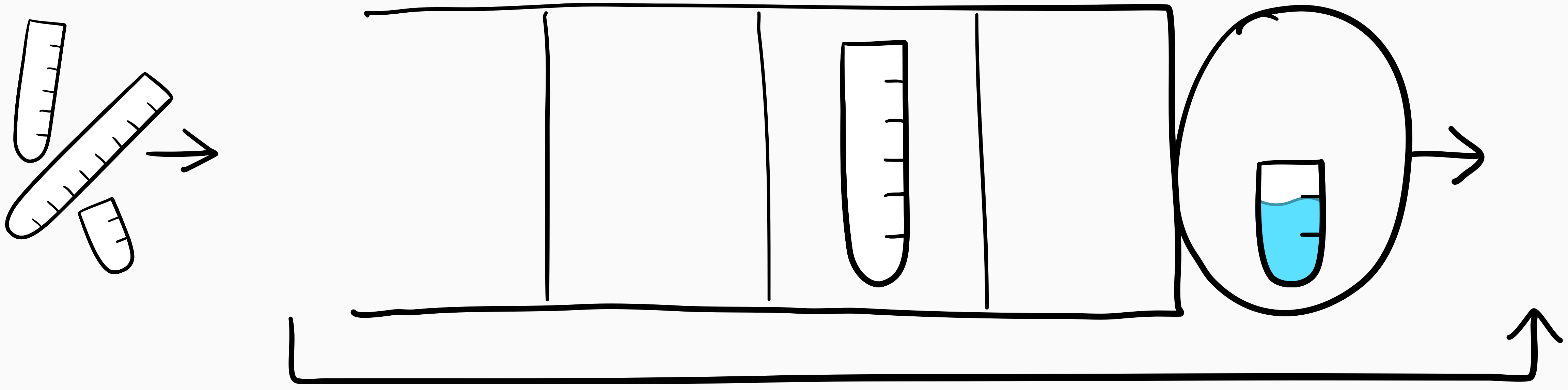


 T = **response time**

 Minimize $\mathbf{E}[T]$?

 Serve short jobs
before long jobs

How should we schedule jobs to minimize delay?

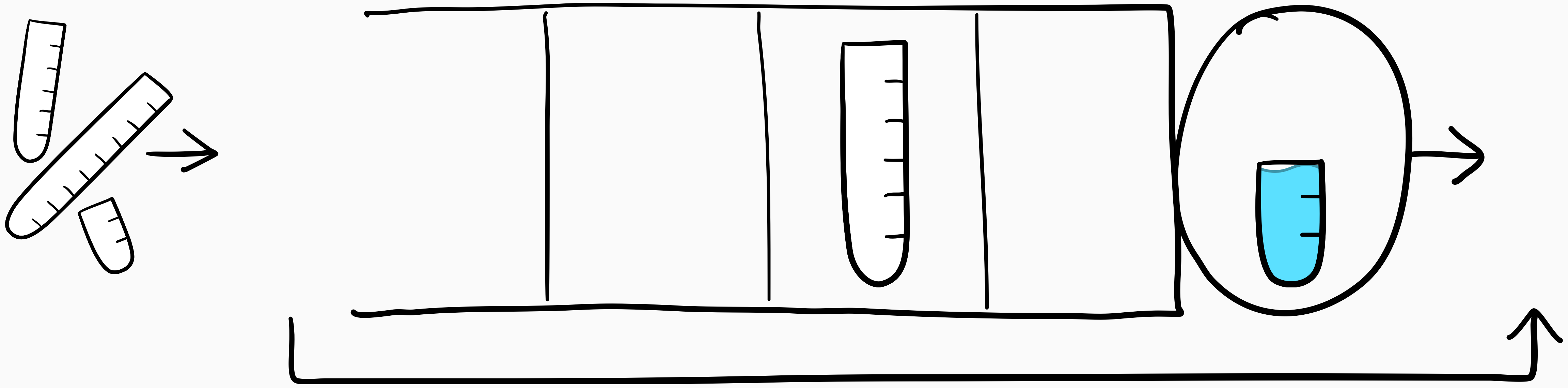


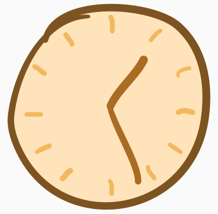
T = **response time**

Minimize $\mathbf{E}[T]$?

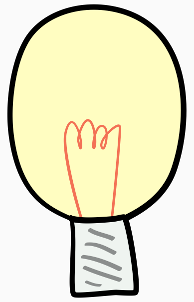
Serve short jobs
before long jobs

How should we schedule jobs to minimize delay?

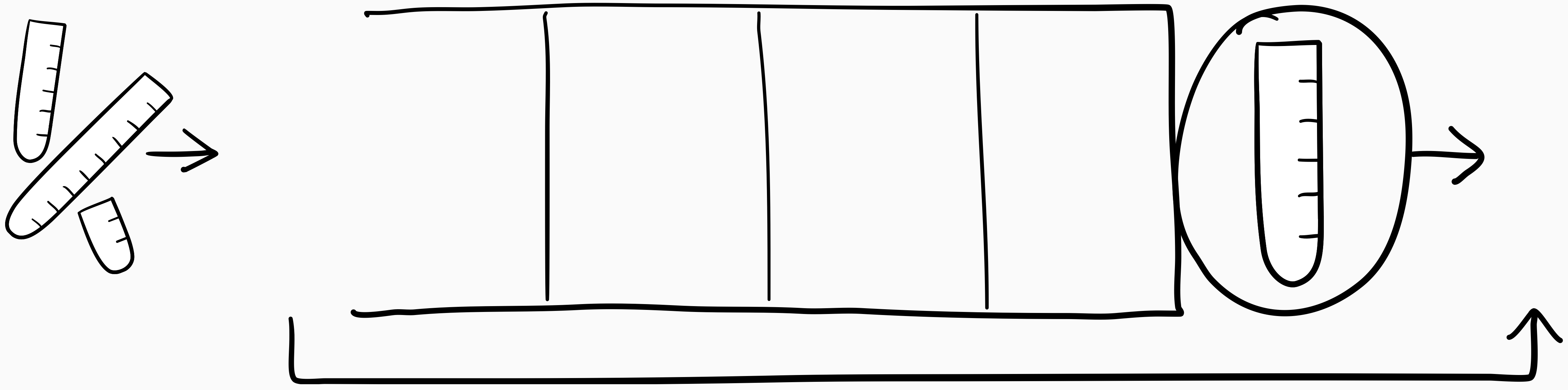


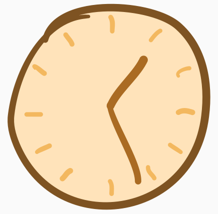
 T = **response time**

 Minimize $\mathbf{E}[T]$?

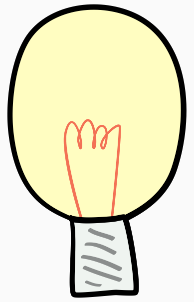
 Serve short jobs before long jobs

How should we schedule jobs to minimize delay?

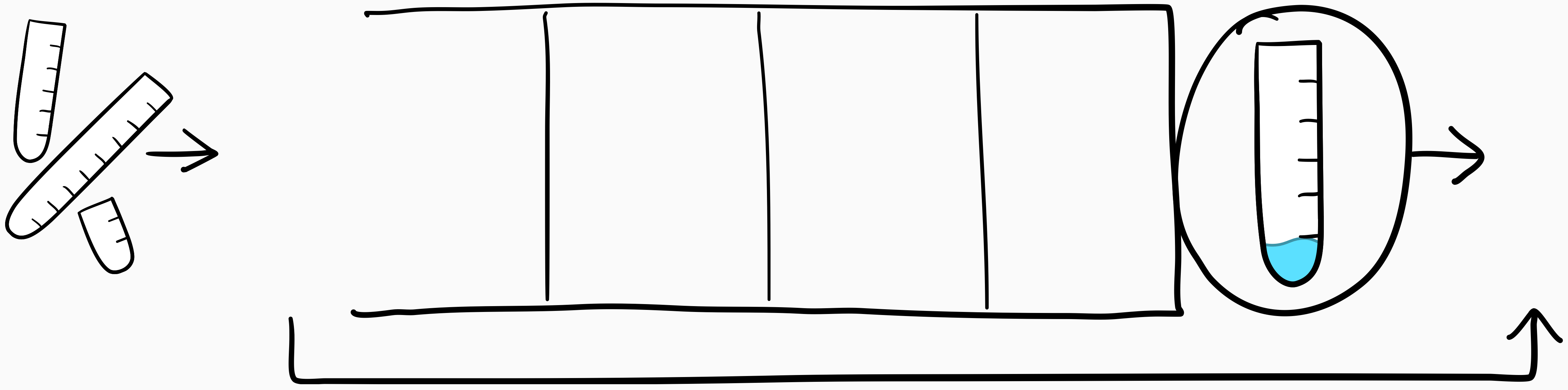


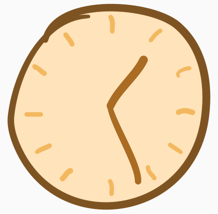
 T = **response time**

 Minimize $\mathbf{E}[T]$?

 Serve short jobs before long jobs

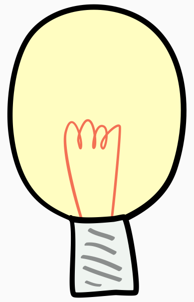
How should we schedule jobs to minimize delay?





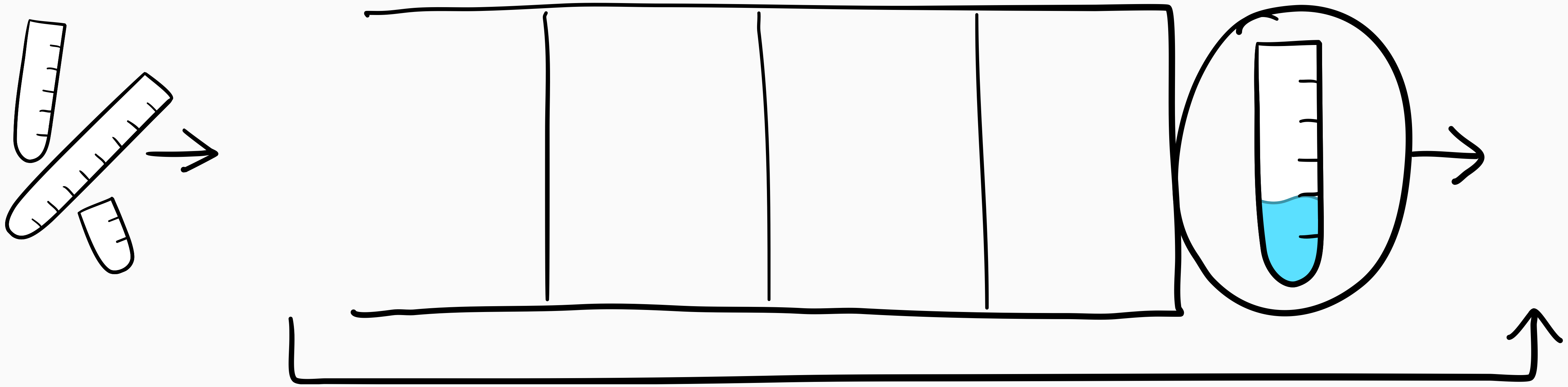
T = **response time**

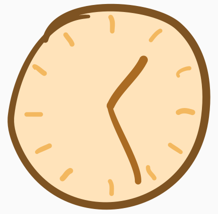
Minimize $\mathbf{E}[T]$?



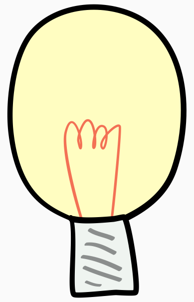
Serve short jobs
before long jobs

How should we schedule jobs to minimize delay?

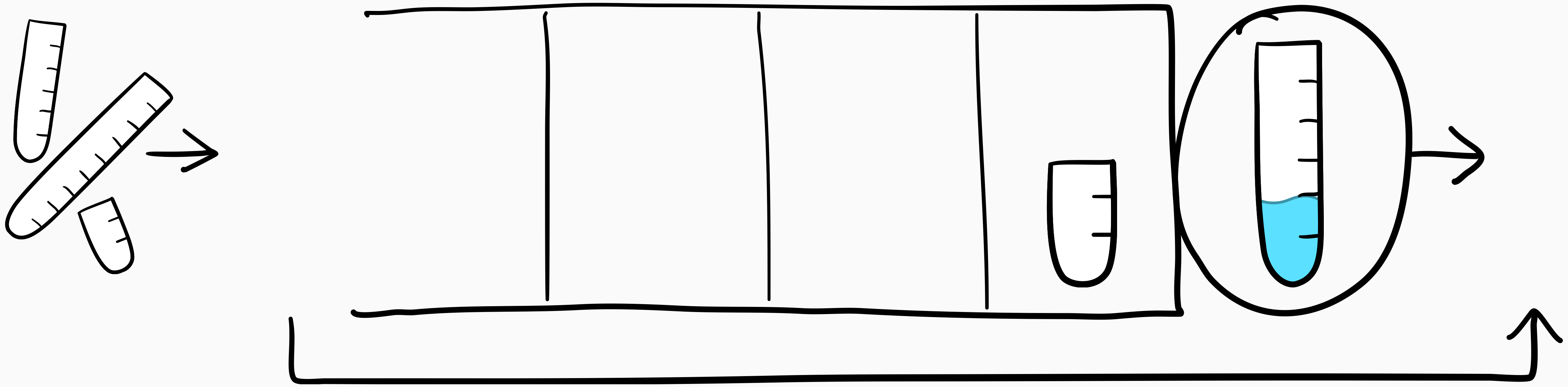


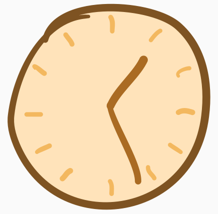
 T = **response time**

 Minimize $\mathbf{E}[T]$?

 Serve short jobs
before long jobs

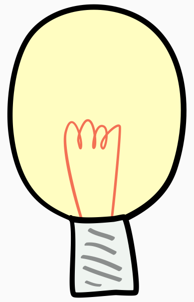
How should we schedule jobs to minimize delay?





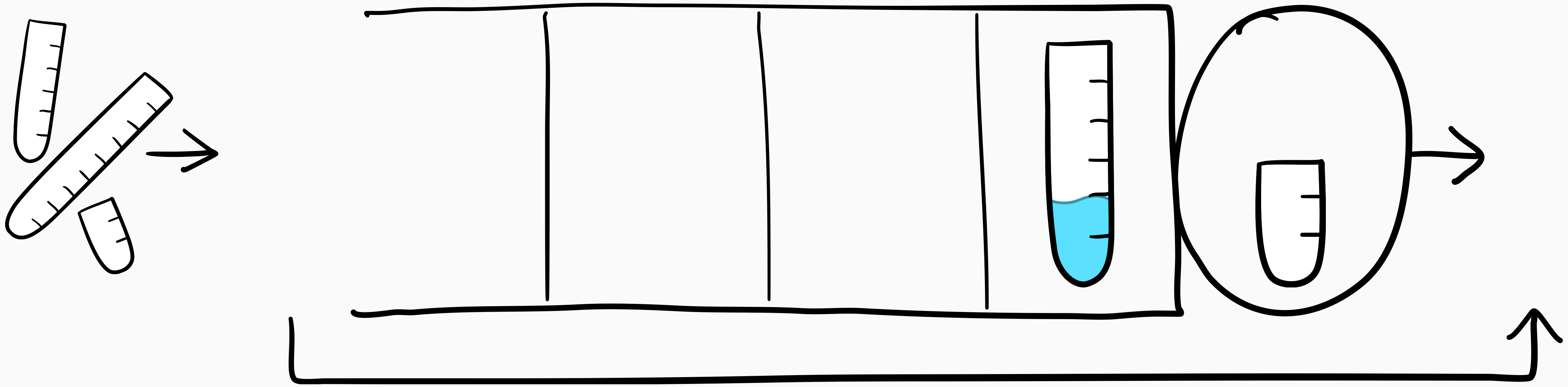
T = **response time**

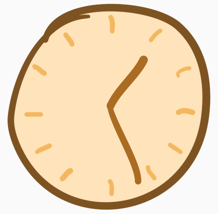
Minimize $\mathbf{E}[T]$?



Serve short jobs
before long jobs

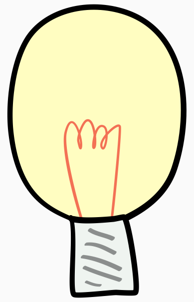
How should we schedule jobs to minimize delay?





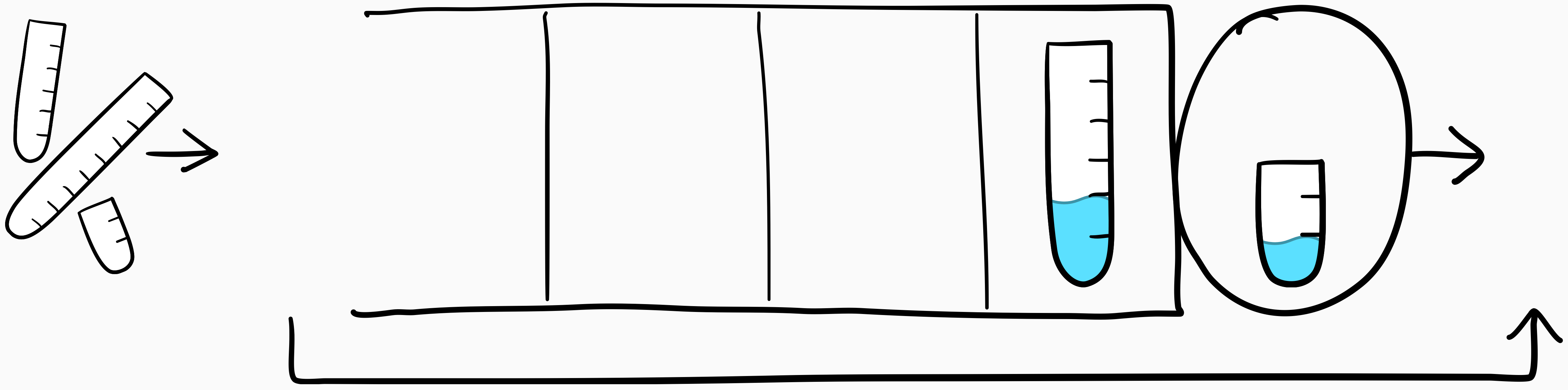
T = **response time**

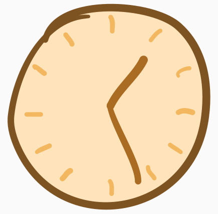
Minimize $\mathbf{E}[T]$?



Serve short jobs
before long jobs

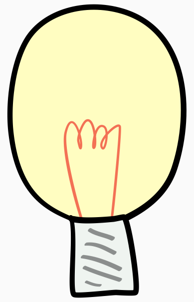
How should we schedule jobs to minimize delay?





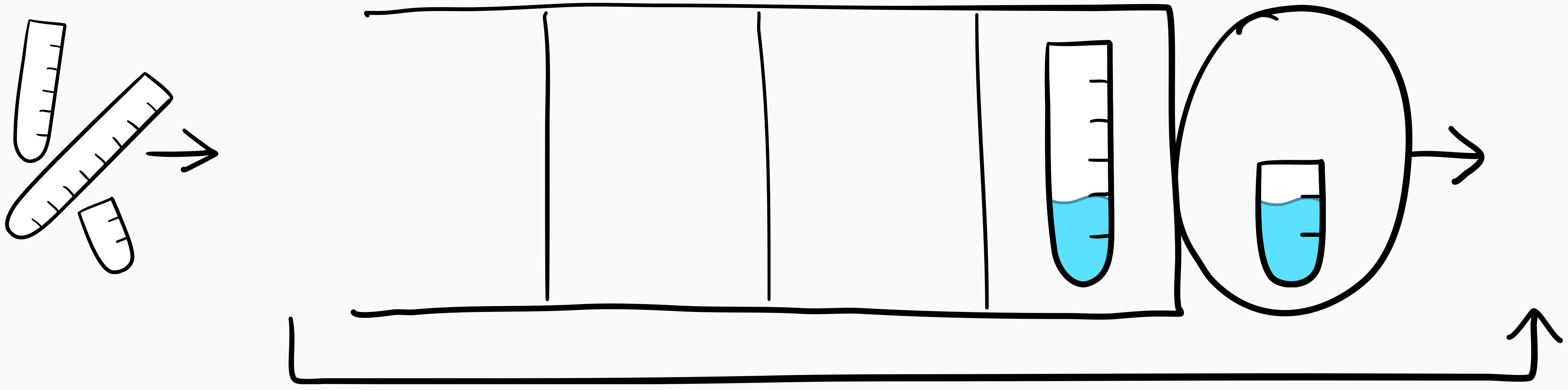
T = **response time**

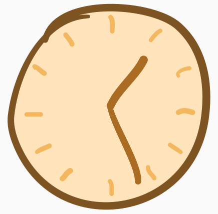
Minimize $\mathbf{E}[T]$?



Serve short jobs
before long jobs

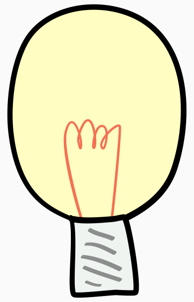
How should we schedule jobs to minimize delay?





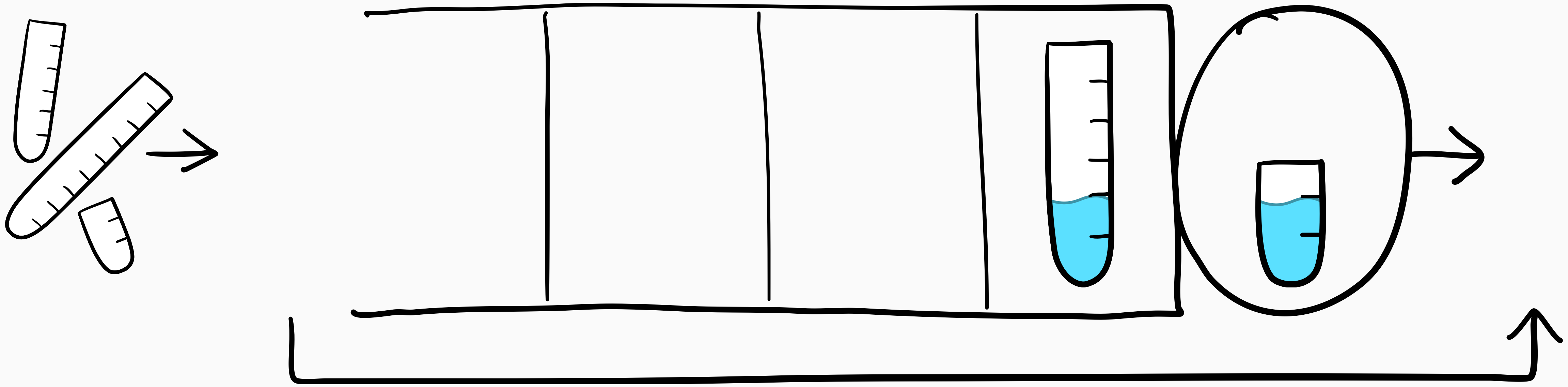
T = **response time**

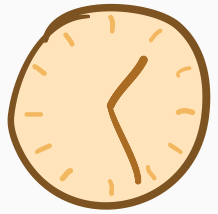
Minimize $\mathbf{E}[T]$?



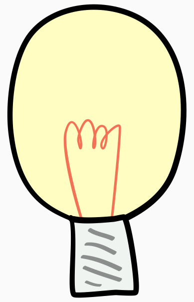
Serve short jobs
before long jobs

How should we schedule jobs to minimize delay?



 T = **response time**

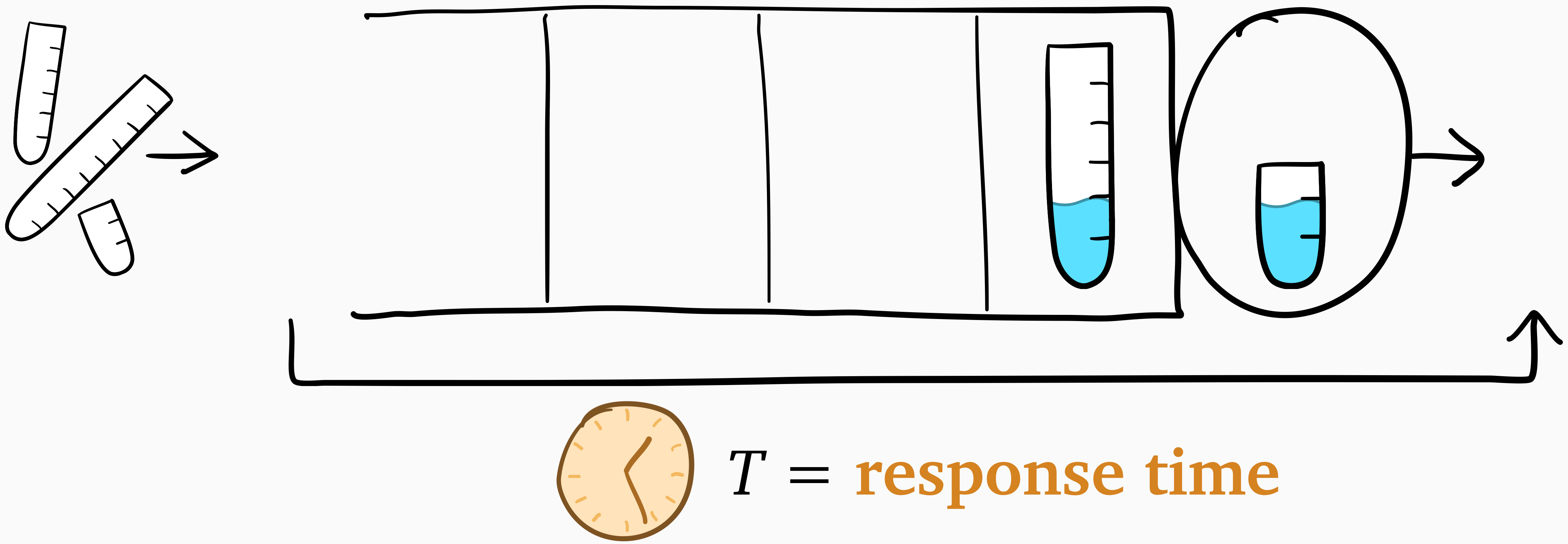
 Minimize $\mathbf{E}[T]$?

 Serve short jobs before long jobs

shortest remaining processing time

 **SRPT:** minimizes $\mathbf{E}[T]$

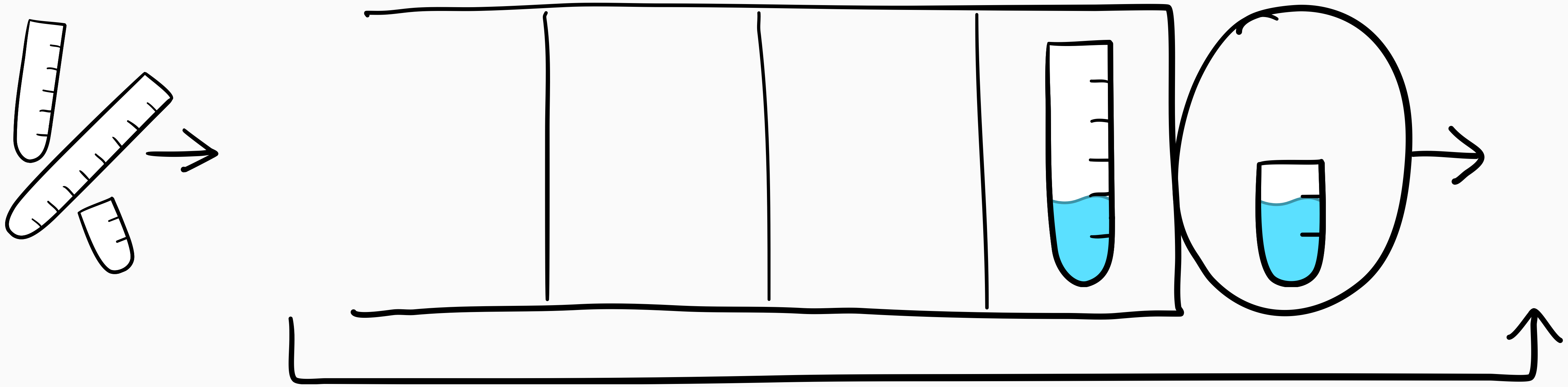
Beyond the mean: tail metrics



T = **response time**

Beyond the mean: tail metrics

$$\text{Minimize} \begin{cases} \mathbf{P}[T > t]? \\ \mathbf{E}[(T - t)^+]? \\ \text{quantiles of } T? \end{cases}$$

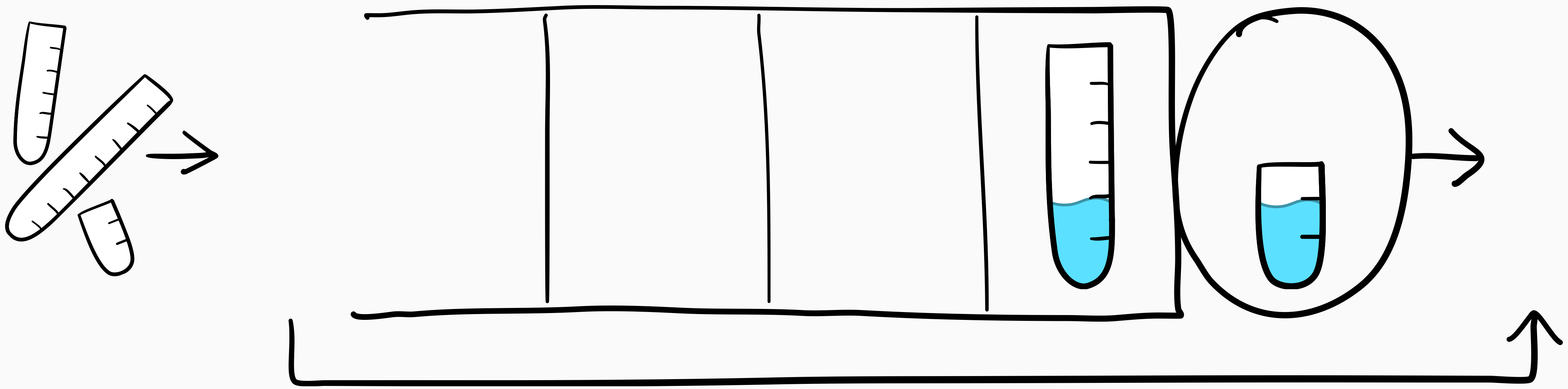


T = **response time**

Beyond the mean: tail metrics

$$\text{Minimize} \begin{cases} \mathbf{P}[T > t]? \\ \mathbf{E}[(T - t)^+]? \\ \text{quantiles of } T? \end{cases}$$

Practice: important



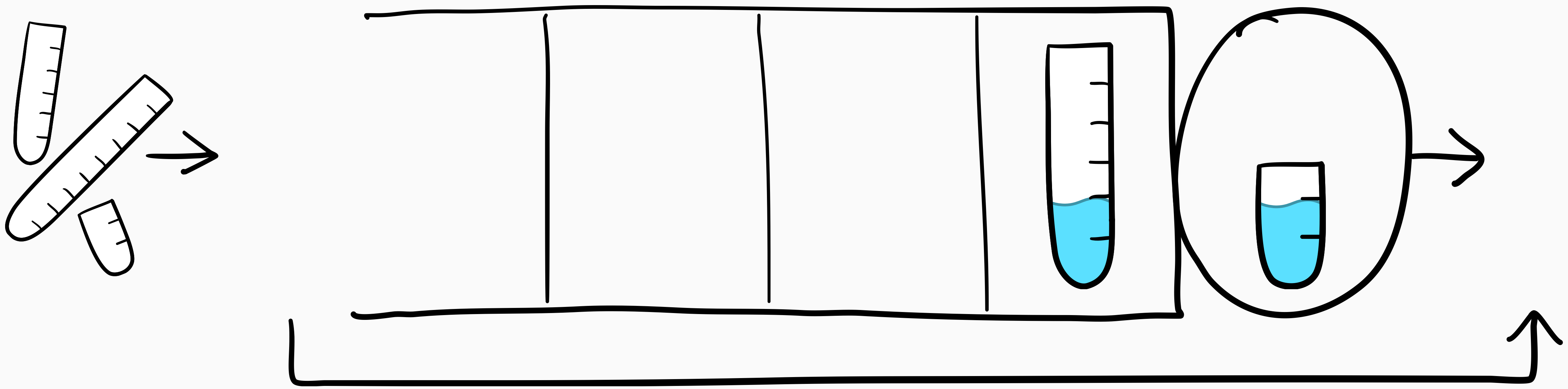
T = **response time**

Beyond the mean: tail metrics

Minimize $\begin{cases} \mathbf{P}[T > t]? \\ \mathbf{E}[(T - t)^+]? \\ \text{quantiles of } T? \end{cases}$

Practice: important

Theory: very hard



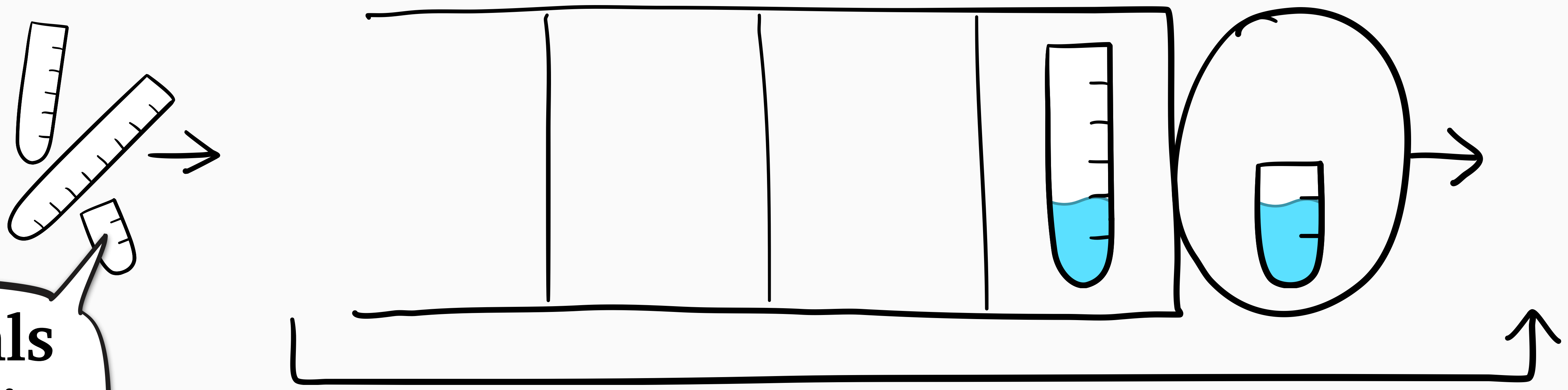
T = **response time**

Beyond the mean: tail metrics

Minimize $\begin{cases} \mathbf{P}[T > t]? \\ \mathbf{E}[(T - t)^{+}]? \\ \text{quantiles of } T? \end{cases}$

Practice: important

Theory: very hard



M/G arrivals

- arrival rate λ
- job size dist. S

T = **response time**

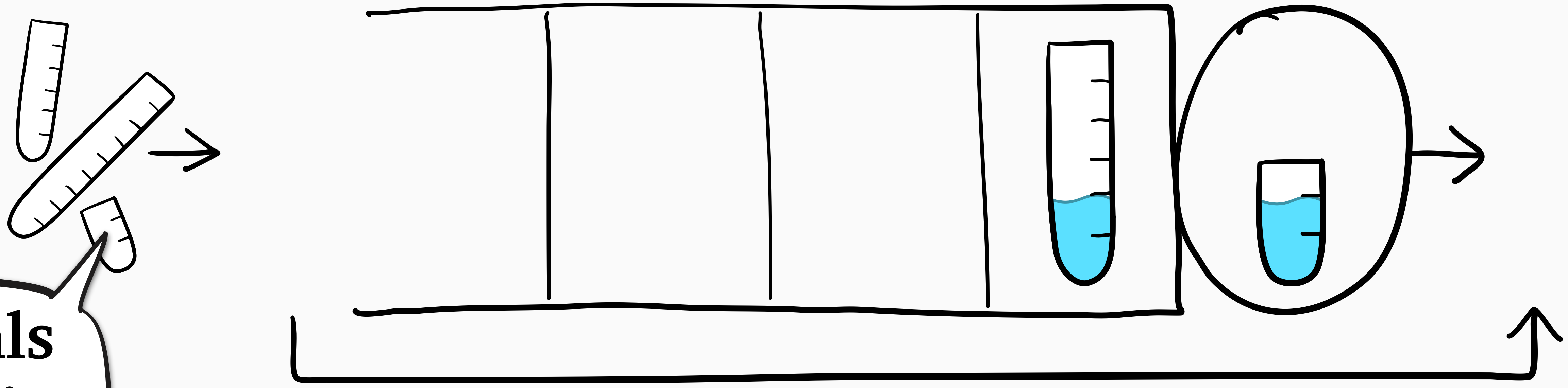
Beyond the mean: tail metrics

Minimize $\begin{cases} \mathbf{P}[T > t]? \\ \mathbf{E}[(T - t)^+]? \\ \text{quantiles of } T? \end{cases}$

Practice: important

Theory: very hard

Tractable:
study $t \to \infty$
asymptotics



M/G arrivals
- arrival rate λ
- job size dist. S

T = **response time**

Beyond the mean: tail metrics

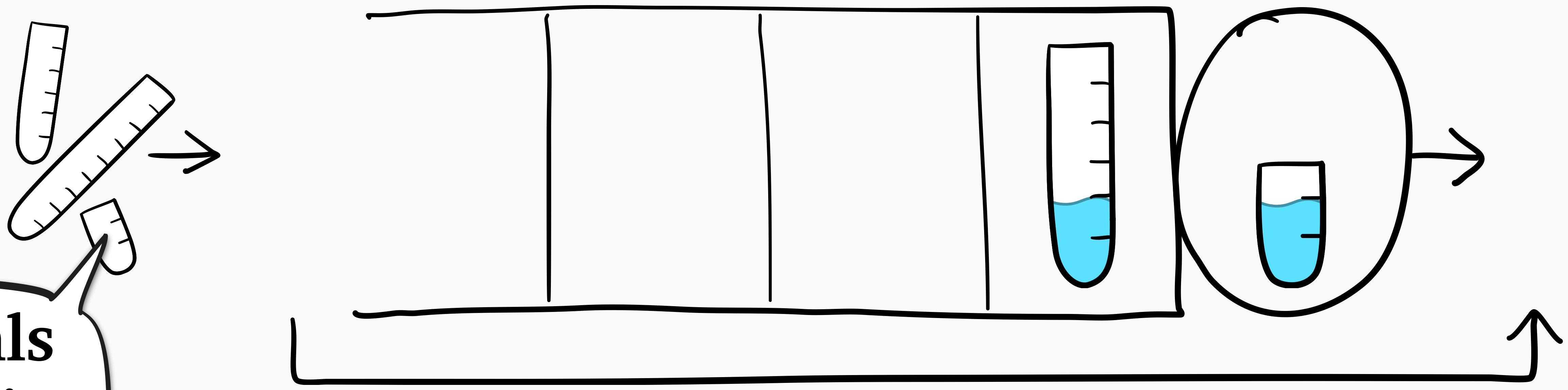
no single t value is most important

Minimize
- $\mathbf{P}[T > t]$?
- $\mathbf{E}[(T - t)^+]$?
- quantiles of T?

Practice: important

Theory: very hard

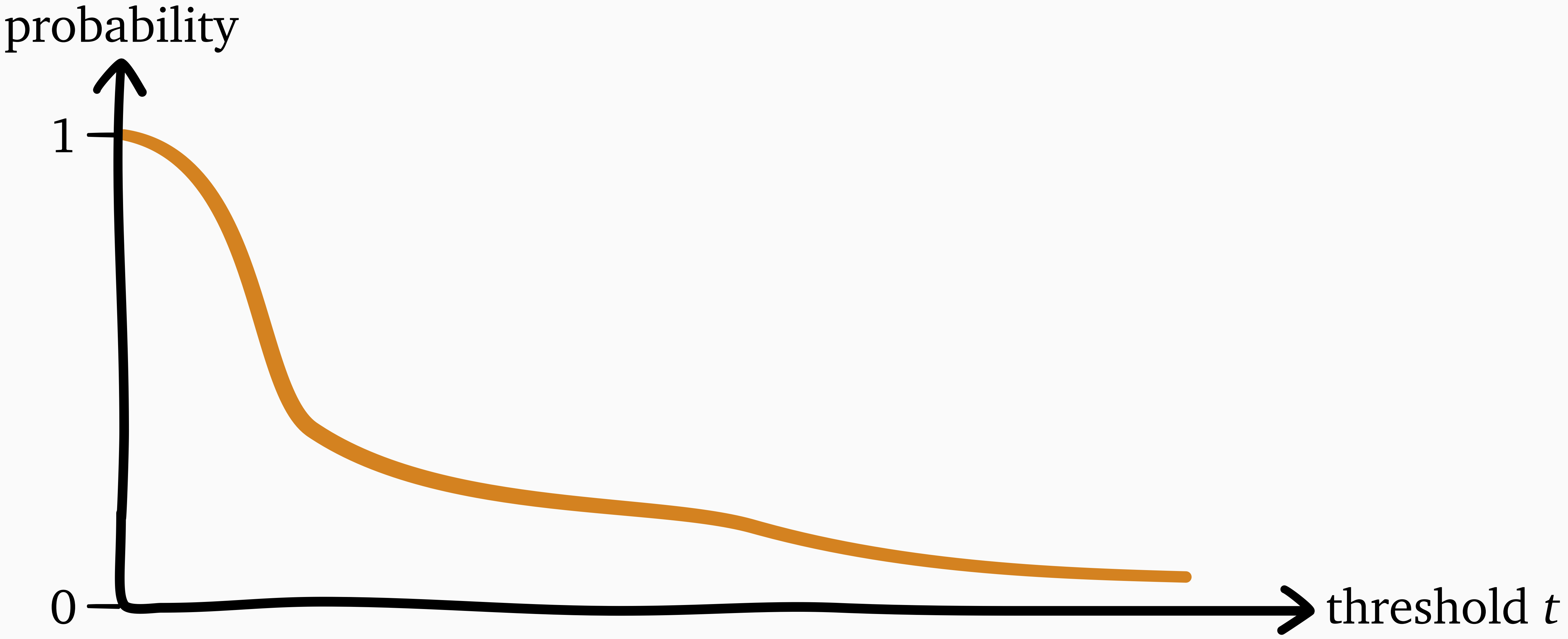
Tractable:
study $t \to \infty$
asymptotics



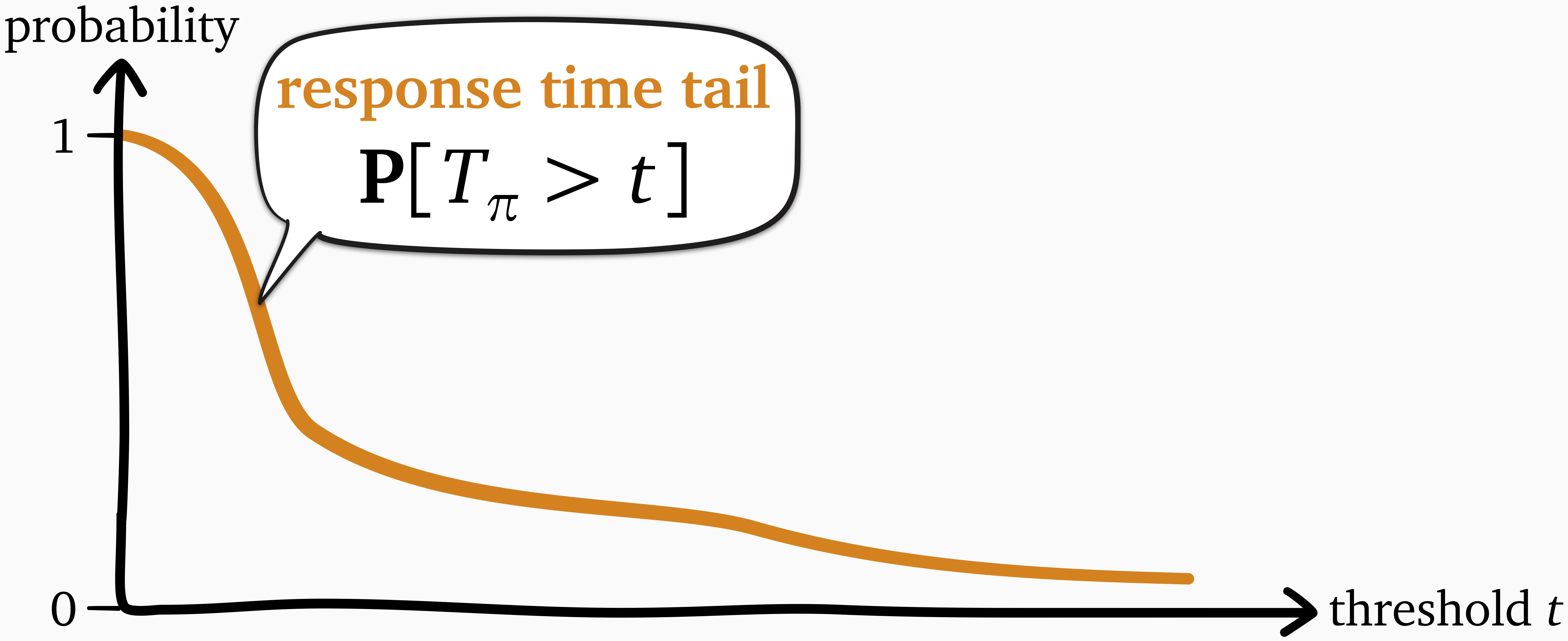
M/G arrivals
- arrival rate λ
- job size dist. S

$T =$ **response time**

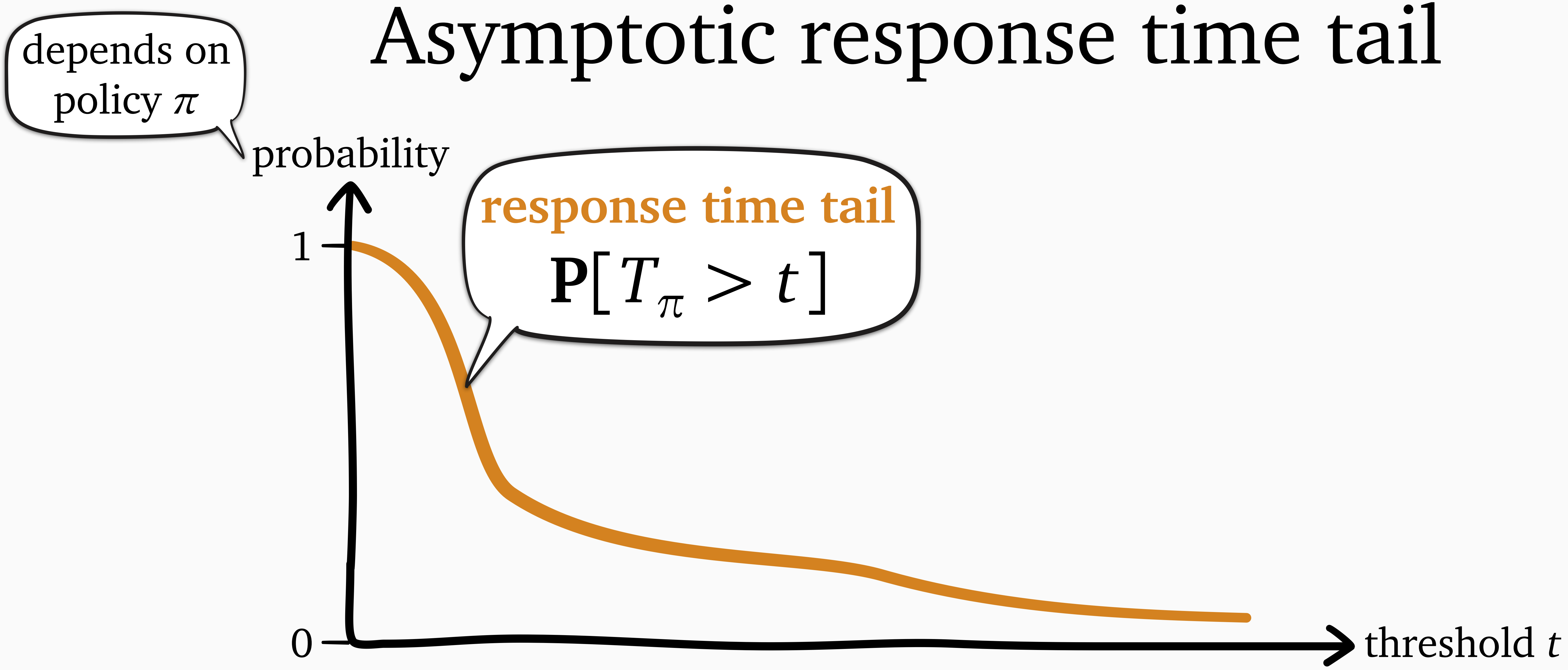
Asymptotic response time tail



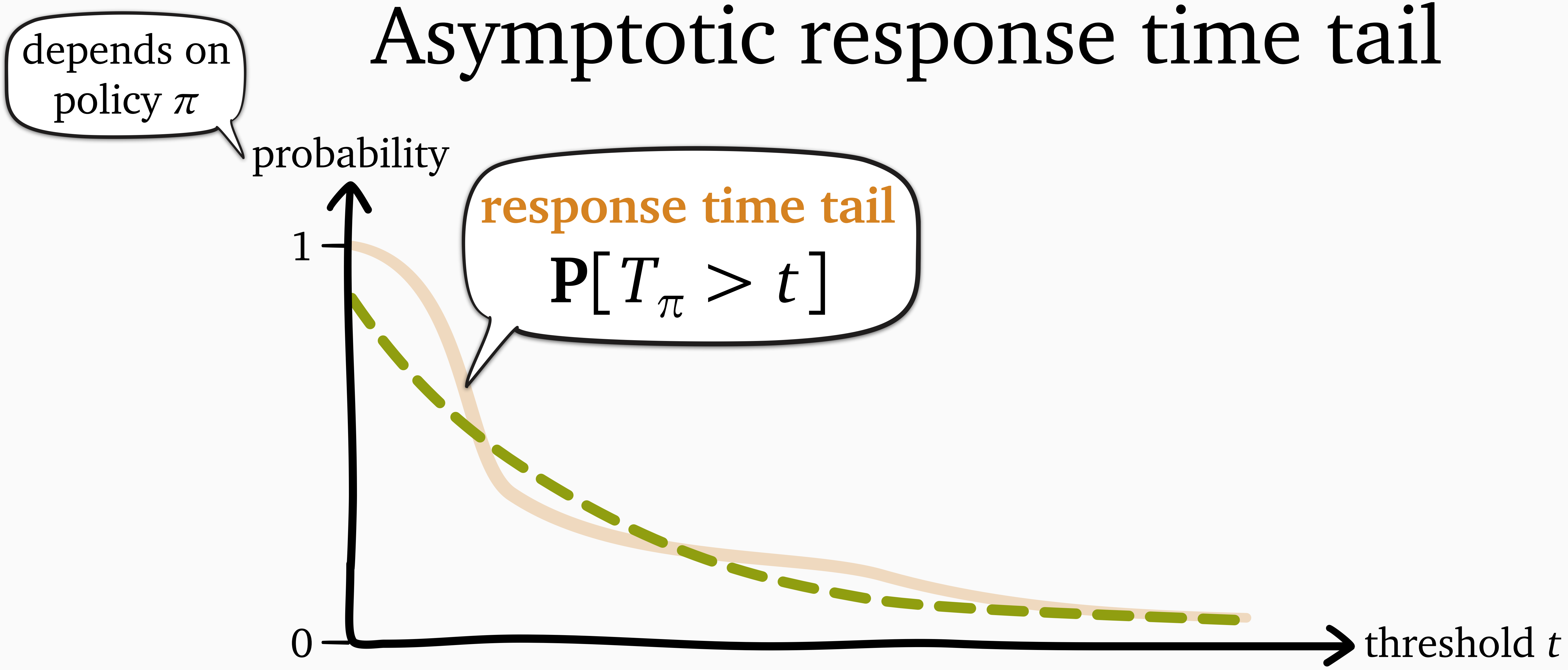
Asymptotic response time tail



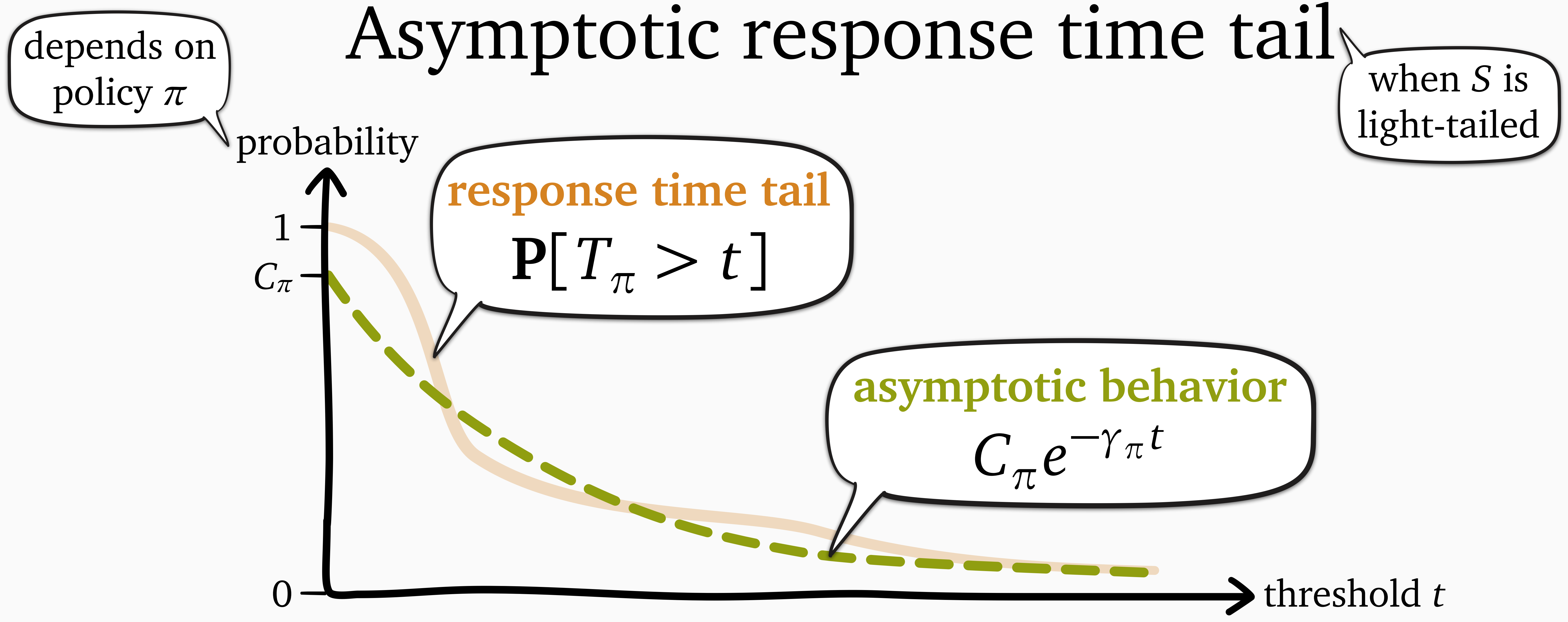
Asymptotic response time tail



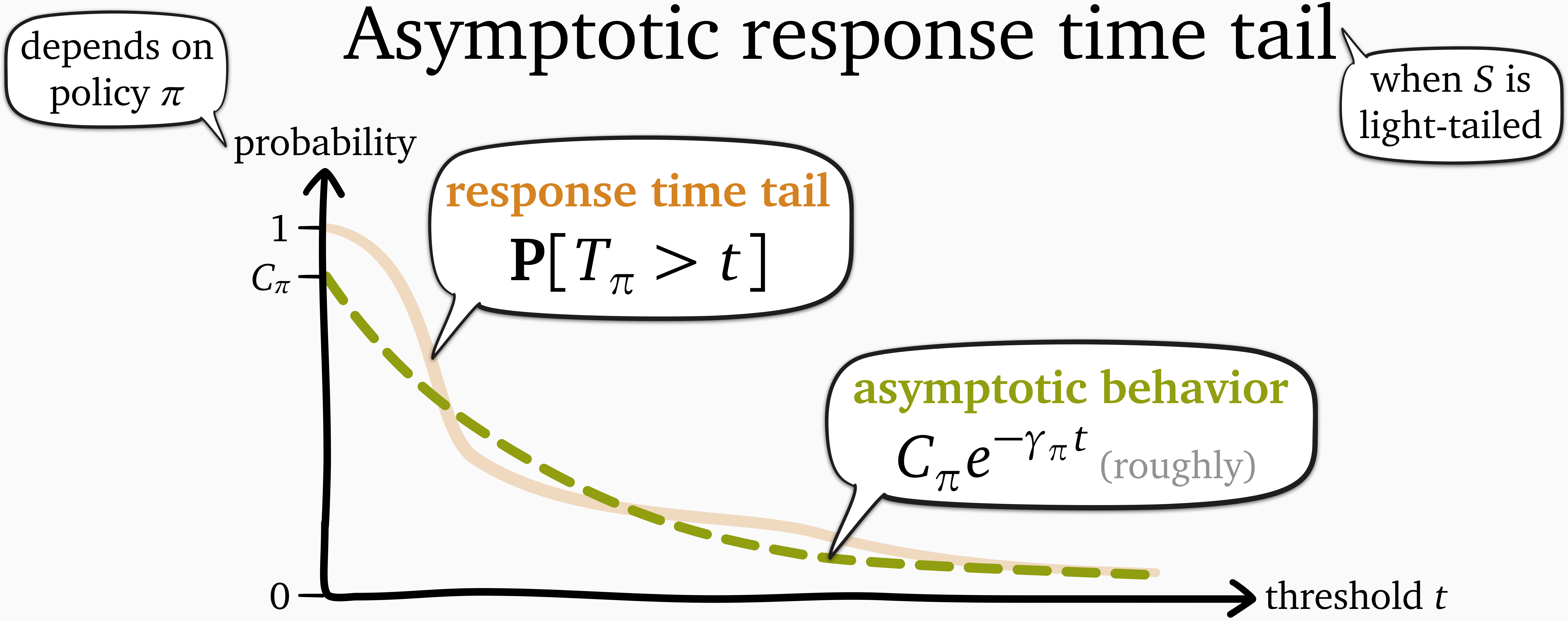
Asymptotic response time tail



Asymptotic response time tail



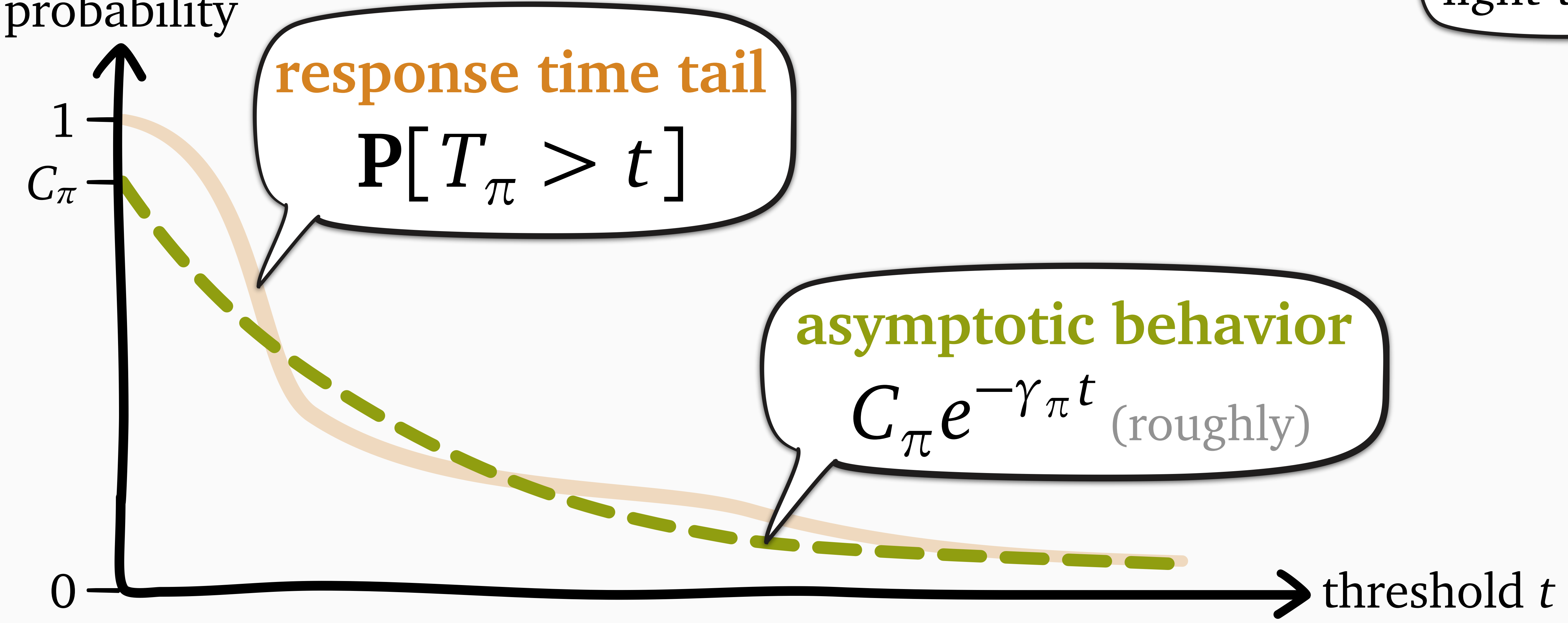
Asymptotic response time tail



Asymptotic response time tail

depends on policy π

when S is light-tailed



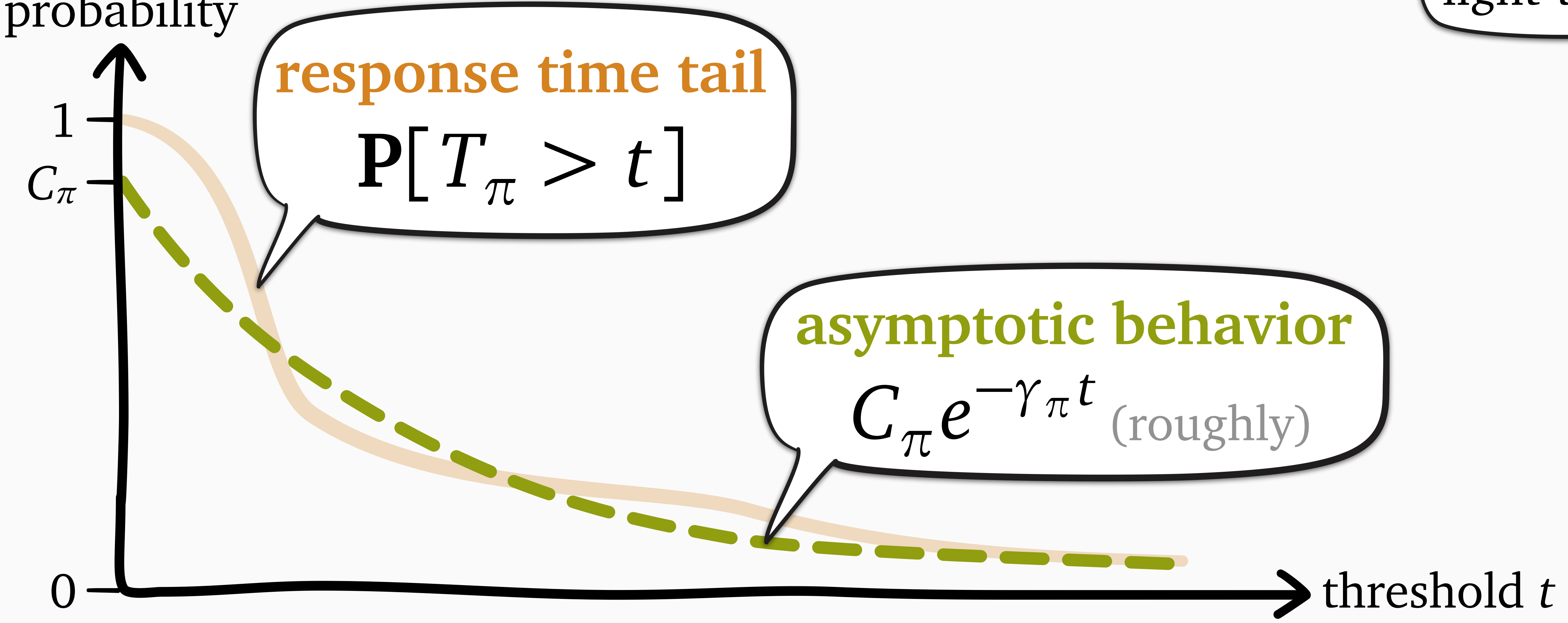
$$\gamma_\pi = \textit{decay rate} \text{ of } \pi$$

$$C_\pi = \textit{tail constant} \text{ of } \pi$$

Asymptotic response time tail

when S is light-tailed

depends on policy π



$\gamma_\pi = \textit{decay rate}$ of π

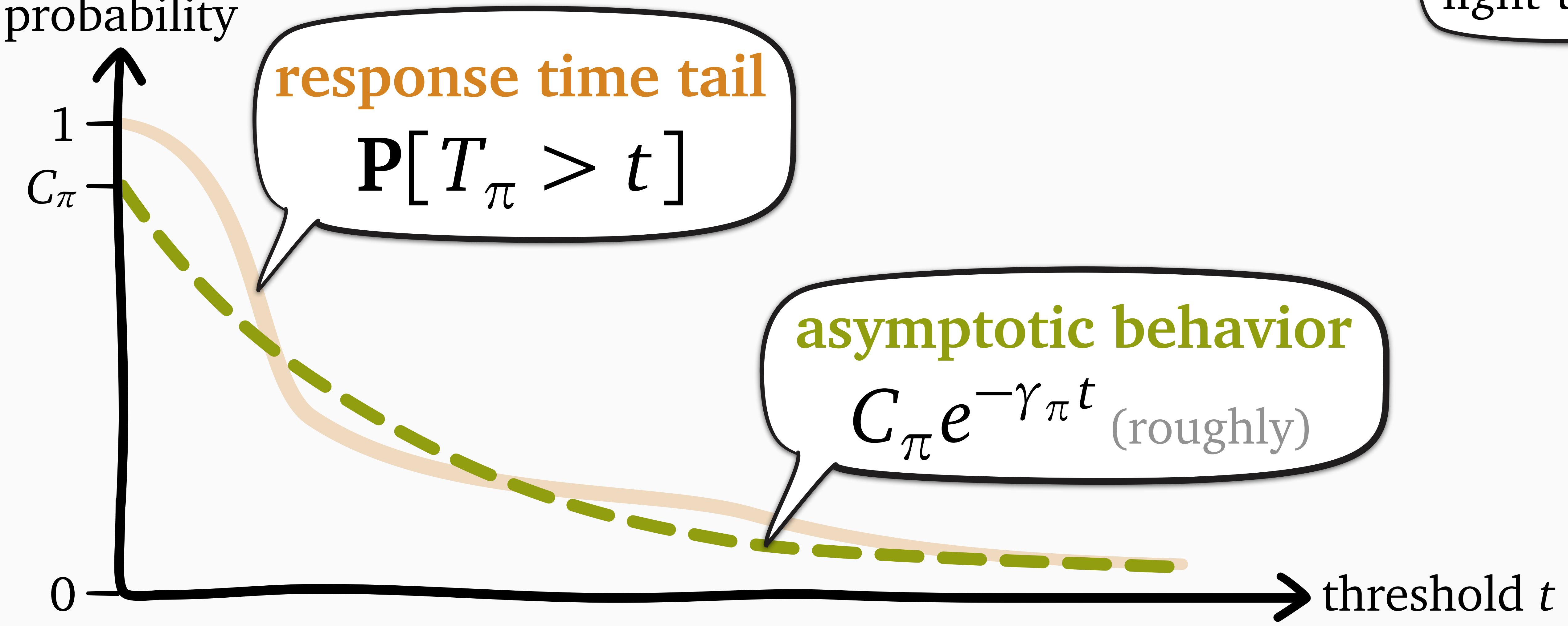
$C_\pi = \textit{tail constant}$ of π

Weak optimality:
optimal γ_π

Asymptotic response time tail

depends on policy π

when S is light-tailed



$\gamma_\pi = \textit{decay rate}$ of π

$C_\pi = \textit{tail constant}$ of π

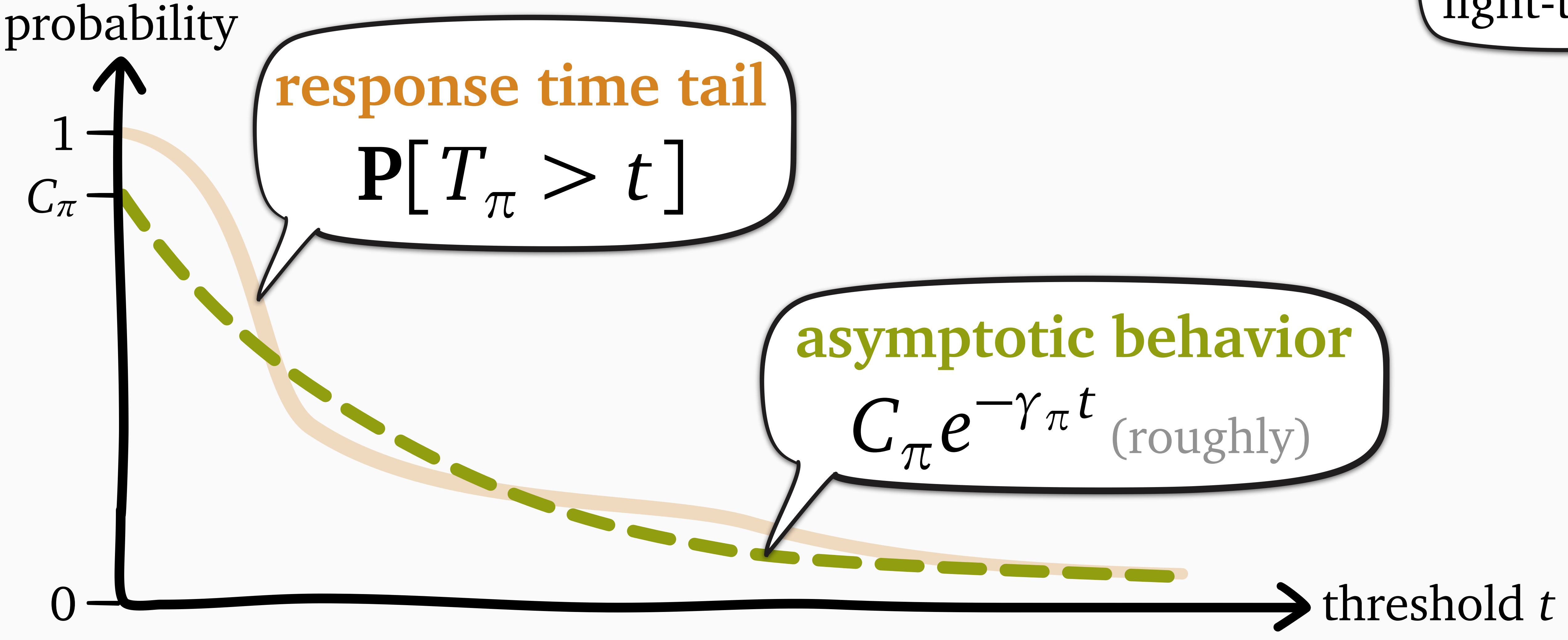
Weak optimality: optimal γ_π

Strong optimality: optimal γ_π and C_π

Asymptotic response time tail

depends on policy π

when S is light-tailed



Weak optimality:
optimal γ_π

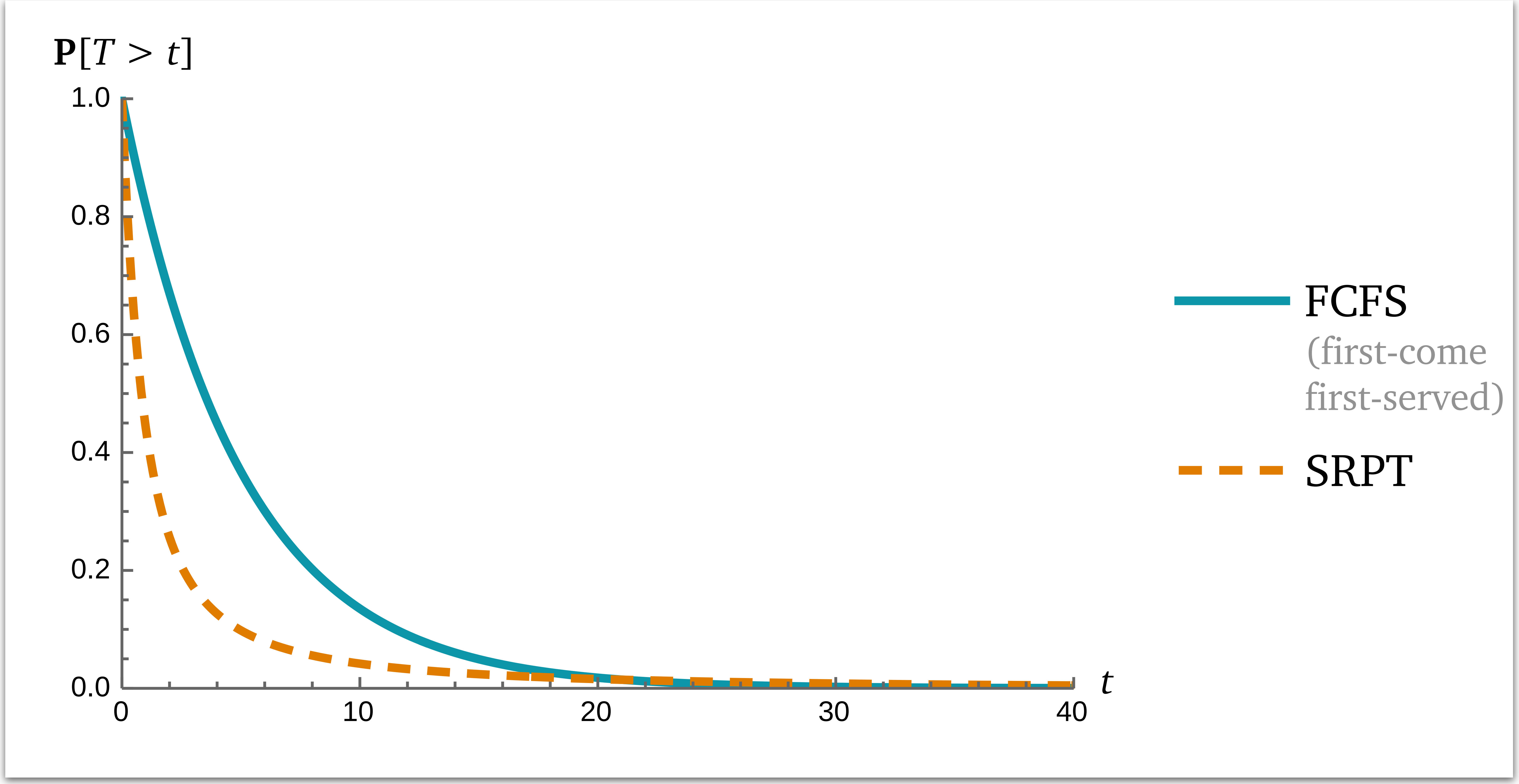
$\gamma_\pi = \textit{decay rate}$ of π

$C_\pi = \textit{tail constant}$ of π

Strong optimality:
optimal γ_π and C_π

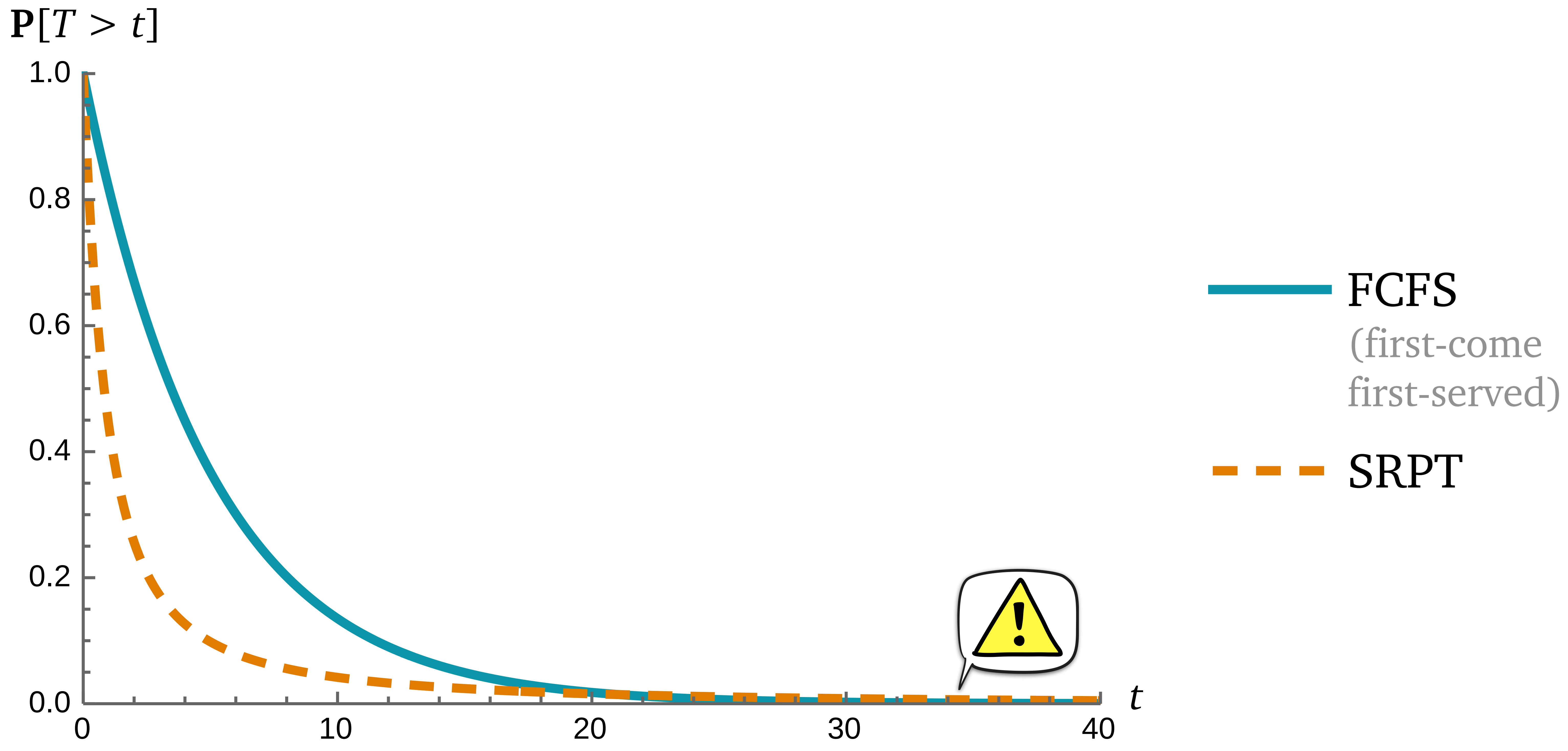
(roughly)

FCFS vs. SRPT

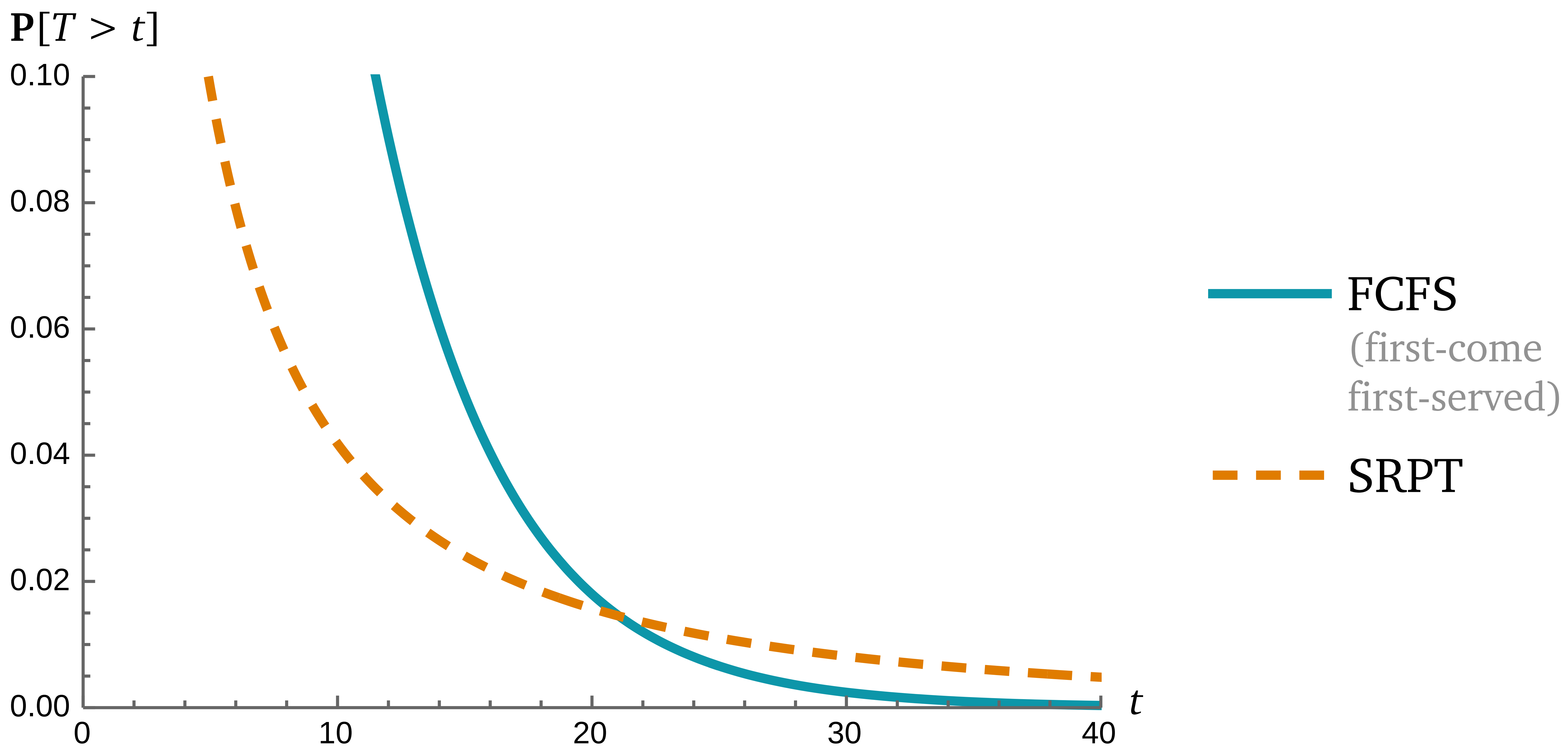


P[T > t]
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
t
FCFS
(first-come first-served)
SRPT

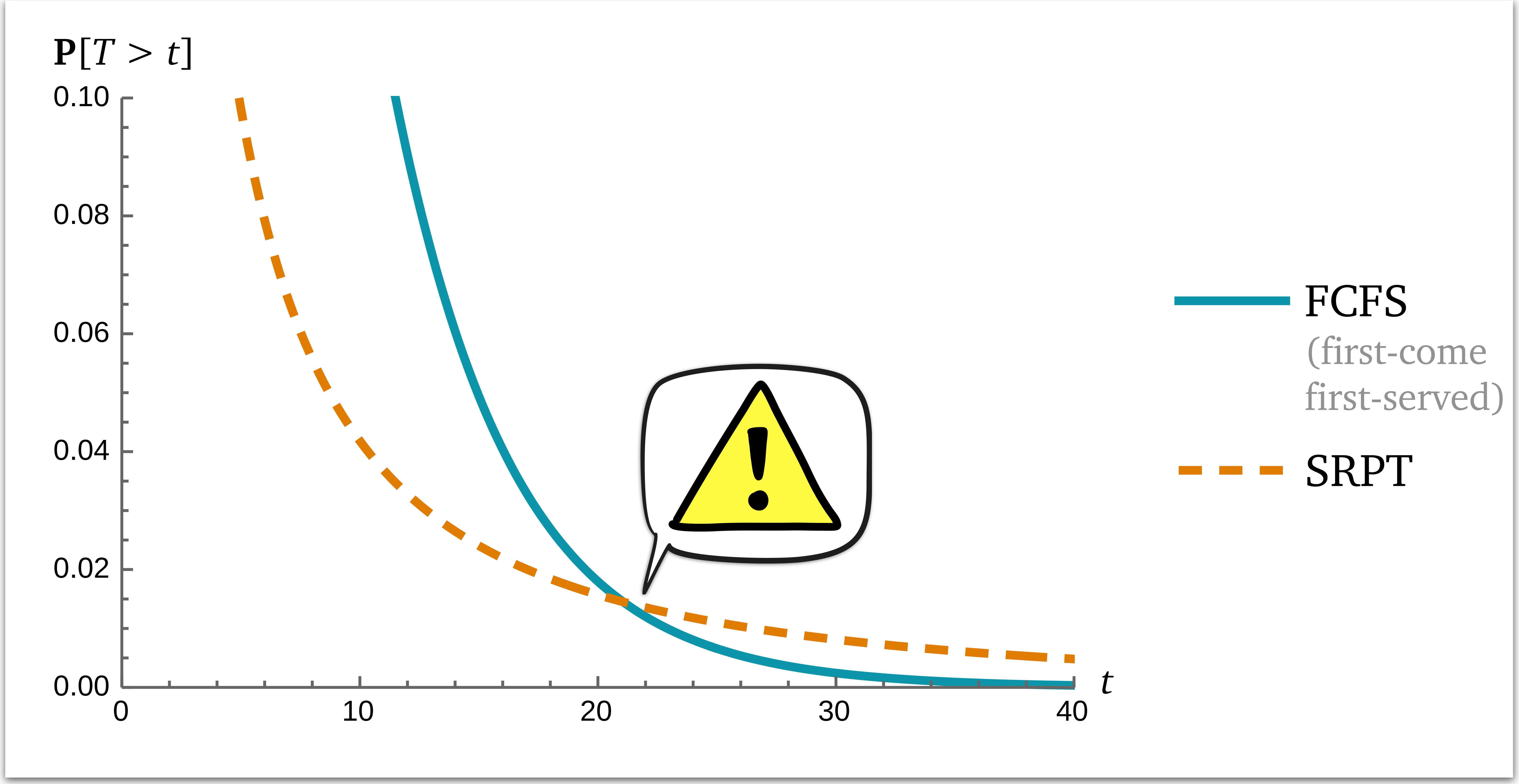
FCFS vs. SRPT


$\mathbf{P}[T > t]$
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
t
FCFS
(first-come first-served)
SRPT

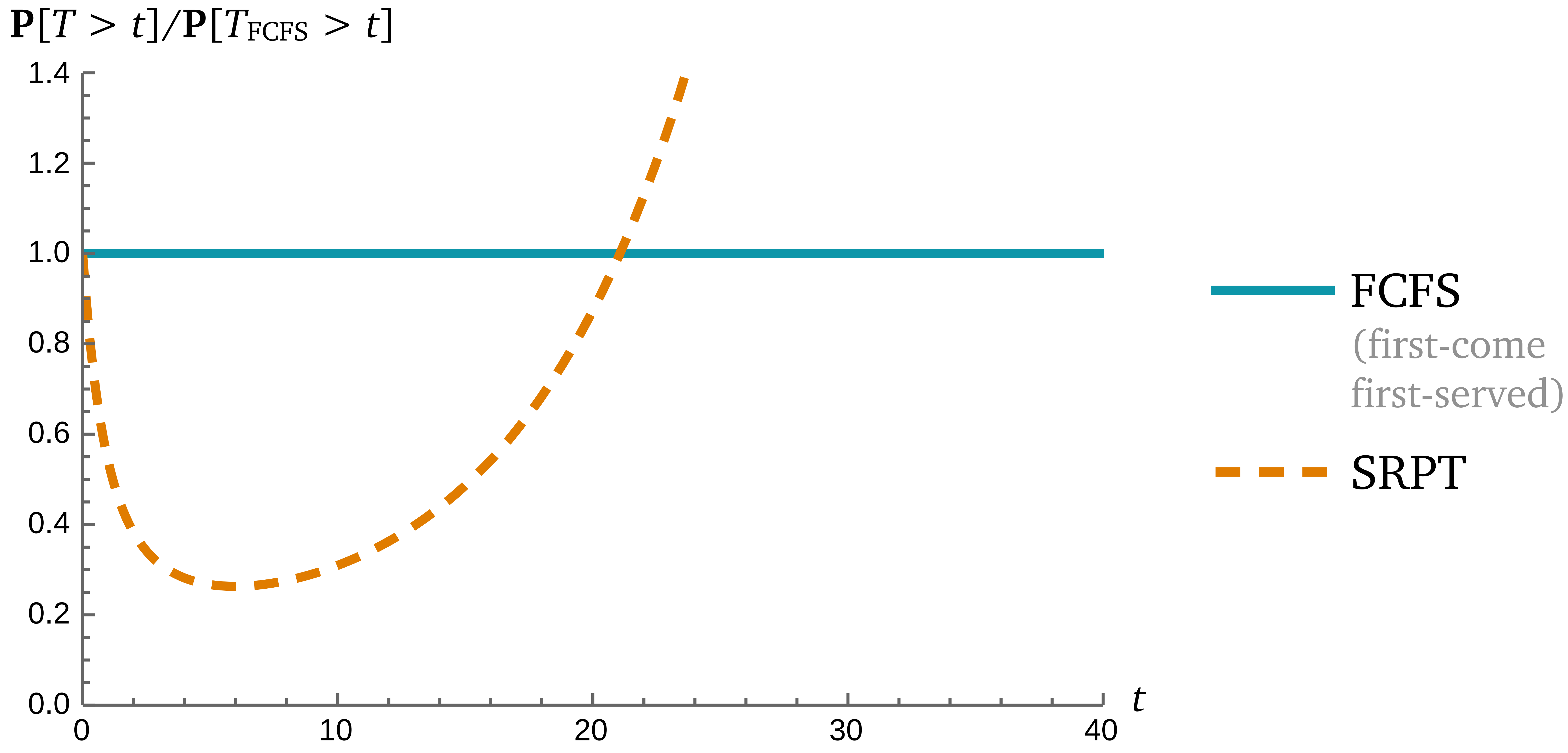
FCFS vs. SRPT


P[T > t]
0.10
0.08
0.06
0.04
0.02
0.00
0
10
20
30
40
t
FCFS
(first-come
first-served)
SRPT

FCFS vs. SRPT

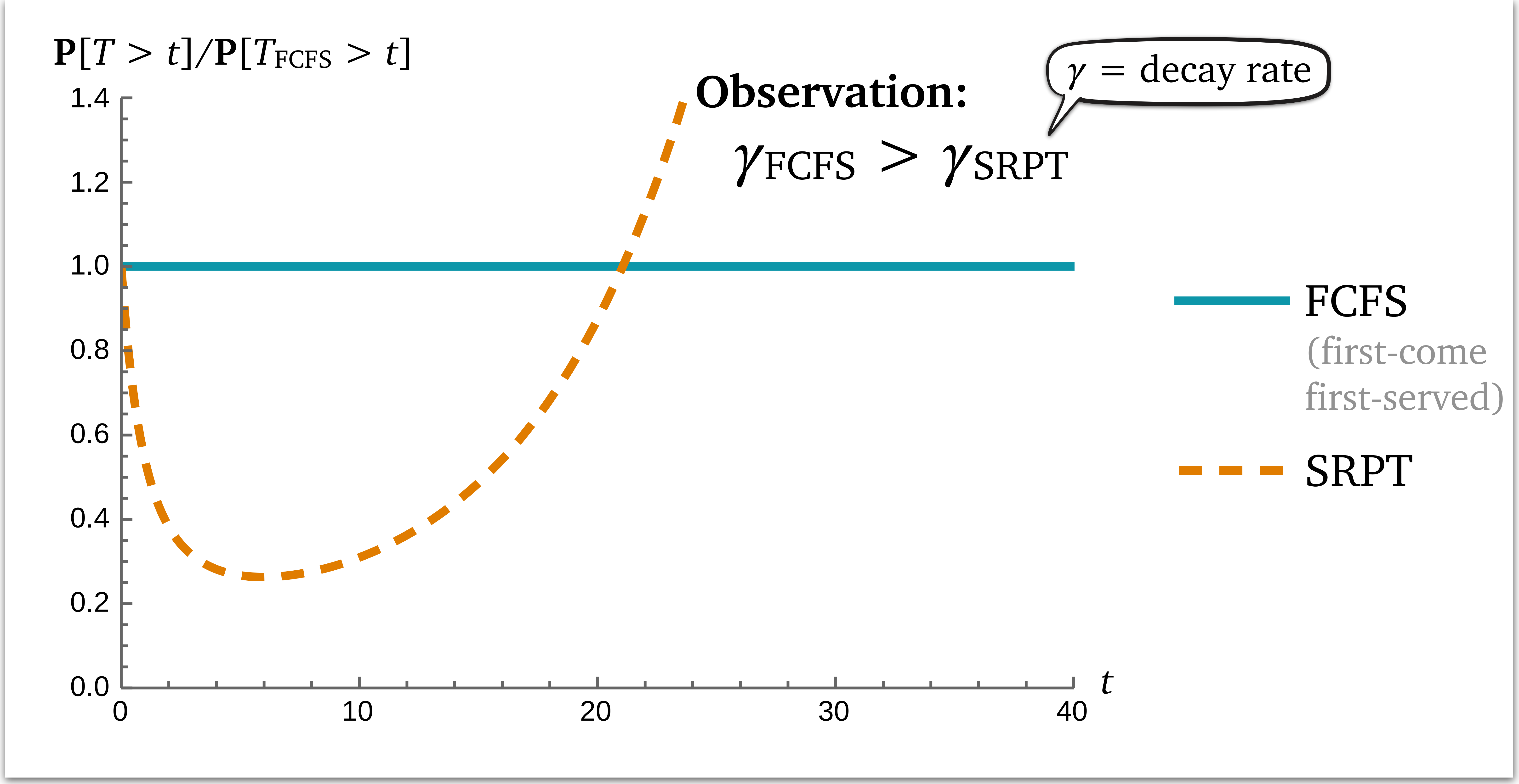

$\mathbf{P}[T > t]$
0.10
0.08
0.06
0.04
0.02
0.00
0
10
20
30
40
t
FCFS
(first-come first-served)
SRPT

FCFS vs. SRPT



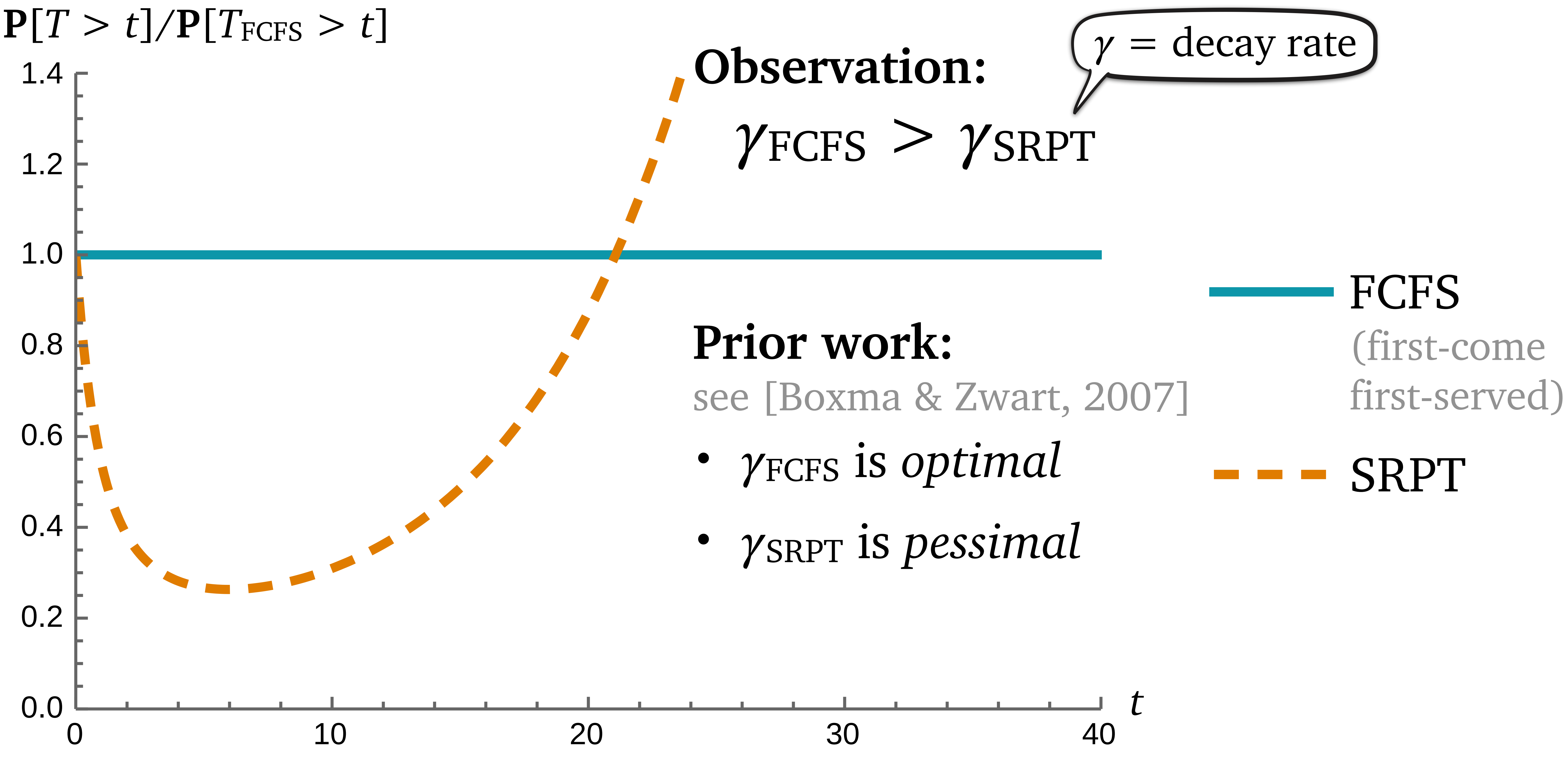
$\mathbf{P}[T > t]/\mathbf{P}[T_{\text{FCFS}} > t]$
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
t
FCFS
(first-come first-served)
SRPT

FCFS vs. SRPT



$\mathbf{P}[T > t]/\mathbf{P}[T_{\text{FCFS}} > t]$
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
t
Observation:
$\gamma_{\text{FCFS}} > \gamma_{\text{SRPT}}$
γ = decay rate
FCFS
(first-come first-served)
SRPT

FCFS vs. SRPT



Observation:

$\gamma_{\text{FCFS}} > \gamma_{\text{SRPT}}$

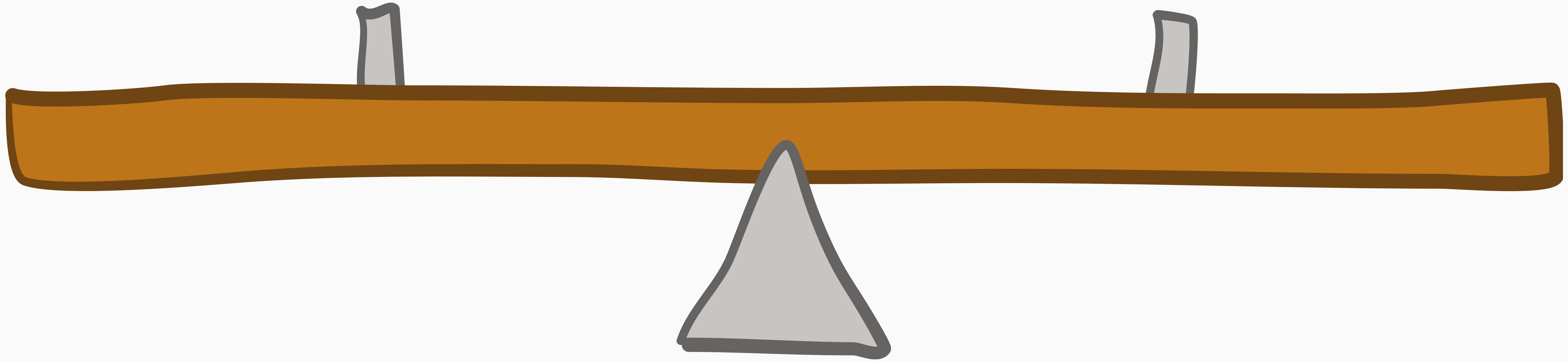
γ = decay rate

Prior work:

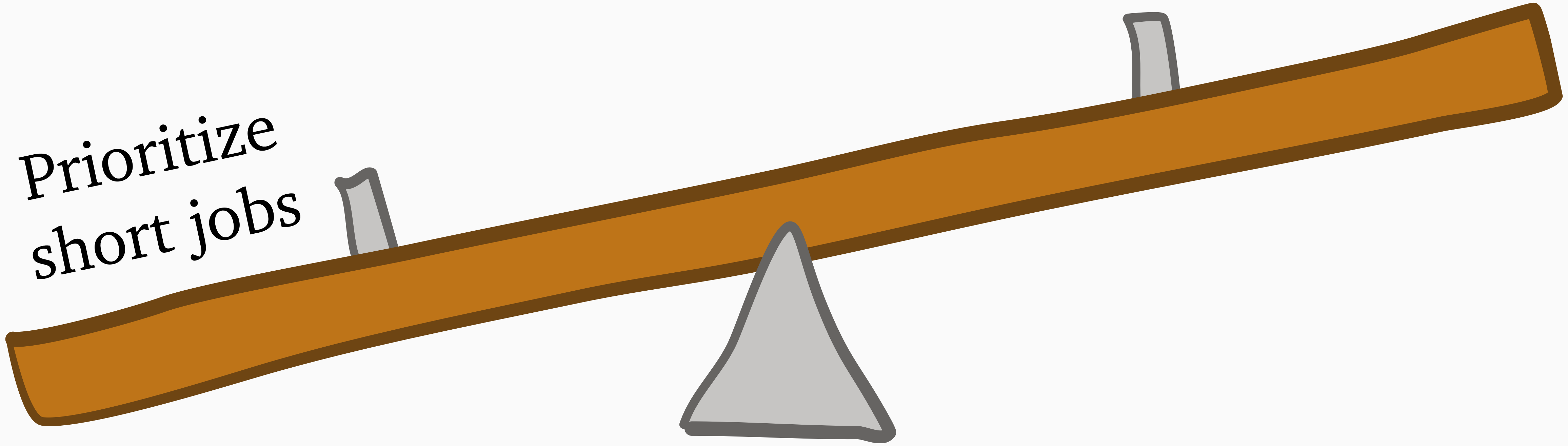
see [Boxma & Zwart, 2007]

- γ_{FCFS} is *optimal*
- γ_{SRPT} is *pessimal*

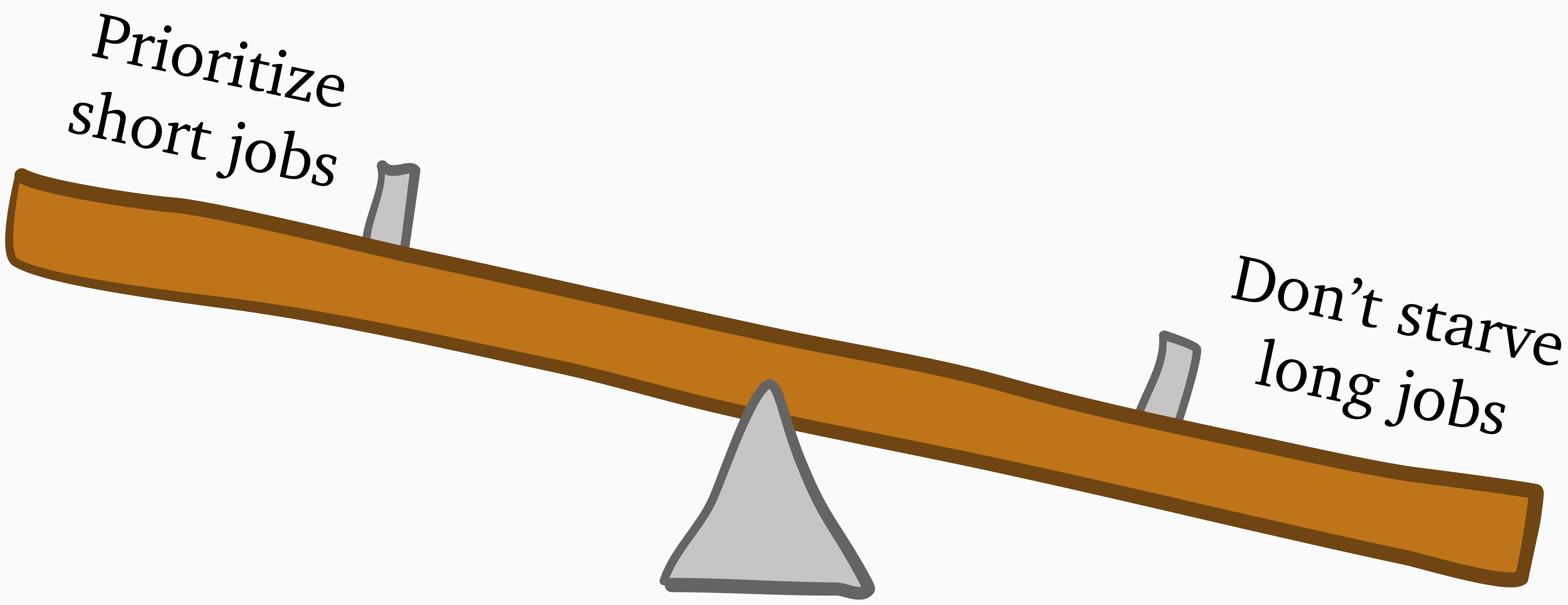
Tradeoff: priority vs. starvation



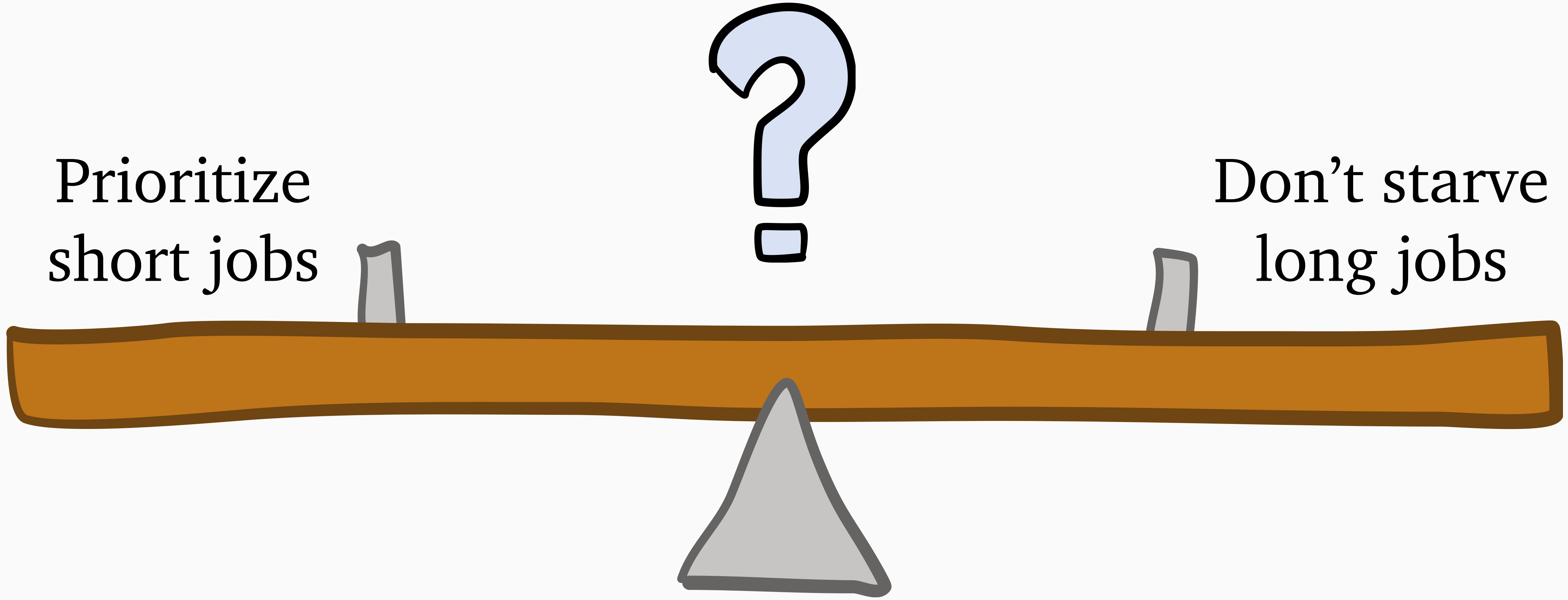
Tradeoff: priority vs. starvation



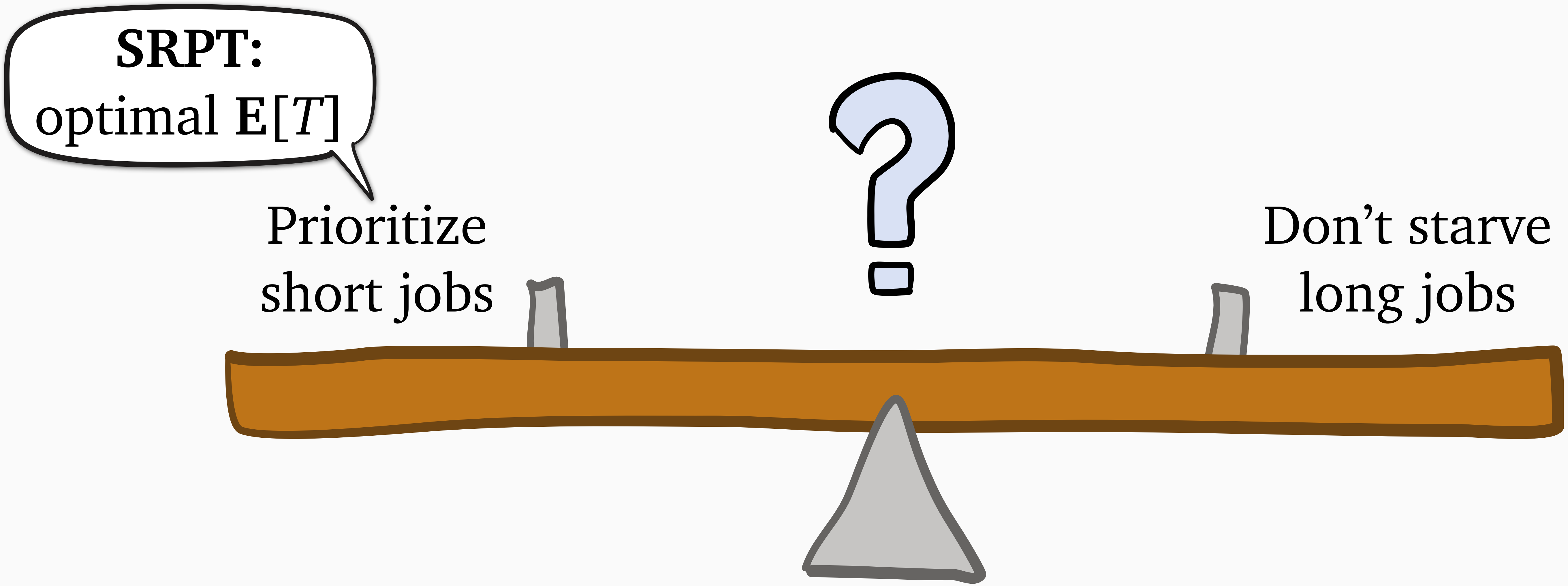
Tradeoff: priority vs. starvation



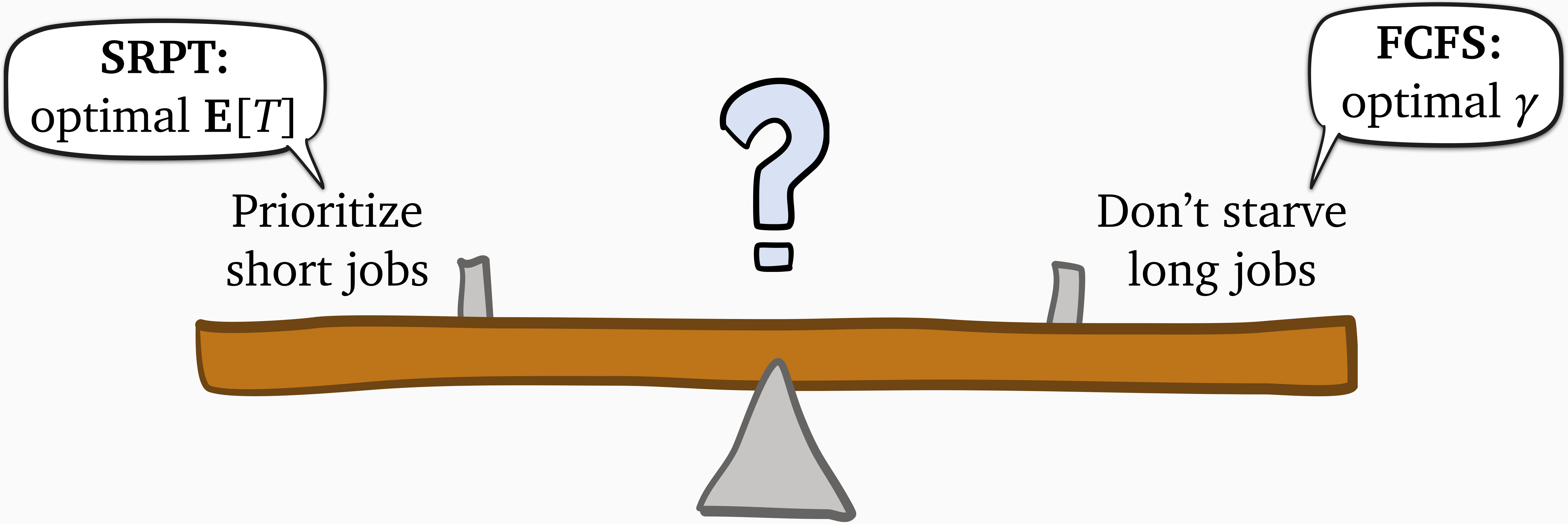
Tradeoff: priority vs. starvation



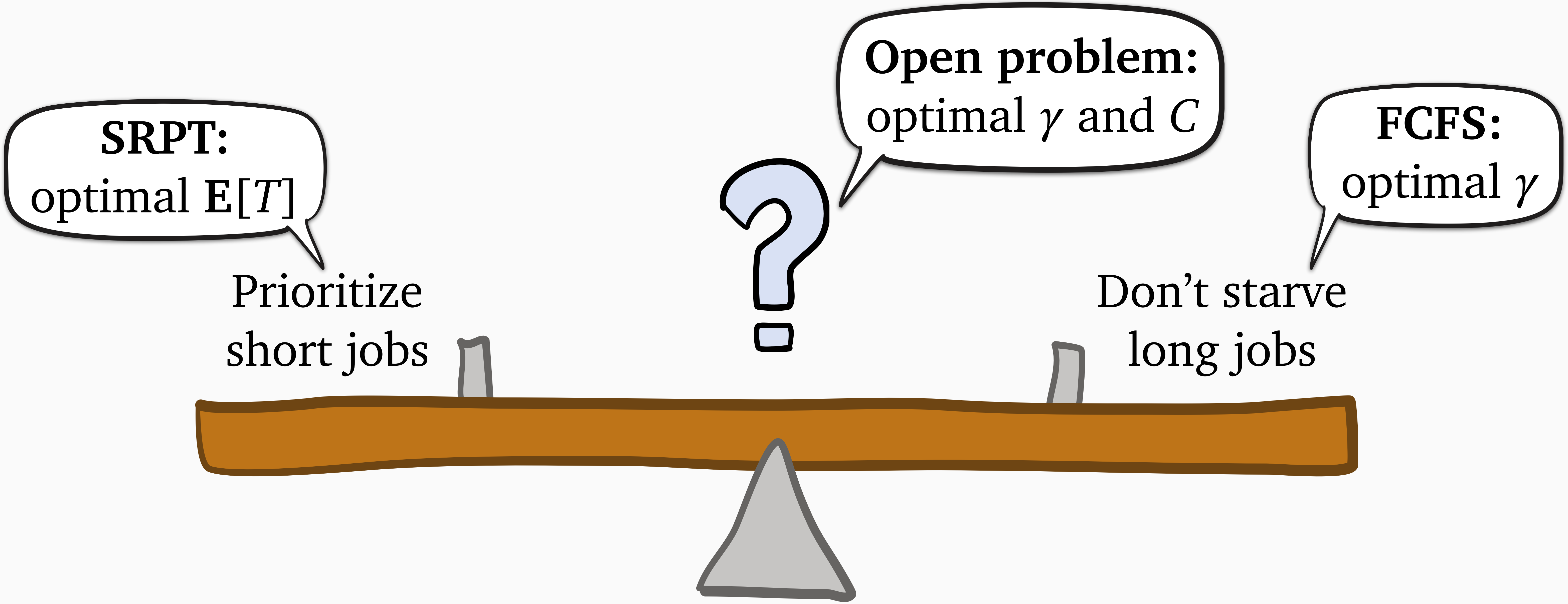
Tradeoff: priority vs. starvation



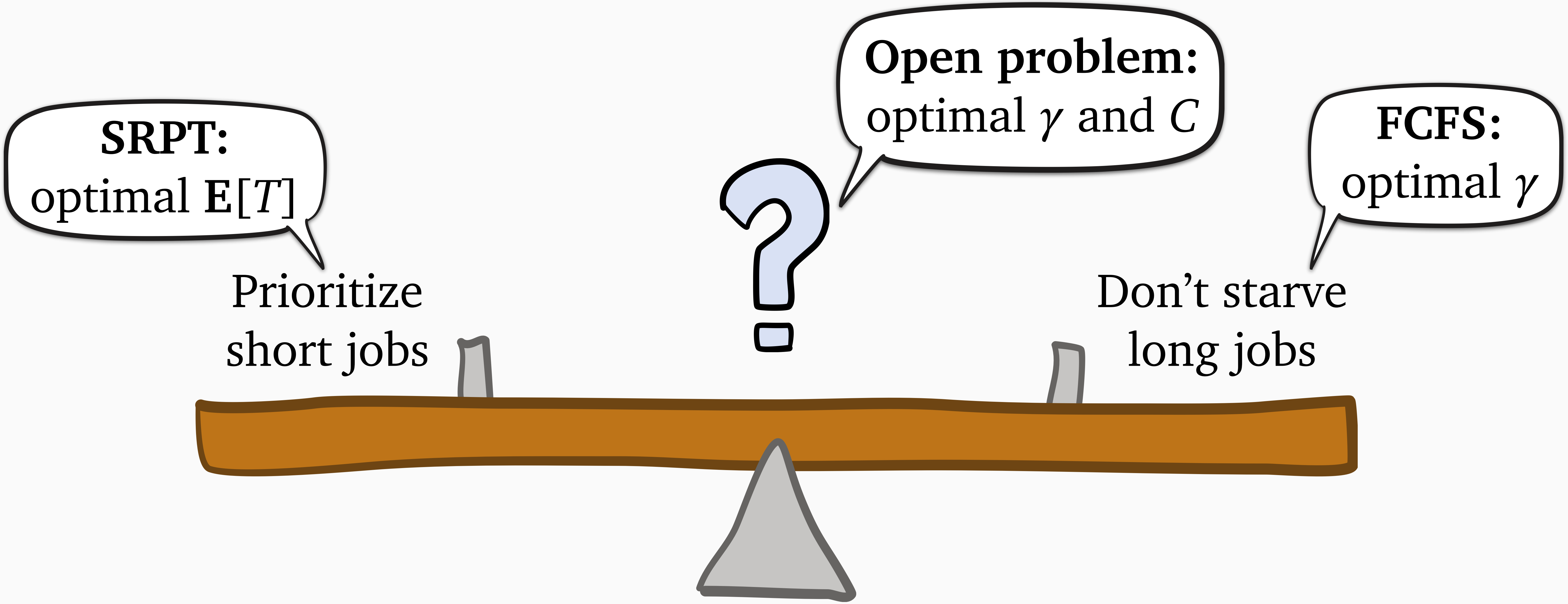
Tradeoff: priority vs. starvation



Tradeoff: priority vs. starvation



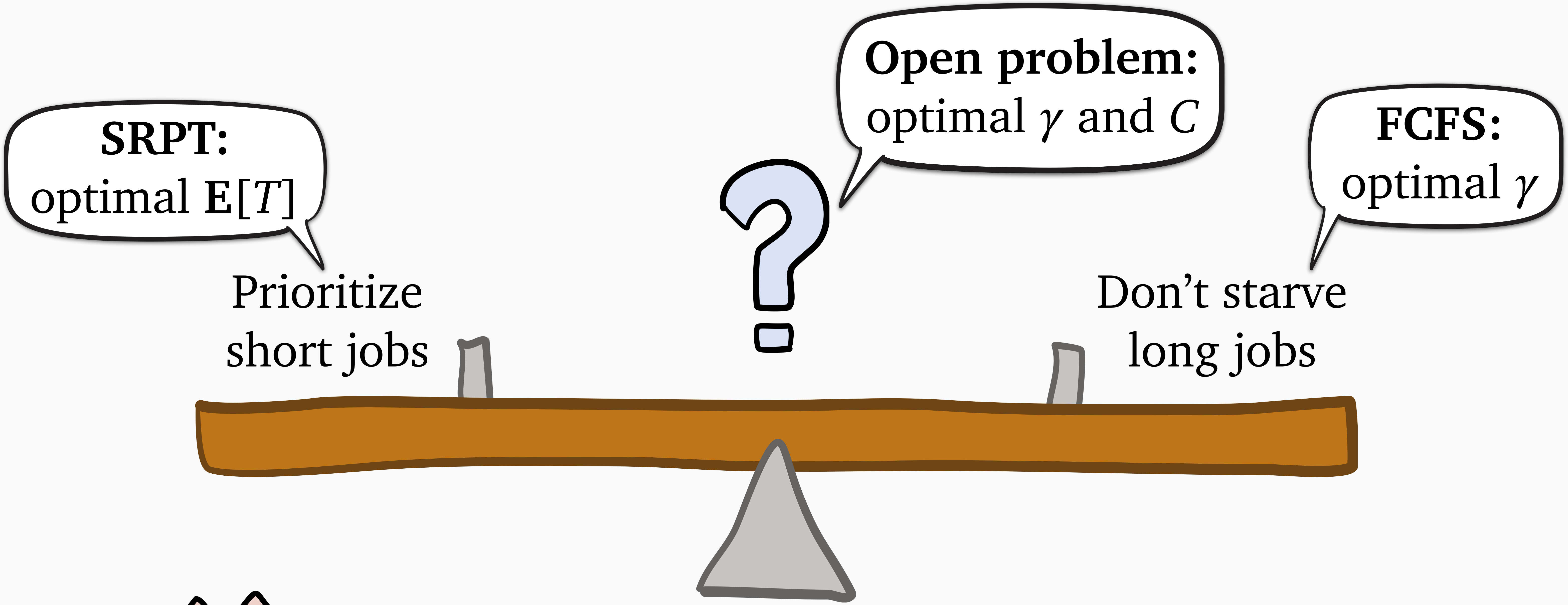
Tradeoff: priority vs. starvation



Conjecture: FCFS optimizes C, too

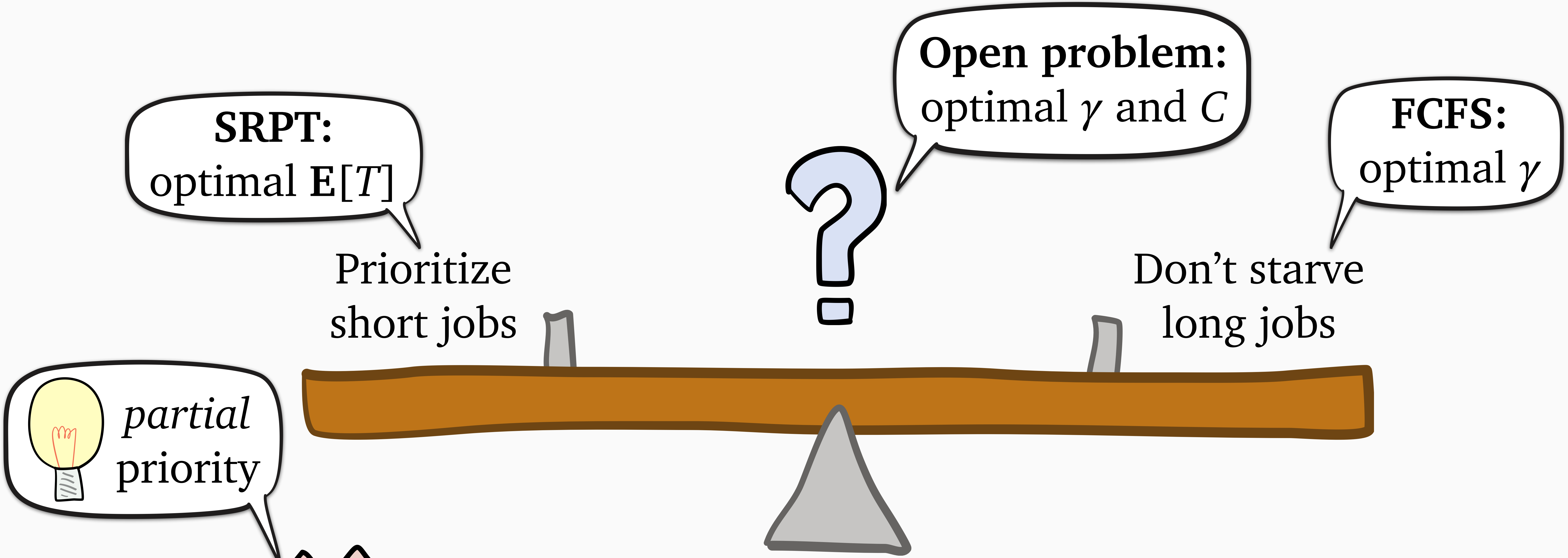
[Wierman & Zwart, 2012]

Tradeoff: priority vs. starvation



Conjecture: FCFS optimizes C, too
[Wierman & Zwart, 2012]

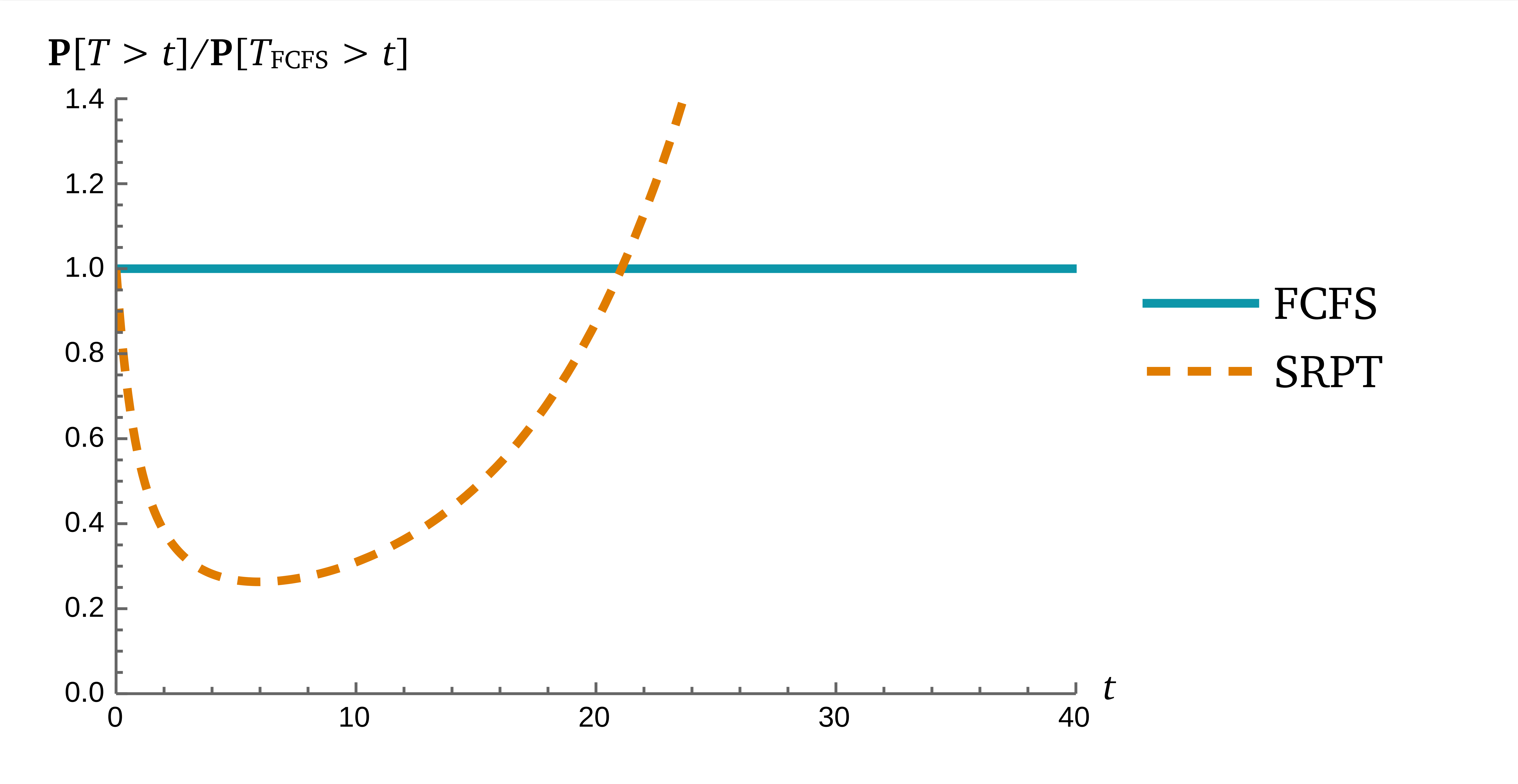
Tradeoff: priority vs. starvation



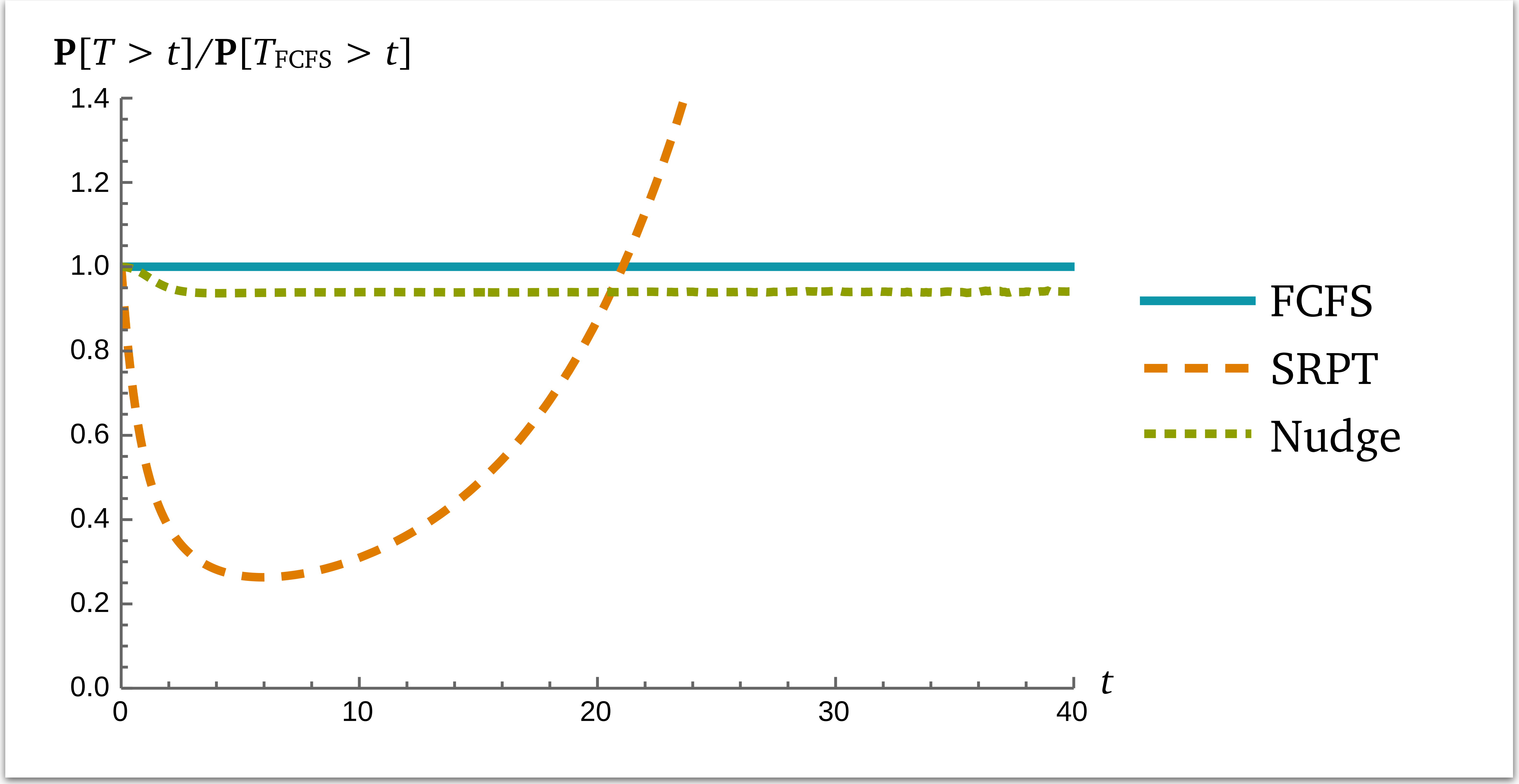
Conjecture: FCFS optimizes C, too

[Wierman & Zwart, 2012]

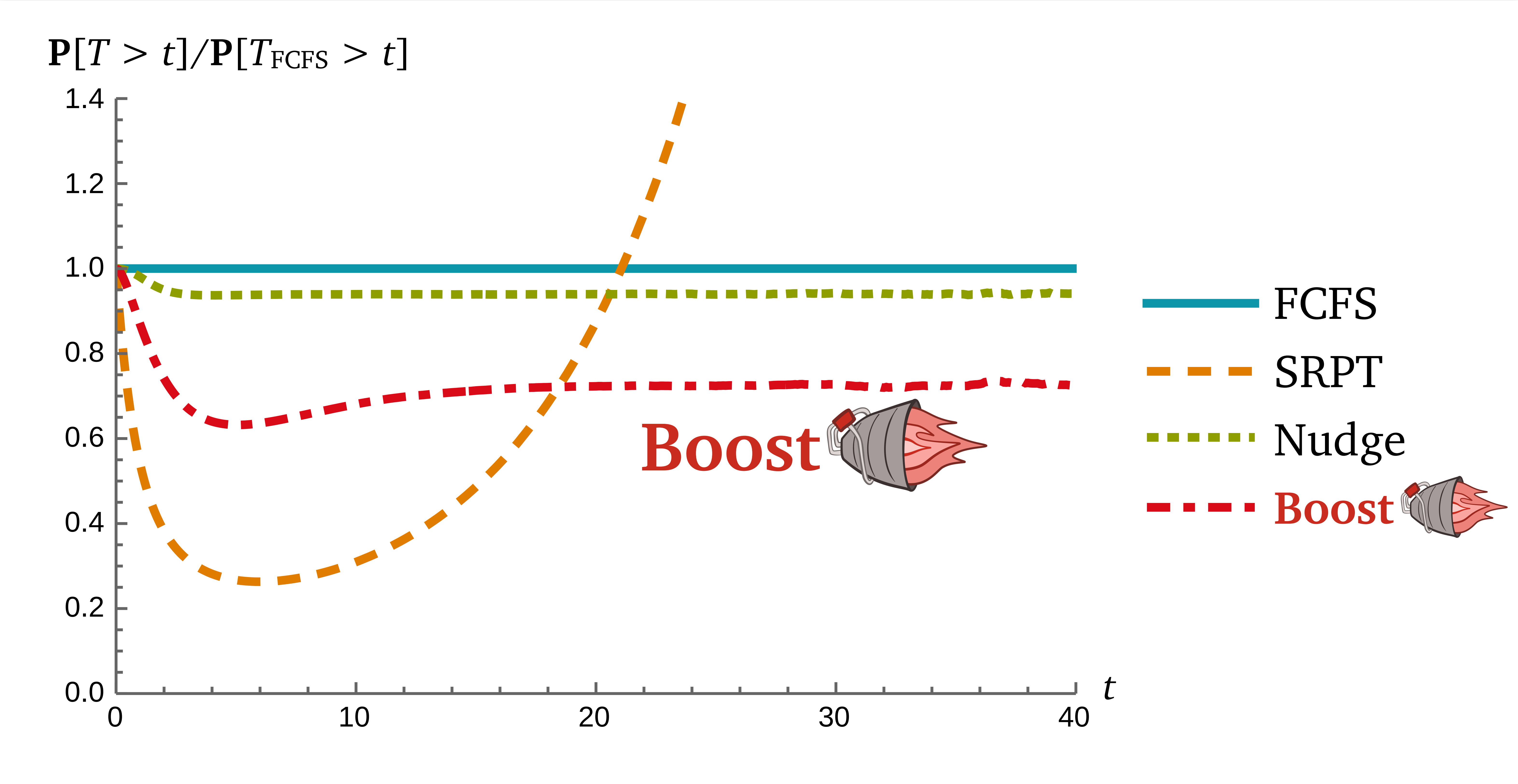
Optimizing the tail constant C



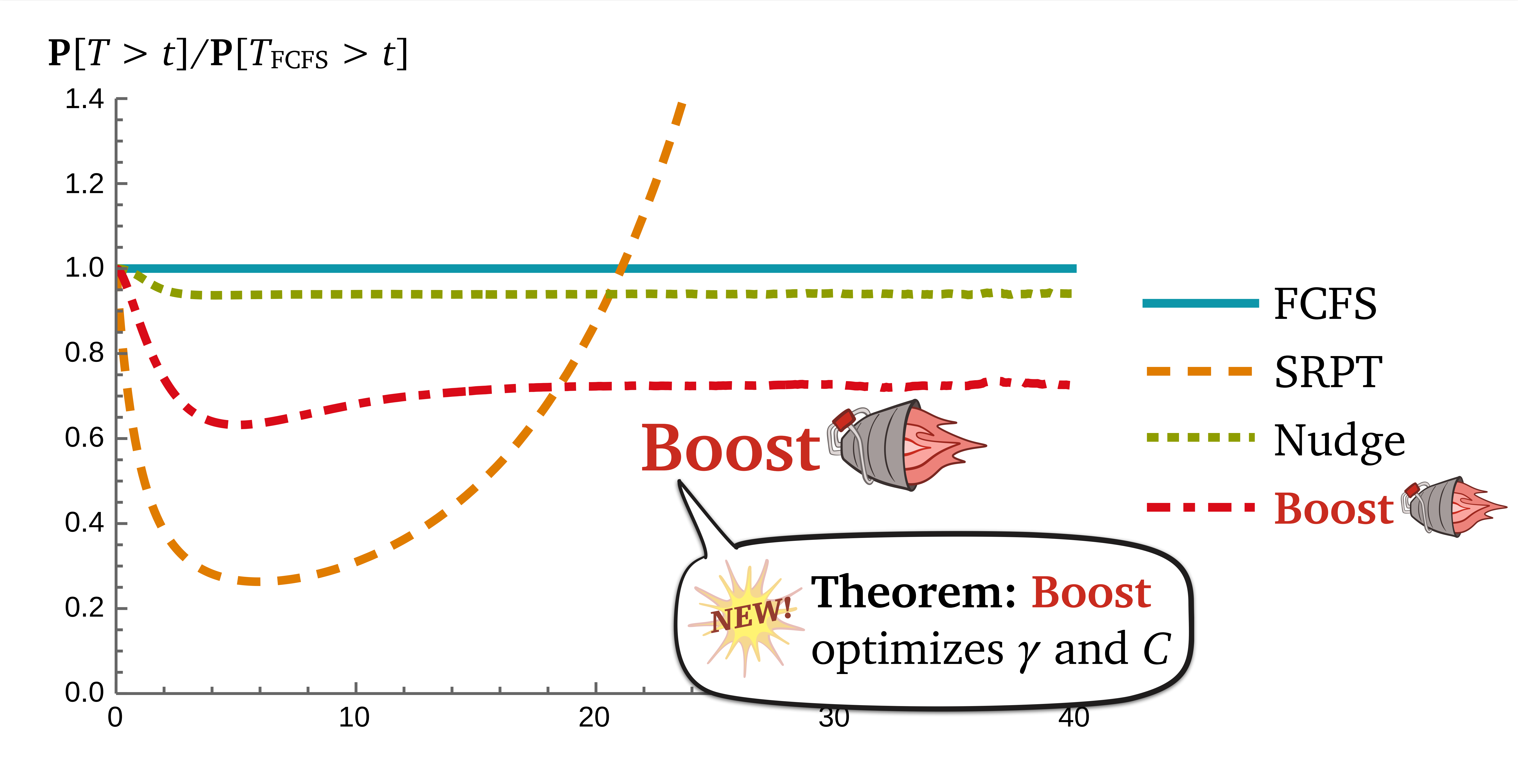
Optimizing the tail constant C



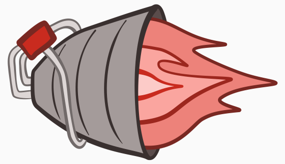
Optimizing the tail constant C

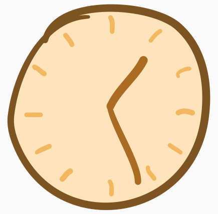


Optimizing the tail constant C



Our contributions:

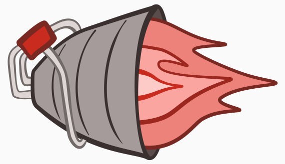
 Design the **Boost** scheduling policy

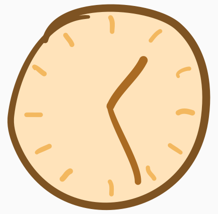
 Analyze **Boost**’s performance

 Prove **Boost** is *strongly tail-optimal* for light-tailed sizes

Our contributions:

actually a *family* of many policies

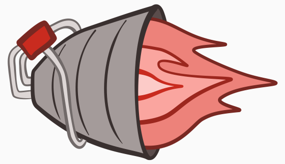
 Design the **Boost** scheduling policy

 Analyze **Boost**'s performance

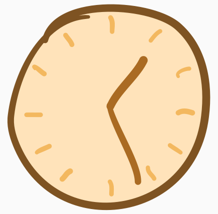
 Prove **Boost** is *strongly tail-optimal* for light-tailed sizes

Our contributions:

actually a *family* of many policies

 Design the **Boost** scheduling policy

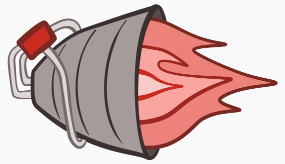
all instances

 Analyze **Boost**'s performance

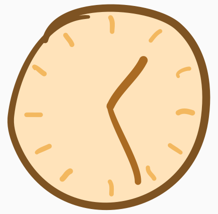
 Prove **Boost** is *strongly tail-optimal* for light-tailed sizes

Our contributions:

actually a *family* of many policies


Design the **Boost** scheduling policy

all instances

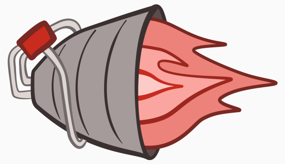

Analyze **Boost**'s performance

specific instance called **γ-Boost**

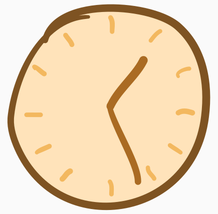

Prove **Boost** is *strongly tail-optimal* for light-tailed sizes

Our contributions:

actually a *family* of many policies

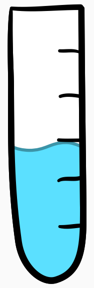
 Design the **Boost** scheduling policy

all instances

 Analyze **Boost**'s performance

specific instance called γ-**Boost**

 Prove **Boost** is *strongly tail-optimal* for light-tailed sizes

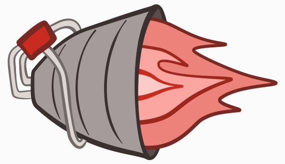


Known job sizes

Yu & Scully. *Strongly Tail-Optimal Scheduling in the Light-Tailed M/G/1.* SIGMETRICS 2024.

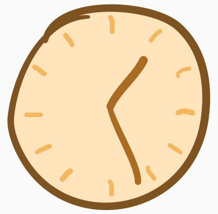
Our contributions:

actually a *family* of many policies



Design the **Boost** scheduling policy

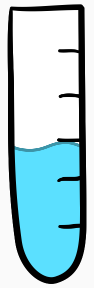
all instances



Analyze **Boost**'s performance

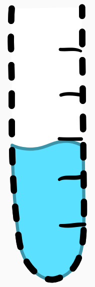
specific instance called **γ-Boost**

Prove **Boost** is *strongly tail-optimal* for light-tailed sizes



Known job sizes

Yu & Scully. *Strongly Tail-Optimal Scheduling in the Light-Tailed M/G/1*. SIGMETRICS 2024.

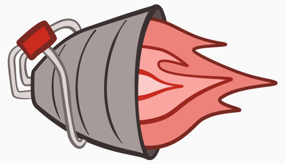


Unknown job sizes

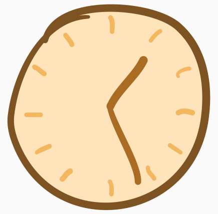
Harlev, Yu, & Scully. *A Gittins Policy for Optimizing Tail Latency*. SIGMETRICS 2025.

Our contributions:

actually a *family* of many policies

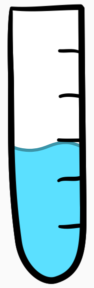

Design the **Boost** scheduling policy

all instances

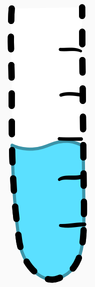

Analyze **Boost**'s performance

specific instance called **γ-Boost**

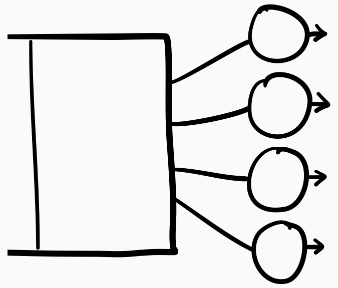

Prove **Boost** is *strongly tail-optimal* for light-tailed sizes


Known job sizes

Yu & Scully. *Strongly Tail-Optimal Scheduling in the Light-Tailed M/G/1*. SIGMETRICS 2024.


Unknown job sizes

Harlev, Yu, & Scully. *A Gittins Policy for Optimizing Tail Latency*. SIGMETRICS 2025.


Multiple servers

Yu, Harlev, Adakroy, & Scully. *A Tale of Two Traffics: Optimizing Tail Latency in the Light-Tailed M/G/k*. SIGMETRICS 2026.

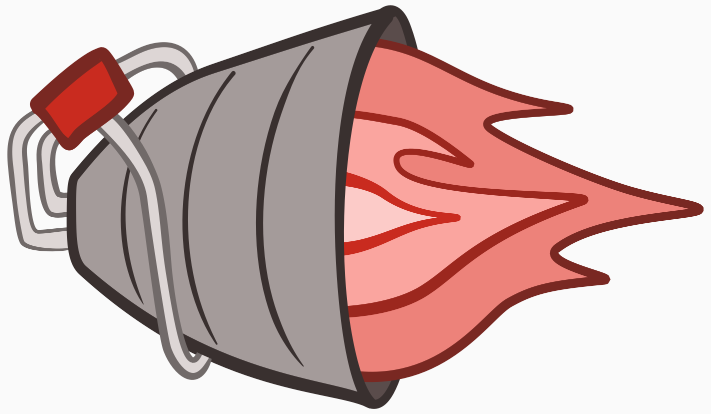
Boost

Why is achieving strong tail optimality hard?

How does the **Boost** policy family work?

How do we achieve strong tail optimality?

Boost



Why is achieving strong tail optimality hard?

How does the Boost policy family work?

How do we achieve strong tail optimality?

Boost

Why did it take so long to beat FCFS?

Why is achieving strong tail optimality hard?

How does the **Boost** policy family work?

How do we achieve strong tail optimality?

Heavy-tailed sizes

"S Pareto-ish" (regularly varying)

$$\mathbf{P}[S > s] \sim As^{-\alpha}$$

Light-tailed sizes

"S exponential-ish or lighter" (class I)

$$\mathbf{P}[S > s] \sim Ae^{-\alpha s}$$

Heavy-tailed sizes

"S Pareto-ish" (regularly varying)

$$\mathbf{P}[S > s] \sim A s^{-\alpha}$$

Light-tailed sizes

"S exponential-ish or lighter" (class I)

$$\mathbf{P}[S > s] \sim A e^{-\alpha s}$$

$$\Downarrow$$

$$\mathbf{P}[T_\pi > t] \sim C_\pi e^{-\gamma_\pi t}$$

Heavy-tailed sizes

"S Pareto-ish" (regularly varying)

$$\mathbf{P}[S > s] \sim A s^{-\alpha}$$

$$\Downarrow$$

$$\mathbf{P}[T_\pi > t] \sim C_\pi t^{-\gamma_\pi}$$

Light-tailed sizes

"S exponential-ish or lighter" (class I)

$$\mathbf{P}[S > s] \sim A e^{-\alpha s}$$

$$\Downarrow$$

$$\mathbf{P}[T_\pi > t] \sim C_\pi e^{-\gamma_\pi t}$$

Heavy-tailed sizes

"S Pareto-ish" (regularly varying)

$$\mathbf{P}[S > s] \sim A s^{-\alpha}$$

$$\mathbf{P}[T_\pi > t] \sim C_\pi t^{-\gamma_\pi}$$

Light-tailed sizes

"S exponential-ish or lighter" (class I)

$$\mathbf{P}[S > s] \sim A e^{-\alpha s}$$

$$\mathbf{P}[T_\pi > t] \sim C_\pi e^{-\gamma_\pi t}$$

$\gamma_\pi = \textit{decay rate}$ of π

$C_\pi = \textit{tail constant}$ of π

Heavy-tailed sizes

"S Pareto-ish" (regularly varying)

$$\mathbf{P}[S > s] \sim As^{-\alpha}$$

$$\mathbf{P}[T_\pi > t] \sim C_\pi t^{-\gamma_\pi}$$

Light-tailed sizes

"S exponential-ish or lighter" (class I)

$$\mathbf{P}[S > s] \sim Ae^{-\alpha s}$$

$$\mathbf{P}[T_\pi > t] \sim C_\pi e^{-\gamma_\pi t}$$

$\gamma_\pi = \textit{decay rate}$ of π

$C_\pi = \textit{tail constant}$ of π

Weak optimality:

maximize γ_π

Heavy-tailed sizes

"S Pareto-ish" (regularly varying)

$$\mathbf{P}[S > s] \sim A s^{-\alpha}$$

$$\Downarrow$$

$$\mathbf{P}[T_\pi > t] \sim C_\pi t^{-\gamma_\pi}$$

Light-tailed sizes

"S exponential-ish or lighter" (class I)

$$\mathbf{P}[S > s] \sim A e^{-\alpha s}$$

$$\Downarrow$$

$$\mathbf{P}[T_\pi > t] \sim C_\pi e^{-\gamma_\pi t}$$

$$\boxed{\begin{aligned} \gamma_\pi &= \textit{decay rate of } \pi \\ C_\pi &= \textit{tail constant of } \pi \end{aligned}}$$

Weak optimality:
maximize γ_π

Strong optimality:
maximize γ_π, minimize C_π

Heavy-tailed sizes

"S Pareto-ish" (regularly varying)

$$\mathbf{P}[S > s] \sim A s^{-\alpha}$$

$$\Downarrow$$

$$\mathbf{P}[T_\pi > t] \sim C_\pi t^{-\gamma_\pi}$$

(roughly)

Light-tailed sizes

"S exponential-ish or lighter" (class I)

$$\mathbf{P}[S > s] \sim A e^{-\alpha s}$$

$$\Downarrow$$

$$\mathbf{P}[T_\pi > t] \sim C_\pi e^{-\gamma_\pi t}$$

(roughly)

$$\boxed{\begin{aligned} \gamma_\pi &= \textit{decay rate of } \pi \\ C_\pi &= \textit{tail constant of } \pi \end{aligned}}$$

Weak optimality:
maximize γ_π

Strong optimality:
maximize γ_π, minimize C_π

(roughly)

What causes large response times?

	Heavy-tailed sizes	Light-tailed sizes

What causes large response times?

	Heavy-tailed sizes	Light-tailed sizes
SRPT, LAS, etc. (least attained service)		
FCFS		

What causes large response times?

	Heavy-tailed sizes	Light-tailed sizes
SRPT, LAS, etc. (least attained service)	**optimal** $\gamma = \alpha$	
FCFS	**pessimal** $\gamma = \alpha - 1$	

What causes large response times?

	Heavy-tailed sizes	Light-tailed sizes
SRPT, LAS, etc. (least attained service)	optimal $\gamma = \alpha$	pessimal γ
FCFS	pessimal $\gamma = \alpha - 1$	optimal γ

What causes large response times?

	Heavy-tailed sizes	Light-tailed sizes
SRPT, LAS, etc. (least attained service)	optimal $\gamma = \alpha$	pessimal γ
FCFS	pessimal $\gamma = \alpha - 1$	optimal γ
Main cause of large T?		

What causes large response times?

	Heavy-tailed sizes	Light-tailed sizes
SRPT, LAS, etc. (least attained service)	optimal $\gamma = \alpha$	pessimal γ
FCFS	pessimal $\gamma = \alpha - 1$	optimal γ
Main cause of large T?	**“Catastrophe”** one giant job	

What causes large response times?

	Heavy-tailed sizes	Light-tailed sizes
SRPT, LAS, etc. (least attained service)	optimal $\gamma = \alpha$	pessimal γ
FCFS	pessimal $\gamma = \alpha - 1$ I'm stuck behind the giant job	optimal γ
Main cause of large T?	**"Catastrophe"** one giant job	

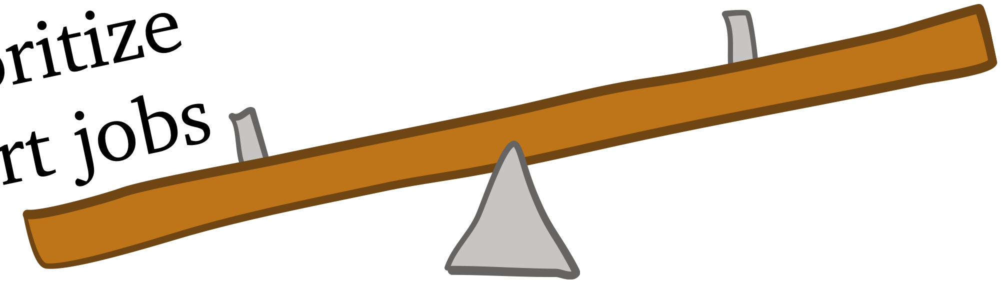
What causes large response times?

	Heavy-tailed sizes	Light-tailed sizes
SRPT, LAS, etc. (least attained service)	optimal $\gamma = \alpha$ I'm the giant job	pessimal γ
FCFS	pessimal $\gamma = \alpha - 1$ I'm stuck behind the giant job	optimal γ
Main cause of large T?	**"Catastrophe"** one giant job	

What causes large response times?

	Heavy-tailed sizes	Light-tailed sizes
SRPT, LAS, etc. (least attained service)	optimal $\gamma = \alpha$ I'm the giant job	pessimal γ
FCFS	pessimal $\gamma = \alpha$ — I'm stuck behind the giant job	
Main cause of large T?	**"Catastrophe"** one giant job	

Prioritize short jobs



[Scully & van Kreveld, 2025]

What causes large response times?

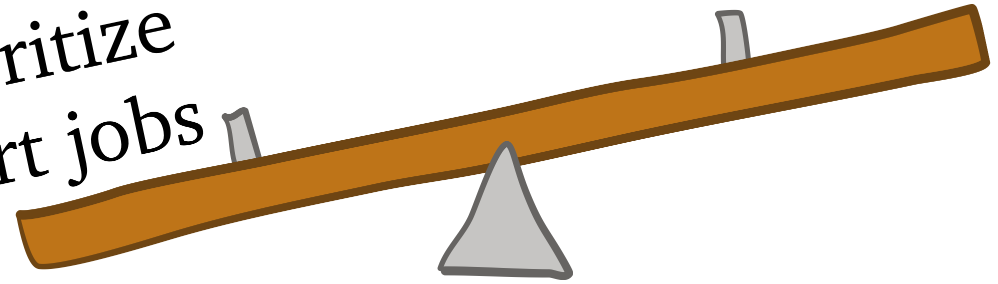
	Heavy-tailed sizes	Light-tailed sizes
SRPT, LAS, etc. (least attained service)	optimal $\gamma = \alpha$	pessimal γ
FCFS	pessimal $\gamma = \alpha$	
Main cause of large T?	**“Catastrophe”** one giant job	

also optimal C for heavy-tailed sizes

I’m the giant job

I’m stuck behind the giant job

Prioritize short jobs



[Scully & van Kreveld, 2025]

What causes large response times?

	Heavy-tailed sizes	Light-tailed sizes
SRPT, LAS, etc. (least attained service)	optimal $\gamma = \alpha$ I'm the giant job	pessimal γ
FCFS	pessimal $\gamma = \alpha - 1$ I'm stuck behind the giant job	optimal γ
Main cause of large T?	**“Catastrophe”** one giant job	

What causes large response times?

	Heavy-tailed sizes	Light-tailed sizes
SRPT, LAS, etc. (least attained service)	optimal $\gamma = \alpha$ I'm the giant job	pessimal γ
FCFS	pessimal $\gamma = \alpha - 1$ I'm stuck behind the giant job	optimal γ
? Main cause of large T?	**"Catastrophe"** one giant job	**"Conspiracy"** lots of biggish jobs

What causes large response times?

	Heavy-tailed sizes	Light-tailed sizes
SRPT, LAS, etc. (least attained service)	optimal $\gamma = \alpha$ I'm the giant job	pessimal γ
FCFS	pessimal $\gamma = \alpha - 1$ I'm stuck behind the giant job	optimal γ I see lots of *work* when I arrive
? Main cause of large T?	**"Catastrophe"** one giant job	**"Conspiracy"** lots of biggish jobs

What causes large response times?

	Heavy-tailed sizes	Light-tailed sizes
SRPT, LAS, etc. (least attained service)	optimal $\gamma = \alpha$ I'm the giant job	pessimal γ I'm a very big job, lots of smaller jobs are passing me
FCFS	pessimal $\gamma = \alpha - 1$ I'm stuck behind the giant job	optimal γ I see lots of *work* when I arrive
Main cause of large T?	**"Catastrophe"** one giant job	**"Conspiracy"** lots of biggish jobs

What causes large response times?

	Heavy-tailed sizes	Light-tailed sizes
SRPT, LAS, etc. (least attained service)	optimal $\gamma = \alpha$	pessimal γ
FCFS	pessimal $\gamma = \alpha - 1$	optimal γ
Main cause of large T?	**“Catastrophe”** one giant job	**“Conspiracy”** lots of biggish jobs

What causes large response times?

	Heavy-tailed sizes	Light-tailed sizes
SRPT, LAS, etc. (least attained service)	optimal $\gamma = \alpha$	pessimal γ
FCFS	pessimal $\gamma = \alpha - 1$	optimal γ
SRPT or LAS with just two buckets		
Main cause of large T?	**“Catastrophe”** one giant job	**“Conspiracy”** lots of biggish jobs

What causes large response times?

	Heavy-tailed sizes	**Light-tailed sizes**
SRPT, LAS, etc. (least attained service)	**optimal** $\gamma = \alpha$	**pessimal** γ
FCFS	**pessimal** $\gamma = \alpha - 1$	**optimal** γ
SRPT or LAS with just two buckets	**pessimal** $\gamma = \alpha - 1$	
 Main cause of large T?	**“Catastrophe”** one giant job	**“Conspiracy”** lots of biggish jobs

What causes large response times?

	Heavy-tailed sizes	Light-tailed sizes
SRPT, LAS, etc. (least attained service)	optimal $\gamma = \alpha$	pessimal γ
FCFS	pessimal $\gamma = \alpha - 1$	optimal γ
SRPT or LAS with just two buckets	pessimal $\gamma = \alpha - 1$	intermediate γ
Main cause of large T?	**"Catastrophe"** one giant job	**"Conspiracy"** lots of biggish jobs

What causes large response times?

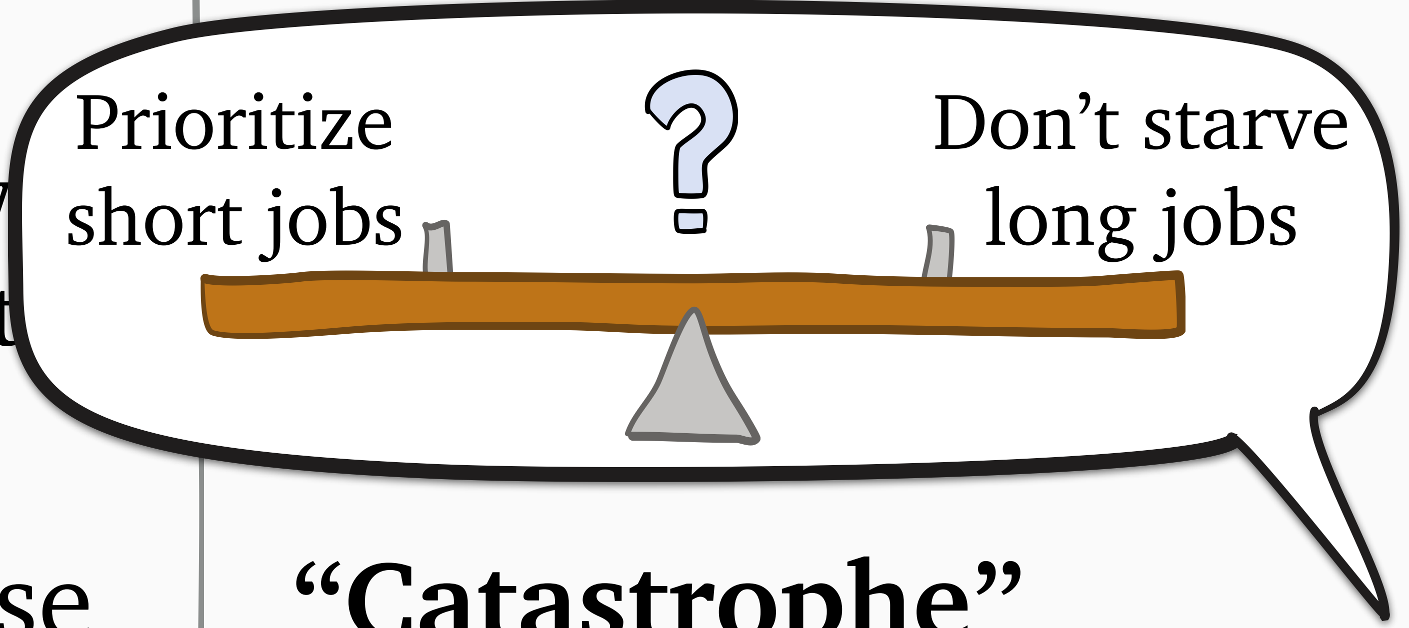
	Heavy-tailed sizes	Light-tailed sizes
SRPT, LAS, etc. (least attained service)	optimal $\gamma = \alpha$	pessimal γ
FCFS	pessimal $\gamma = \alpha - 1$	optimal γ
SRPT or LAS with just two buckets	pessimal $\gamma = \alpha - 1$	intermediate γ
Main cause of large T?	**“Catastrophe”** one giant job	**“Conspiracy”** lots of biggish jobs

I’m in bucket 2,
lots of bucket 1 jobs
are passing me

What causes large response times?

	Heavy-tailed sizes	Light-tailed sizes
SRPT, LAS, etc. (least attained service)	optimal $\gamma = \alpha$	pessimal γ
FCFS	pessimal $\gamma = \alpha - 1$	optimal γ
SRPT or LAS w just two bucket		intermediate γ
Main cause of large T?	**“Catastrophe”** one giant job	**“Conspiracy”** lots of biggish jobs

I’m in bucket 2,
lots of bucket 1 jobs
are passing me



What causes large response times?

Heavy-tailed sizes | Light-tailed sizes

SRPT, LAS, etc.
(least attained service

FCFS

SRPT or LAS w
just two bucket

⚠ Takeaway:
for **light-tailed sizes**,
avoid strict priorities

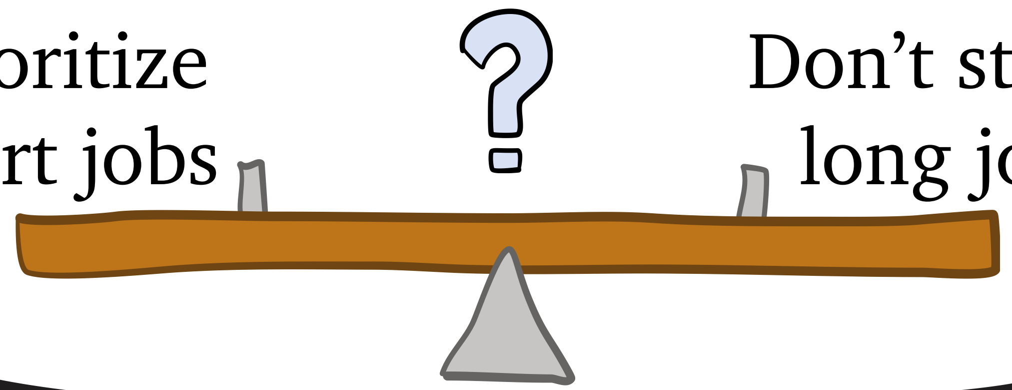
...essimal γ

...ptimal γ

intermediate γ

I'm in bucket 2,
lots of bucket 1 jobs
are passing me

Prioritize
short jobs



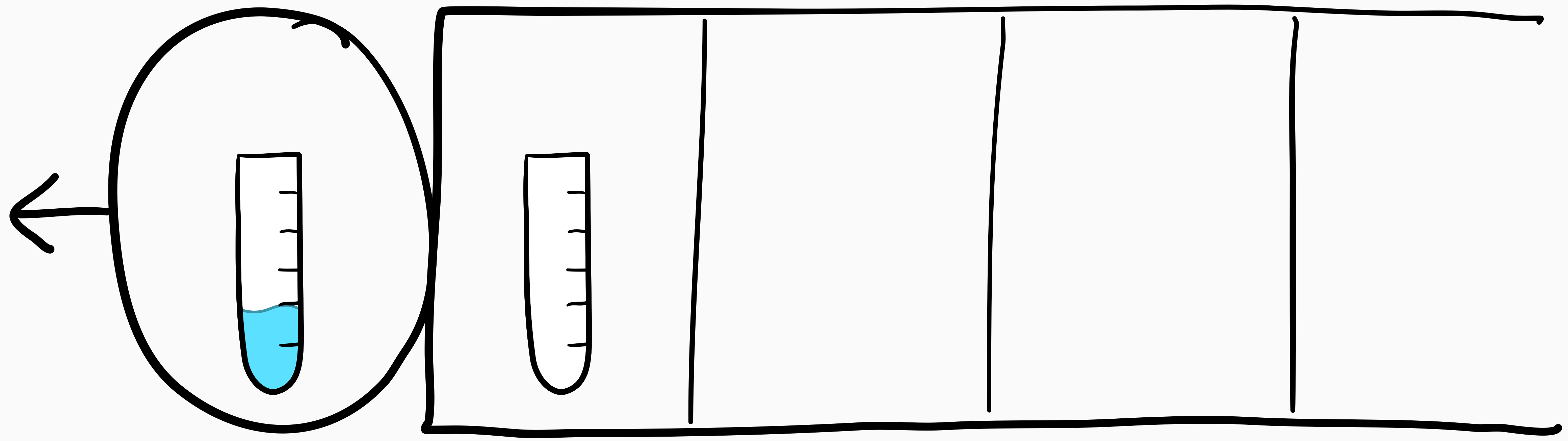
Don't starve
long jobs

Main cause of large T?

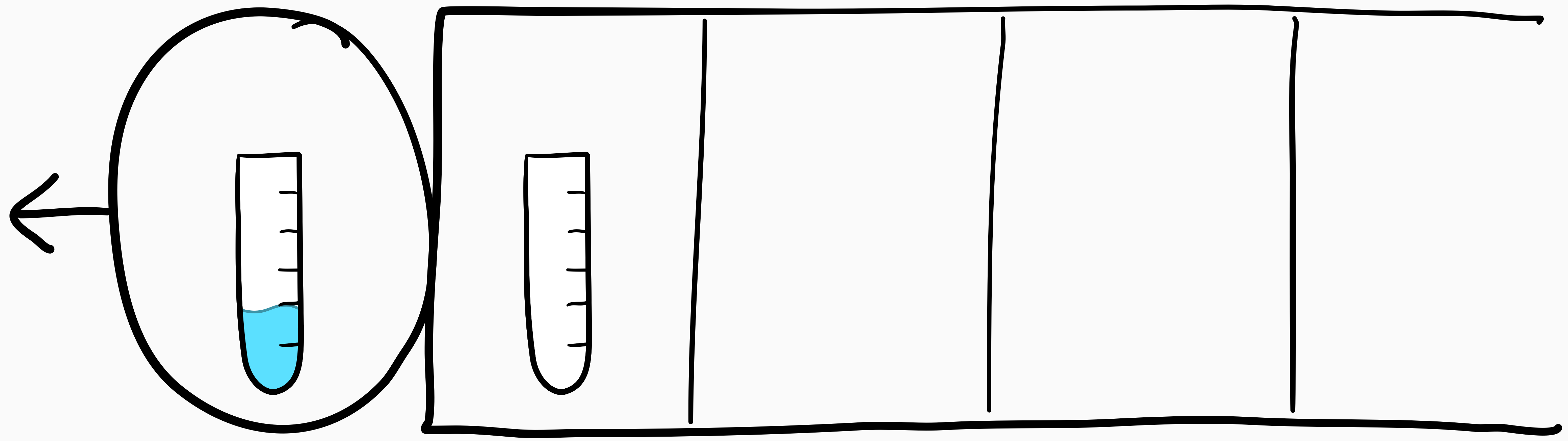
"Catastrophe"
one giant job

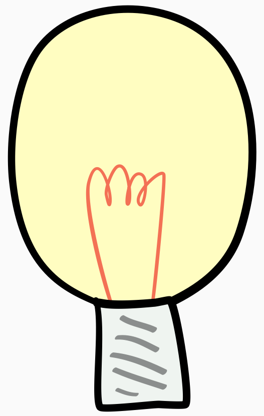
"Conspiracy"
lots of biggish jobs

Can we beat FCFS?



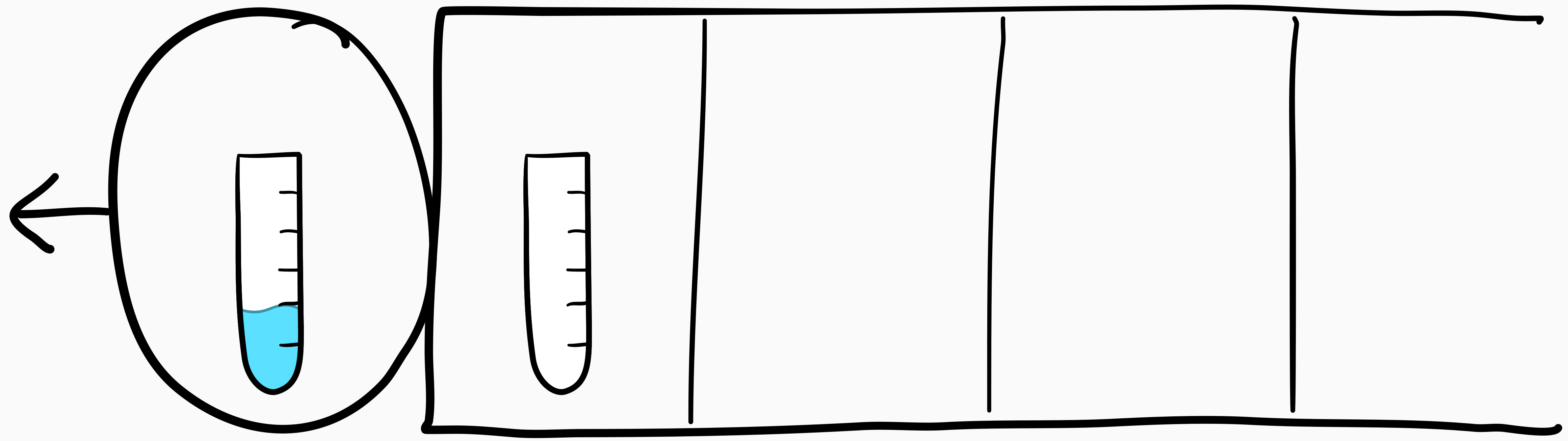
Can we beat FCFS?

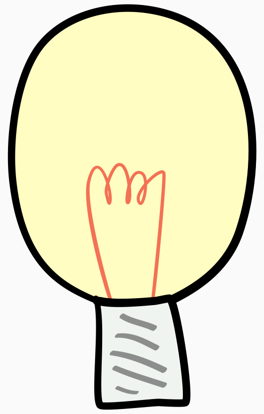




Nudge [Grosof et al., 2021]

Can we beat FCFS?

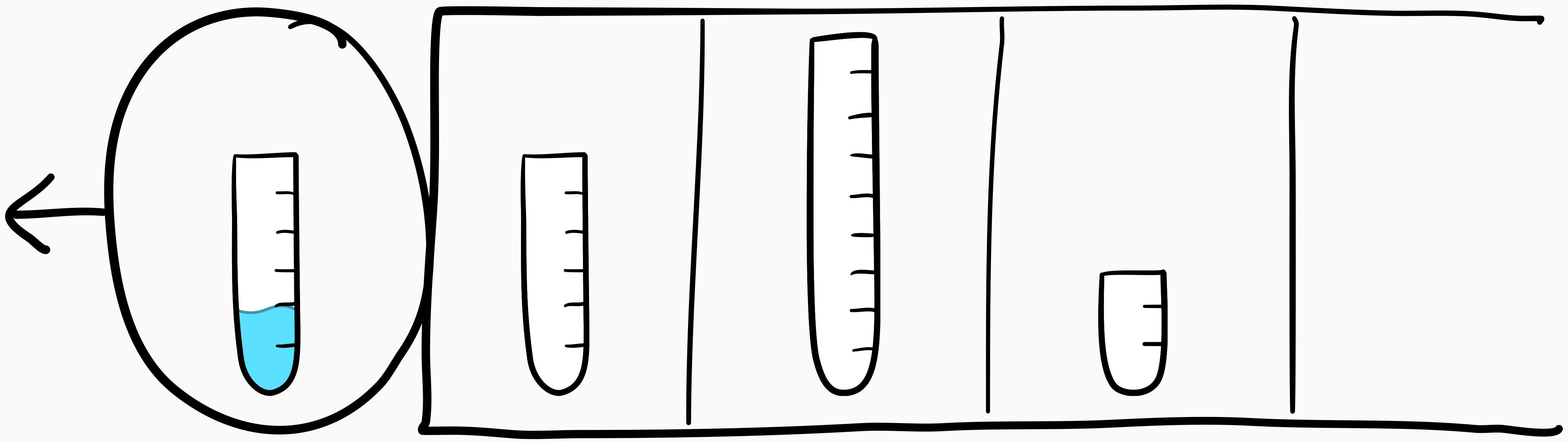


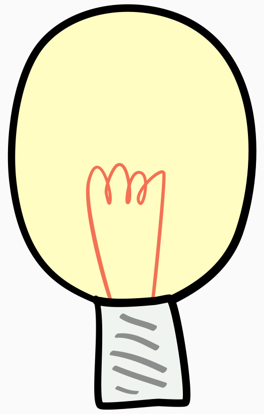


Nudge [Grosof et al., 2021]

- small job can pass one large job

Can we beat FCFS?

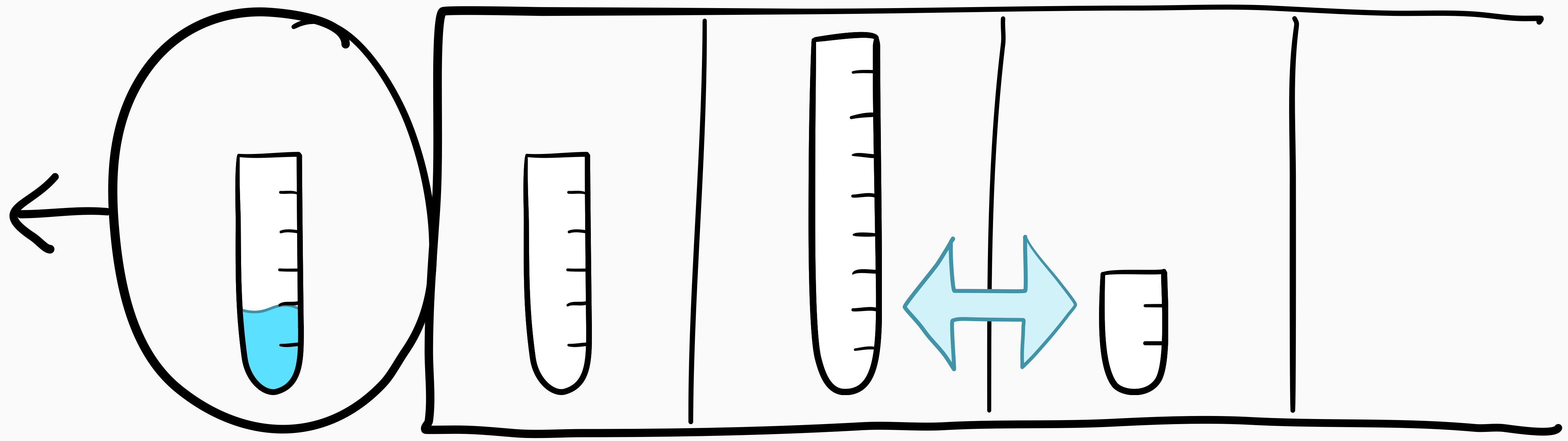


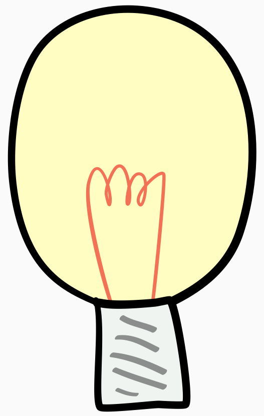


Nudge [Grosof et al., 2021]

- small job can pass one large job

Can we beat FCFS?

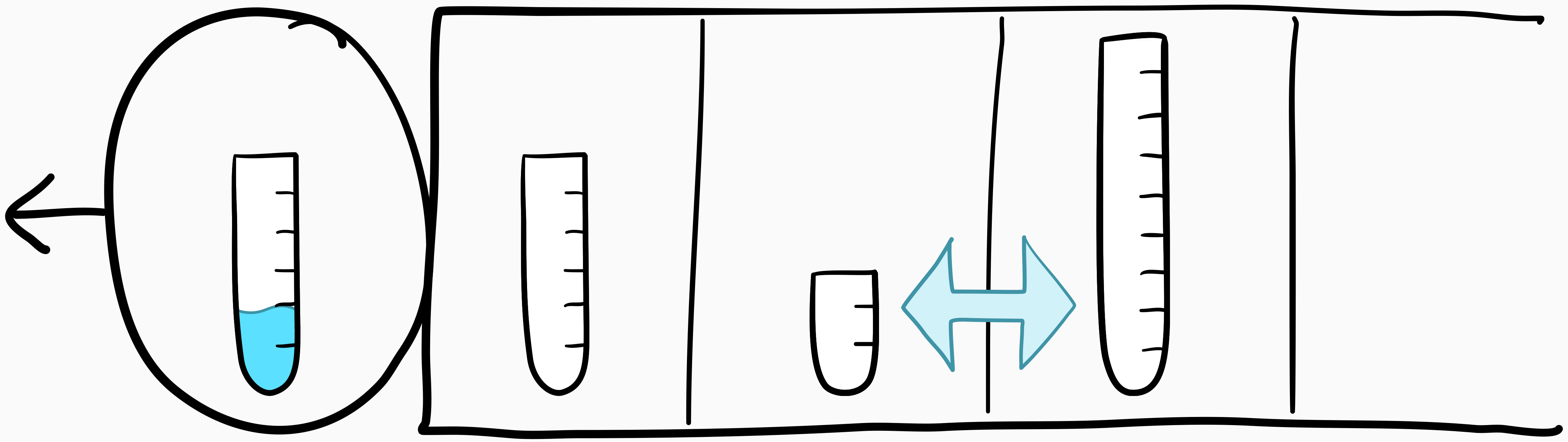


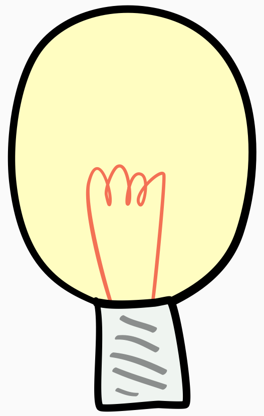


Nudge [Grosof et al., 2021]

- small job can pass one large job

Can we beat FCFS?

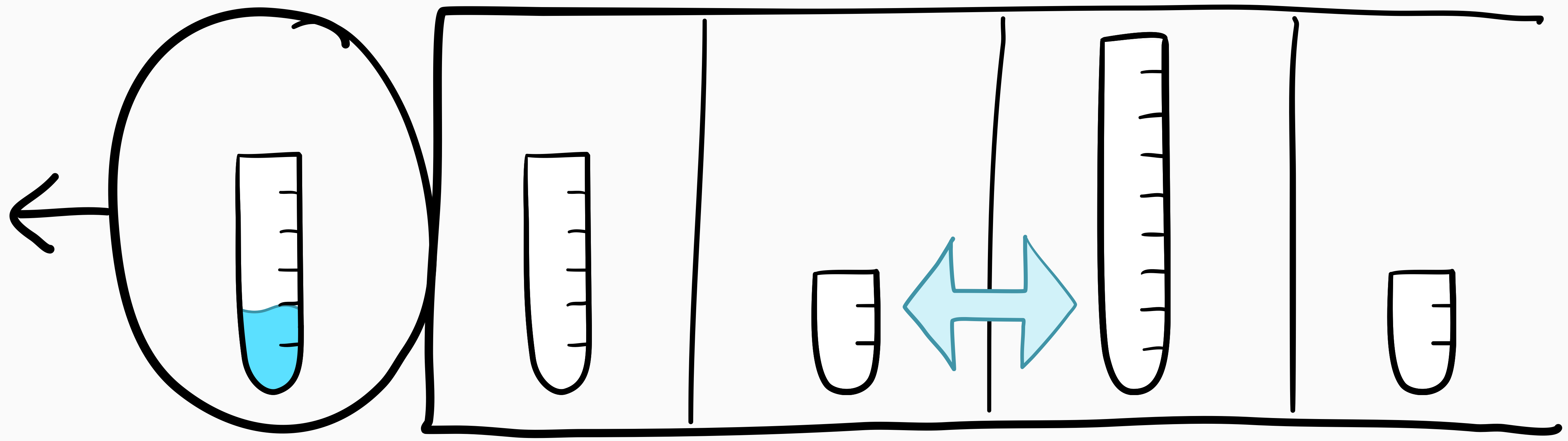


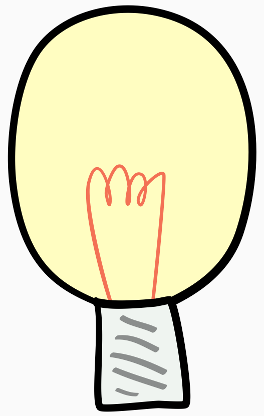


Nudge [Grosof et al., 2021]

- small job can pass one large job

Can we beat FCFS?

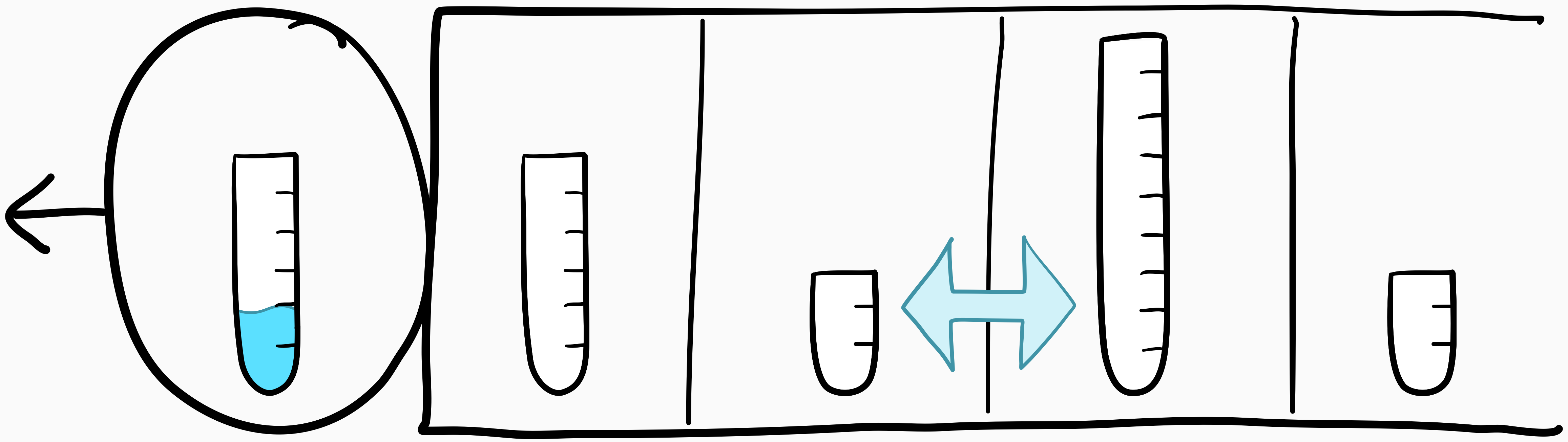


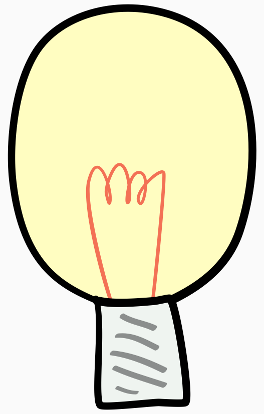


Nudge [Grosof et al., 2021]

- small job can pass one large job

Can we beat FCFS?

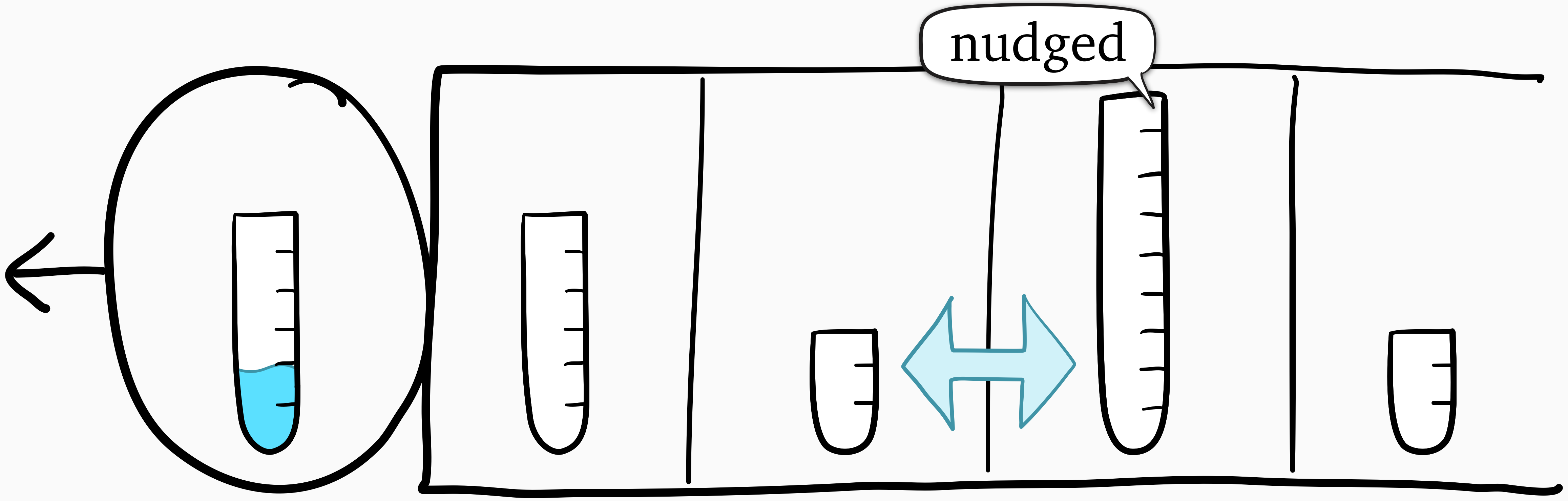


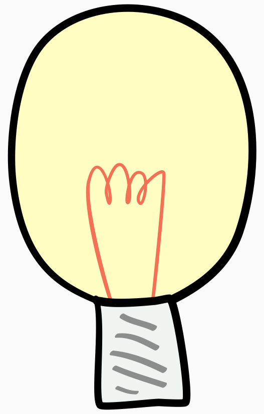


Nudge [Grosof et al., 2021]

- small job can pass one large job
- large job can't be passed twice

Can we beat FCFS?

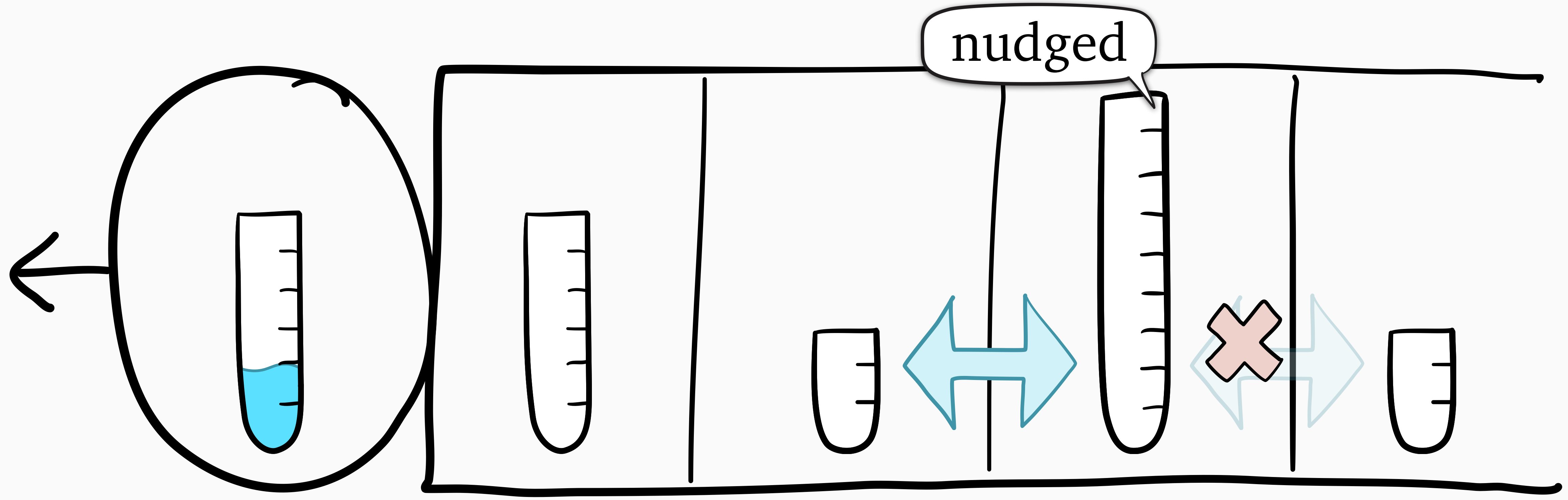


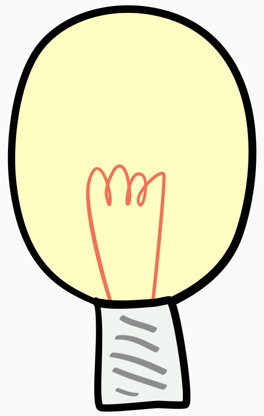


Nudge [Grosof et al., 2021]

- small job can pass one large job
- large job can't be passed twice

Can we beat FCFS?

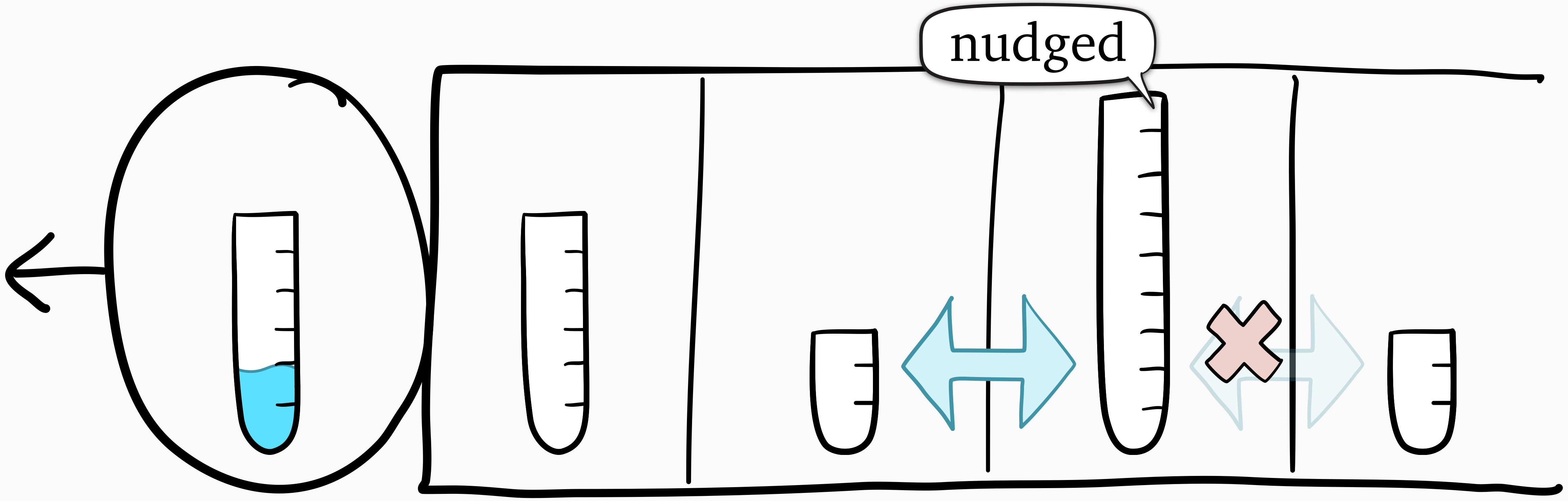


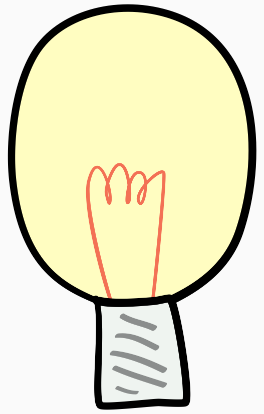


Nudge [Grosof et al., 2021]

- small job can pass one large job
- large job can't be passed twice

Can we beat FCFS?





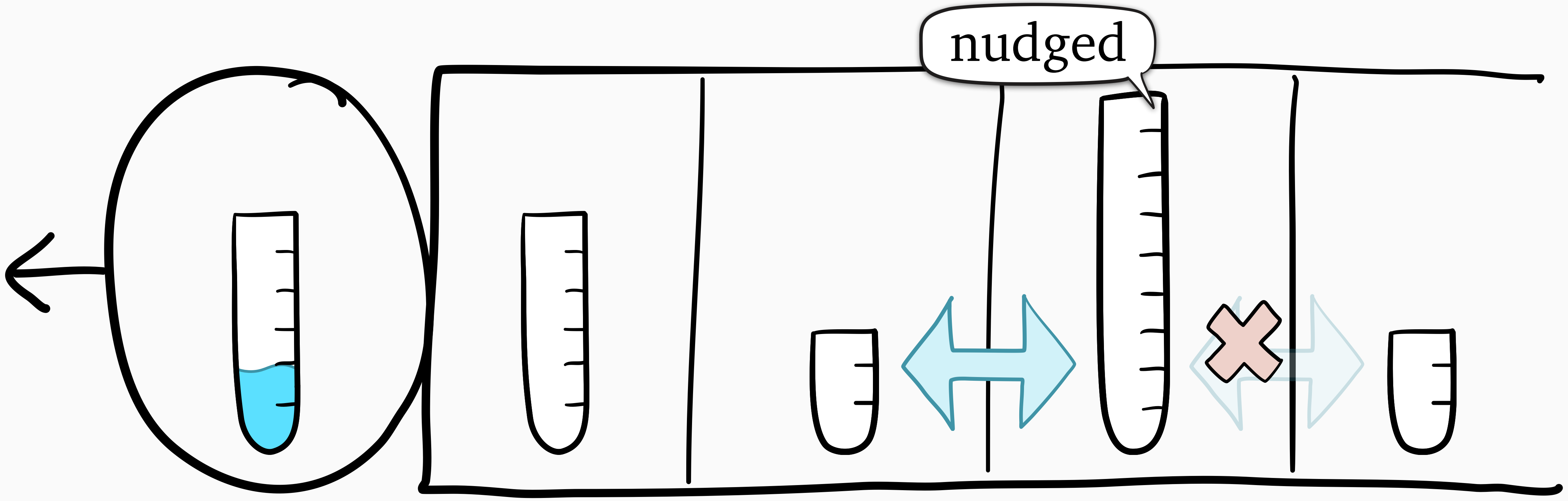
Nudge [Grosof et al., 2021]

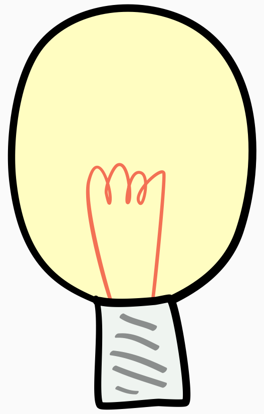
- small job can pass one large job
- large job can't be passed twice

Theorem:

$C_{\text{Nudge}} < C_{\text{FCFS}}$

Can we beat FCFS?





Nudge [Grosof et al., 2021]

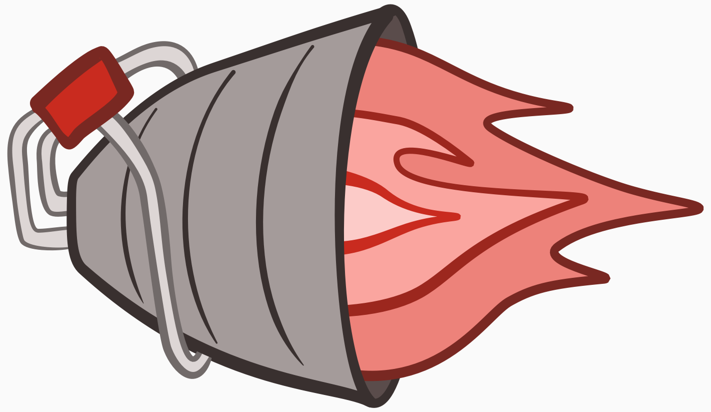
- small job can pass one large job
- large job can't be passed twice

Theorem:

$C_{\text{Nudge}} < C_{\text{FCFS}}$

More complex variants get even lower C

[Van Houdt, 2022; Charlet & Van Houdt, 2024]

Boost 

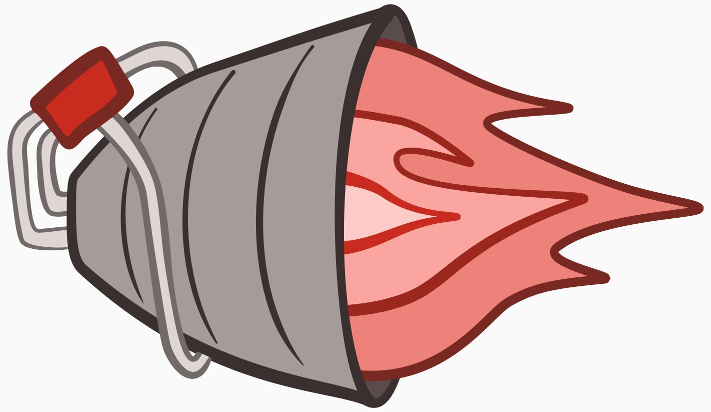
Why did it take so long to beat FCFS?

Why is achieving strong tail optimality hard?

How does the **Boost** policy family work?

How do we achieve strong tail optimality?

Boost



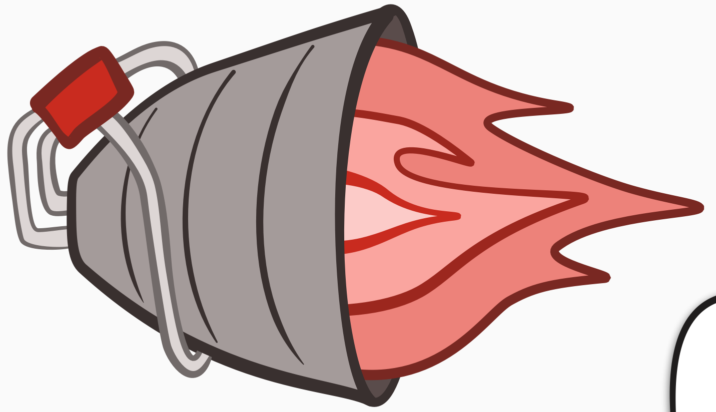
Why did it take so long to beat FCFS?

Why is achieving strong tail optimality hard?

How does the **Boost** policy family work?

How do we achieve strong tail optimality?

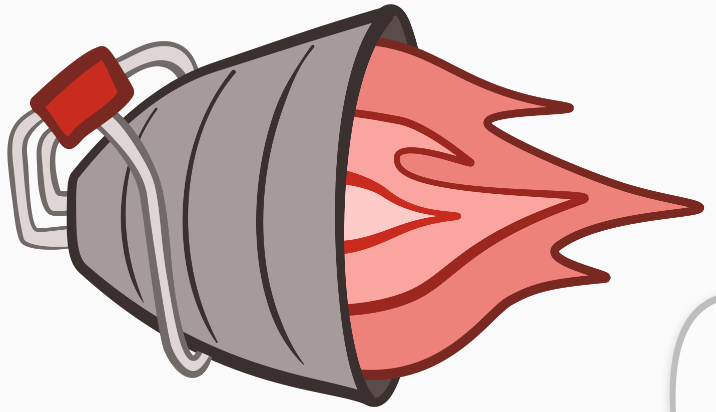
Boost



Need to avoid strict priorities

- Why did it take so long to beat FCFS?
- Why is achieving strong tail optimality hard?
- How does the **Boost** policy family work?
- How do we achieve strong tail optimality?

Boost

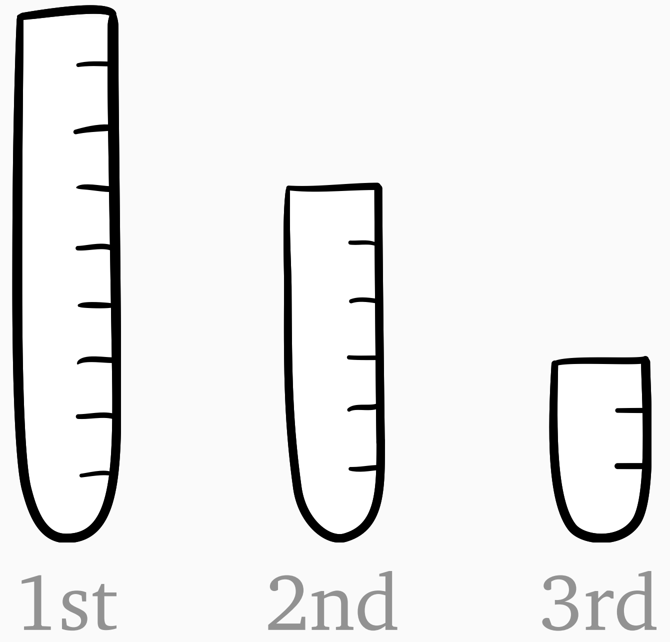


Need to avoid strict priorities

- Why did it take so long to beat FCFS?
- Why is achieving strong tail optimality hard?
- How does the **Boost** policy family work?
- How do we achieve strong tail optimality?

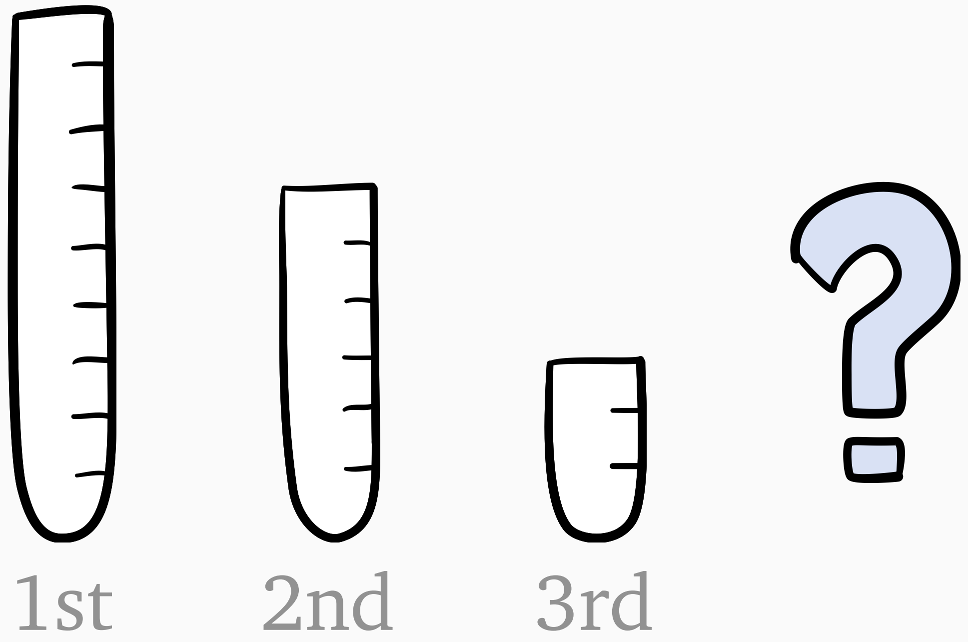
Can we beat Nudge?

Can we beat Nudge?



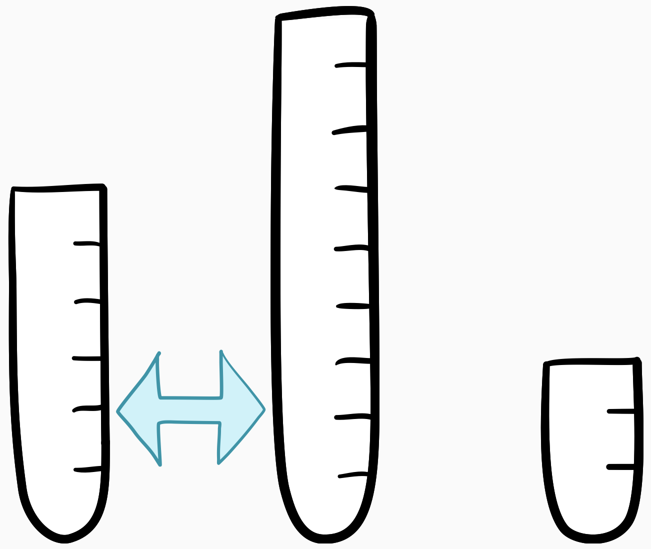
How to handle range of sizes?

Can we beat Nudge?

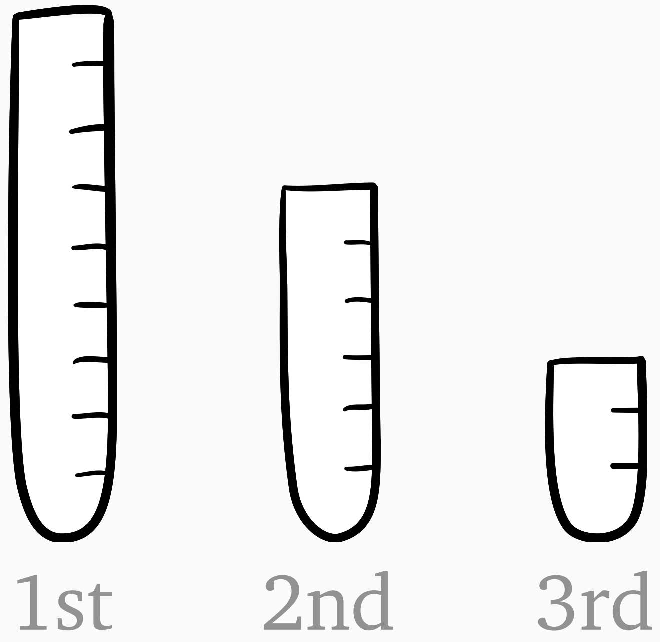


How to handle range of sizes?

medium = small?

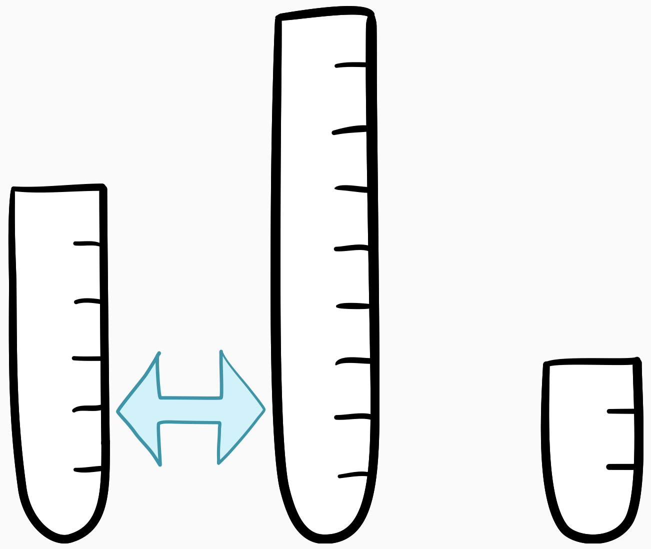


Can we beat Nudge?

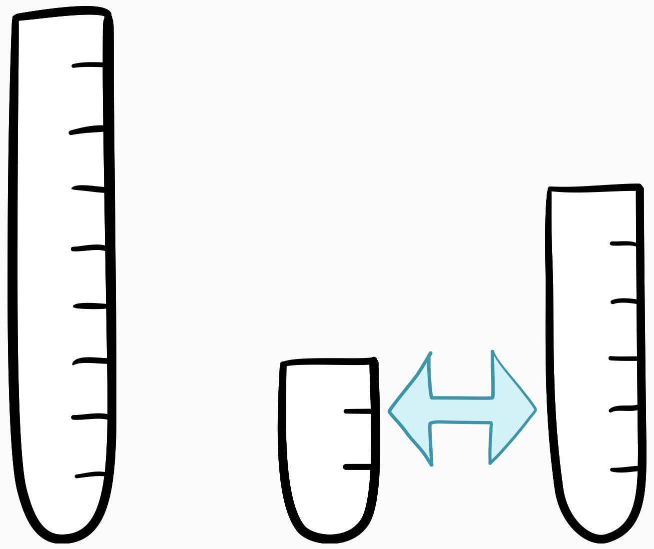


How to handle range of sizes?

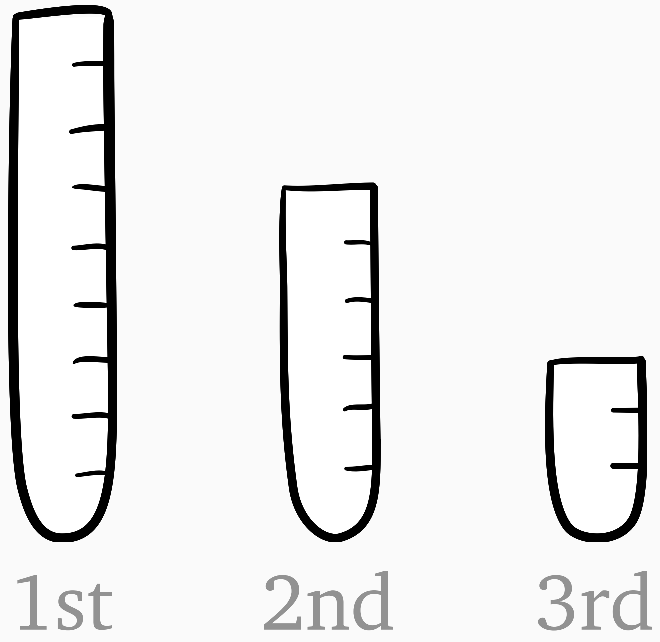
medium = small?



medium = large?

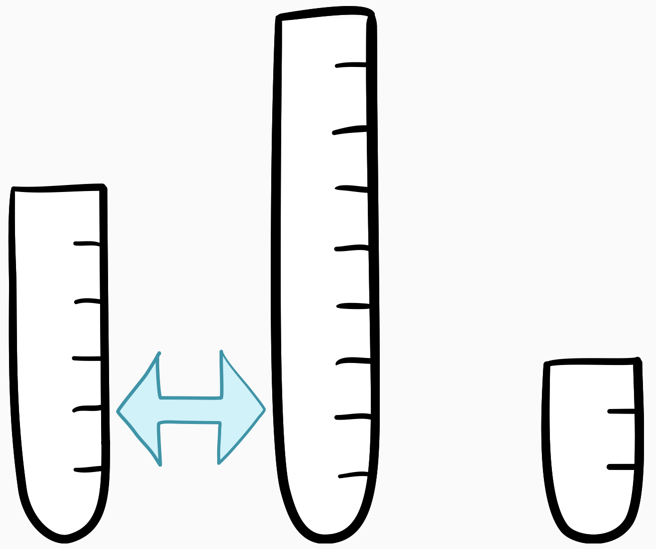


Can we beat Nudge?

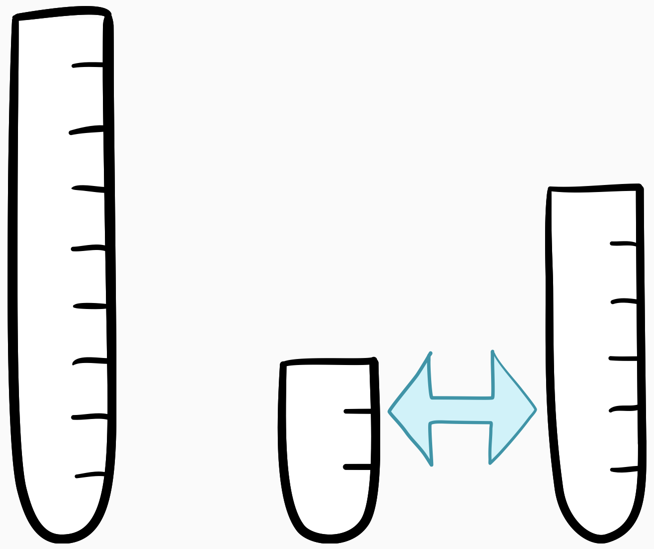


How to handle range of sizes?

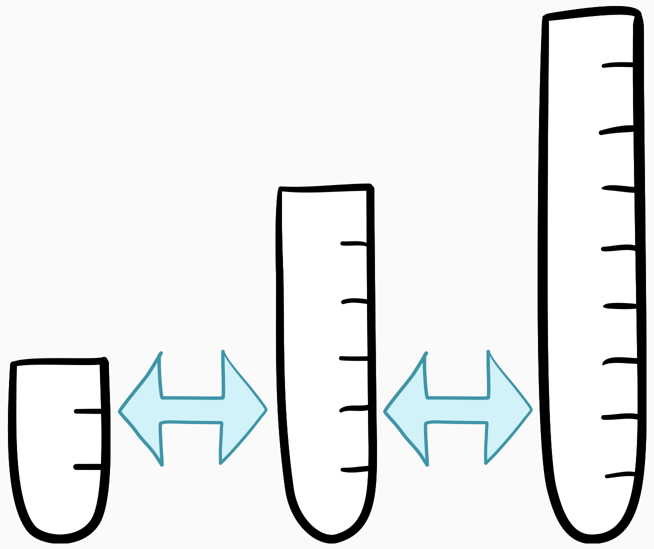
medium = small?



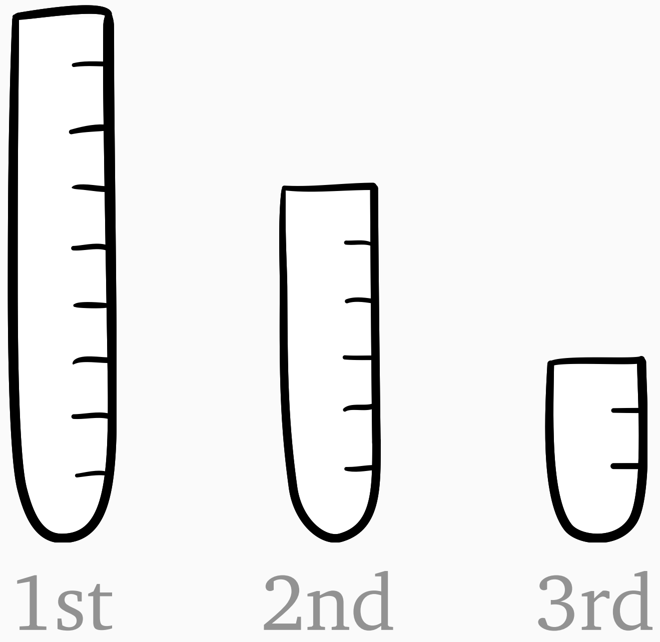
medium = large?



something else?



Can we beat Nudge?



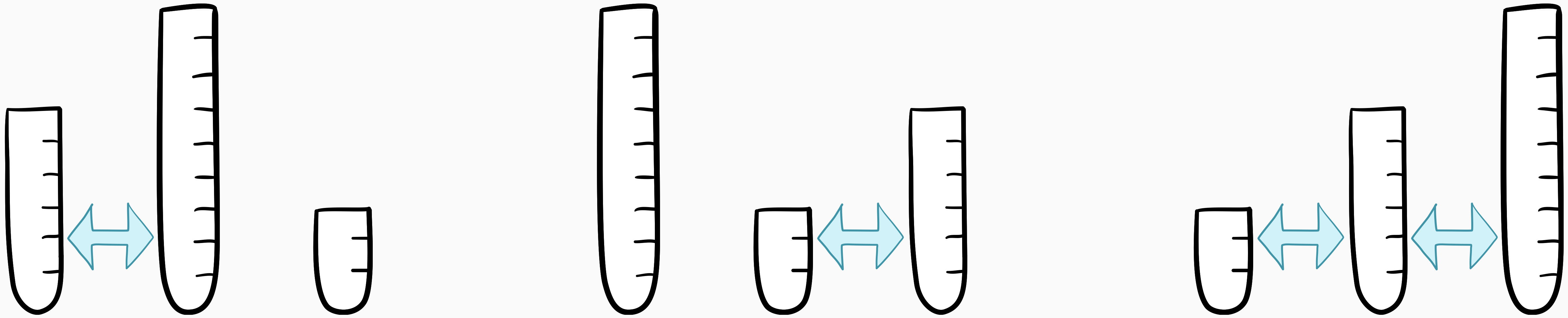
How to handle range of sizes?

What info could help us decide?

medium = small?

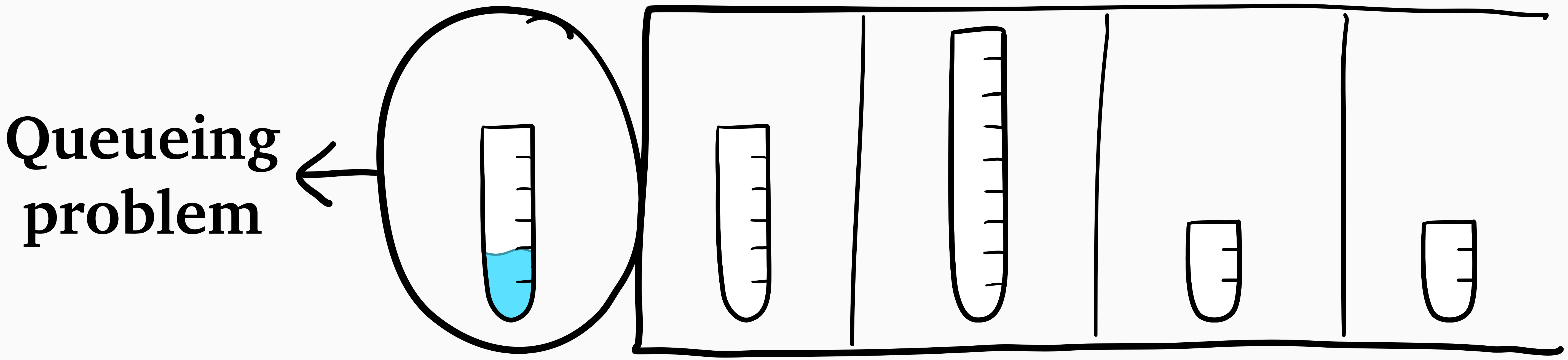
medium = large?

something else?

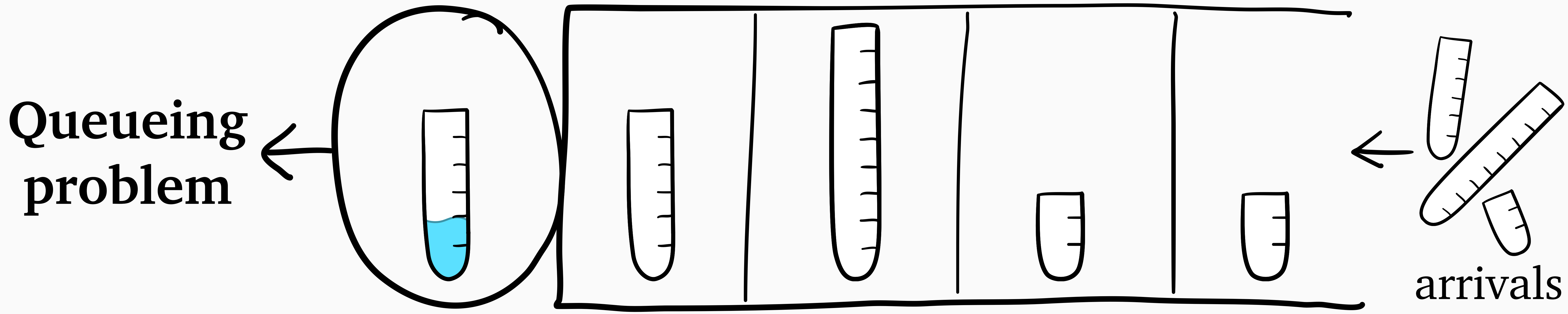


Where do optimal policies come from?

Where do optimal policies come from?

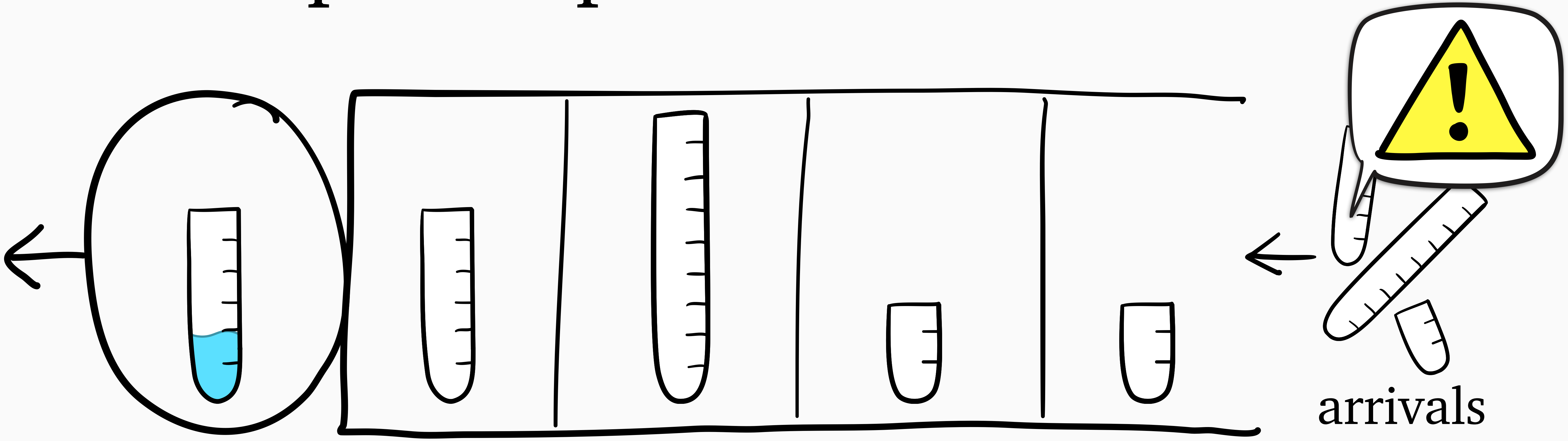


Where do optimal policies come from?



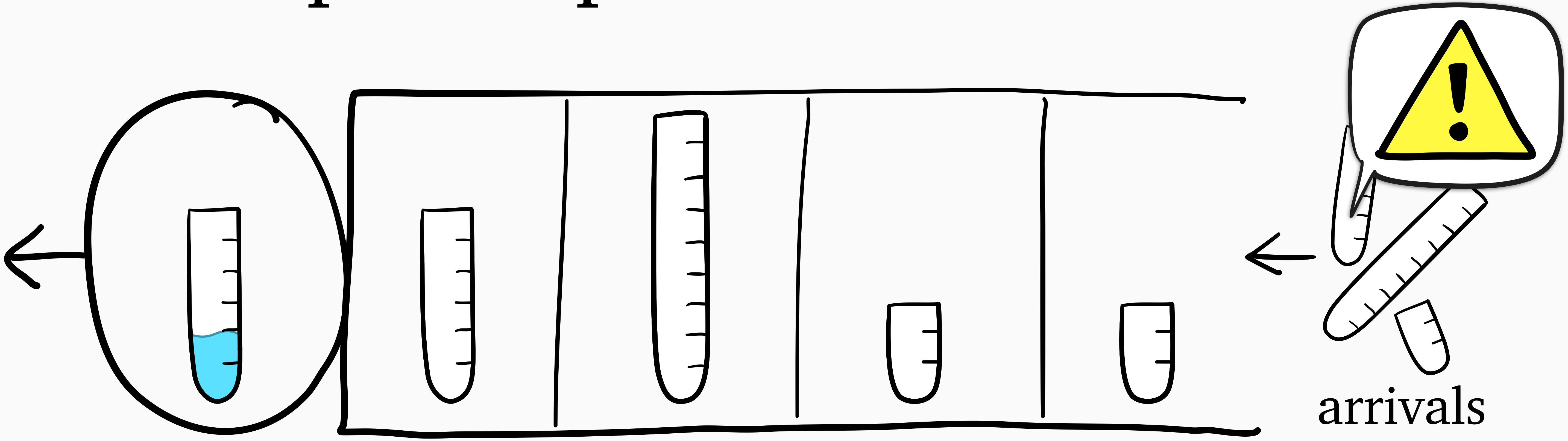
Where do optimal policies come from?

Queueing problem

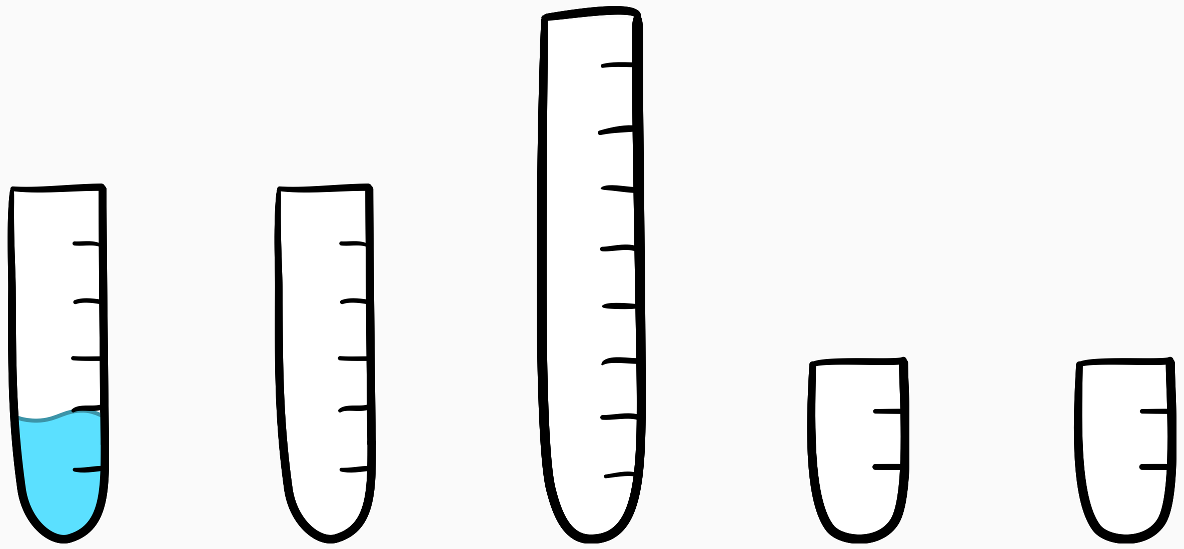


Where do optimal policies come from?

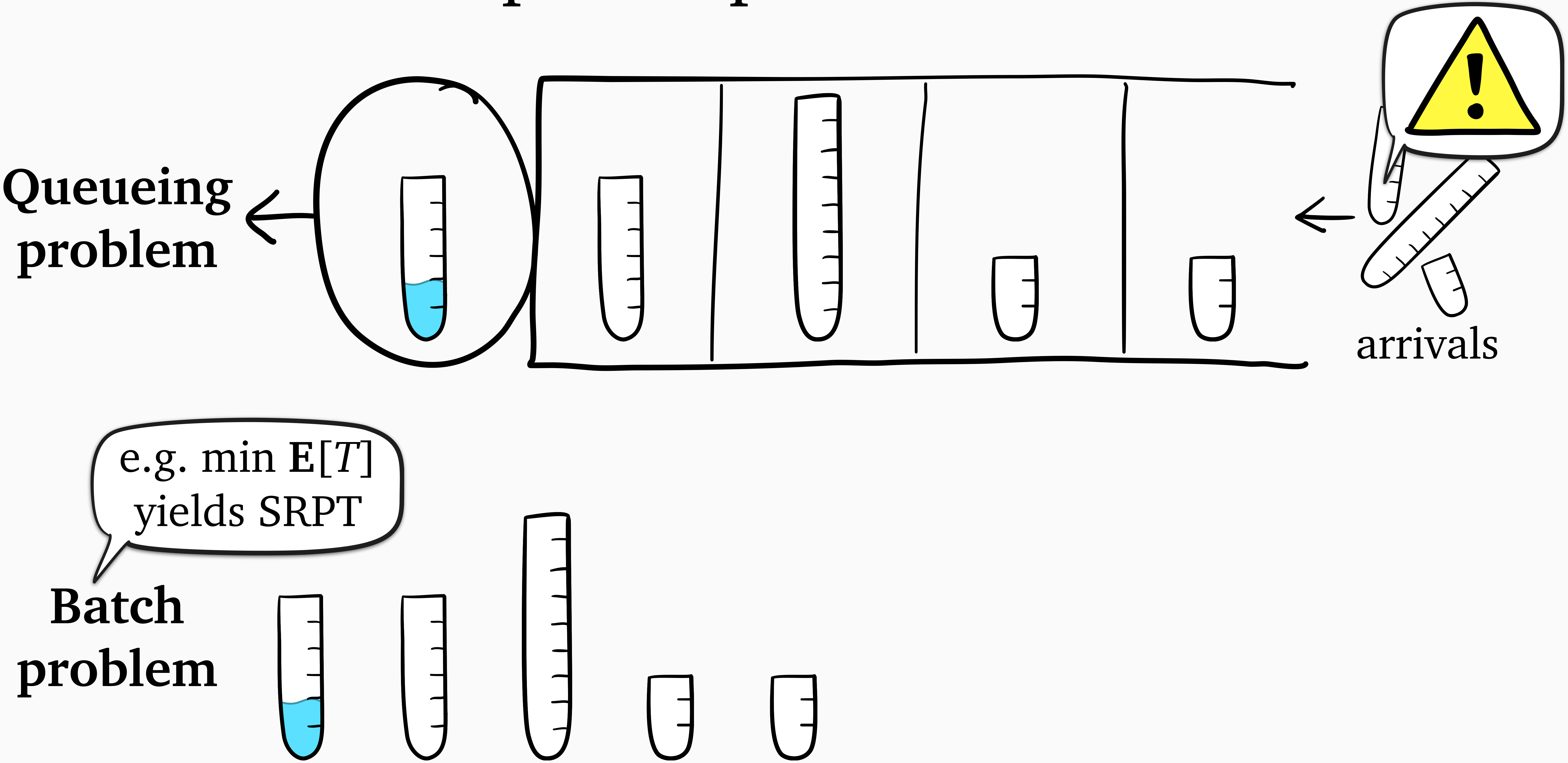
Queueing problem



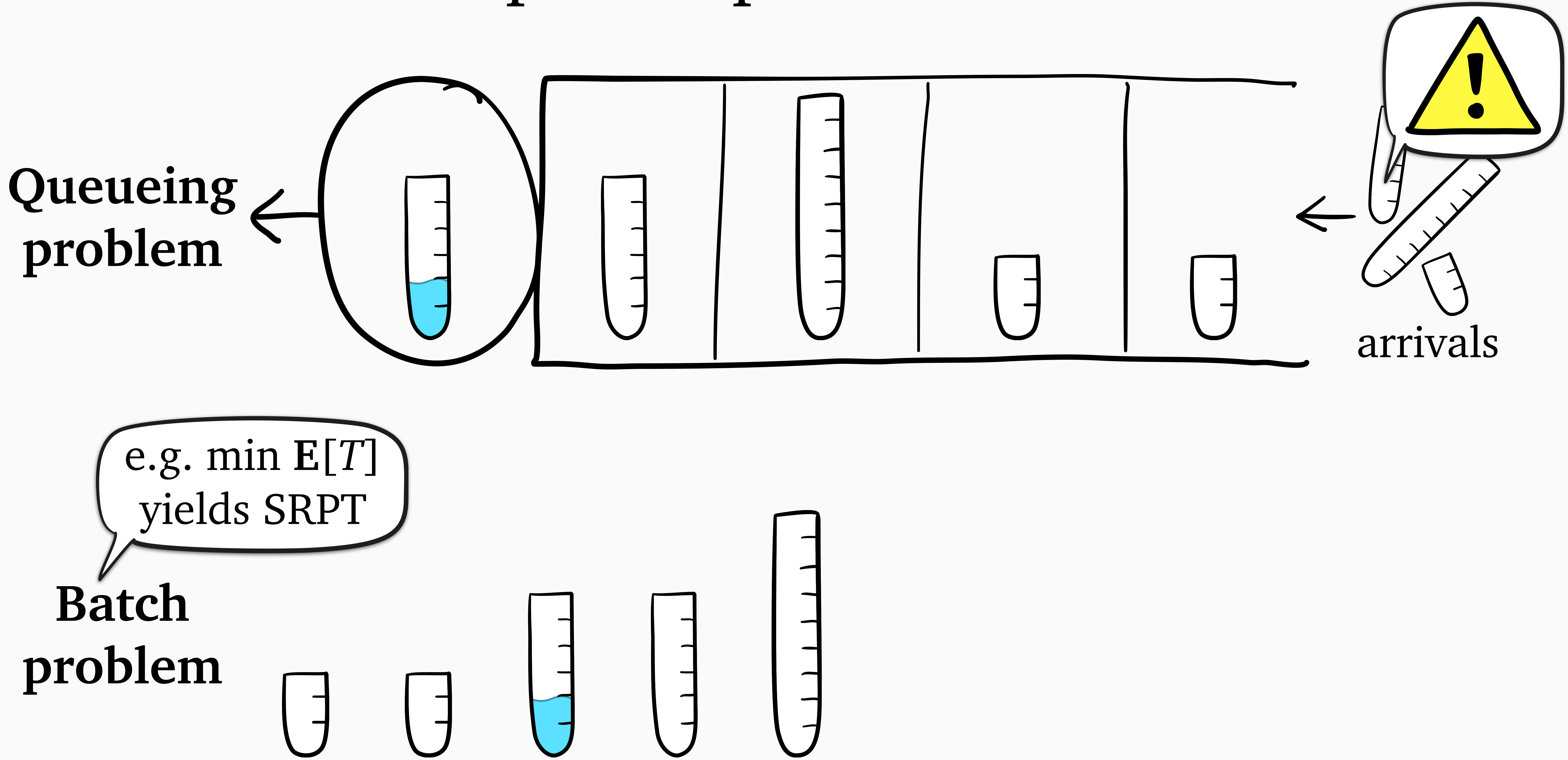
Batch problem



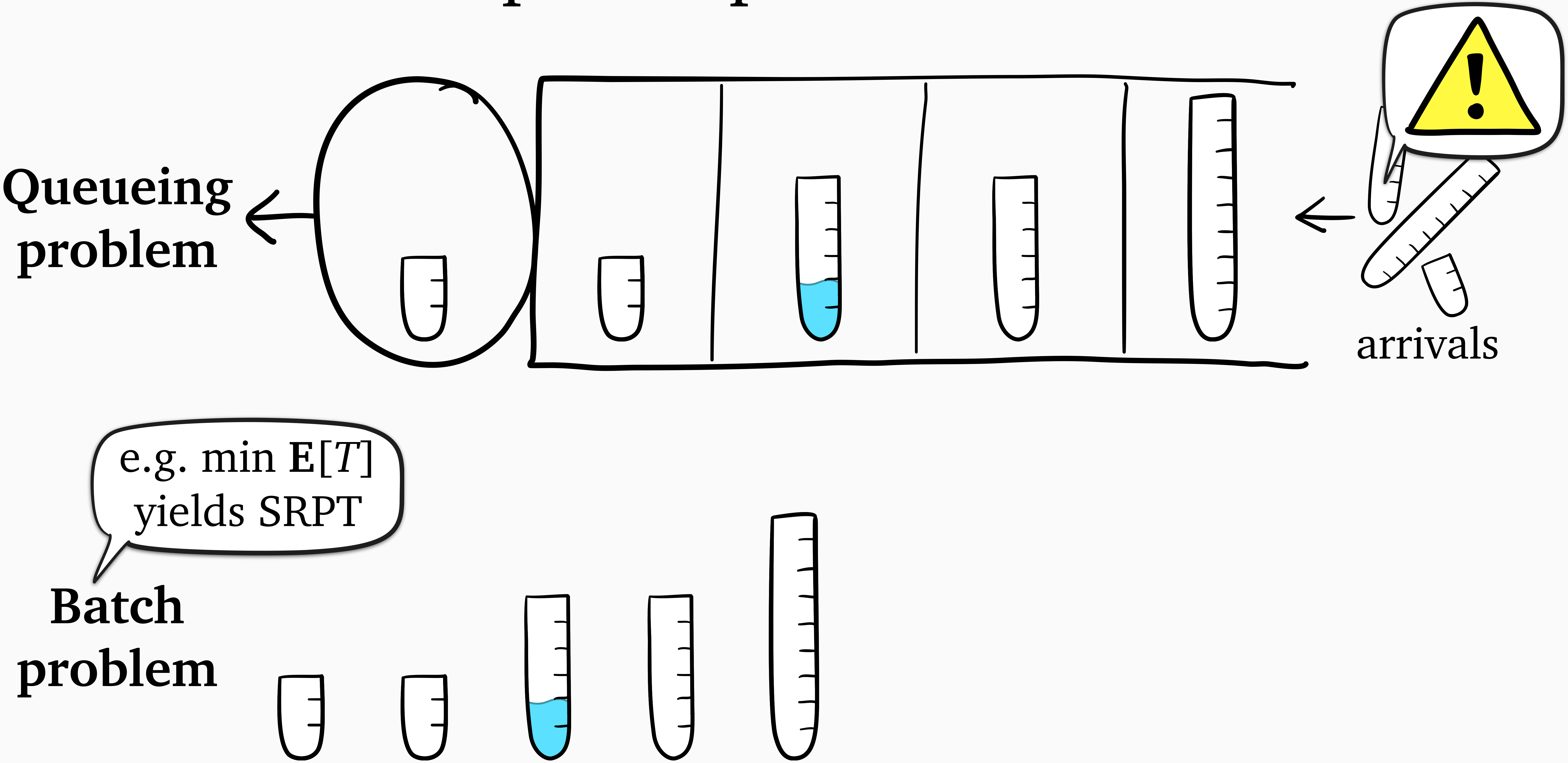
Where do optimal policies come from?



Where do optimal policies come from?

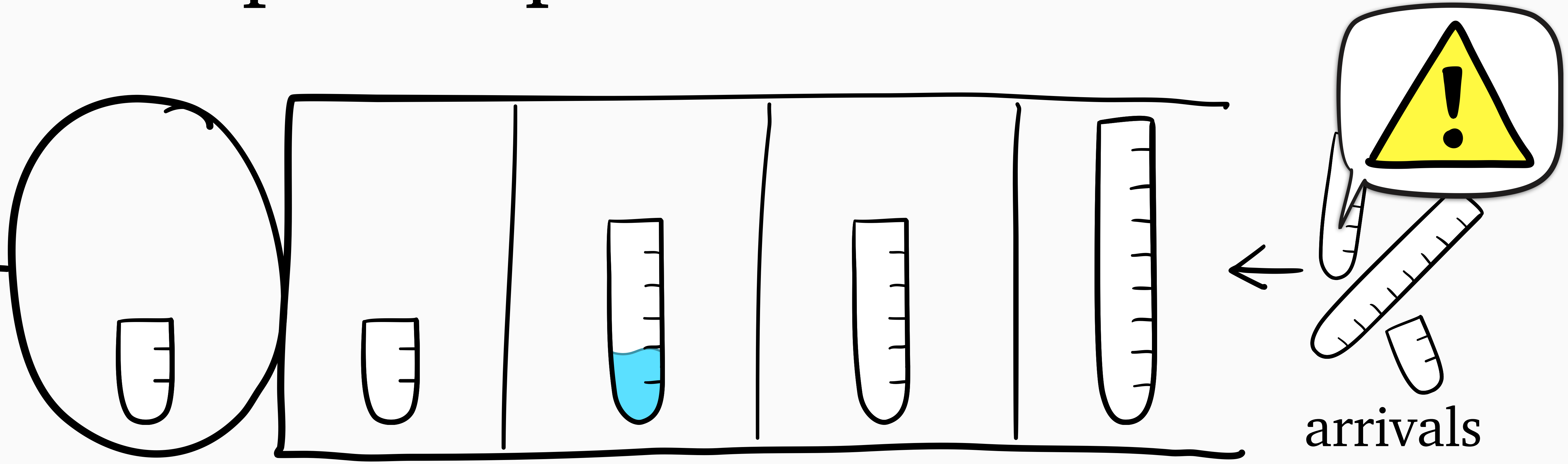


Where do optimal policies come from?



Where do optimal policies come from?

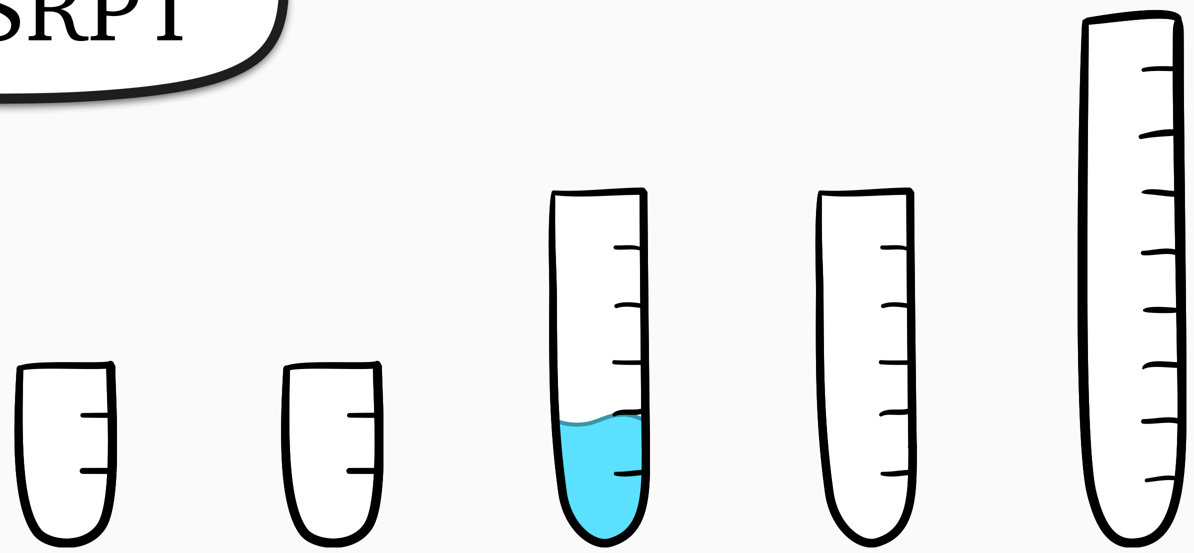
Queueing problem



arrivals

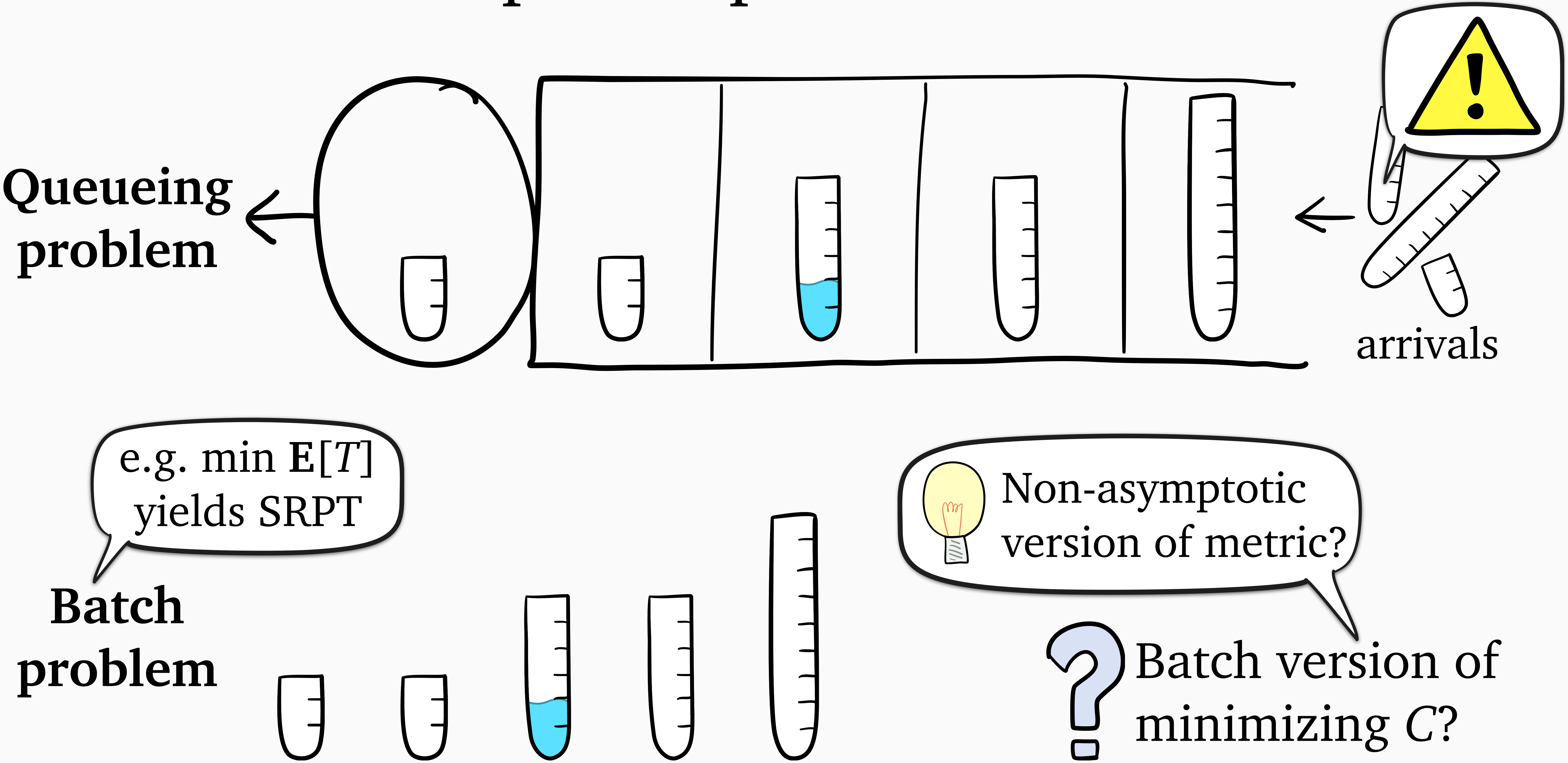
e.g. min $\mathbf{E}[T]$ yields SRPT

Batch problem

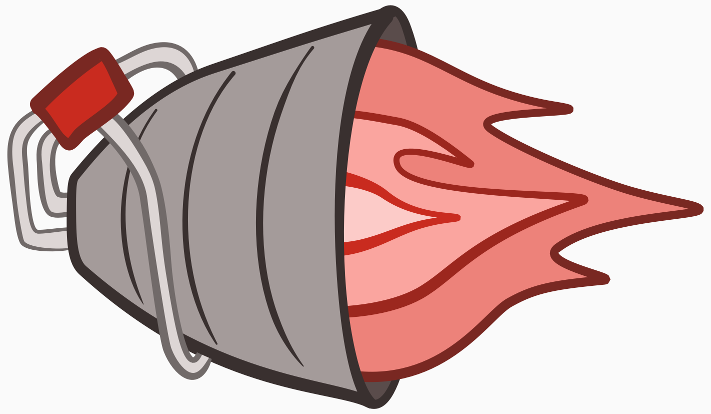


Batch version of minimizing C?

Where do optimal policies come from?



Boost



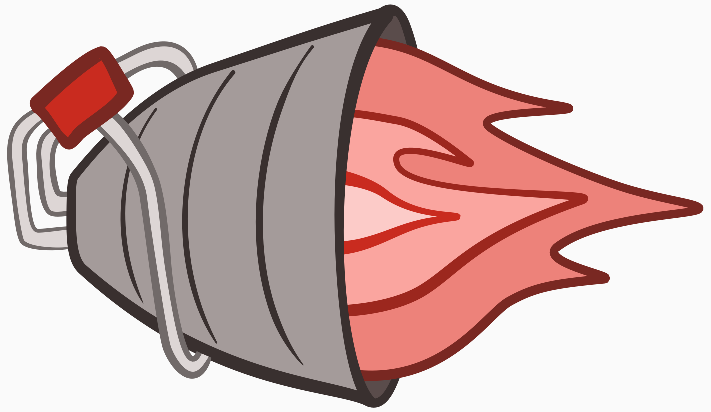
Why did it take so long to beat FCFS?

Why is achieving strong tail optimality hard?

How does the **Boost** policy family work?

How do we achieve strong tail optimality?

Boost



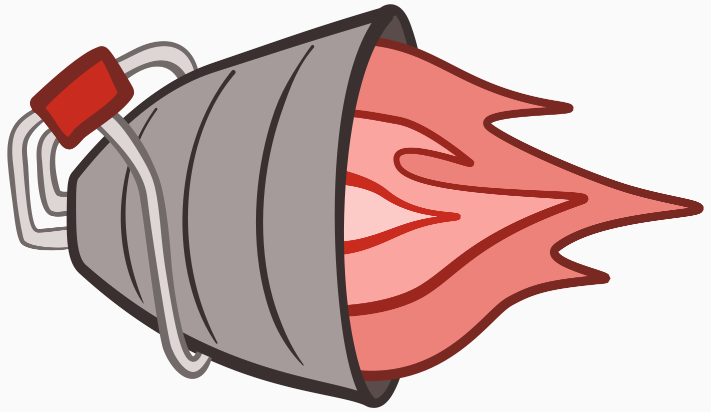
Why did it take so long to beat FCFS?

Why is achieving strong tail optimality hard?

How does the **Boost** policy family work?

How do we achieve strong tail optimality?

Boost



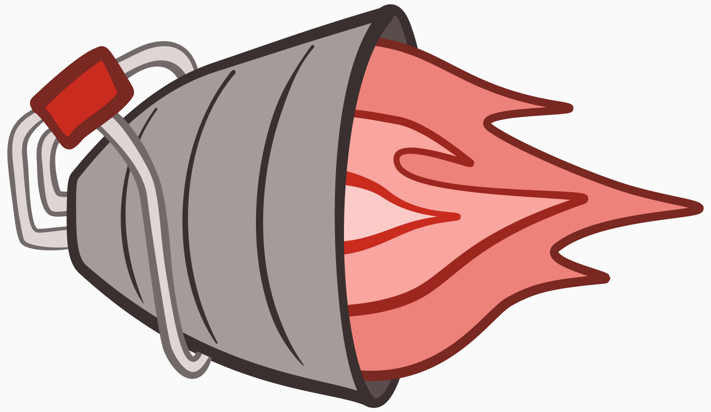
- Why did it take so long to beat FCFS?
- Why is achieving strong tail optimality hard?

How to handle range of sizes?

Batch version of minimizing C?

- How does the **Boost** policy family work?
- How do we achieve strong tail optimality?

Boost



Why did it take so long to beat FCFS?

How to handle range of sizes?

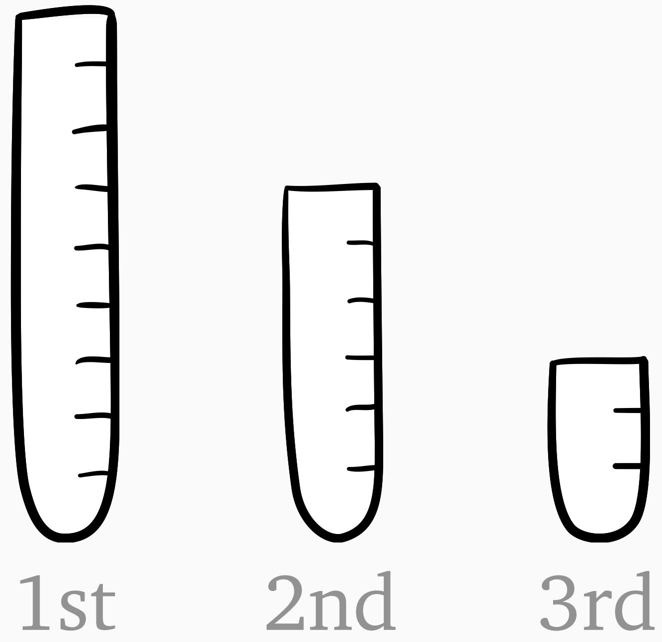
Why is achieving strong tail optimality hard?

Batch version of minimizing C?

How does the **Boost** policy family work?

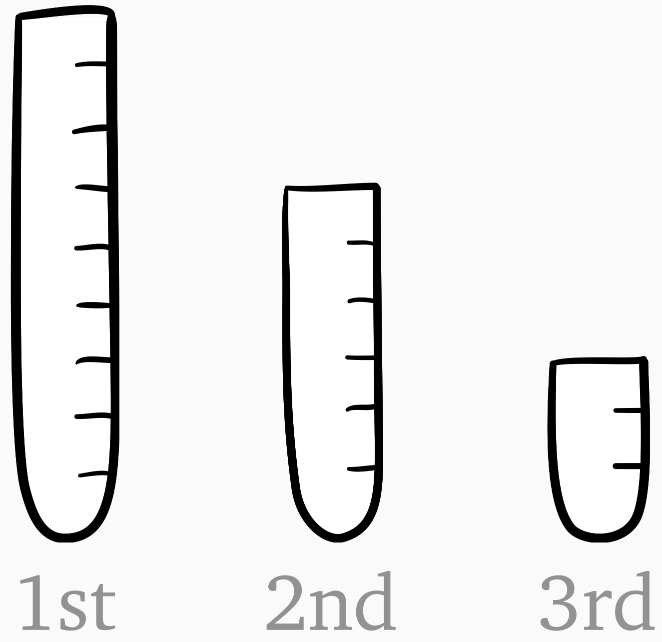
How do we achieve strong tail optimality?

Key information:



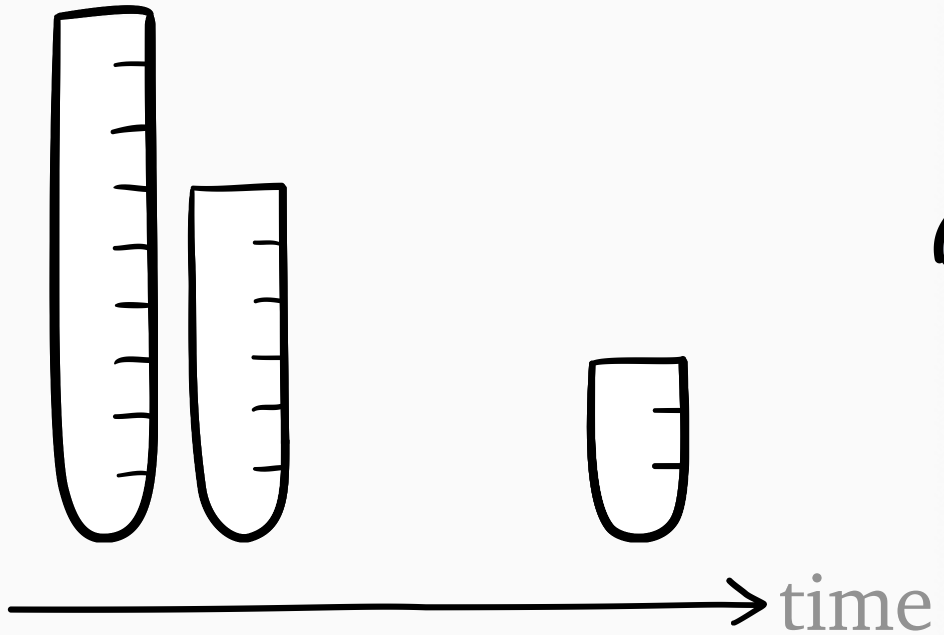
How to handle range of sizes?

Key information: *arrival times*



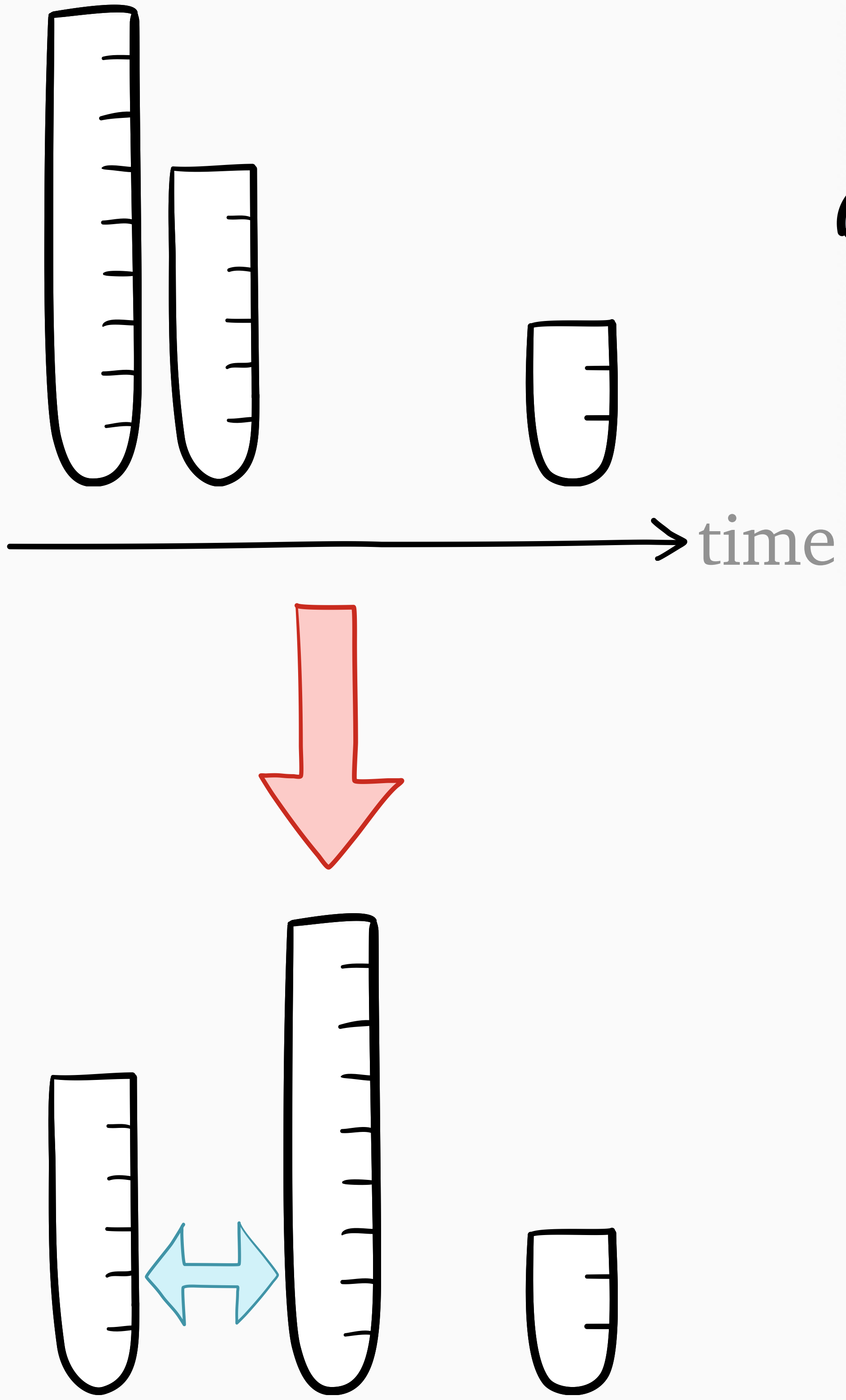
How to handle
range of sizes?

Key information: *arrival times*



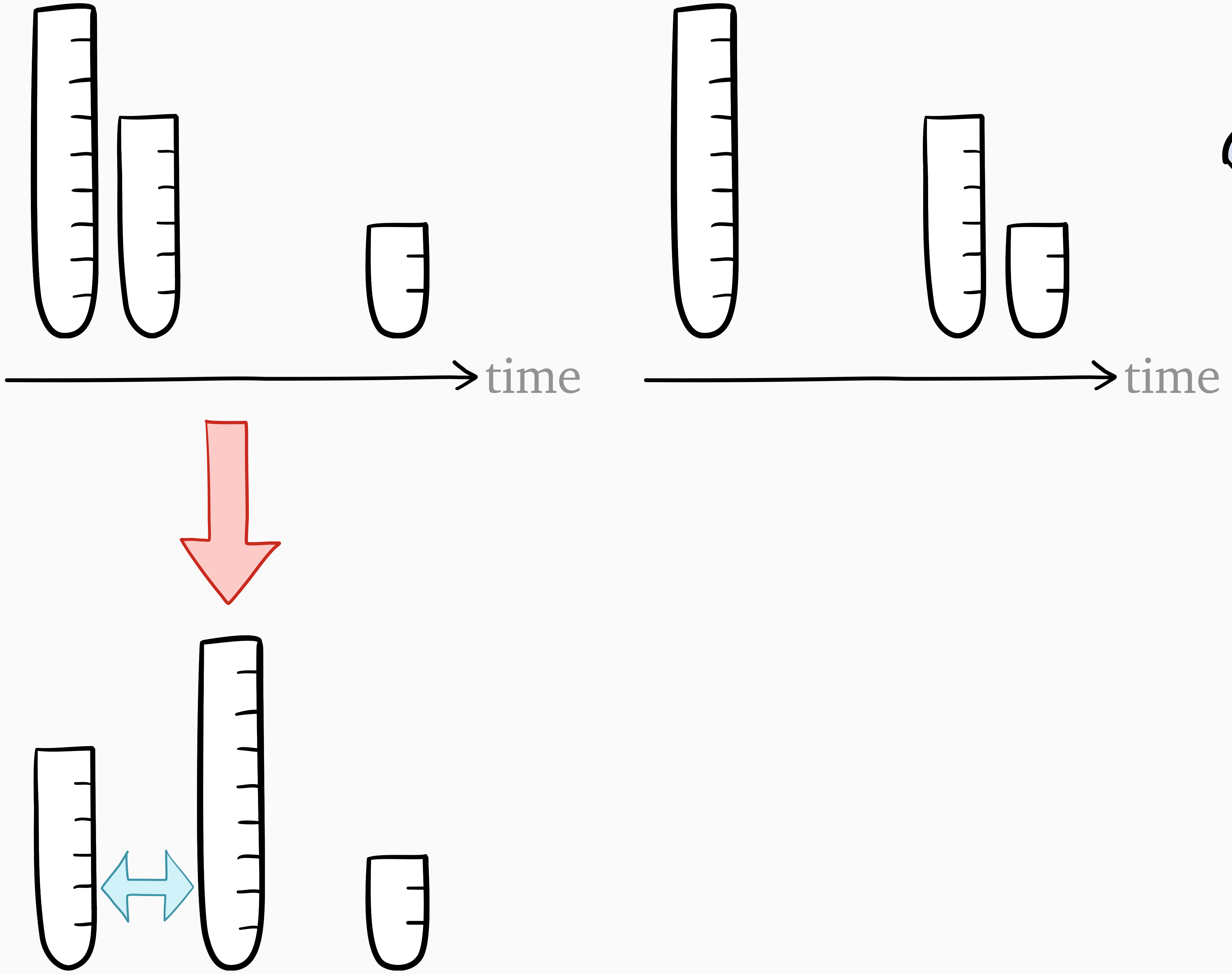
How to handle range of sizes?

Key information: *arrival times*



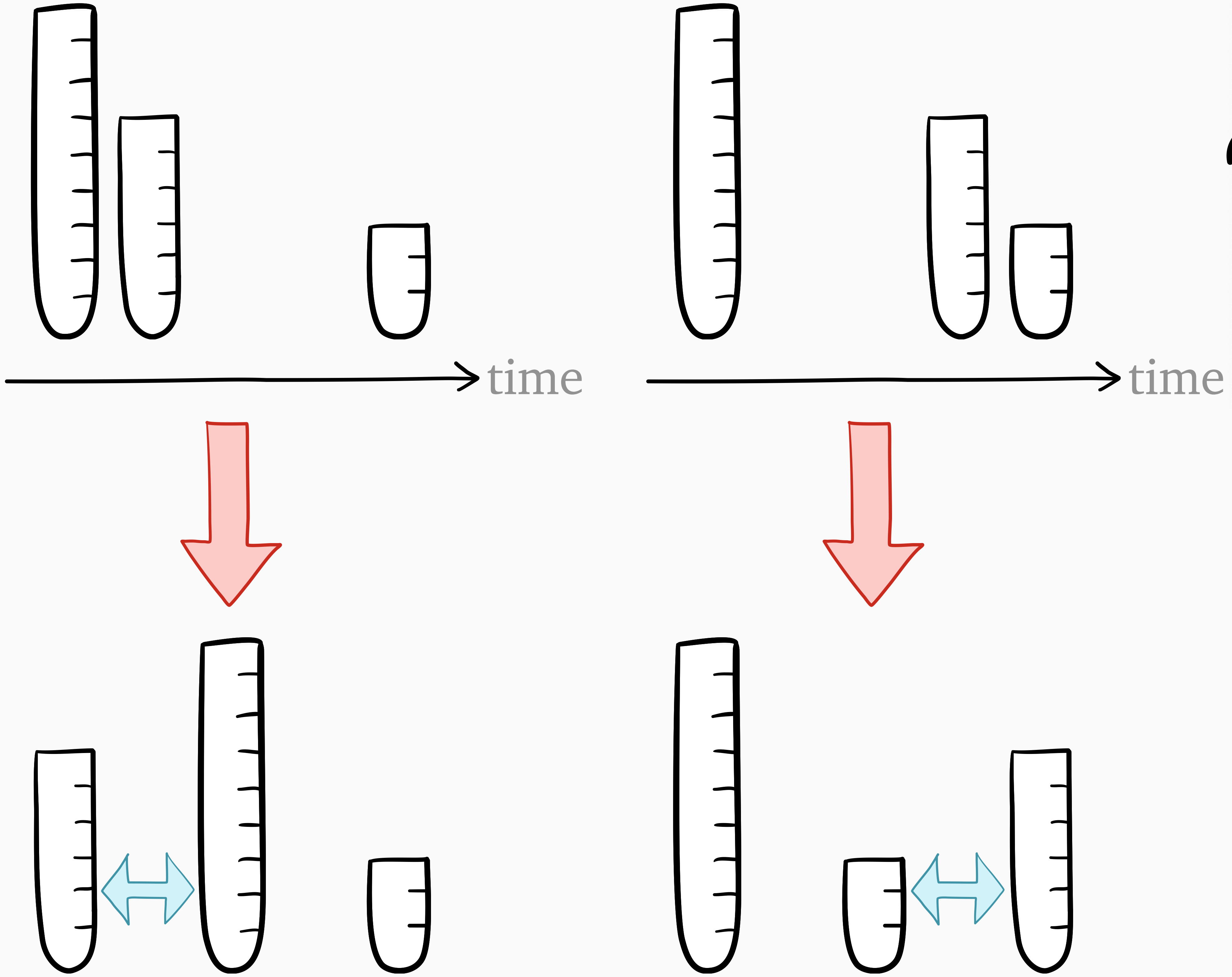
How to handle
range of sizes?

Key information: *arrival times*



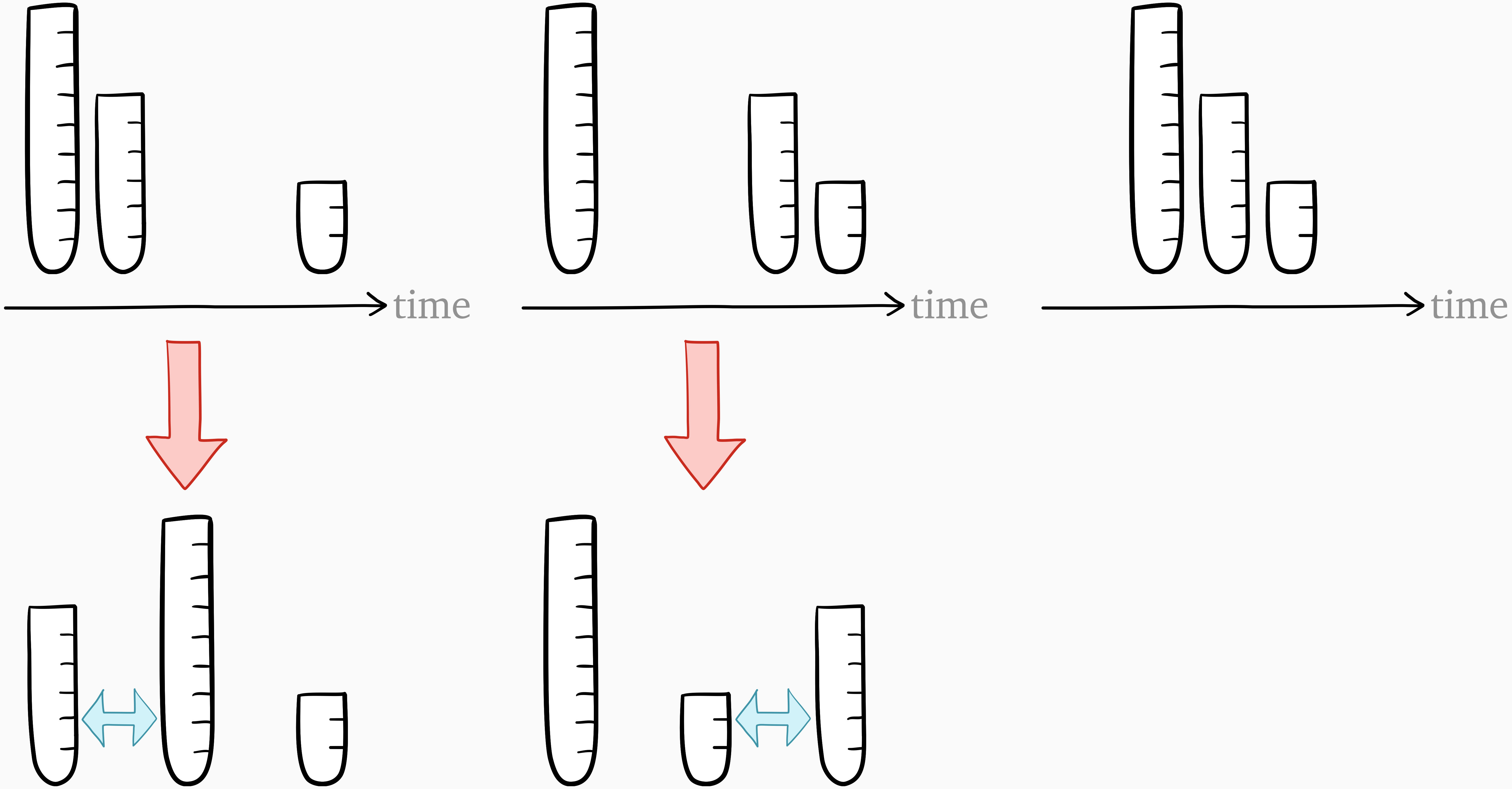
How to handle range of sizes?

Key information: *arrival times*

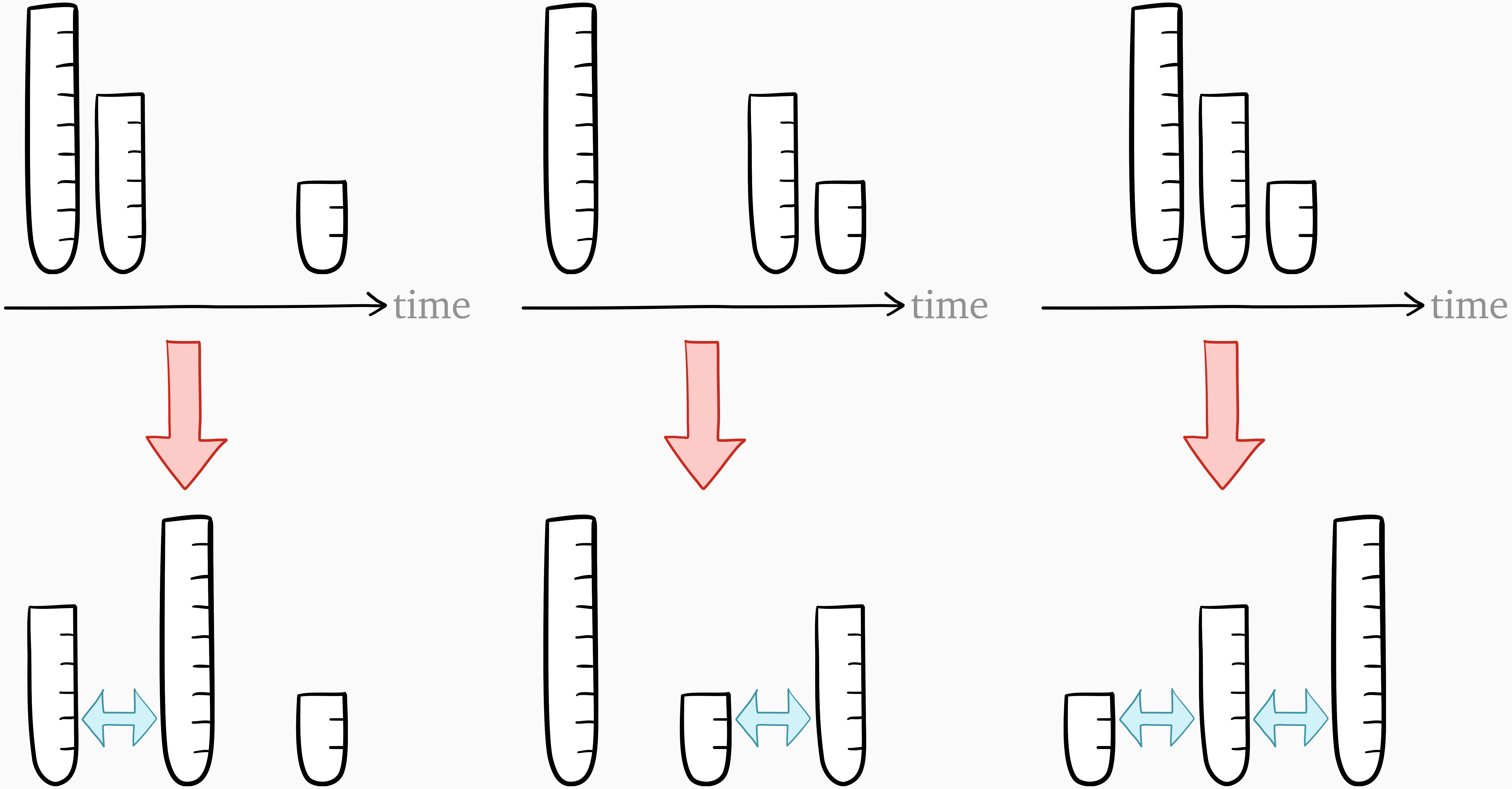


How to handle range of sizes?

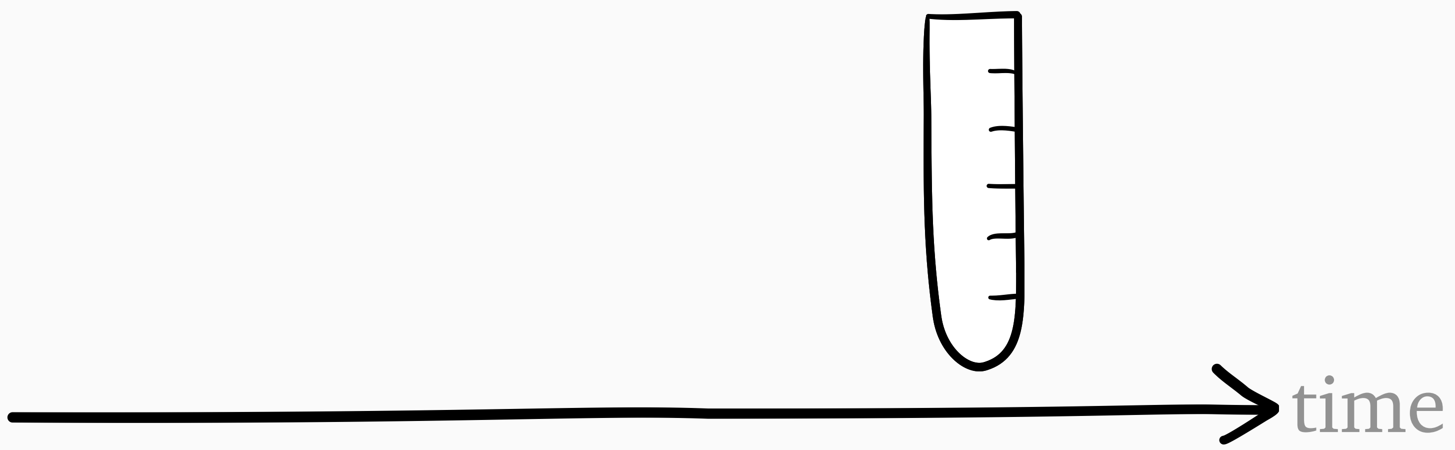
Key information: *arrival times*



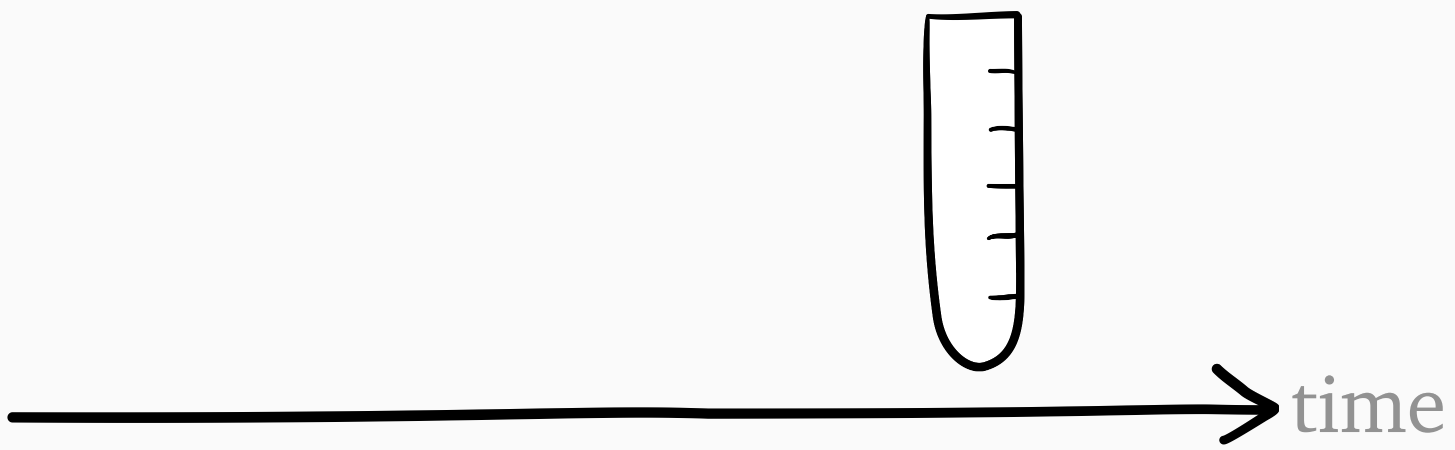
Key information: *arrival times*



Combining arrival time and size



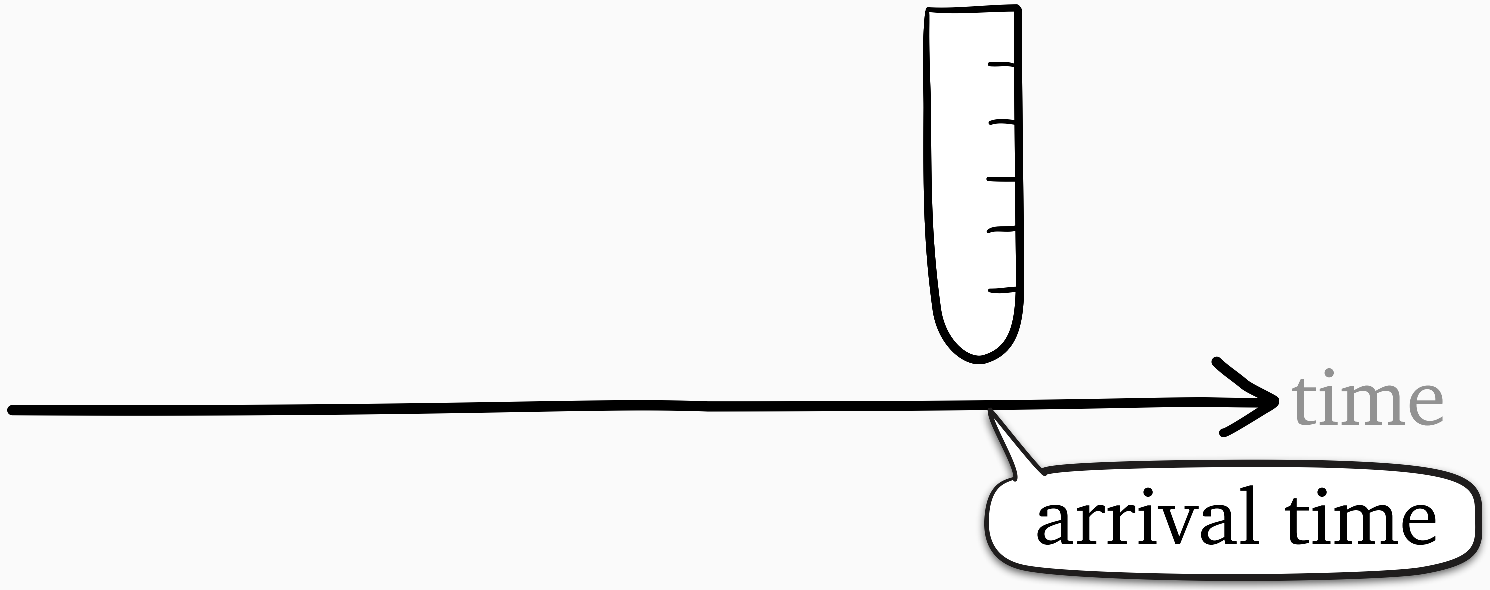
Combining arrival time and size



boosted arrival time

$= \text{arrival time} - \boldsymbol{b}(\text{size})$

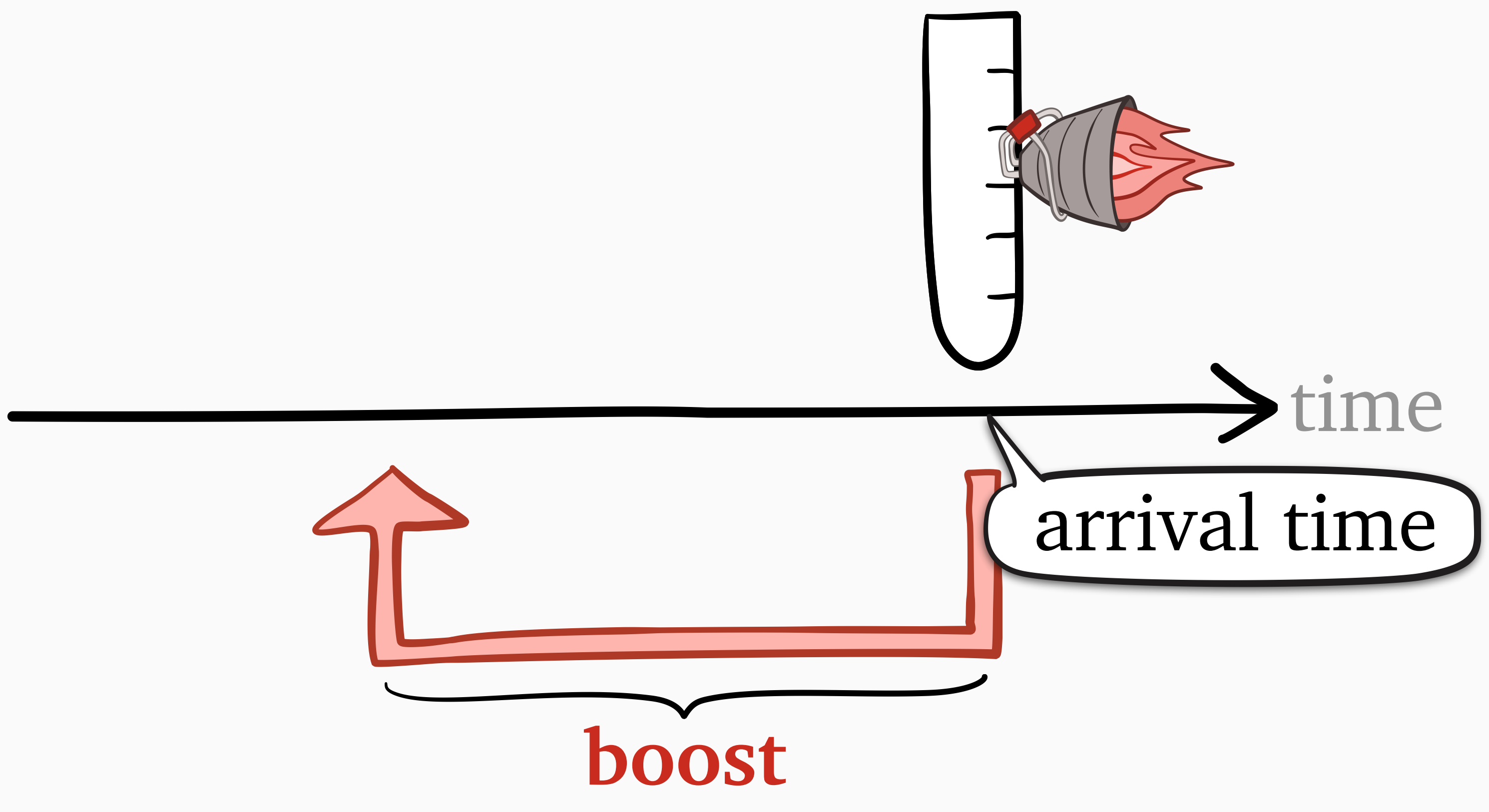
Combining arrival time and size



boosted arrival time

$= \text{arrival time} - \boldsymbol{b}(\text{size})$

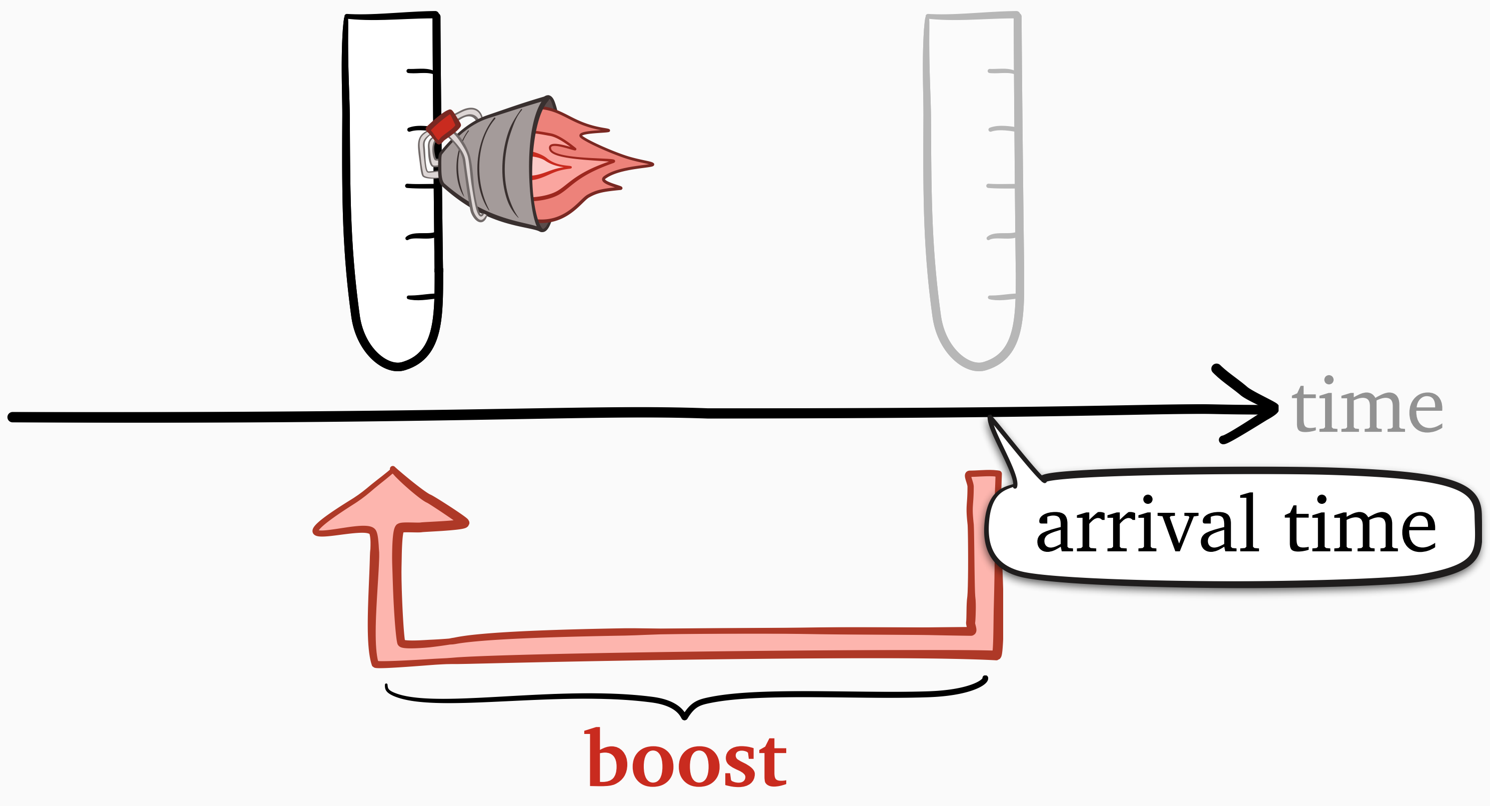
Combining arrival time and size



boosted arrival time

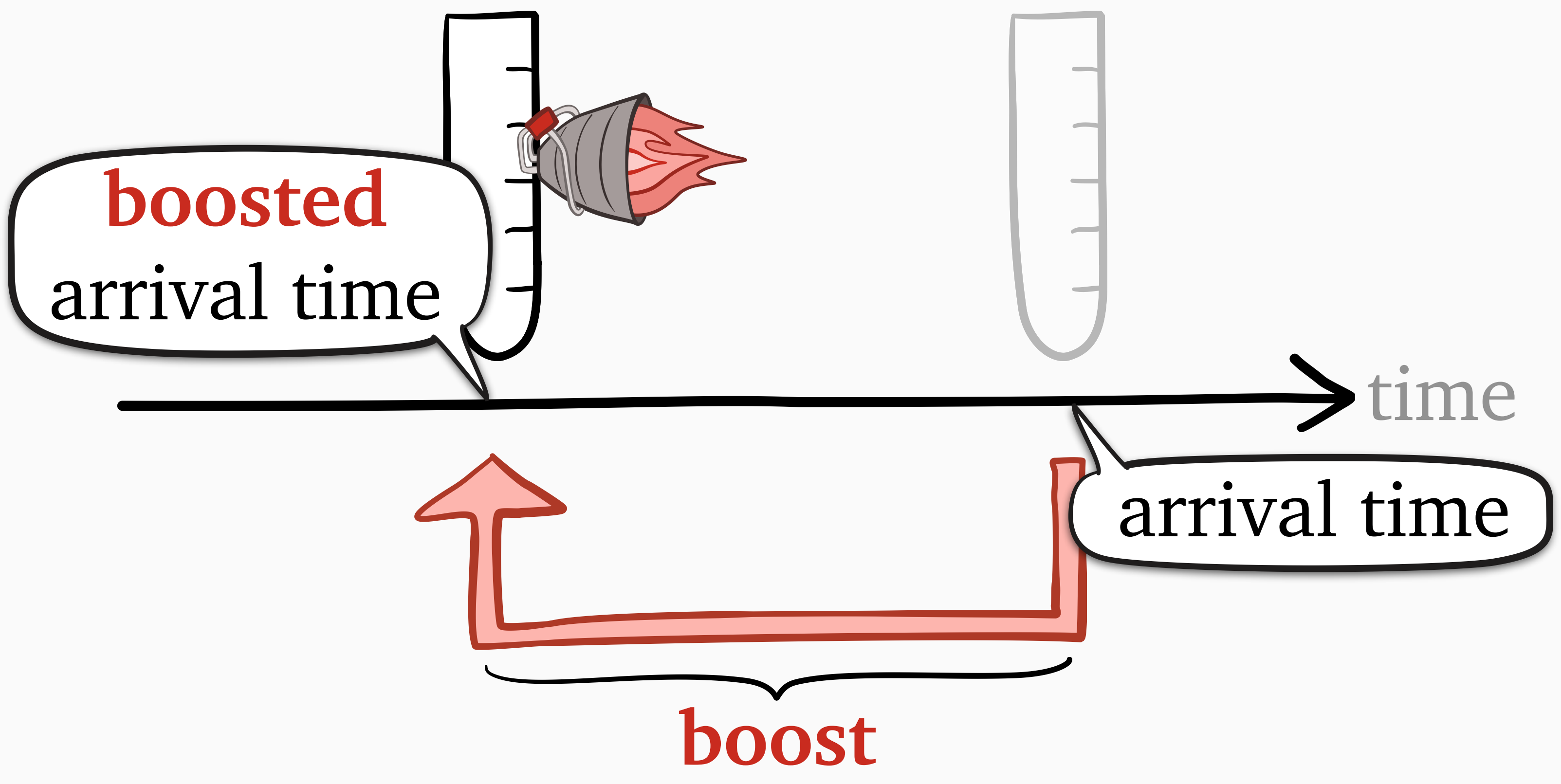
$$= \text{arrival time} - \boldsymbol{b}(\text{size})$$

Combining arrival time and size



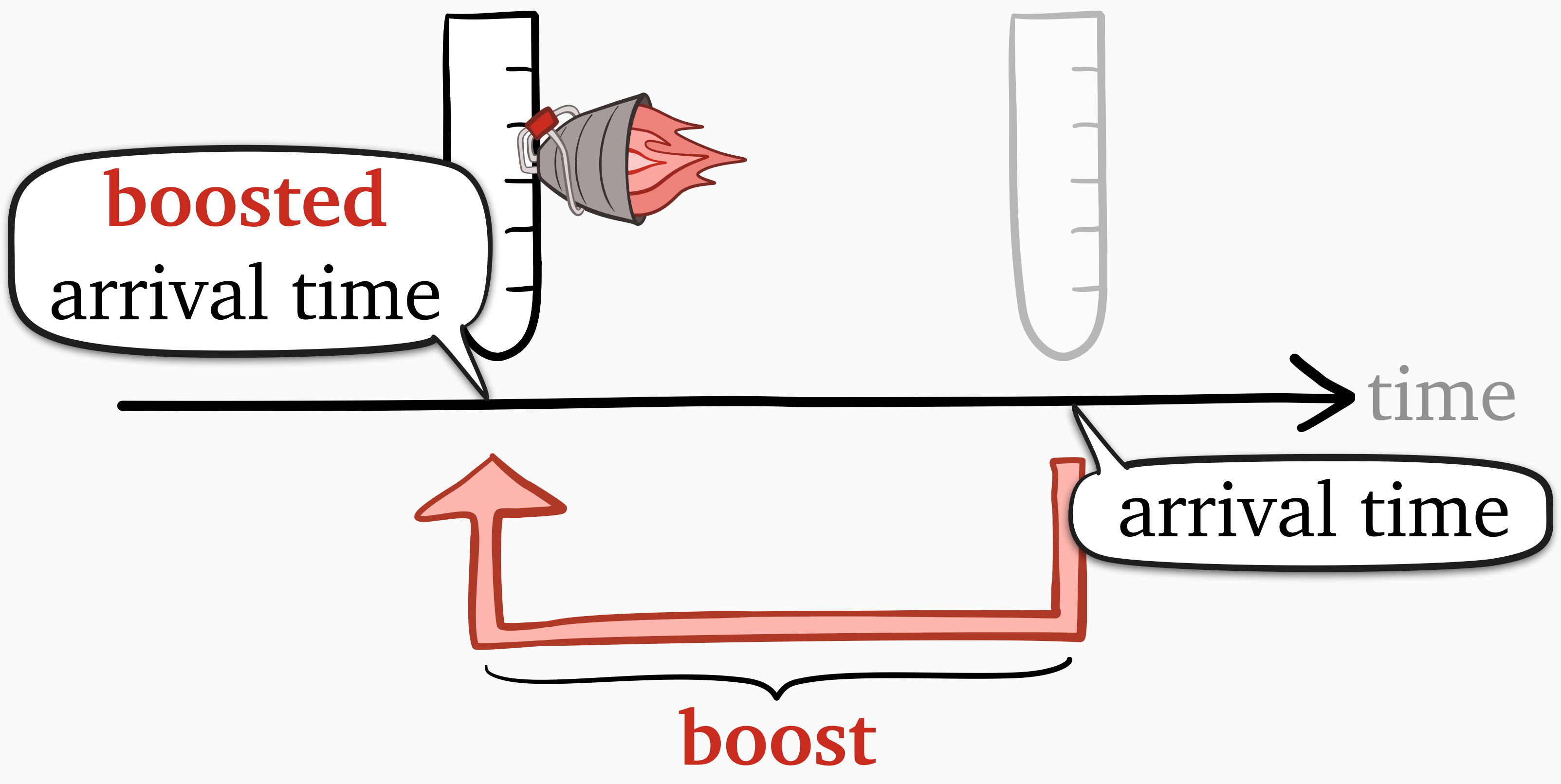
boosted arrival time
$= \text{arrival time} - \boldsymbol{b}(\text{size})$

Combining arrival time and size



boosted arrival time
$$= \text{arrival time} - \boldsymbol{b}(\text{size})$$

Combining arrival time and size

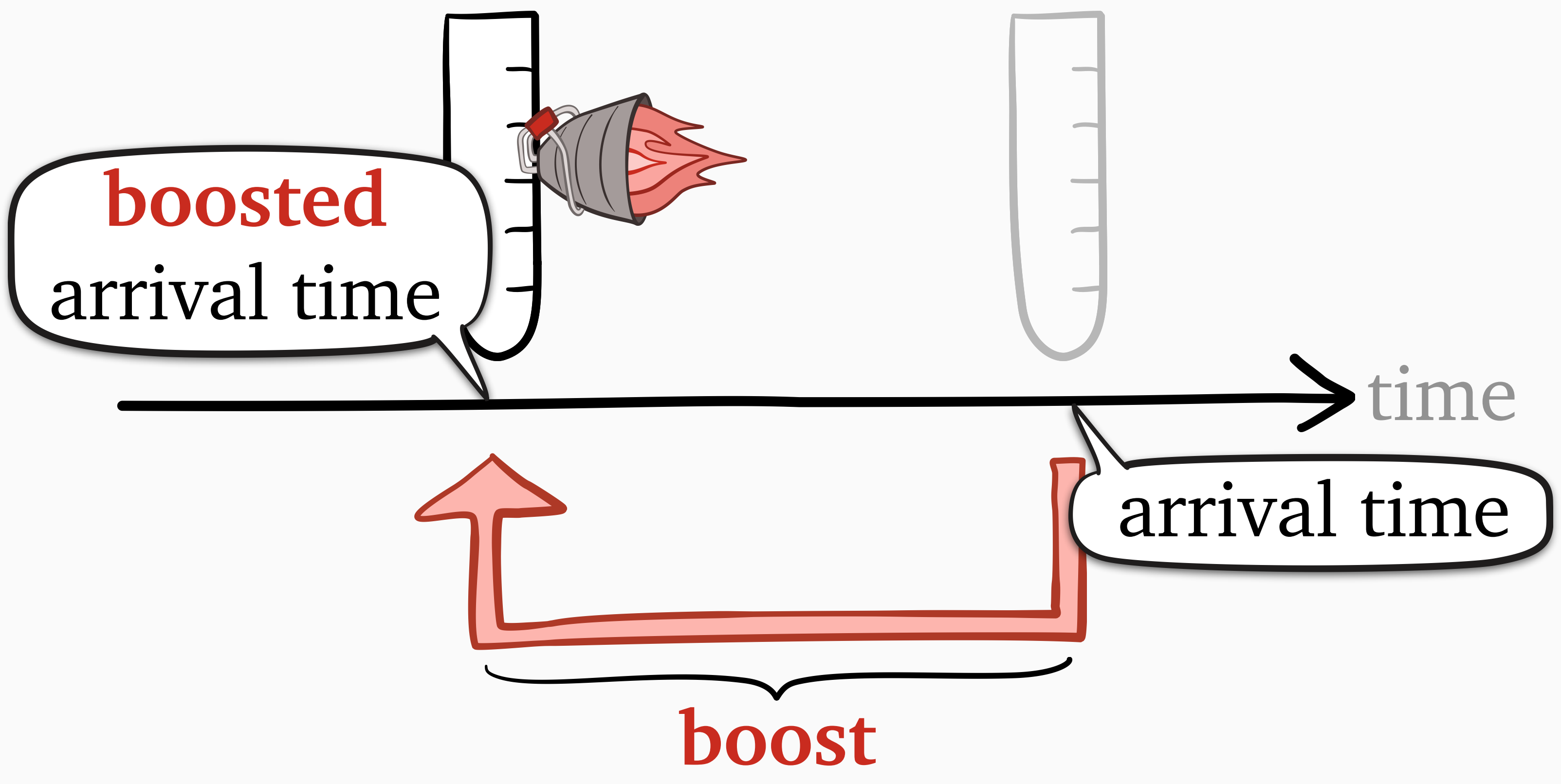


boosted arrival time

= arrival time − $\boldsymbol{b}$(size)

smaller sizes get bigger boosts

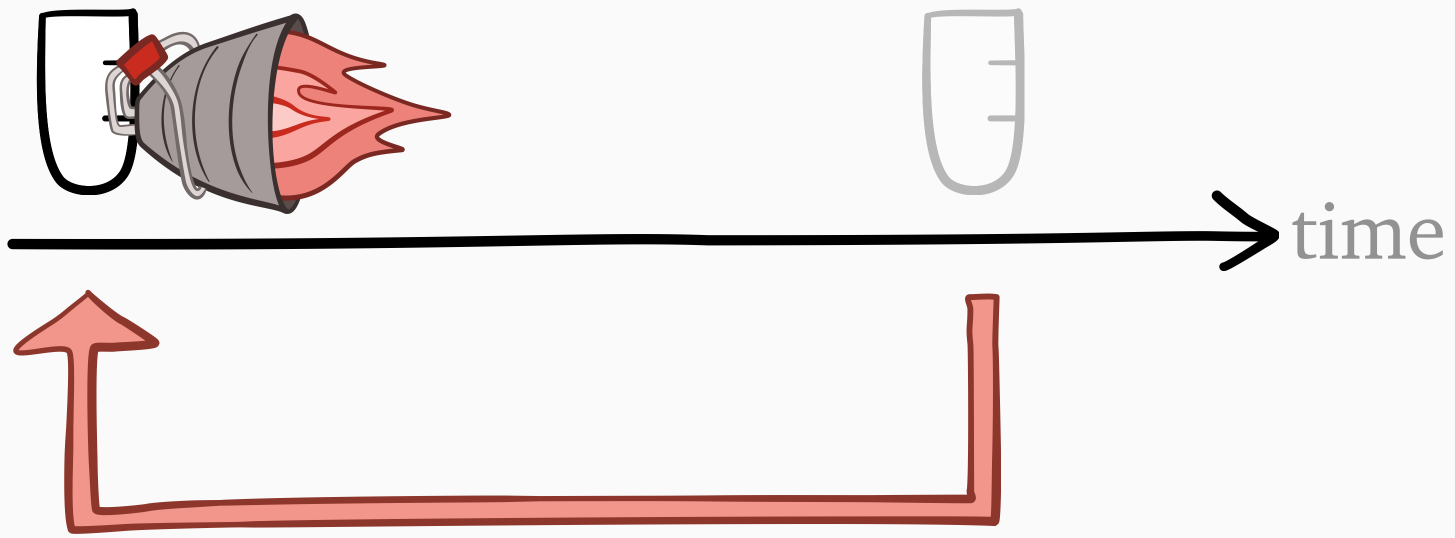
Combining arrival time and size



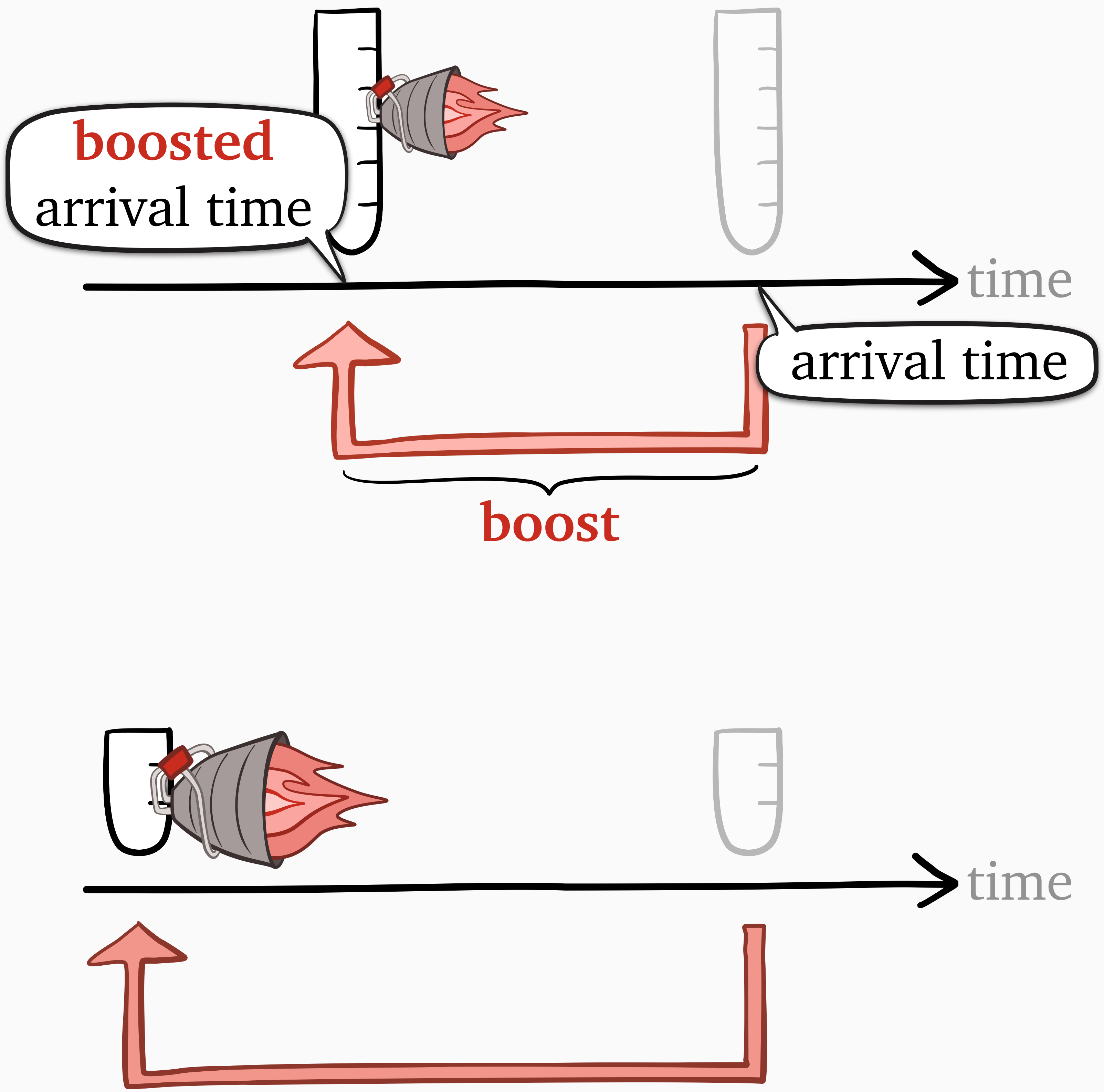
boosted arrival time

= arrival time − $\boldsymbol{b}$(size)

smaller sizes get bigger boosts

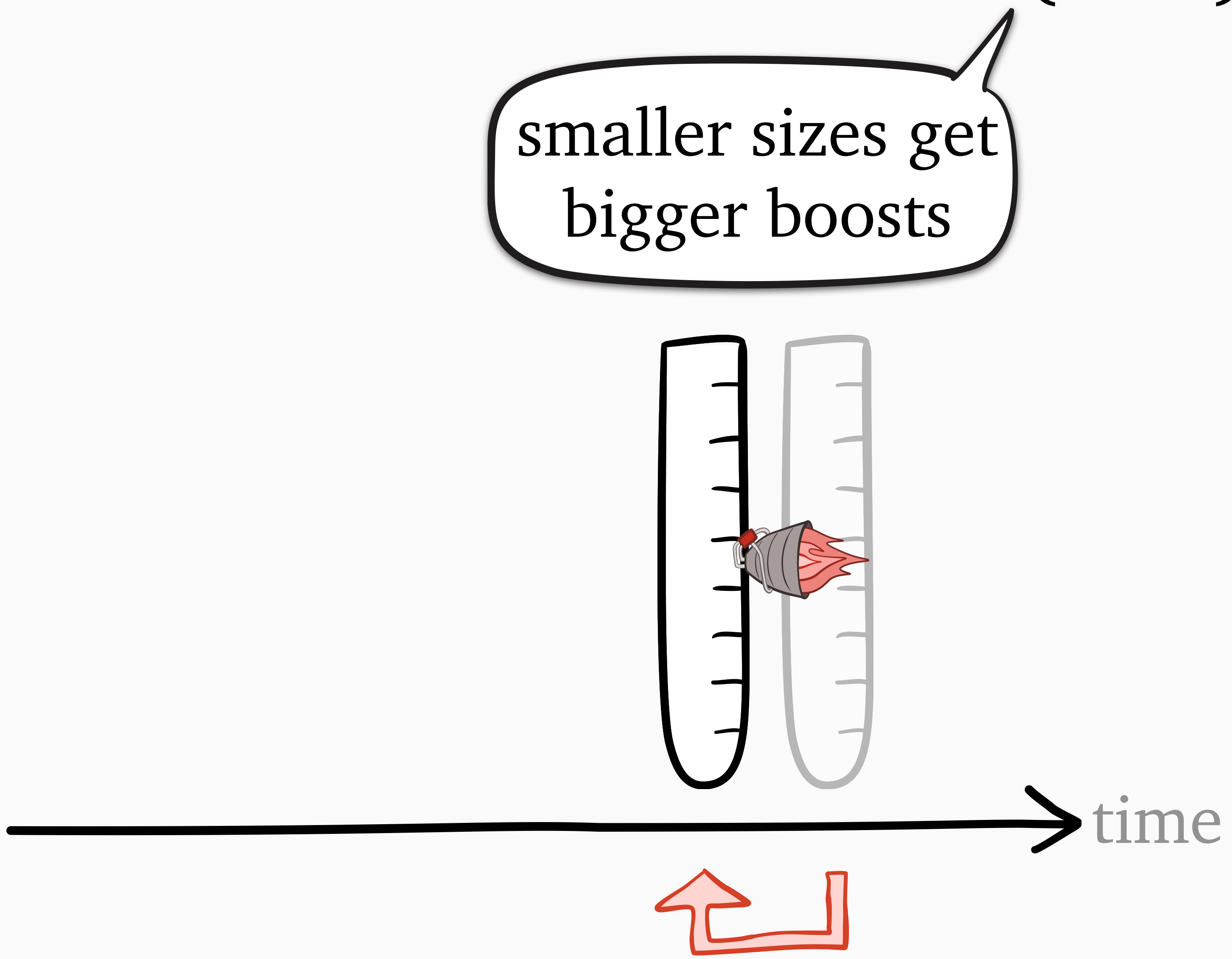


Combining arrival time and size

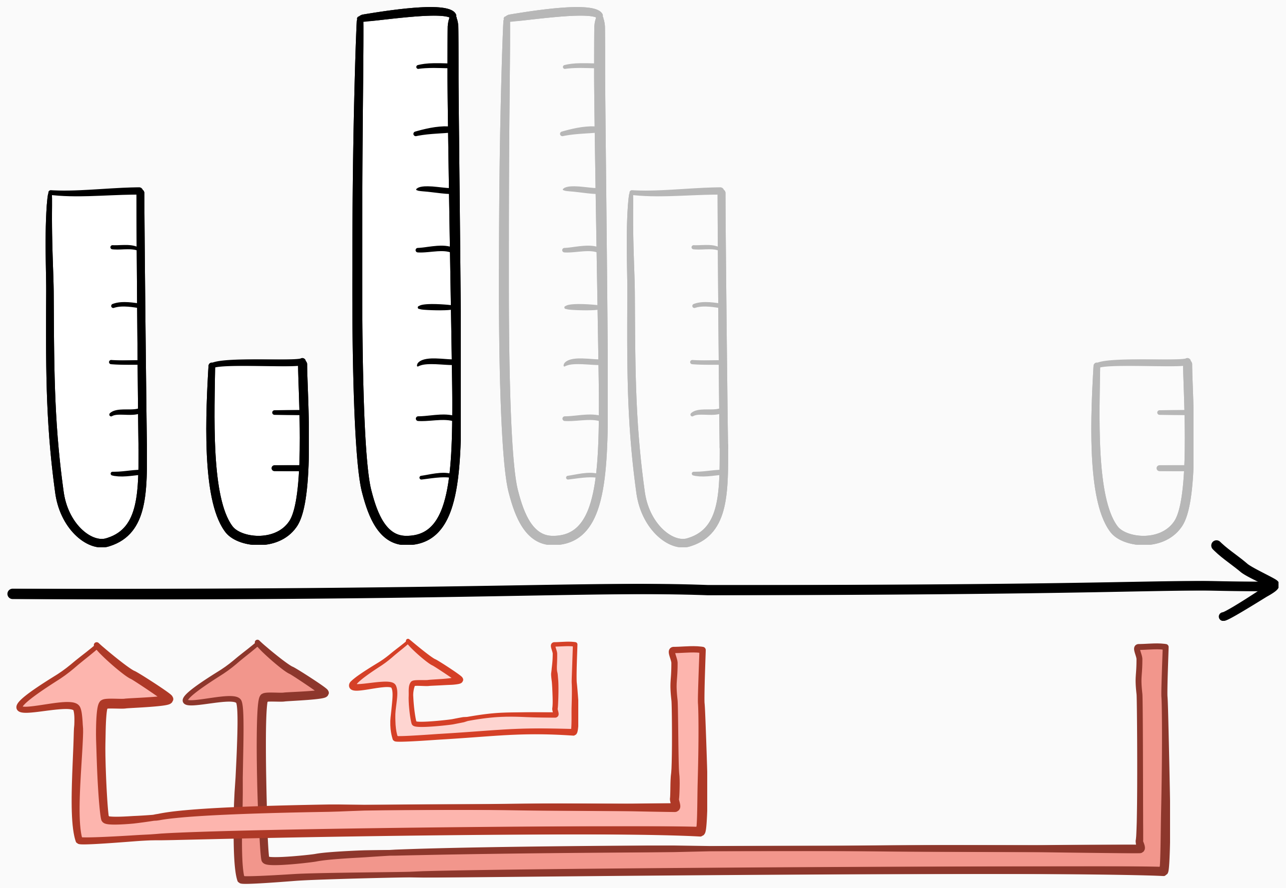


boosted arrival time

$= \text{arrival time} - b(\text{size})$



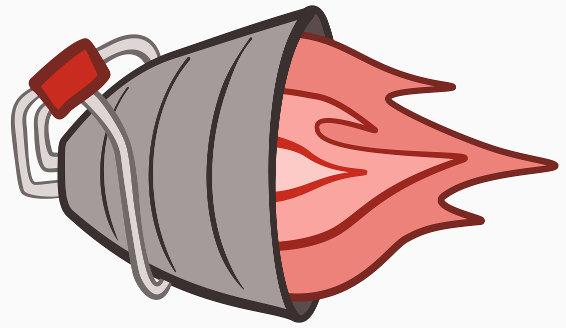
Boost policies



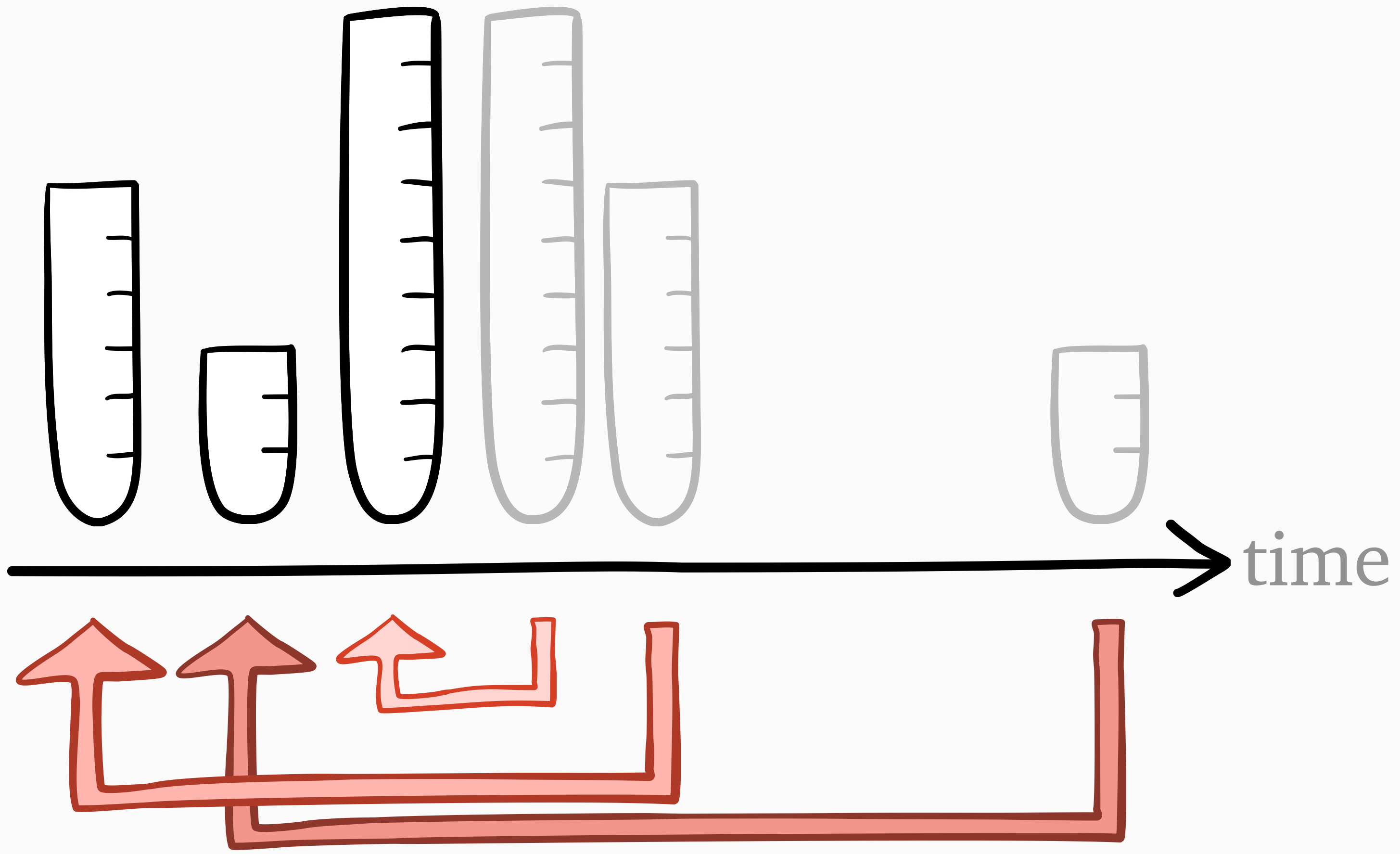
boosted arrival time

$= \text{arrival time} - \boldsymbol{b}(\text{size})$

Boost policies

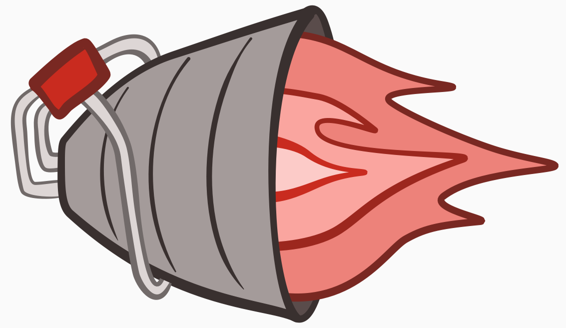


Scheduling rule: always serve job of
*minimum **boosted** arrival time*

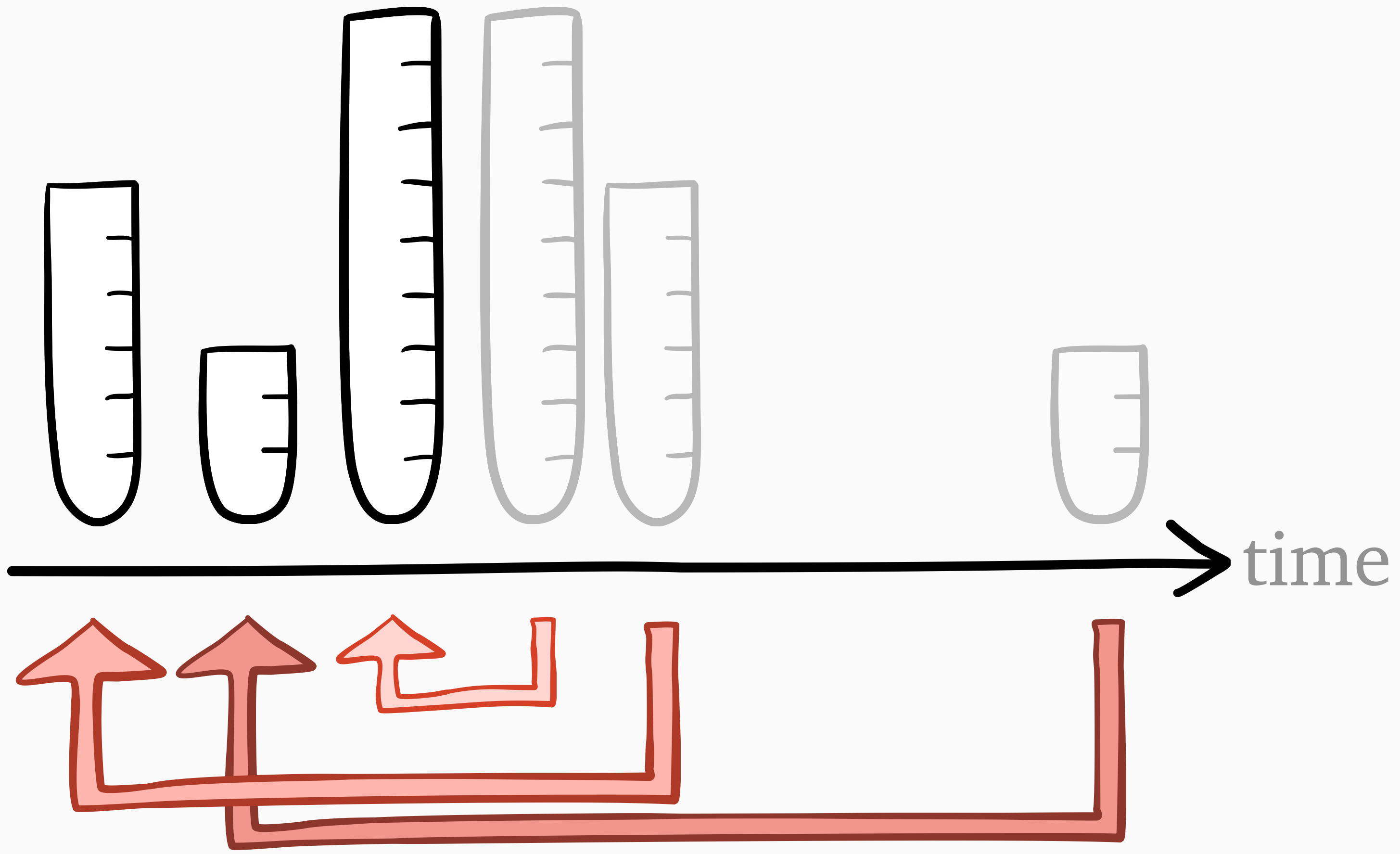


boosted arrival time
= arrival time − ***b***(size)

Boost policies



Scheduling rule: always serve job of
*minimum **boosted** arrival time*

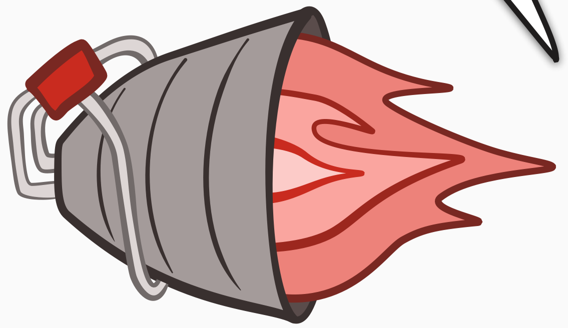


boosted arrival time
= arrival time − b(size)

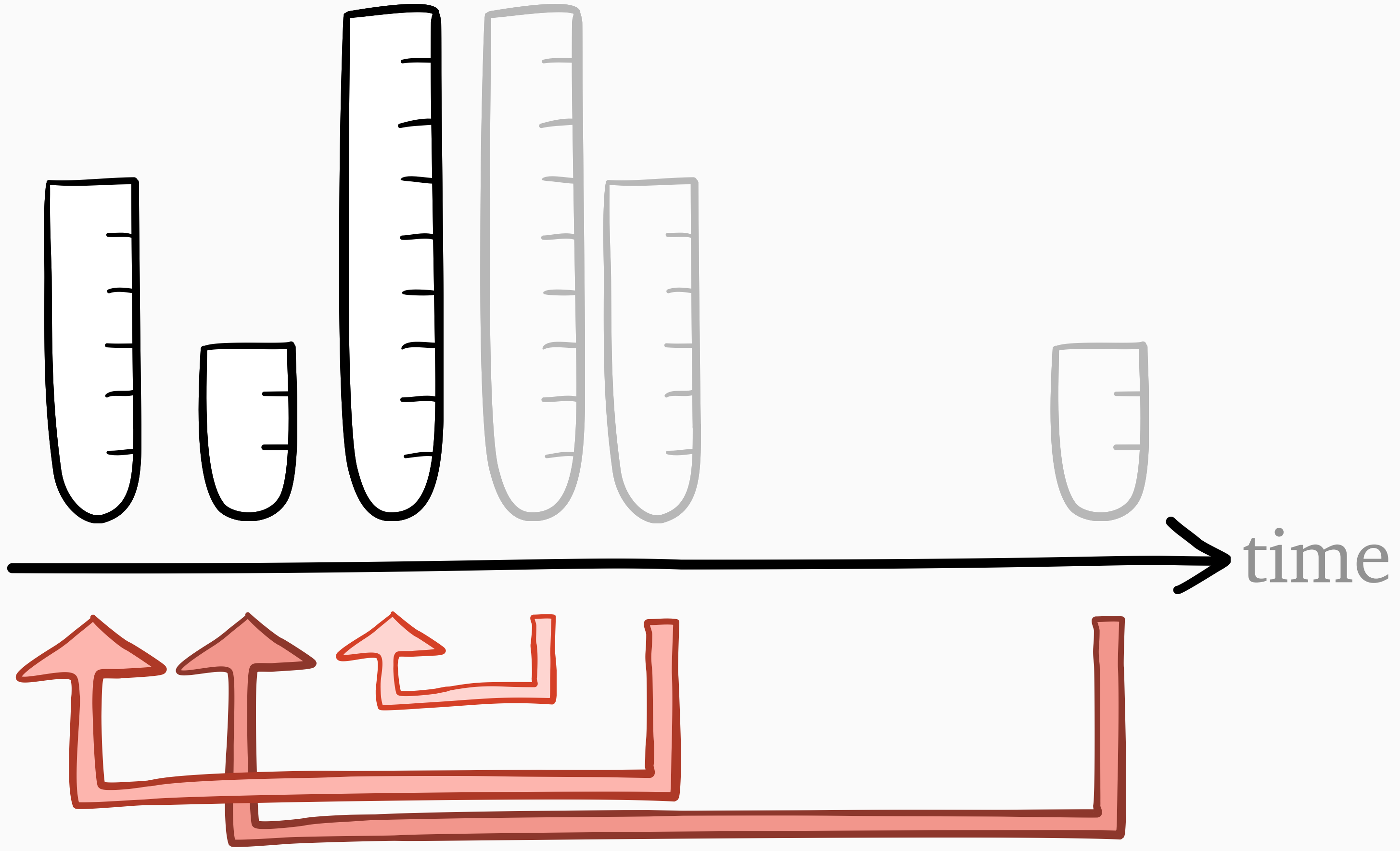
can vary choice of
boost function

Boost policies

can be preemptive
or nonpreemptive



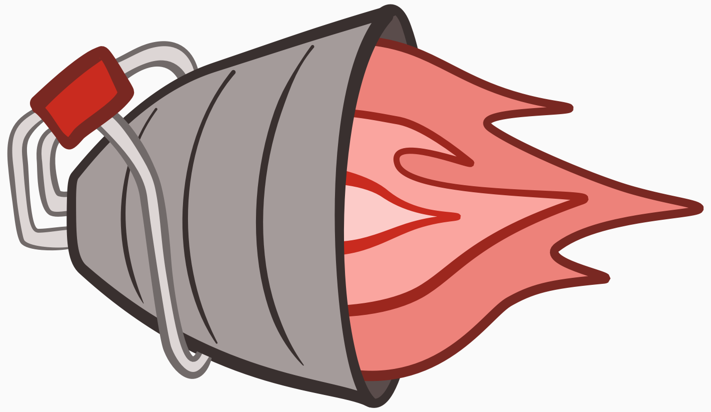
Scheduling rule: always serve job of
*minimum **boosted** arrival time*



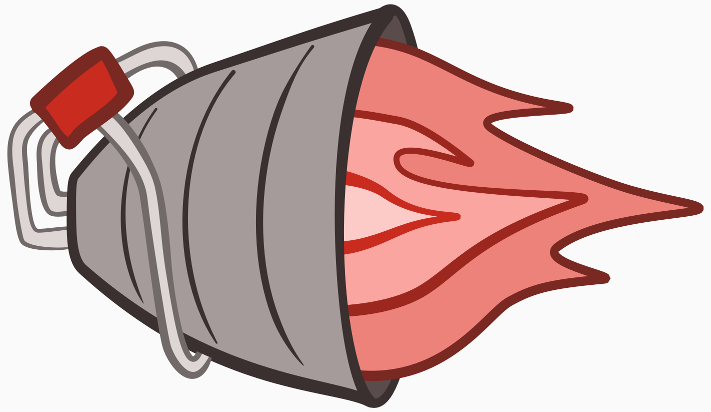
boosted arrival time
$= \text{arrival time} - \boldsymbol{b}(\text{size})$

can vary choice of
boost function

Boost

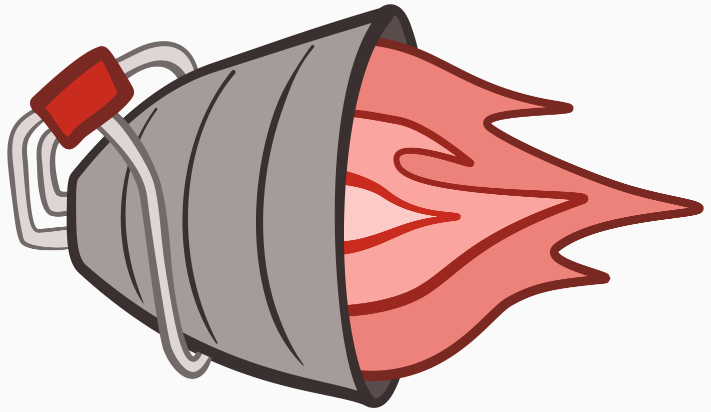


- Why did it take so long to beat FCFS?
- Why is achieving strong tail optimality hard?
- How does the **Boost** policy family work?
- How do we achieve strong tail optimality?

Boost 

- Why did it take so long to beat FCFS?
- Why is achieving strong tail optimality hard?
- How does the **Boost** policy family work?
- How do we achieve strong tail optimality?

Boost



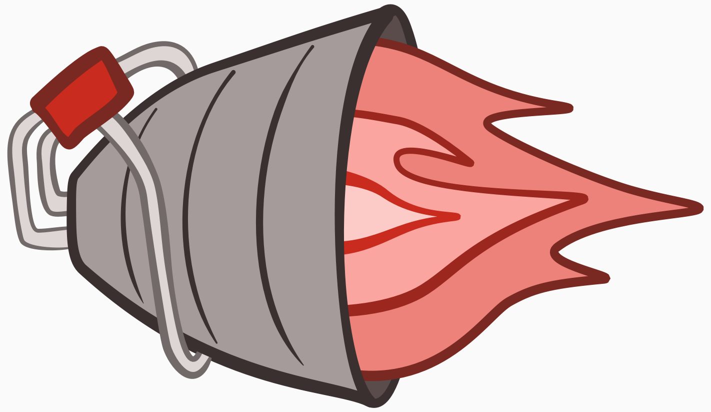
Why did it take so long to beat FCFS?

Why is achieving strong tail optimality hard?

How does the **Boost** policy family work?

How do we achieve strong tail optimality?

Boost



- Why did it take so long to beat FCFS?
- Why is achieving strong tail optimality hard?
- How does the **Boost** policy family work?
- How do we achieve strong tail optimality?

What's the right **boost** function?

Transforming the problem

$$C = \lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t]$$

Transforming the problem

$$C = \lim_{t \to \infty} e^{\gamma t} \mathbf{P}[T > t]$$

Finite batch problem?

Transforming the problem

$$C = \lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t] = \lim_{\theta\to\gamma} \frac{\gamma-\theta}{\gamma}\mathbf{E}[e^{\theta T}]$$

final value theorem

Finite batch problem?

Transforming the problem

$$C = \underbrace{\lim_{t \to \infty} e^{\gamma t} \mathbf{P}[T > t]}_{\text{“}\infty \cdot \mathbf{P}[T > \infty]\text{”}} = \lim_{\theta \to \gamma} \frac{\gamma - \theta}{\gamma} \mathbf{E}[e^{\theta T}]$$

Finite batch problem?

Transforming the problem

$$C = \underbrace{\lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t]}_{\text{“}\infty \cdot \mathbf{P}[T > \infty]\text{”}} = \lim_{\theta\to\gamma}\frac{\gamma-\theta}{\gamma}\mathbf{E}[e^{\theta T}]$$

Always zero in batch setting

Finite batch problem?

Transforming the problem

$$C = \underbrace{\lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t]}_{\text{“}\infty \cdot \mathbf{P}[T > \infty]\text{”}} = \underbrace{\lim_{\theta\to\gamma} \frac{\gamma-\theta}{\gamma}\mathbf{E}[e^{\theta T}]}_{\text{“}0 \cdot \mathbf{E}[e^{\gamma T}]\text{”}}$$

Always zero in batch setting

Finite batch problem?

Transforming the problem

$$C = \underbrace{\lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t]}_{\text{“}\infty \cdot \mathbf{P}[T > \infty]\text{”}} = \underbrace{\lim_{\theta\to\gamma} \frac{\gamma-\theta}{\gamma}\mathbf{E}[e^{\theta T}]}_{\text{“}0 \cdot \mathbf{E}[e^{\gamma T}]\text{”}}$$

Finite batch problem?

Always zero in batch setting

Makes sense in batch setting!

Transforming the problem

$$C = \underbrace{\lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t]}_{\text{“}\infty \cdot \mathbf{P}[T > \infty]\text{”}} = \underbrace{\lim_{\theta\to\gamma} \frac{\gamma-\theta}{\gamma}\mathbf{E}[e^{\theta T}]}_{\text{“}0 \cdot \mathbf{E}[e^{\gamma T}]\text{”}}$$

Finite batch problem?

Always zero in batch setting

Makes sense in batch setting!

$t_i = d_i - a_i$

$a_i =$ arrival time of job i

$d_i =$ departure time of job i

Transforming the problem

$$C = \underbrace{\lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t]}_{\text{“}\infty \cdot \mathbf{P}[T > \infty]\text{”}} = \underbrace{\lim_{\theta\to\gamma} \frac{\gamma-\theta}{\gamma}\mathbf{E}[e^{\theta T}]}_{\text{“}0 \cdot \mathbf{E}[e^{\gamma T}]\text{”}}$$

Always zero in batch setting

Makes sense in batch setting!

Finite batch problem?

$t_i = d_i - a_i$

a_i = arrival time of job i

d_i = departure time of job i

Batch problem: minimize

$$\mathbf{E}[e^{\gamma T}] = \frac{1}{n}\sum_{i=1}^{n} e^{\gamma t_i} = \frac{1}{n}\sum_{i=1}^{n} e^{-\gamma a_i} e^{\gamma d_i}$$

Transforming the problem

$$C = \underbrace{\lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t]}_{\text{“}\infty \cdot \mathbf{P}[T > \infty]\text{”}} = \underbrace{\lim_{\theta\to\gamma} \frac{\gamma - \theta}{\gamma}\mathbf{E}[e^{\theta T}]}_{\text{“}0 \cdot \mathbf{E}[e^{\gamma T}]\text{”}}$$

Finite batch problem?

Always zero in batch setting

Makes sense in batch setting!

almost classic problem

Batch problem: minimize

$t_i = d_i - a_i$

$a_i =$ arrival time of job i

$d_i =$ departure time of job i

$$\mathbf{E}[e^{\gamma T}] = \frac{1}{n}\sum_{i=1}^{n} e^{\gamma t_i} = \frac{1}{n}\sum_{i=1}^{n} e^{-\gamma a_i} e^{\gamma d_i}$$

Solving the batch problem

$$t_i = d_i - a_i$$
$$a_i = \text{arrival time of job } i$$
$$d_i = \text{departure time of job } i$$

Batch problem: minimize

$$\mathbf{E}[e^{\gamma T}] = \frac{1}{n}\sum_{i=1}^{n} e^{\gamma t_i} = \frac{1}{n}\sum_{i=1}^{n} e^{-\gamma a_i} e^{\gamma d_i}$$

Solving the batch problem

$t_i = d_i - a_i$

$a_i =$ arrival time of job i

$d_i =$ departure time of job i

Batch problem: minimize

$$\mathbf{E}[e^{\gamma T}] = \frac{1}{n}\sum_{i=1}^{n} e^{\gamma t_i} = \frac{1}{n}\sum_{i=1}^{n} e^{-\gamma a_i} e^{\gamma d_i}$$

Classic metric: mean weighted discounted departure time

$$\frac{1}{n}\sum_{i=1}^{n} w_i e^{-\theta d_i}$$

Solving the batch problem

$t_i = d_i - a_i$

$a_i =$ arrival time of job i

$d_i =$ departure time of job i

Batch problem: minimize

$$\mathbf{E}[e^{\gamma T}] = \frac{1}{n}\sum_{i=1}^{n} e^{\gamma t_i} = \frac{1}{n}\sum_{i=1}^{n} e^{-\gamma a_i} e^{\gamma d_i}$$

Classic metric: mean weighted discounted departure time

$$\frac{1}{n}\sum_{i=1}^{n} w_i e^{-\theta d_i}$$

Solving the batch problem

$t_i = d_i - a_i$

a_i = arrival time of job i

d_i = departure time of job i

can't start i before a_i

Batch problem: minimize

$$\mathbf{E}[e^{\gamma T}] = \frac{1}{n}\sum_{i=1}^{n} e^{\gamma t_i} = \frac{1}{n}\sum_{i=1}^{n} e^{-\gamma a_i} e^{\gamma d_i}$$

Classic metric: mean weighted discounted departure time

$$\frac{1}{n}\sum_{i=1}^{n} w_i e^{-\theta d_i}$$

Solving the batch problem

$t_i = d_i - a_i$

$a_i =$ arrival time of job i

$d_i =$ departure time of job i

can't start i before a_i

Batch problem: minimize

$$\mathbf{E}[e^{\gamma T}] = \frac{1}{n}\sum_{i=1}^{n} e^{\gamma t_i} = \frac{1}{n}\sum_{i=1}^{n} e^{-\gamma a_i} e^{\gamma d_i}$$

$\gamma > 0$

Classic metric: mean weighted discounted departure time

$$\frac{1}{n}\sum_{i=1}^{n} w_i e^{-\theta d_i}$$

Solving the batch problem

$t_i = d_i - a_i$

$a_i =$ arrival time of job i

$d_i =$ departure time of job i

can't start i before a_i

negative discount rate

Batch problem: minimize

$$\mathbf{E}[e^{\gamma T}] = \frac{1}{n}\sum_{i=1}^{n} e^{\gamma t_i} = \frac{1}{n}\sum_{i=1}^{n} e^{-\gamma a_i} e^{\gamma d_i}$$

$\gamma > 0$

Classic metric: mean weighted discounted departure time

$$\frac{1}{n}\sum_{i=1}^{n} w_i e^{-\theta d_i}$$

Solving the batch problem

$t_i = d_i - a_i$

$a_i =$ arrival time of job i

$d_i =$ departure time of job i

can't start i before a_i

negative discount rate

Batch problem: minimize

$$\mathbf{E}[e^{\gamma T}] = \frac{1}{n}\sum_{i=1}^{n} e^{\gamma t_i} = \frac{1}{n}\sum_{i=1}^{n} e^{-\gamma a_i} e^{\gamma d_i}$$

$\gamma > 0$

Classic metric: mean weighted discounted departure time

$$\frac{1}{n}\sum_{i=1}^{n} w_i e^{-\theta d_i}$$

Relaxation solved by (sign-flipped) WDSPT, which is **Boost** with

$$b(s) = \frac{1}{\gamma}\log\frac{1}{1-e^{-\gamma s}}$$

Solving the batch problem

$t_i = d_i - a_i$

$a_i =$ arrival time of job i

$d_i =$ departure time of job i

can't start i before a_i

negative *discount rate*

Batch problem: minimize

$$\mathbf{E}[e^{\gamma T}] = \frac{1}{n}\sum_{i=1}^{n} e^{\gamma t_i} = \frac{1}{n}\sum_{i=1}^{n} e^{-\gamma a_i} e^{\gamma d_i}$$

$\gamma > 0$

Classic metric: mean weighted discounted departure time

$$\frac{1}{n}\sum_{i=1}^{n} w_i e^{-\theta d_i}$$

Relaxation solved by (sign-flipped) WDSPT, which is **Boost** with

$$b(s) = \frac{1}{\gamma}\log\frac{1}{1-e^{-\gamma s}}$$

γ-Boost

Solving the batch problem

$t_i = d_i - a_i$

$a_i =$ arrival time of job i

$d_i =$ departure time of job i

can't start i before a_i

negative discount rate

Batch problem: minimize

$$\mathbf{E}[e^{\gamma T}] = \frac{1}{n}\sum_{i=1}^{n} e^{\gamma t_i} = \frac{1}{n}\sum_{i=1}^{n} e^{-\gamma a_i} e^{\gamma d_i}$$

$\gamma > 0$

Classic metric: mean weighted discounted departure time

$$\frac{1}{n}\sum_{i=1}^{n} w_i e^{-\theta d_i}$$

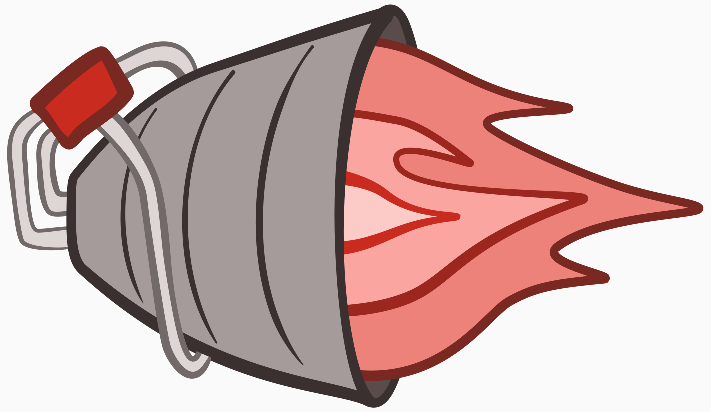
Relaxation solved by (sign-flipped) WDSPT, which is **Boost** with

$$b(s) = \frac{1}{\gamma}\log\frac{1}{1-e^{-\gamma s}}$$

γ-Boost

Unknown sizes: swap WDSPT for *Gittins*

Boost

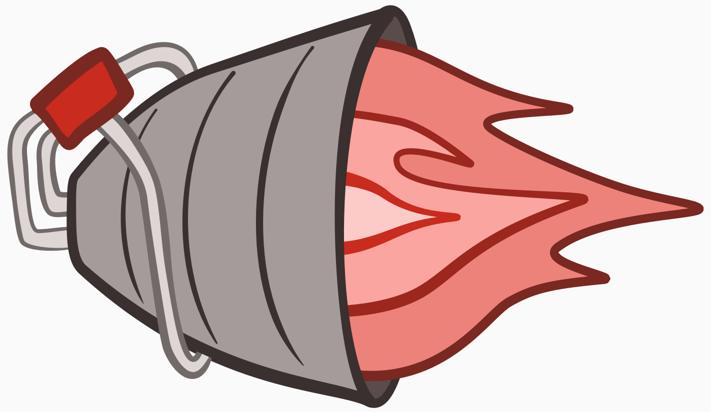


- Why did it take so long to beat FCFS?
- Why is achieving strong tail optimality hard?
- How does the **Boost** policy family work?
- How do we achieve strong tail optimality?

Boost

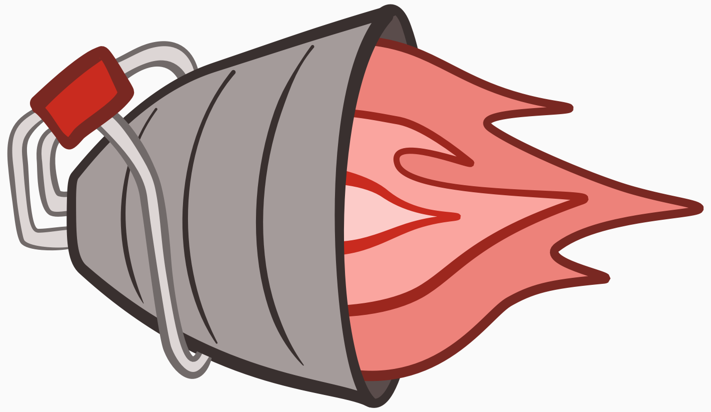
- Why did it take so long to beat FCFS?
- Why is achieving strong tail optimality hard?
- How does the **Boost** policy family work?
- How do we achieve strong tail optimality?

Boost



- Why did it take so long to beat FCFS?
- Why is achieving strong tail optimality hard?
- How does the **Boost** policy family work?
- How do we achieve strong tail optimality?

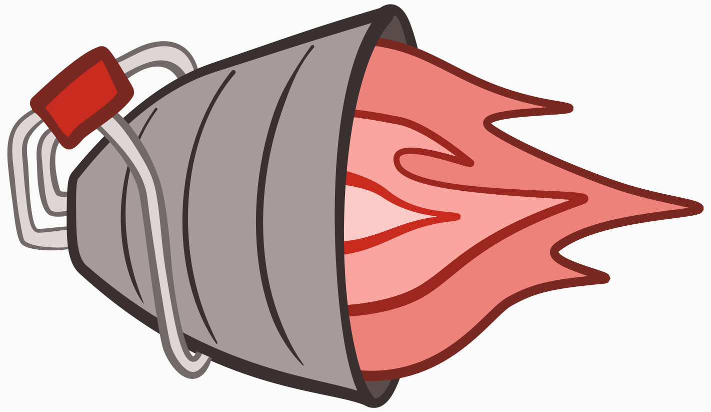
Boost



- Why did it take so long to beat FCFS?
- Why is achieving strong tail optimality hard?
- How does the **Boost** policy family work?
- How do we achieve strong tail optimality?

Should you use **Boost** today?

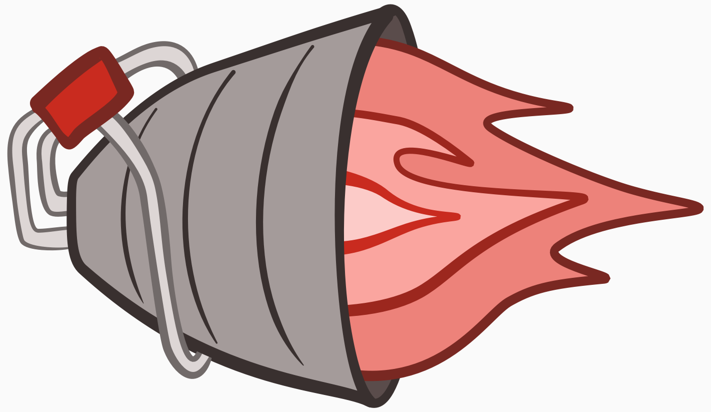
Boost



- Why did it take so long to beat FCFS?
- Why is achieving strong tail optimality hard?
- How does the **Boost** policy family work?
- How do we achieve strong tail optimality?

Should you use **Boost** today?

- Naturally *robust to noise*

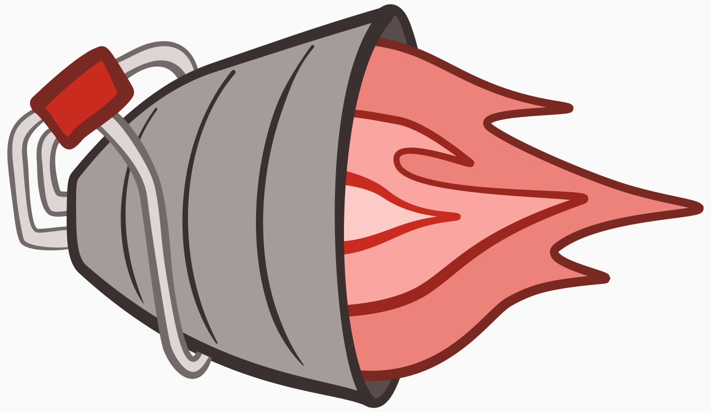
Boost 

- Why did it take so long to beat FCFS?
- Why is achieving strong tail optimality hard?
- How does the **Boost** policy family work?
- How do we achieve strong tail optimality?

Should you use **Boost** today?

- Naturally *robust to noise*
- Can adapt to *unknown job sizes* [Harlev et al., 2025]

Boost



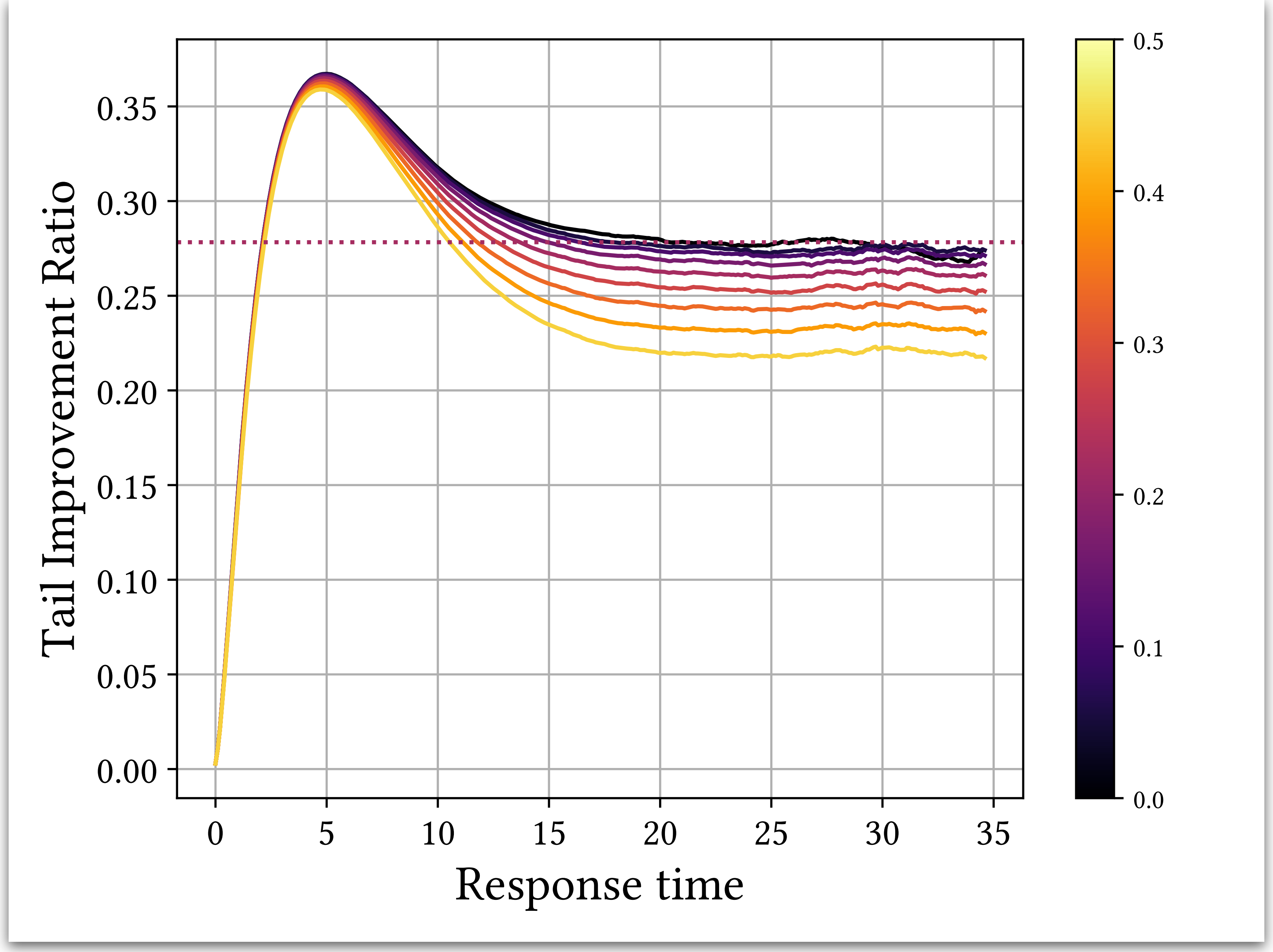
- Why did it take so long to beat FCFS?
- Why is achieving strong tail optimality hard?
- How does the **Boost** policy family work?
- How do we achieve strong tail optimality?

Should you use **Boost** today?

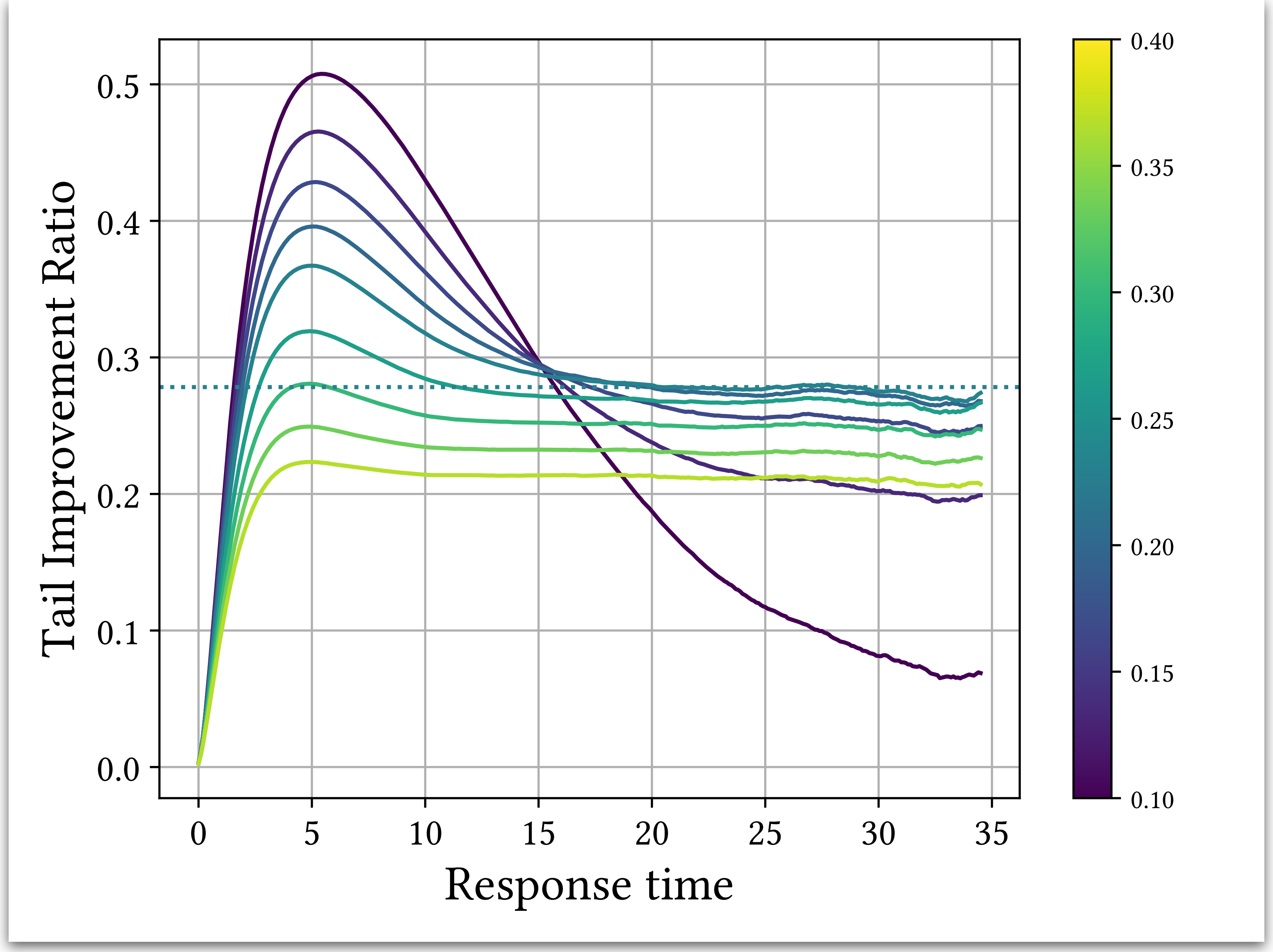
- Naturally *robust to noise*
- Can adapt to *unknown job sizes* [Harlev et al., 2025]
- Can adapt to *multiple servers* [Yu et al., 2026]

Boost is naturally robust

Noisy size information



Misspecified γ



Boost in multiserver systems

Boost in multiserver systems

Theorem: γ**-Boost** optimizes γ and C in heavy-traffic limit (load ≈ 1)

Boost in multiserver systems

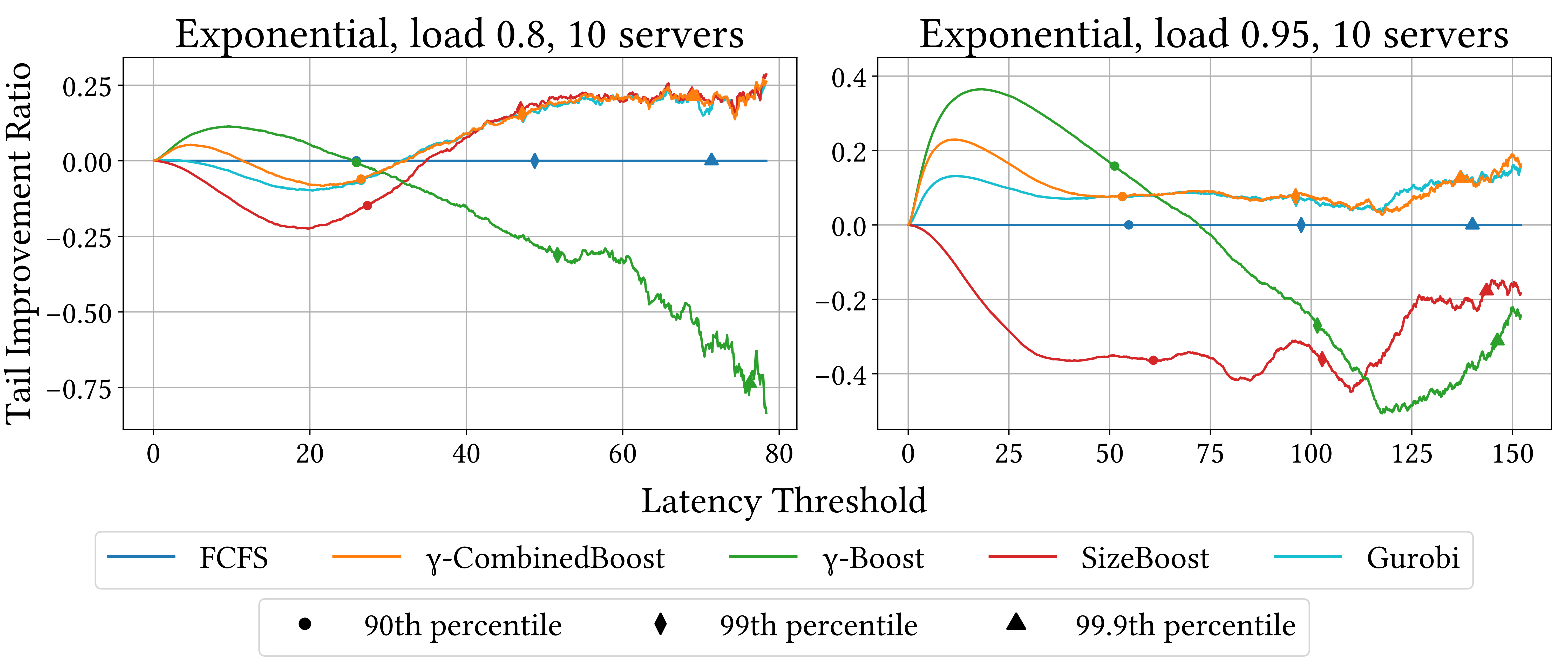
load < 1?

Theorem: γ-**Boost** optimizes γ and C in heavy-traffic limit (load ≈ 1)

Boost in multiserver systems

load < 1?

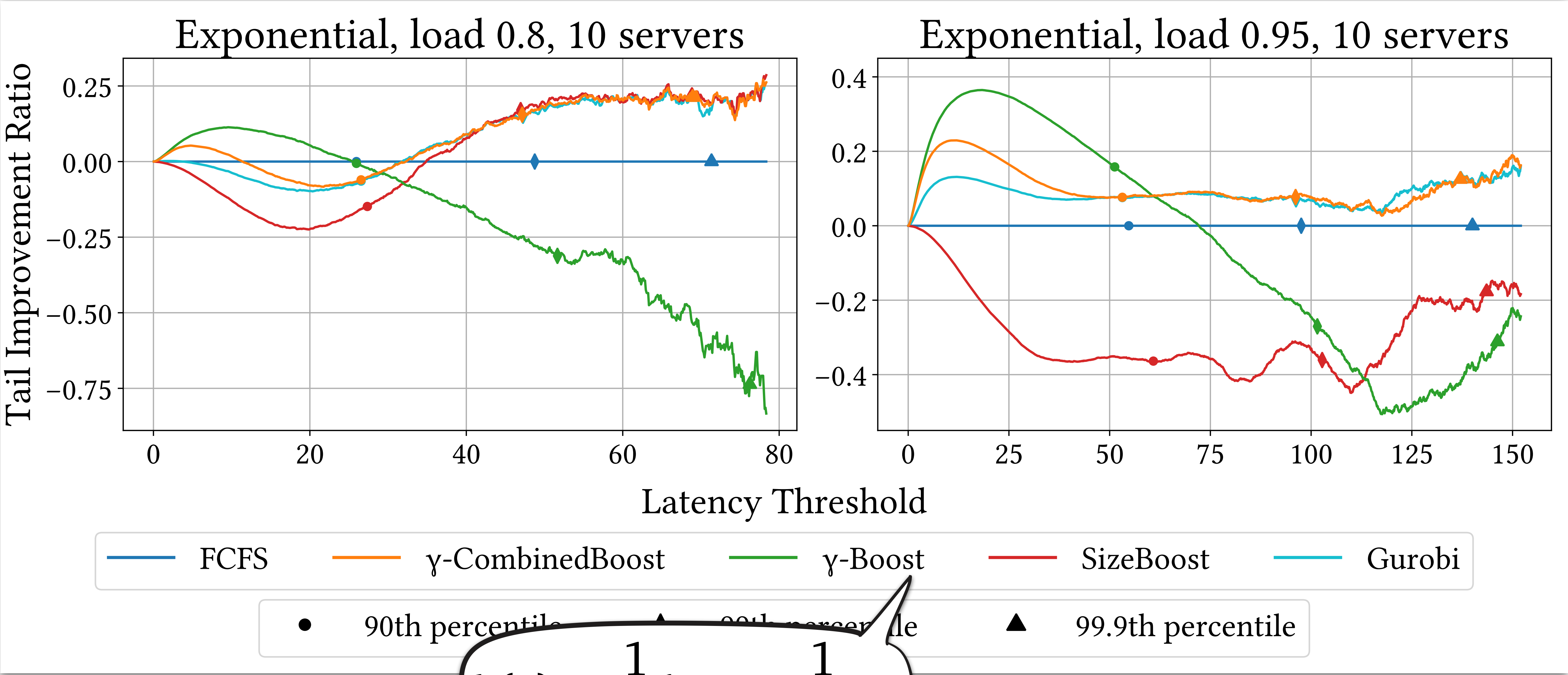
Theorem: γ-Boost optimizes γ and C in heavy-traffic limit (load $\approx$ 1)



Boost in multiserver systems

load < 1?

Theorem: γ-Boost optimizes γ and C in heavy-traffic limit (load $\approx$ 1)

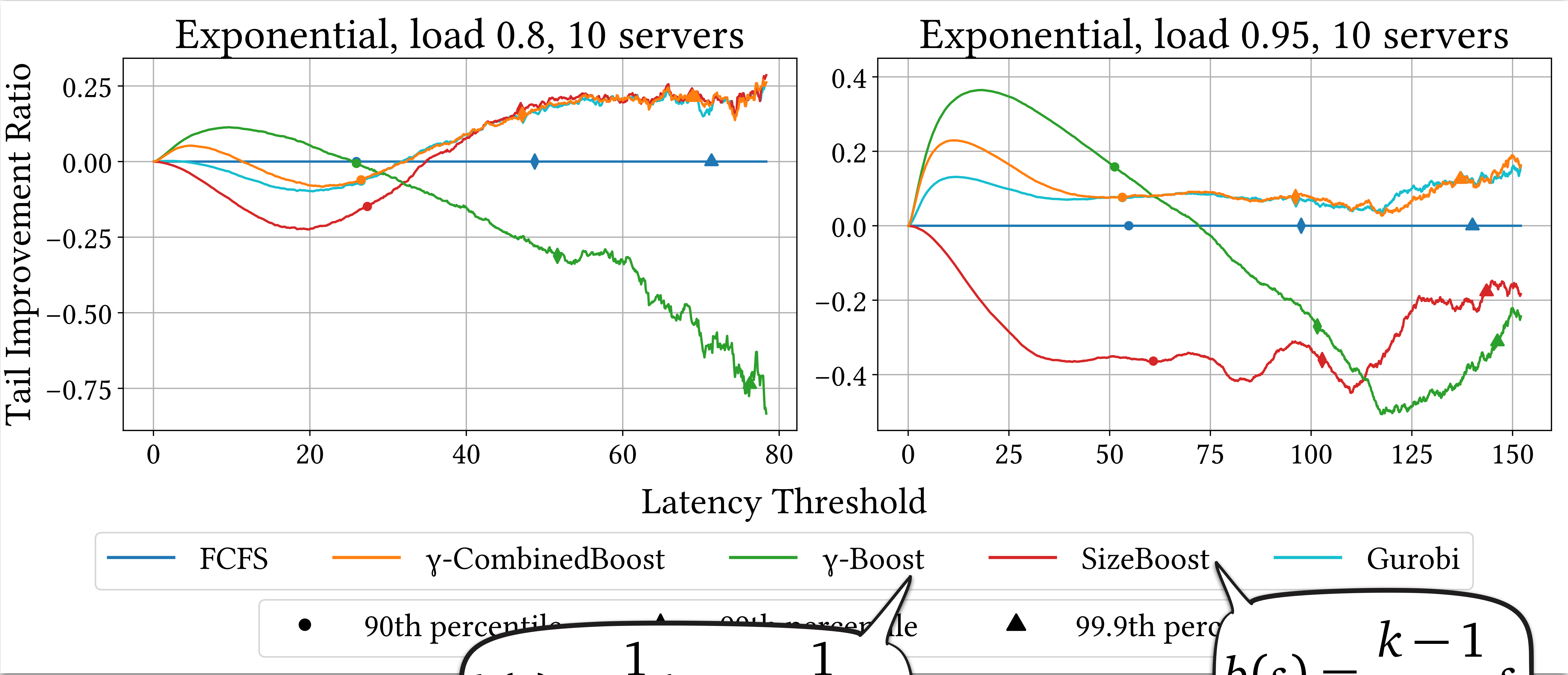


$$b(s) = \frac{1}{\gamma} \log \frac{1}{1 - e^{-\gamma s}}$$

Boost in multiserver systems

load < 1?

Theorem: γ-**Boost** optimizes γ and C in heavy-traffic limit (load $\approx$ 1)



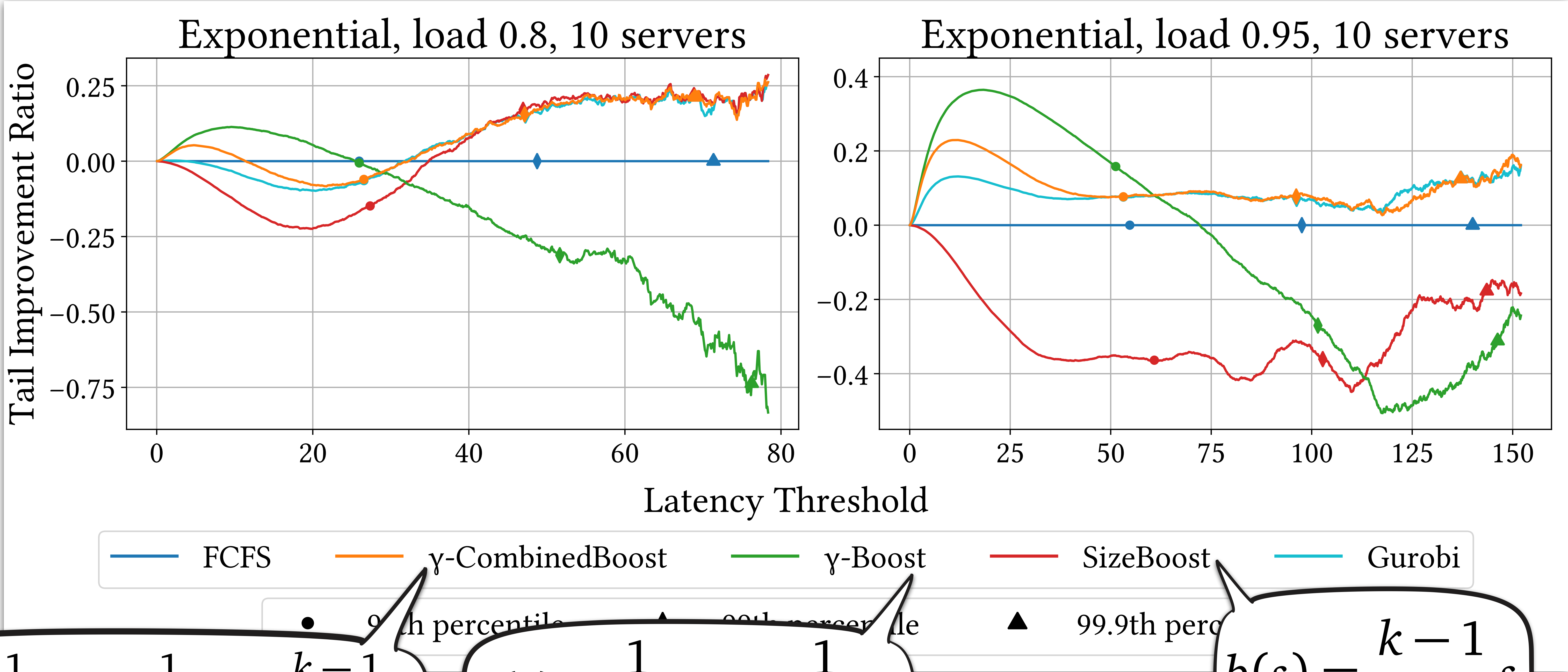
$$b(s) = \frac{1}{\gamma} \log \frac{1}{1 - e^{-\gamma s}}$$

$$b(s) = \frac{k-1}{k} s$$

Boost in multiserver systems

load < 1?

Theorem: γ-Boost optimizes γ and C in heavy-traffic limit (load $\approx$ 1)

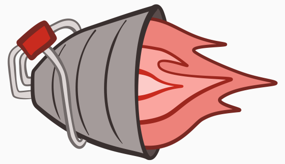


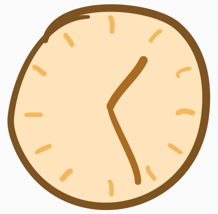
$$b(s) = \frac{1}{\gamma}\log\frac{1}{1-e^{-\gamma s}} + \frac{k-1}{k}s$$

$$b(s) = \frac{1}{\gamma}\log\frac{1}{1-e^{-\gamma s}}$$

$$b(s) = \frac{k-1}{k}s$$

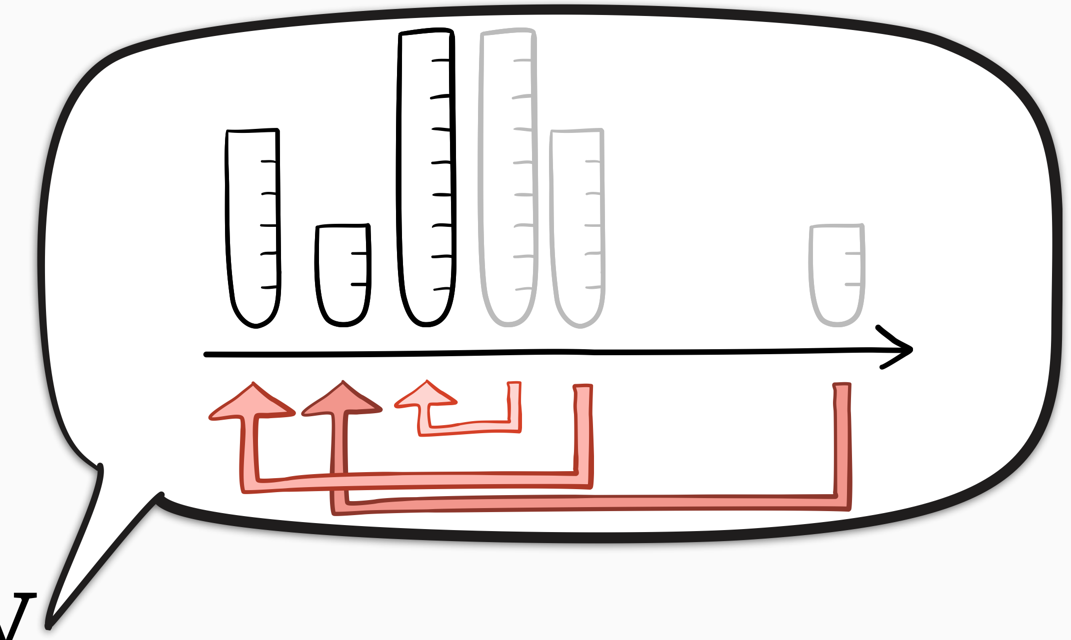
Our contributions:

 Design the **Boost** scheduling policy

 Analyze **Boost**'s performance

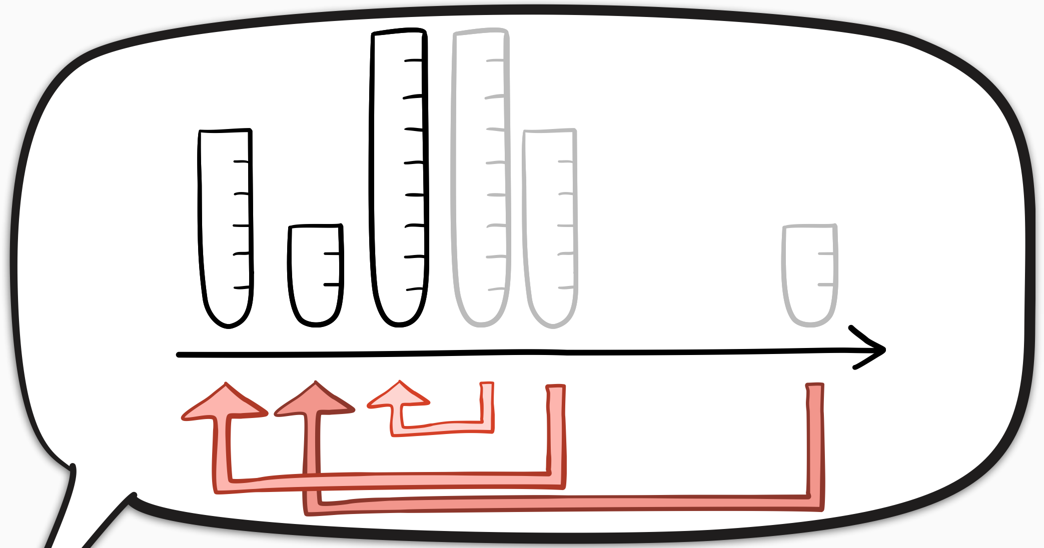
 Prove **Boost** is *strongly tail-optimal* for light-tailed sizes

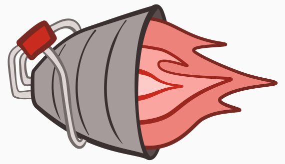
Our contributions:



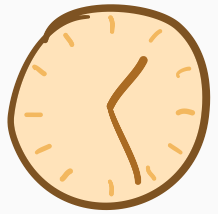
- Design the **Boost** scheduling policy
- Analyze **Boost**'s performance
- Prove **Boost** is *strongly tail-optimal* for light-tailed sizes

Our contributions:



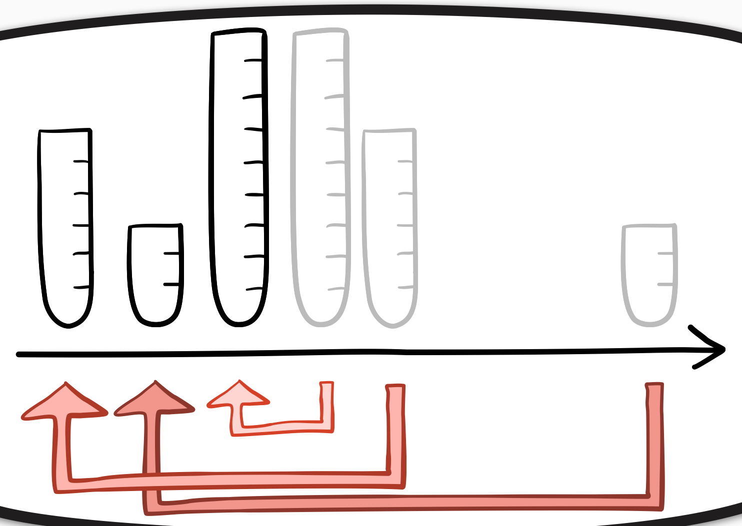
 Design the **Boost** scheduling policy

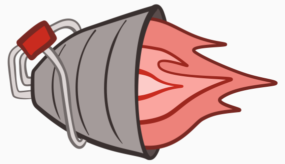
compute C_{Boost}

 Analyze **Boost**’s performance

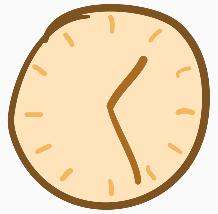
 Prove **Boost** is *strongly tail-optimal* for light-tailed sizes

Our contributions:



 Design the **Boost** scheduling policy

compute C_{Boost}

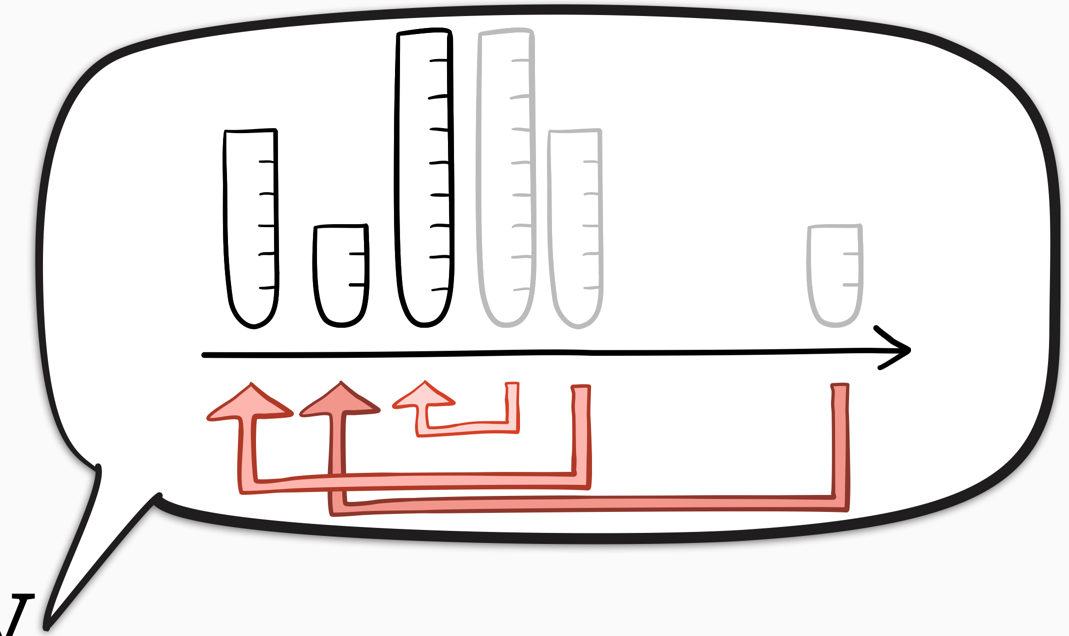
 Analyze **Boost**'s performance

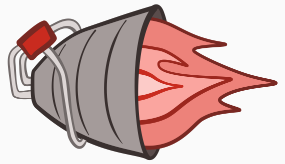
γ-Boost:

$$b(s) = \frac{1}{\gamma} \log \frac{1}{1 - e^{-\gamma s}}$$

 Prove **Boost** is *strongly tail-optimal* for light-tailed sizes

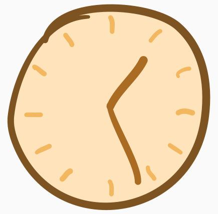
Our contributions:





Design the **Boost** scheduling policy

compute C_{Boost}

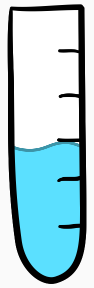


Analyze **Boost**'s performance

γ-Boost:

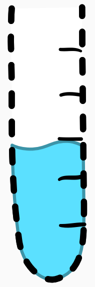
$$b(s) = \frac{1}{\gamma} \log \frac{1}{1 - e^{-\gamma s}}$$

Prove **Boost** is *strongly tail-optimal* for light-tailed sizes



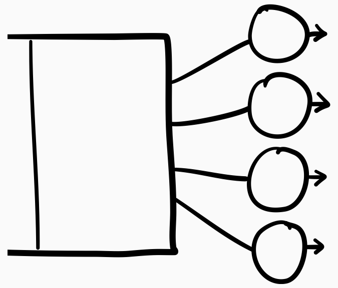
Known job sizes

Yu & Scully. *Strongly Tail-Optimal Scheduling in the Light-Tailed M/G/1*. SIGMETRICS 2024.



Unknown job sizes

Harlev, Yu, & Scully. *A Gittins Policy for Optimizing Tail Latency*. SIGMETRICS 2025.



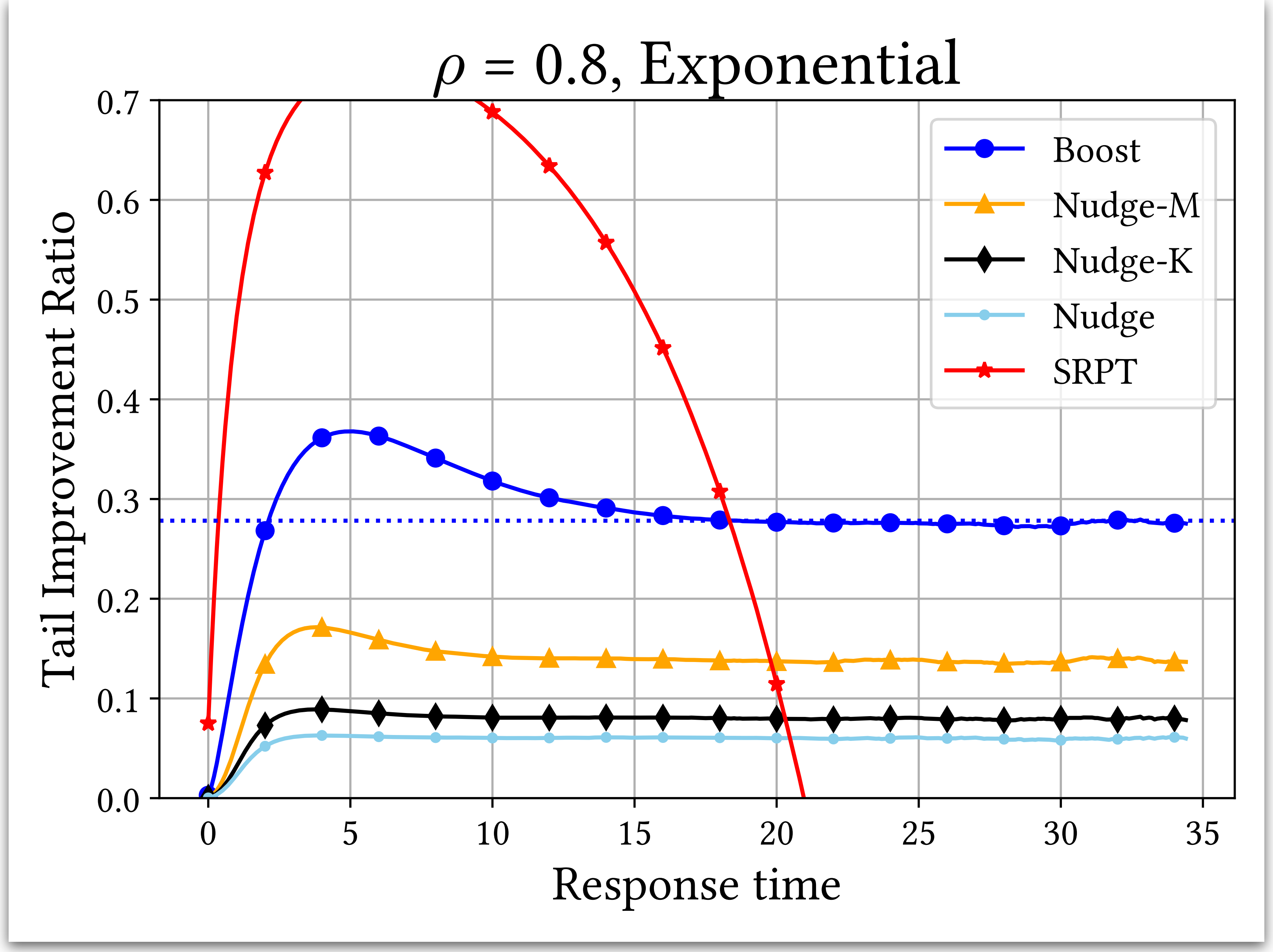
Multiple servers

Yu, Harlev, Adakroy, & Scully. *A Tale of Two Traffics: Optimizing Tail Latency in the Light-Tailed M/G/k*. SIGMETRICS 2026.

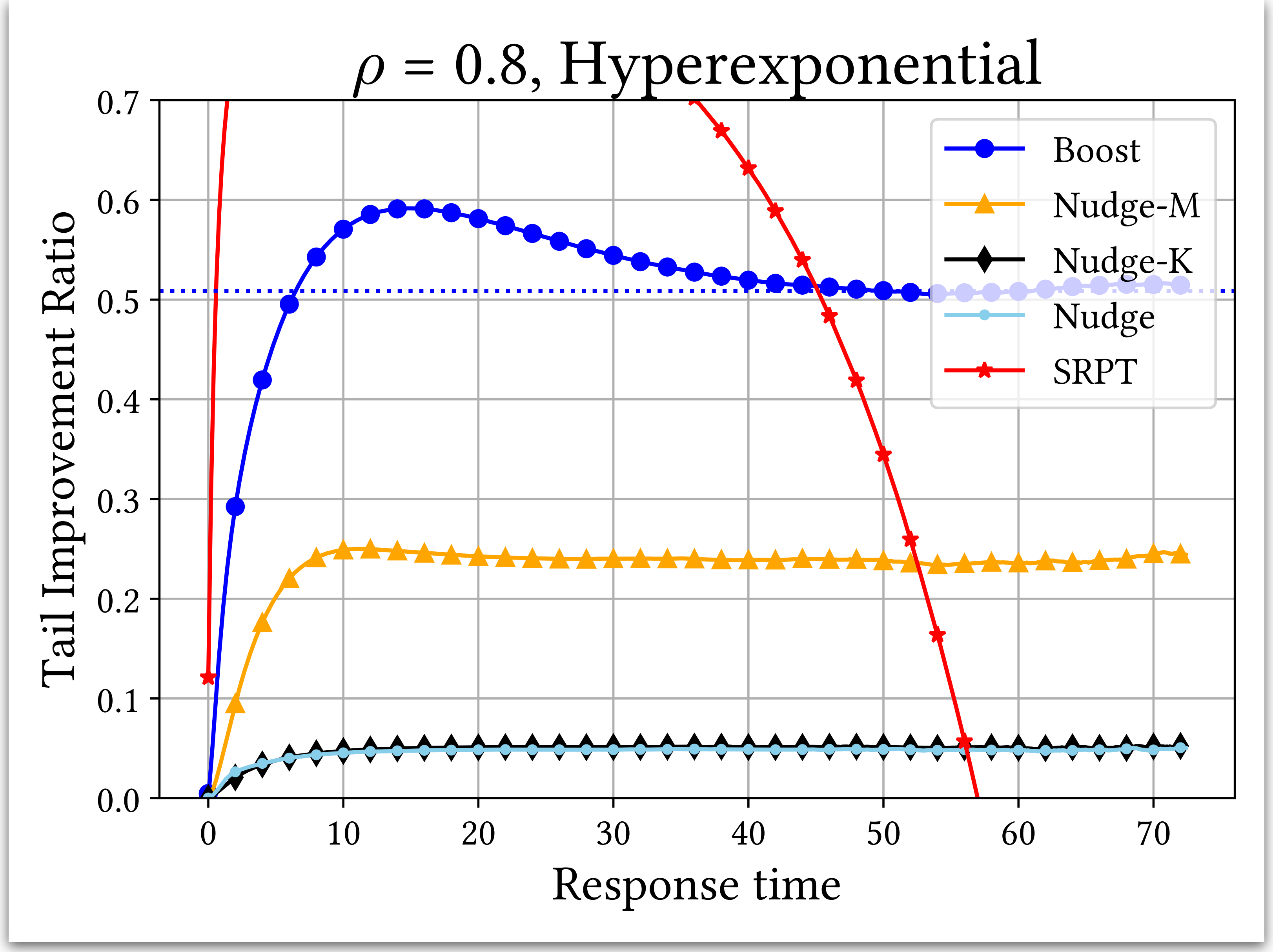
Bonus slides

Impact of job size variance

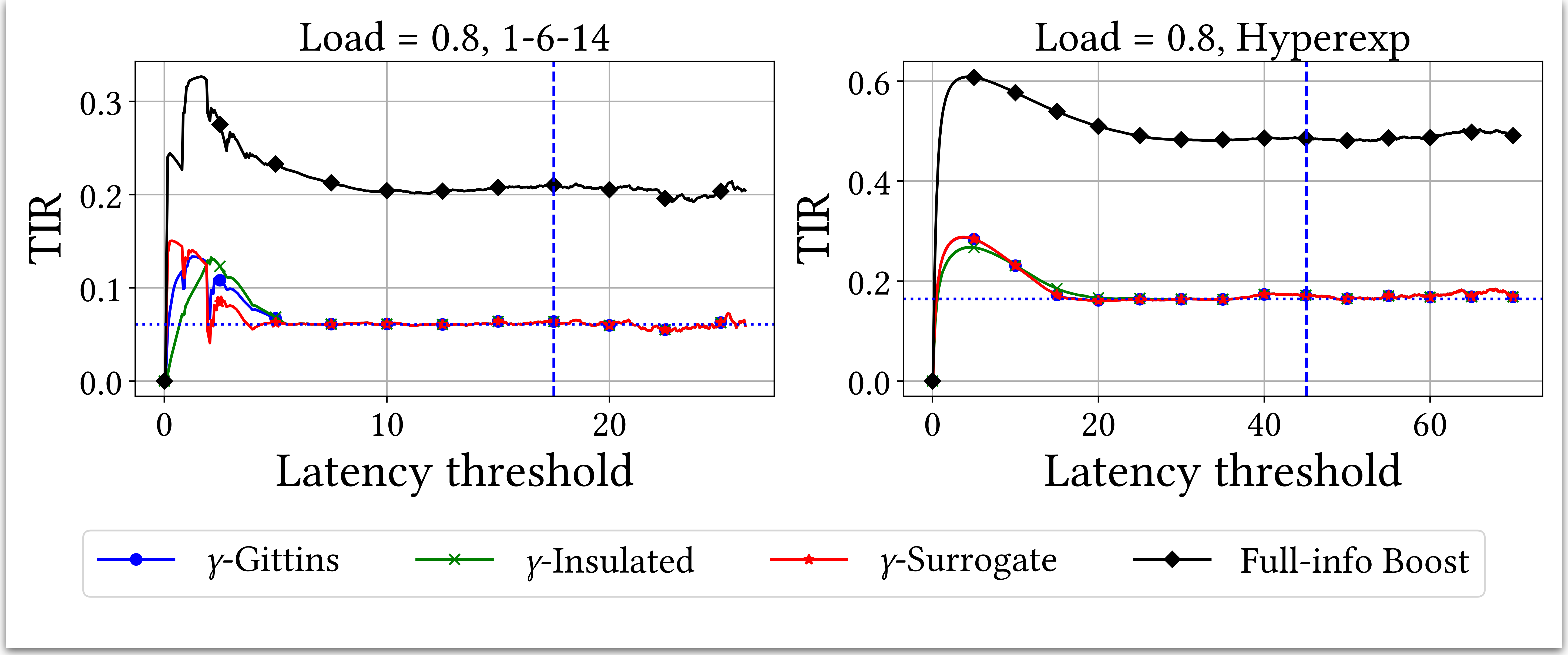
Low variance



High variance



Unknown job sizes



Why is **Boost** *weakly* tail optimal?

Why is **Boost** *weakly* tail optimal?

$\mathbf{P}[T > t] \sim Ce^{-\gamma t}$

Why is Boost *weakly* tail optimal?

$$\mathbf{P}[T > t] \sim Ce^{-\gamma t} \quad \Longrightarrow \quad C = \lim_{t \to \infty} e^{\gamma t}\mathbf{P}[T > t]$$

Why is **Boost** *weakly* tail optimal?

$$\mathbf{P}[T > t] \sim C e^{-\gamma t} \quad \Rightarrow \quad C = \lim_{t \to \infty} e^{\gamma t} \mathbf{P}[T > t] = \lim_{\theta \to \gamma} \frac{\gamma - \theta}{\gamma} \mathbf{E}[e^{\theta T}]$$

final value theorem

Why is **Boost** *weakly* tail optimal?

$$\mathbf{P}[T > t] \sim Ce^{-\gamma t} \quad \Rightarrow \quad C = \lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t] = \lim_{\theta\to\gamma} \frac{\gamma - \theta}{\gamma}\mathbf{E}[e^{\theta T}]$$

final value theorem

FCFS

$$T_{\text{FCFS}} = W + S$$

Why is **Boost** *weakly* tail optimal?

$$\mathbf{P}[T > t] \sim Ce^{-\gamma t} \quad \Rightarrow \quad C = \lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t] = \lim_{\theta\to\gamma} \frac{\gamma - \theta}{\gamma}\mathbf{E}[e^{\theta T}]$$

final value theorem

FCFS

$$T_{\text{FCFS}} = W + S$$

work

Why is **Boost** *weakly* tail optimal?

$$\mathbf{P}[T > t] \sim Ce^{-\gamma t} \quad \Rightarrow \quad C = \lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t] = \lim_{\theta\to\gamma} \frac{\gamma-\theta}{\gamma}\mathbf{E}[e^{\theta T}]$$

final value theorem

FCFS

$$T_{\text{FCFS}} = W + S$$

work

$$\Downarrow$$

$$C_{\text{FCFS}} = C_W \mathbf{E}[e^{\gamma S}]$$

Why is Boost *weakly* tail optimal?

$$\mathbf{P}[T > t] \sim C e^{-\gamma t} \quad \Rightarrow \quad C = \lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t] = \lim_{\theta\to\gamma} \frac{\gamma - \theta}{\gamma}\mathbf{E}[e^{\theta T}]$$

final value theorem

FCFS

$$T_{\text{FCFS}} = W + S$$

work

$$C_{\text{FCFS}} = C_W \mathbf{E}[e^{\gamma S}]$$

$$\lim_{t\to\infty} e^{\gamma t}\mathbf{P}[W > t]$$

Why is Boost *weakly* tail optimal?

$$\mathbf{P}[T > t] \sim C e^{-\gamma t} \quad \Rightarrow \quad C = \lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t] = \lim_{\theta\to\gamma} \frac{\gamma-\theta}{\gamma}\mathbf{E}[e^{\theta T}]$$

final value theorem

FCFS

$$T_{\text{FCFS}} = W + S$$

work

$$\Downarrow$$

$$C_{\text{FCFS}} = C_W \mathbf{E}[e^{\gamma S}]$$

$$\lim_{t\to\infty} e^{\gamma t}\mathbf{P}[W > t]$$

Boost

$$T_{\text{Boost}} \approx W + S - b(S) + V$$

Why is Boost *weakly* tail optimal?

$$\mathbf{P}[T > t] \sim Ce^{-\gamma t} \quad \Rightarrow \quad C = \lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t] = \lim_{\theta\to\gamma} \frac{\gamma-\theta}{\gamma}\mathbf{E}[e^{\theta T}]$$

final value theorem

FCFS

$$T_{\text{FCFS}} = W + S$$

work

$$C_{\text{FCFS}} = C_W \mathbf{E}[e^{\gamma S}]$$

$$\lim_{t\to\infty} e^{\gamma t}\mathbf{P}[W > t]$$

Boost

boost function

$$T_{\text{Boost}} \approx W + S - b(S) + V$$

Why is Boost *weakly* tail optimal?

$$\mathbf{P}[T > t] \sim Ce^{-\gamma t} \quad \Rightarrow \quad C = \lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t] = \lim_{\theta\to\gamma} \frac{\gamma-\theta}{\gamma}\mathbf{E}[e^{\theta T}]$$

final value theorem

FCFS

$$T_{\text{FCFS}} = W + S$$

work

$$C_{\text{FCFS}} = C_W \mathbf{E}[e^{\gamma S}]$$

$$\lim_{t\to\infty} e^{\gamma t}\mathbf{P}[W > t]$$

Boost

boost function

$$T_{\text{Boost}} \approx W + S - b(S) + V$$

crossing work

Why is **Boost** *weakly* tail optimal?

$$\mathbf{P}[T > t] \sim C e^{-\gamma t}$$

$$C = \lim_{t \to \infty} e^{\gamma t} \mathbf{P}[T > t] = \lim_{\theta \to \gamma} \frac{\gamma - \theta}{\gamma} \mathbf{E}[e^{\theta T}]$$

final value theorem

FCFS

$$T_{\text{FCFS}} = W + S$$

work

$$C_{\text{FCFS}} = C_W \mathbf{E}[e^{\gamma S}]$$

$$\lim_{t \to \infty} e^{\gamma t} \mathbf{P}[W > t]$$

Boost

boost function

$$T_{\text{Boost}} \approx W + S - b(S) + V$$

crossing work

$$C_{\text{Boost}} = C_W \mathbf{E}[e^{\gamma(S - b(S))}]\, \mathbf{E}[e^{\gamma V}]$$

Why is Boost *weakly* tail optimal?

$$\mathbf{P}[T > t] \sim Ce^{-\gamma t} \quad \Rightarrow \quad C = \lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t] = \lim_{\theta\to\gamma} \frac{\gamma-\theta}{\gamma}\mathbf{E}[e^{\theta T}]$$

final value theorem

FCFS

$$T_{\text{FCFS}} = W + S$$

work

$$C_{\text{FCFS}} = C_W \mathbf{E}[e^{\gamma S}]$$

$$\lim_{t\to\infty} e^{\gamma t}\mathbf{P}[W > t]$$

Boost

boost function

$$T_{\text{Boost}} \approx W + S - b(S) + V$$

crossing work

$$C_{\text{Boost}} = C_W \mathbf{E}[e^{\gamma(S-b(S))}]\,\mathbf{E}[e^{\gamma V}]$$

? finite?

Why is Boost *weakly* tail optimal?

$$\mathbf{P}[T > t] \sim C e^{-\gamma t} \quad \Rightarrow \quad C = \lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t] = \lim_{\theta\to\gamma} \frac{\gamma-\theta}{\gamma}\mathbf{E}[e^{\theta T}]$$

final value theorem

Boost

boost function

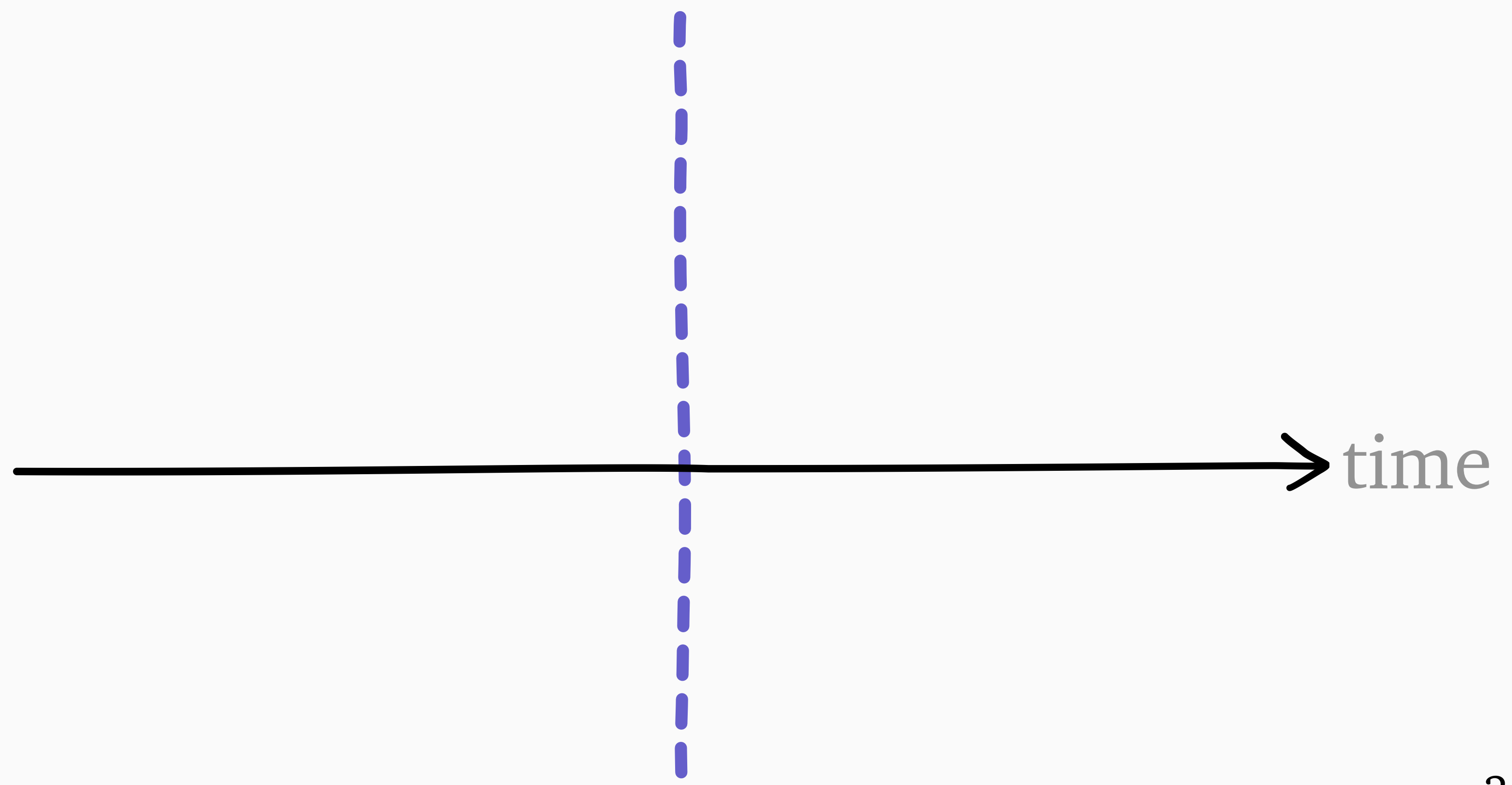
$$T_{\text{Boost}} \approx W + S - b(S) + V$$

crossing work

$$C_{\text{Boost}} = C_W \mathbf{E}[e^{\gamma(S-b(S))}]\,\mathbf{E}[e^{\gamma V}]$$

finite?

$V =$ **crossing work**



Why is **Boost** *weakly* tail optimal?

$$\mathbf{P}[T > t] \sim Ce^{-\gamma t} \quad \Rightarrow \quad C = \lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t] = \lim_{\theta\to\gamma} \frac{\gamma-\theta}{\gamma}\mathbf{E}[e^{\theta T}]$$

final value theorem

Boost

boost function

$$T_{\text{Boost}} \approx W + S - b(S) + V$$

crossing work

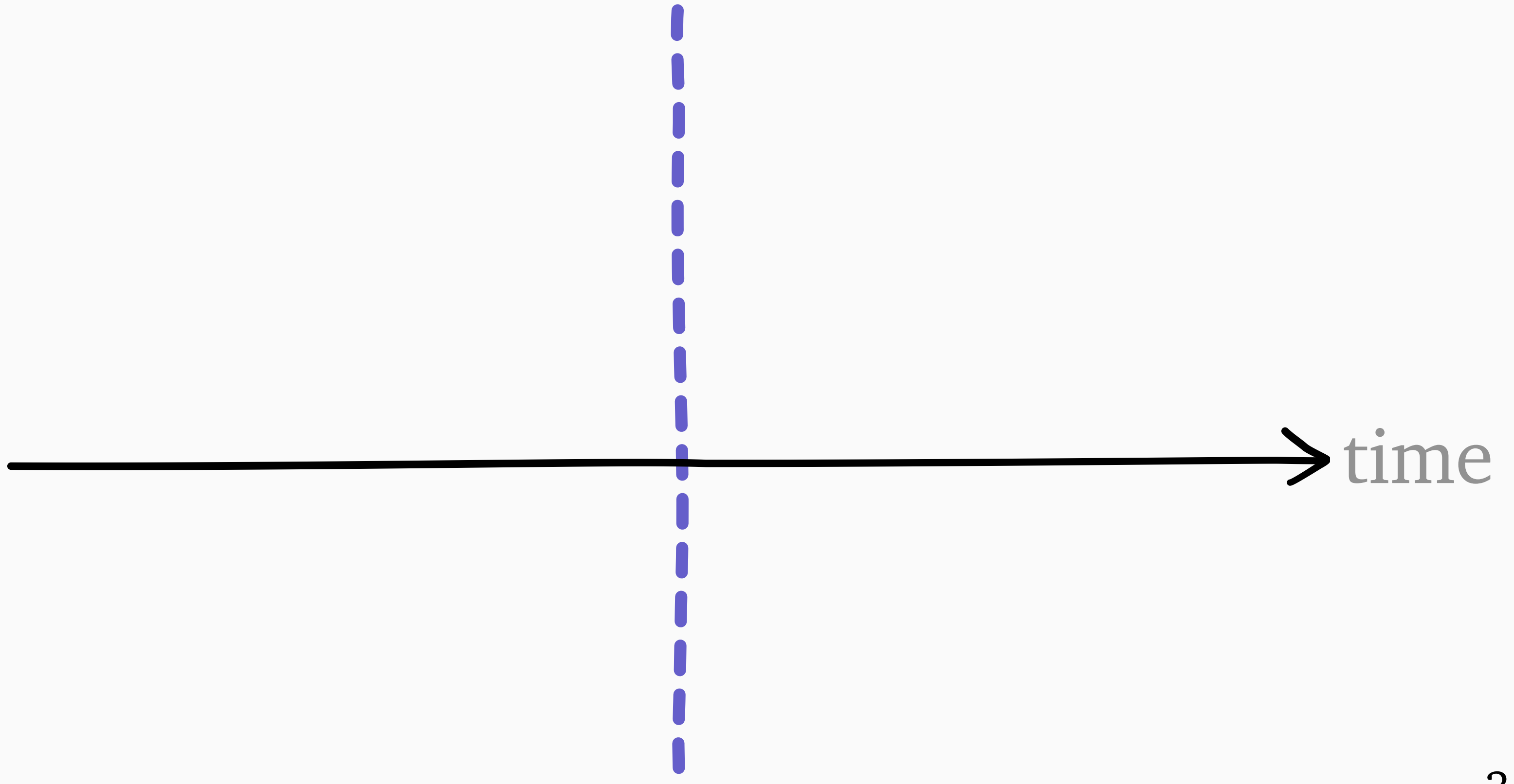
$$\Downarrow$$

$$C_{\text{Boost}} = C_W \mathbf{E}[e^{\gamma(S-b(S))}]\mathbf{E}[e^{\gamma V}]$$

? finite?

V = **crossing work**

work that "boosts past" a given time



Why is **Boost** *weakly* tail optimal?

$$\mathbf{P}[T > t] \sim Ce^{-\gamma t} \quad \Rightarrow \quad C = \lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t] = \lim_{\theta\to\gamma} \frac{\gamma-\theta}{\gamma}\mathbf{E}[e^{\theta T}]$$

final value theorem

Boost

boost function

$$T_{\text{Boost}} \approx W + S - b(S) + V$$

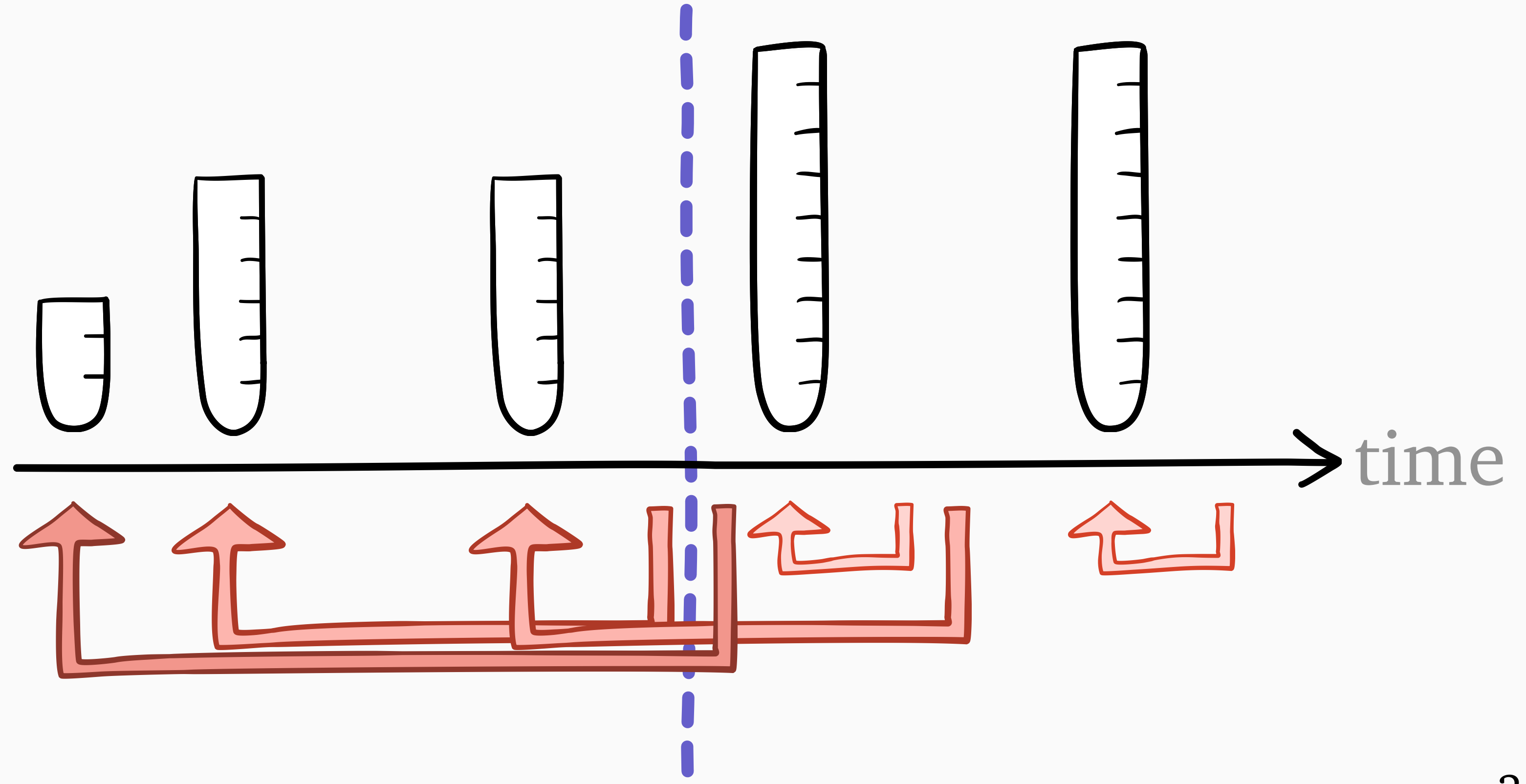
crossing work

$$C_{\text{Boost}} = C_W \mathbf{E}[e^{\gamma(S-b(S))}]\mathbf{E}[e^{\gamma V}]$$

finite?

V = **crossing work**

work that “boosts past” a given time



Why is **Boost** *weakly* tail optimal?

$$\mathbf{P}[T > t] \sim Ce^{-\gamma t}$$

$$C = \lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t] = \lim_{\theta\to\gamma} \frac{\gamma - \theta}{\gamma}\mathbf{E}[e^{\theta T}]$$

final value theorem

Boost

boost function

$$T_{\text{Boost}} \approx W + S - b(S) + V$$

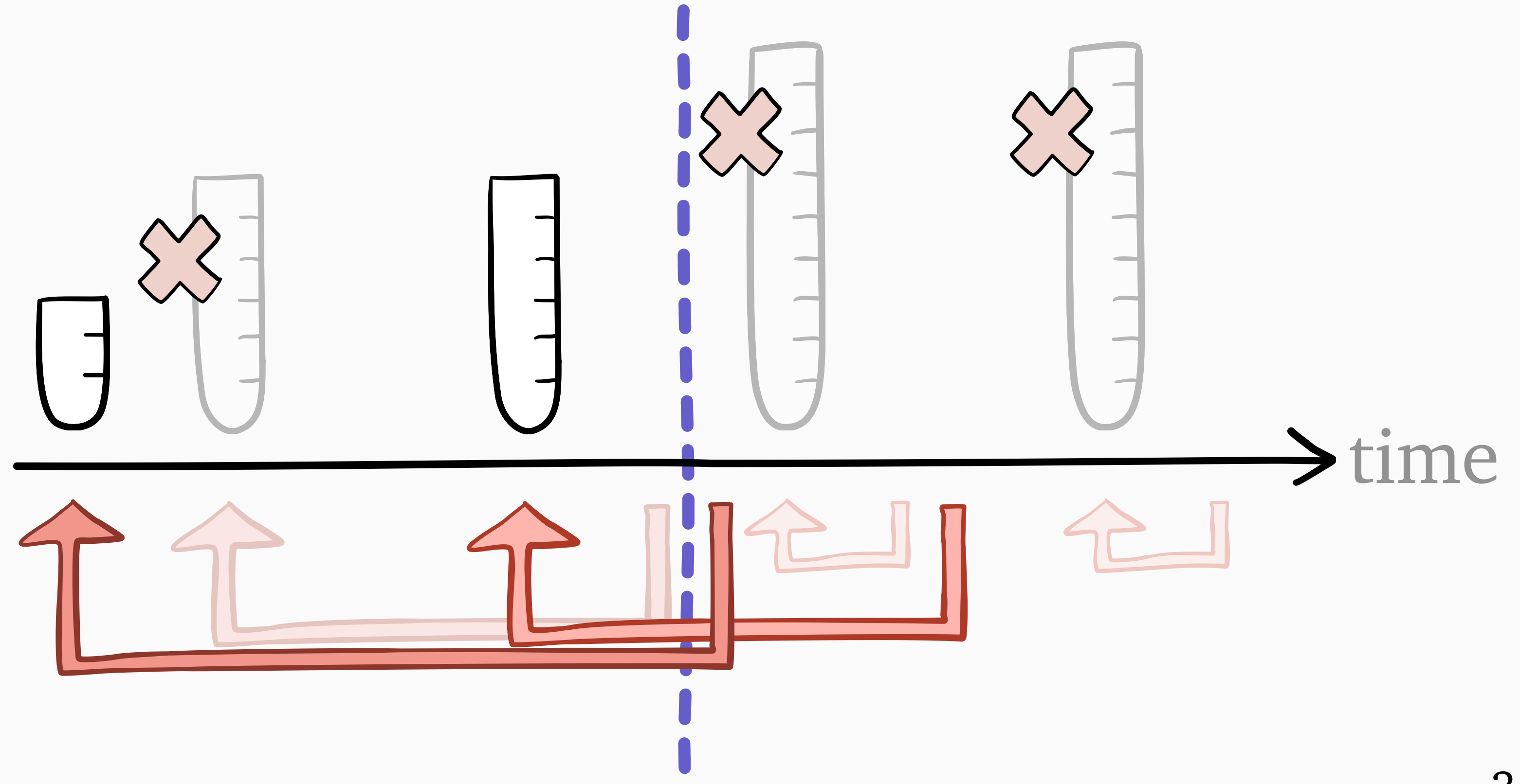
crossing work

$$C_{\text{Boost}} = C_W \mathbf{E}[e^{\gamma(S-b(S))}]\mathbf{E}[e^{\gamma V}]$$

finite?

V = crossing work

work that “boosts past” a given time



Why is Boost *weakly* tail optimal?

$$\mathbf{P}[T > t] \sim Ce^{-\gamma t} \quad \Rightarrow \quad C = \lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t] = \lim_{\theta\to\gamma} \frac{\gamma-\theta}{\gamma}\mathbf{E}[e^{\theta T}]$$

final value theorem

Boost

boost function

$$T_{\text{Boost}} \approx W + S - b(S) + V$$

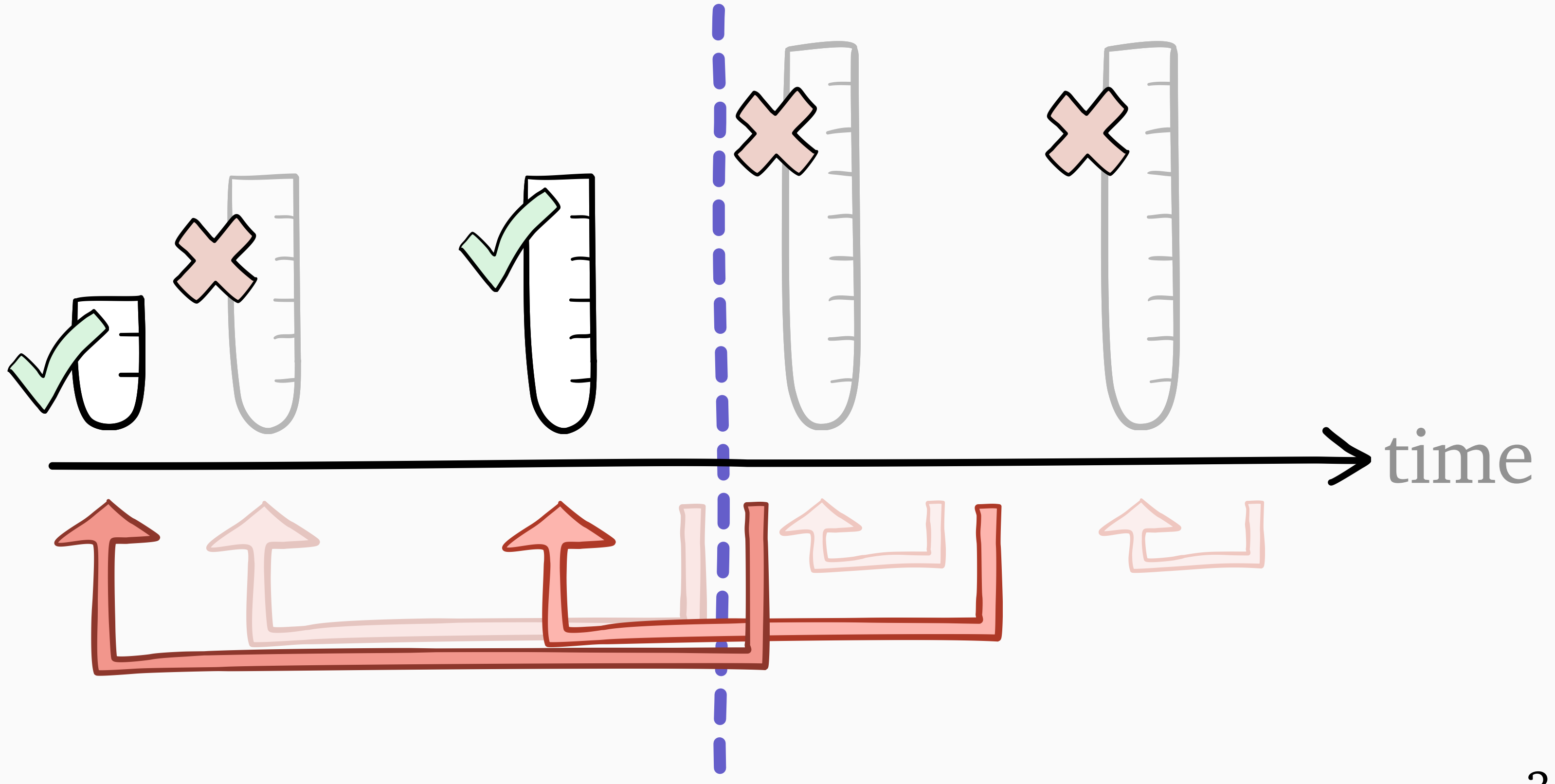
crossing work

$$C_{\text{Boost}} = C_W \mathbf{E}[e^{\gamma(S-b(S))}]\mathbf{E}[e^{\gamma V}]$$

? finite?

V = **crossing work**

work that "boosts past" a given time



Why is **Boost** *weakly* tail optimal?

$$\mathbf{P}[T > t] \sim Ce^{-\gamma t} \quad \Rightarrow \quad C = \lim_{t\to\infty} e^{\gamma t}\mathbf{P}[T > t] = \lim_{\theta\to\gamma} \frac{\gamma-\theta}{\gamma}\mathbf{E}[e^{\theta T}]$$

final value theorem

Boost

boost function

$$T_{\text{Boost}} \approx W + S - b(S) + V$$

crossing work

$$C_{\text{Boost}} = C_W \mathbf{E}[e^{\gamma(S-b(S))}]\mathbf{E}[e^{\gamma V}]$$

Lemma: finite if $b(s) = O(1/s)$

V = **crossing work**

work that "boosts past" a given time