

# Genuinely Distributed Byzantine Machine Learning

El-Mahdi El-Mhamdi   Rachid Guerraoui   Arsany Guirguis  
Lê Nguyễn Hoàng   Sébastien Rouault  
`first.last@epfl.ch`

Swiss Federal Institute of Technology (EPFL)

August 6, 2020

**EPFL**

**PODC 2020**

# The Big Picture



Machine learning (ML) tackles ***critical tasks...***

# The Big Picture



Machine learning (ML) tackles ***critical tasks...***  
...so ML should be made ***robust***



# The Big Picture



Machine learning (ML) tackles **critical tasks**...  
...so ML should be made **robust**



Literature: robust when  $\left\{ \begin{array}{l} \text{using} \\ \text{training} \end{array} \right.$  the model

# The Big Picture



Machine learning (ML) tackles **critical tasks**...  
...so ML should be made **robust**



Literature: robust when **training** the model

# The Big Picture



Machine learning (ML) tackles **critical tasks**...  
...so ML should be made **robust**



Literature: robust when **training** the model  
4y ago ————

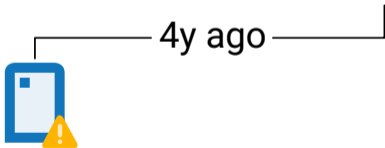
# The Big Picture



Machine learning (ML) tackles ***critical tasks***...  
...so ML should be made ***robust***



Literature: robust when ***training*** the model



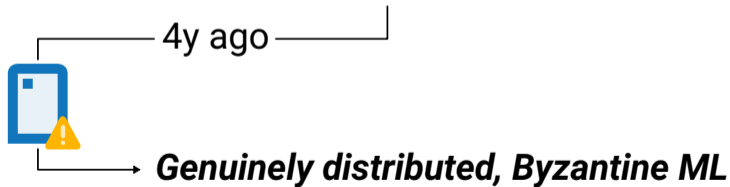
# The Big Picture



Machine learning (ML) tackles **critical tasks**...  
...so ML should be made **robust**

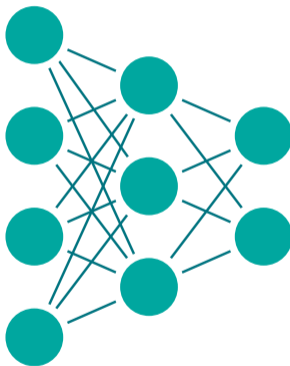


Literature: robust when **training** the model





# Machine learning (ML)

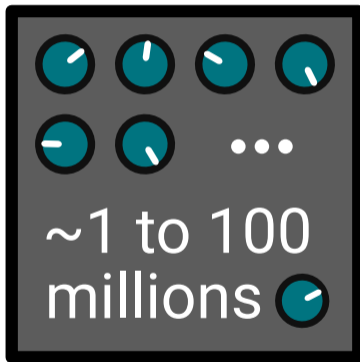


Boat

Goat

...

# Machine learning (ML)

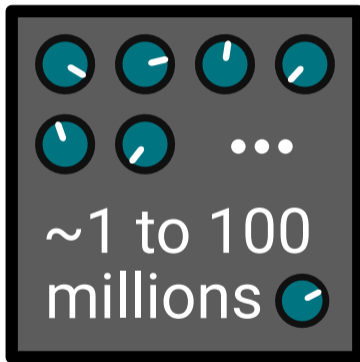


Boat

Goat

...

# Machine learning (ML)

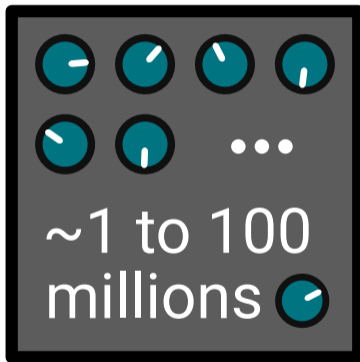


Krust

Zr0m

...

# Machine learning (ML)

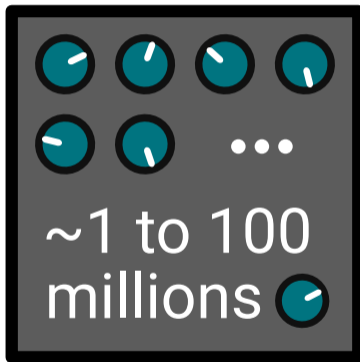


Brust

GOrm

...

# Machine learning (ML)

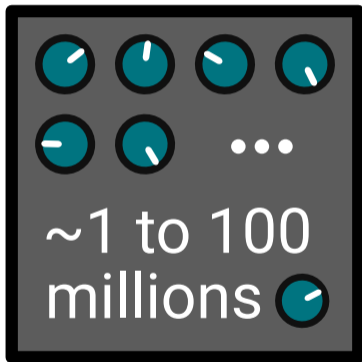


Bost

GOat

...

# Machine learning (ML)



Boat

Goat

...

# Stochastic Gradient Descent (SGD)



Training loop:

1. Estimate gradient
2. Turn potentiometers following the gradient
3. Loop back to step 1.

# Stochastic Gradient Descent (SGD)

$$\begin{pmatrix} 4.2 \\ -0.5 \\ -1.0 \\ 0.8 \\ -5.7 \\ 0.3 \\ \vdots \end{pmatrix}$$

Training loop:

1. Estimate gradient
2. Turn potentiometers following the gradient
3. Loop back to step 1.



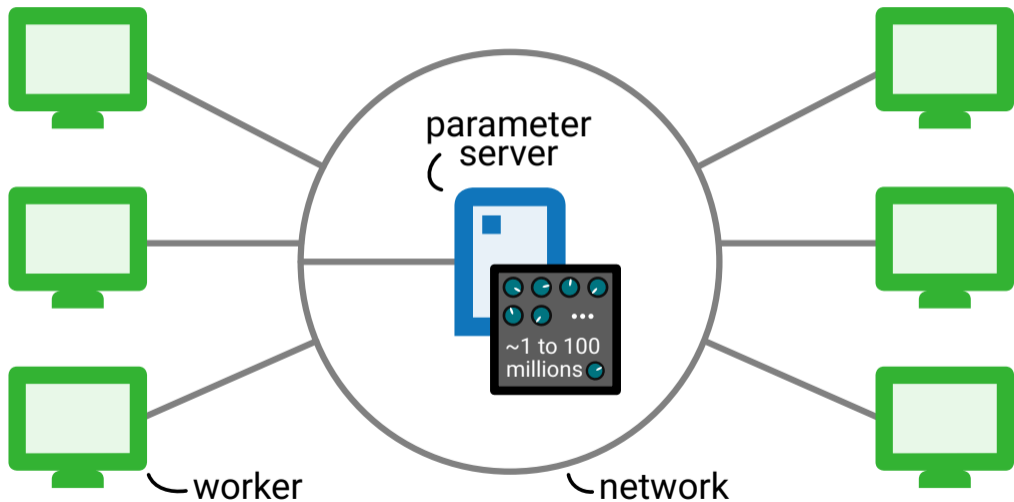
# Stochastic Gradient Descent (SGD)

$$\begin{pmatrix} 4.2 \\ -0.5 \\ -1.0 \\ 0.8 \\ -5.7 \\ 0.3 \\ \vdots \end{pmatrix}$$

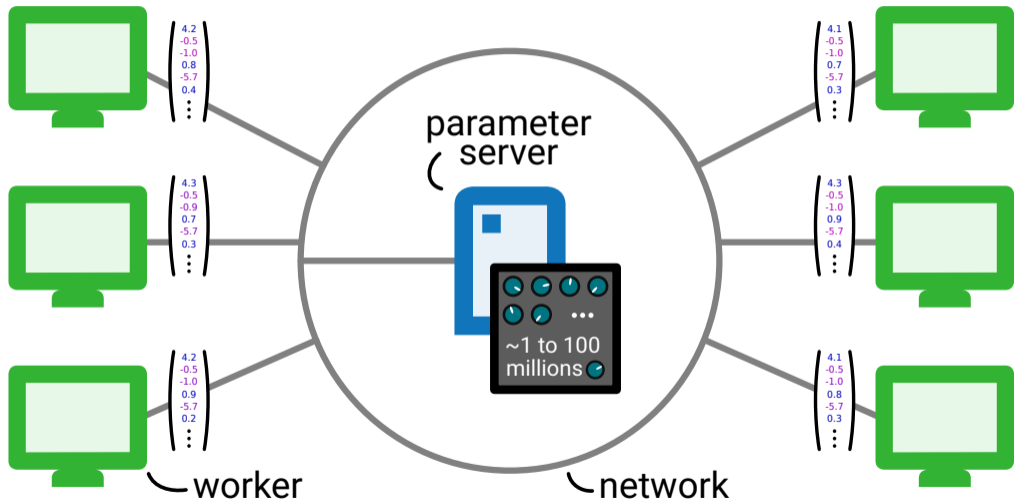
Training loop:

1. **Estimate** gradient
2. Turn potentiometers following the gradient
3. Loop back to step 1.

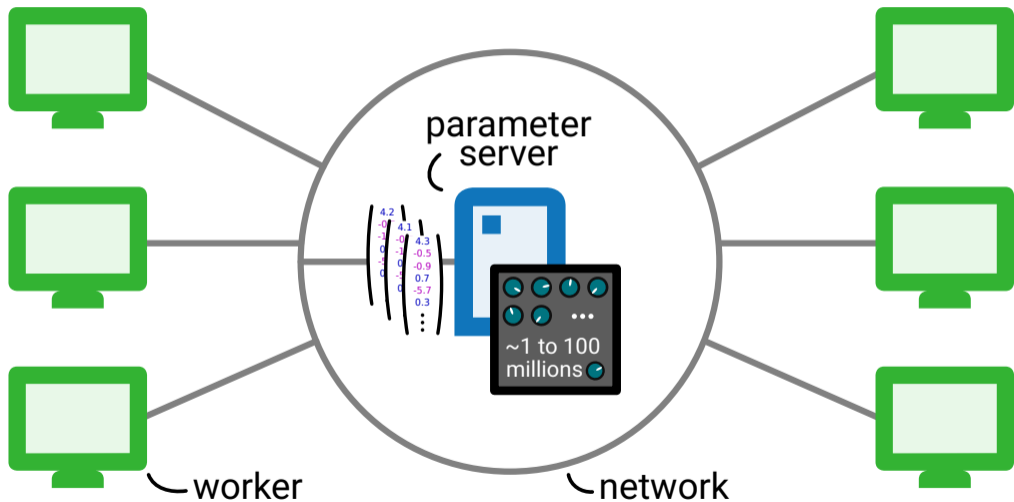
# Distributed SGD



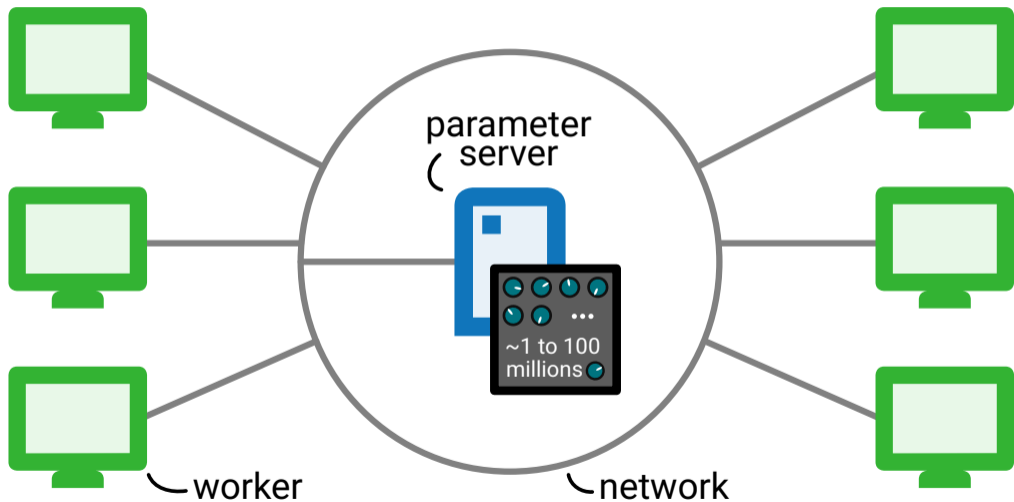
# Distributed SGD



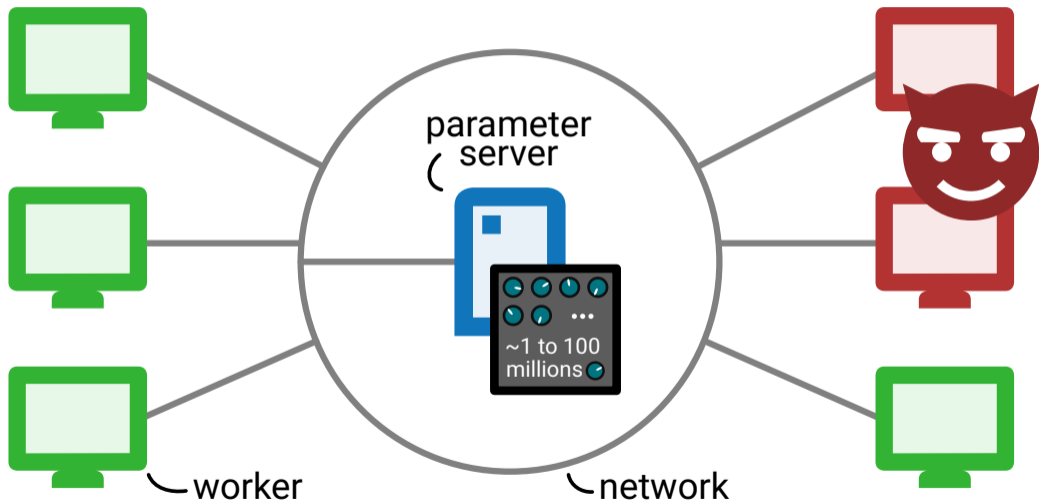
# Distributed SGD



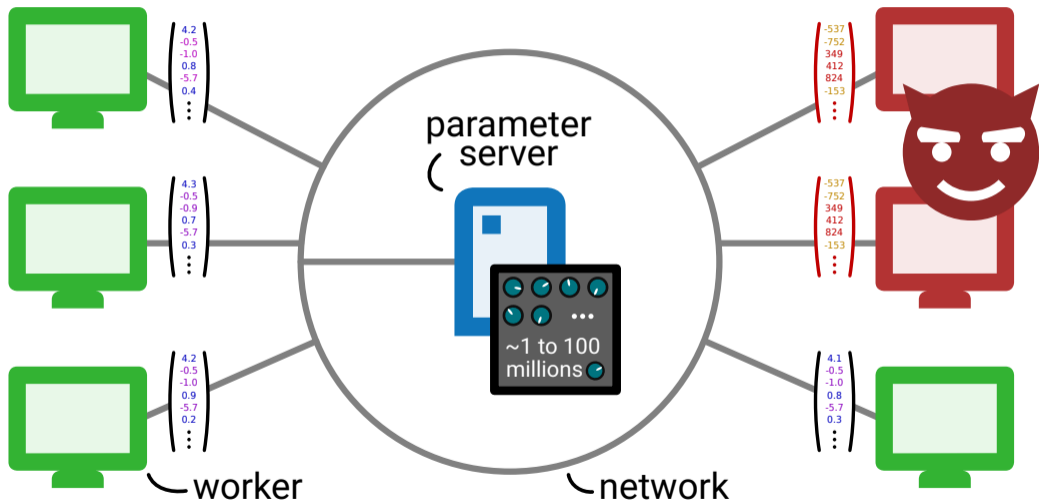
# Distributed SGD



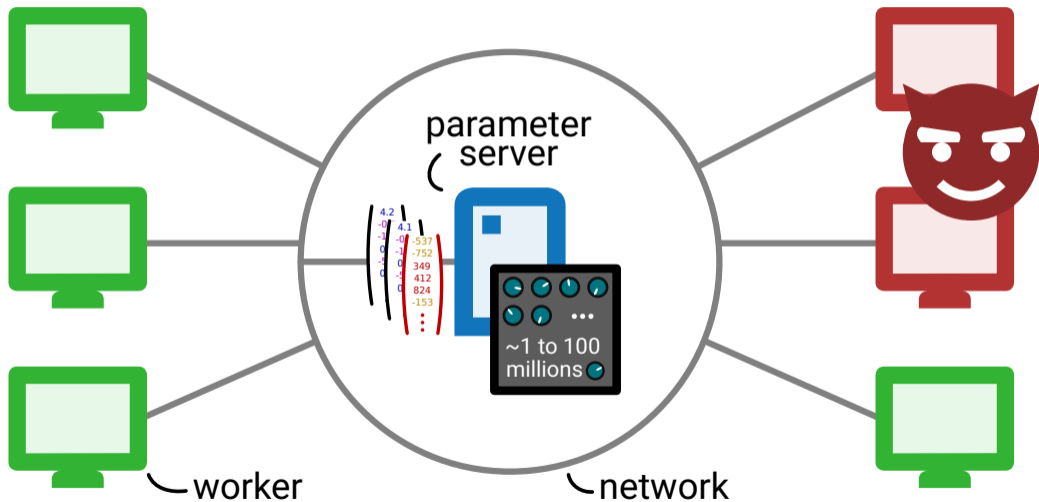
# Distributed, Byzantine SGD



# Distributed, Byzantine SGD

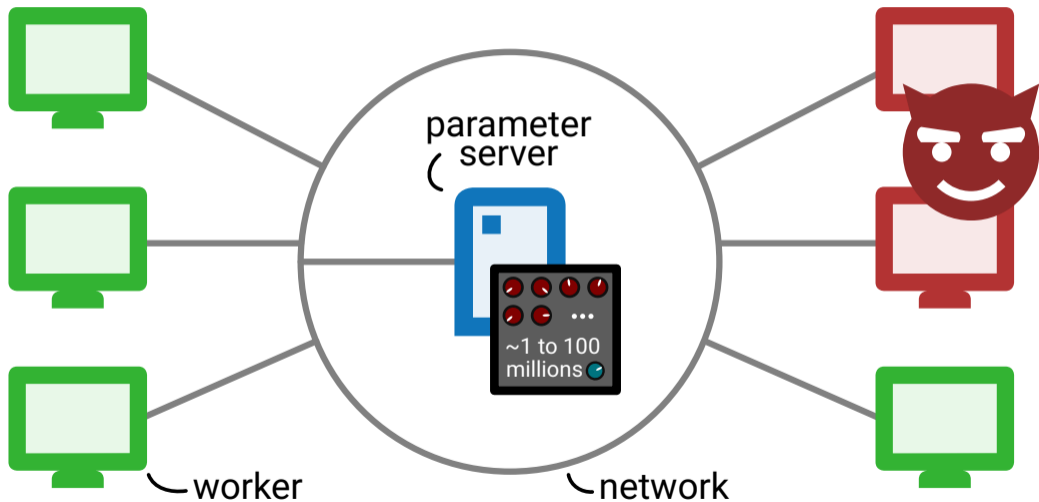


# Distributed, Byzantine SGD





# Distributed, Byzantine SGD

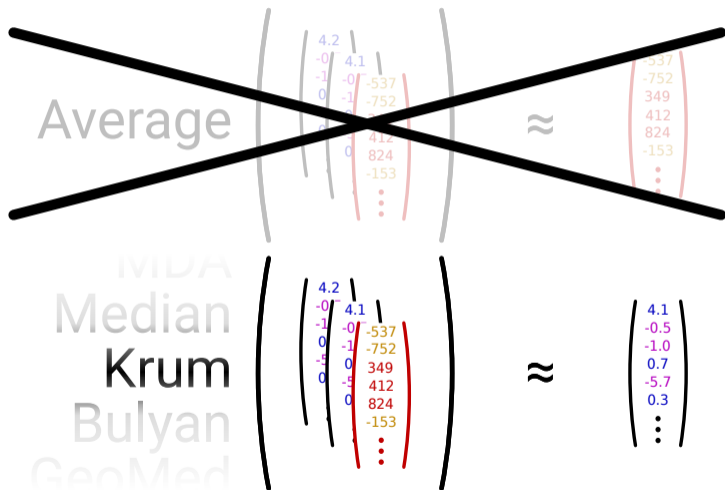


# Byzantine-resilient SGD

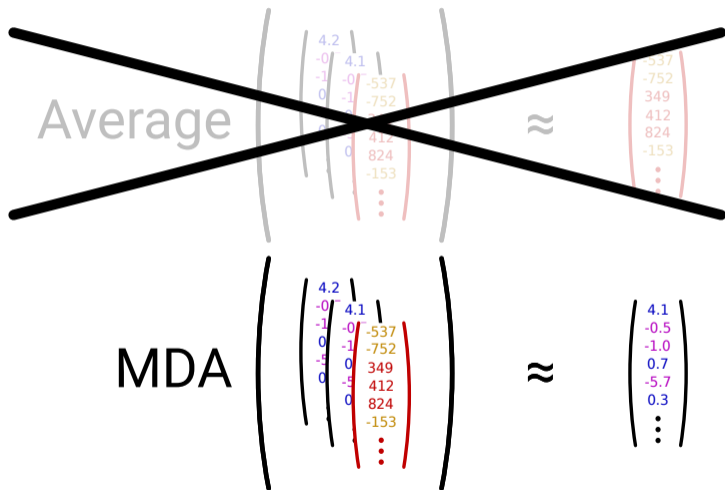
Average

$$\left( \begin{array}{c} \left( \begin{array}{c} 4.2 \\ -0.1 \\ -1.1 \\ 0 \\ -1 \\ 0 \\ 0 \end{array} \right) \\ \left( \begin{array}{c} 4.1 \\ -0.1 \\ -1 \\ 0 \\ -1 \\ 0 \end{array} \right) \\ \left( \begin{array}{c} -537 \\ -752 \\ 349 \\ 412 \\ 824 \\ -153 \\ \vdots \end{array} \right) \end{array} \right) \approx \left( \begin{array}{c} -537 \\ -752 \\ 349 \\ 412 \\ 824 \\ -153 \\ \vdots \end{array} \right)$$

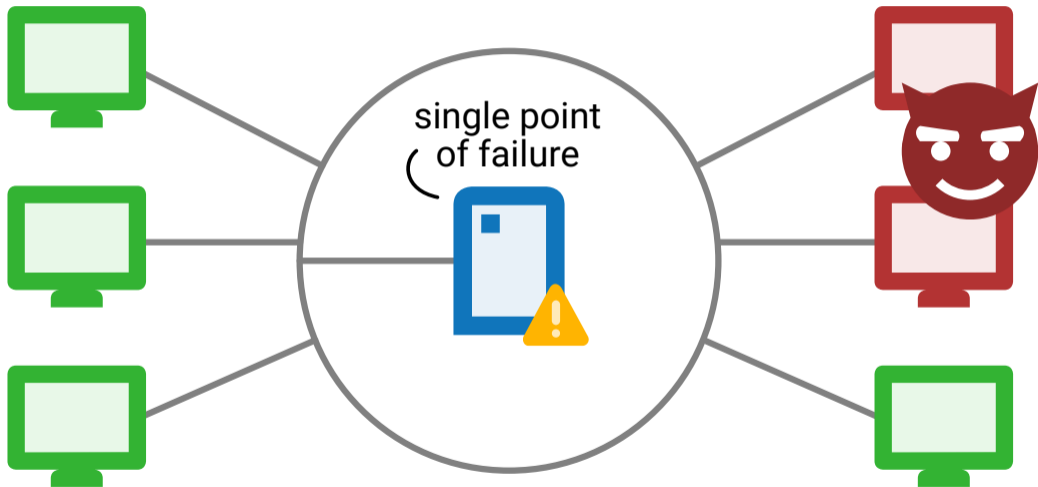
# Byzantine-resilient SGD



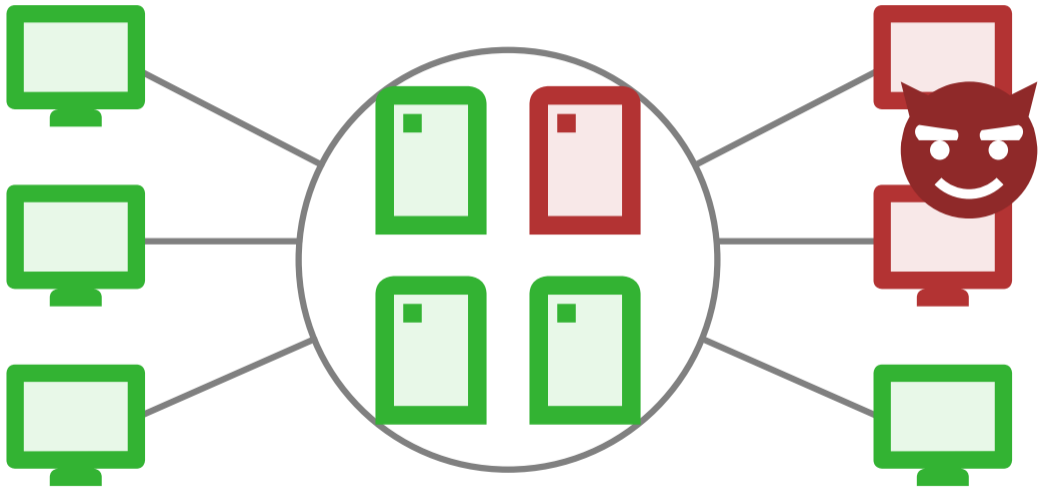
# Byzantine-resilient SGD



# Problem



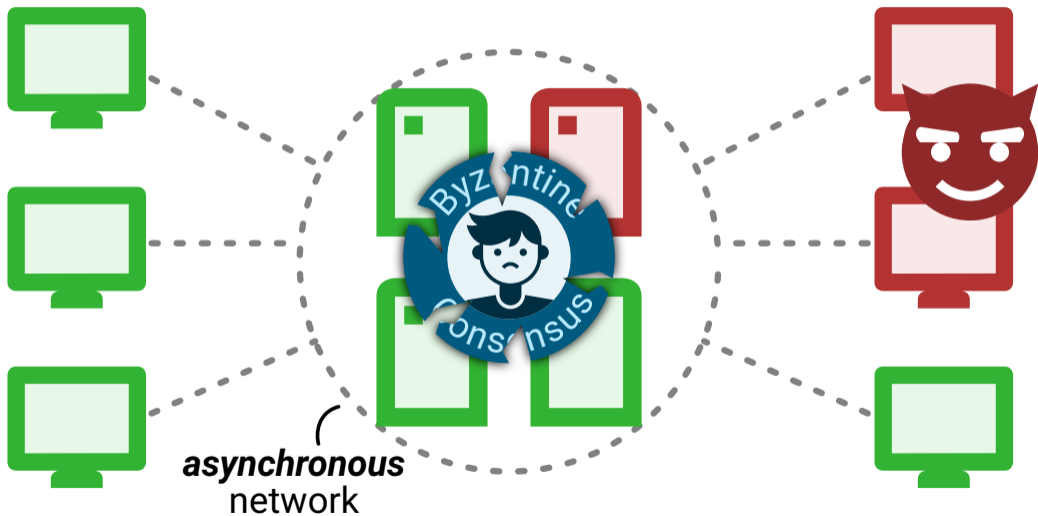
# Problem... solution



# Problem... solution

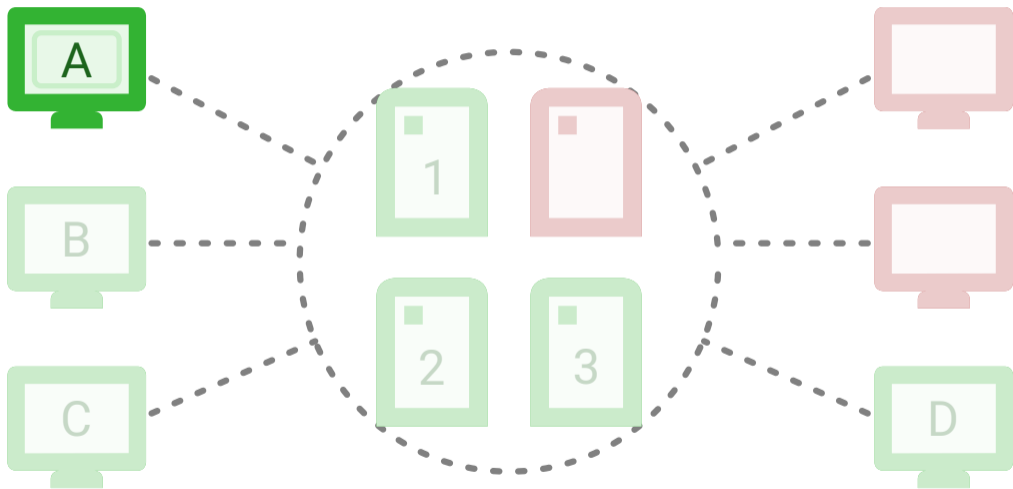


# Problem... solution... nope

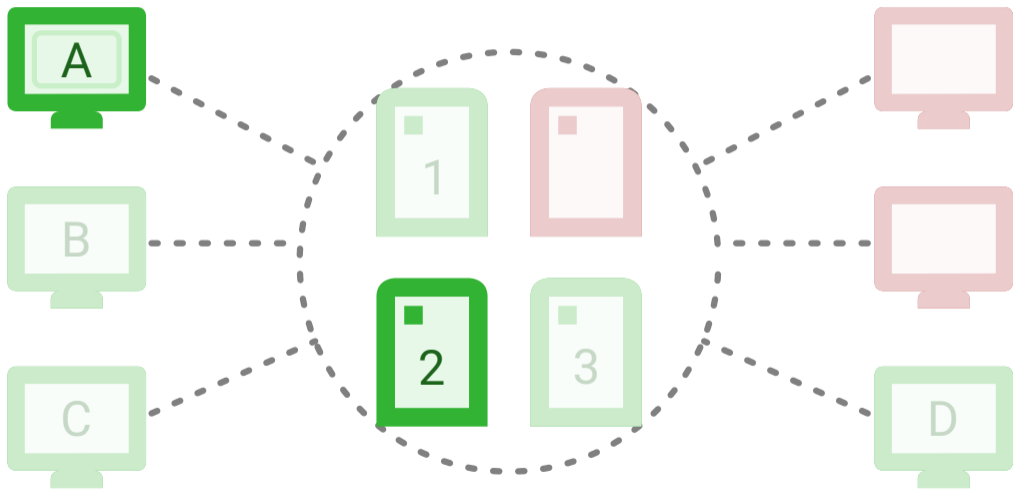




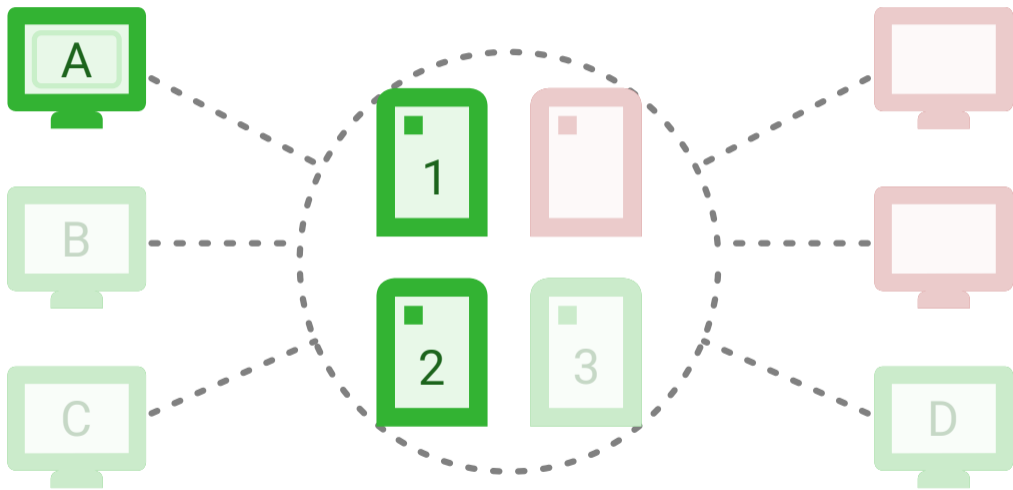
# Key problem: divergence



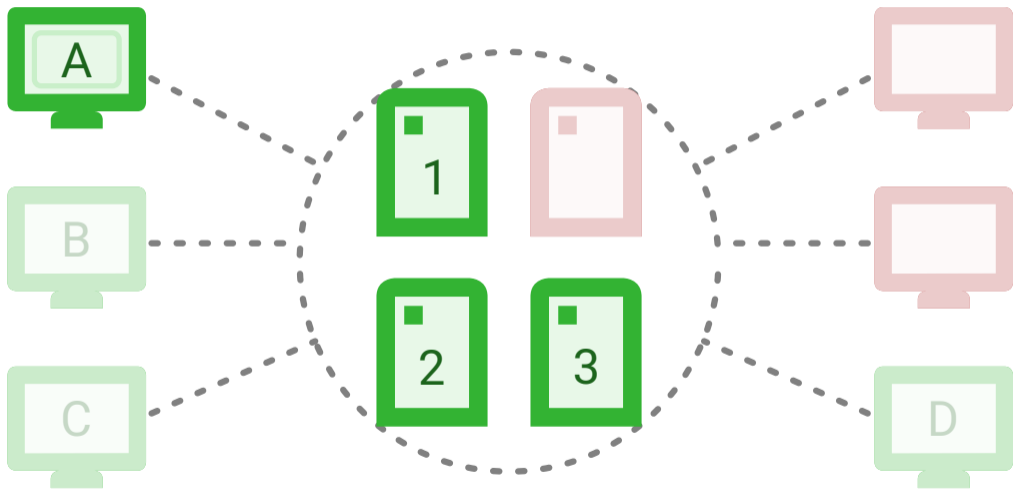
# Key problem: divergence



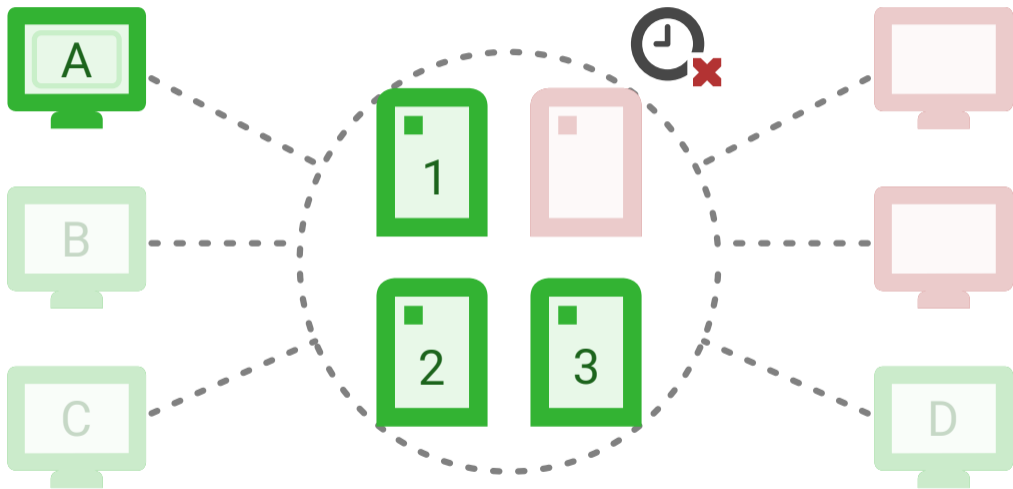
# Key problem: divergence



# Key problem: divergence



# Key problem: divergence



# Key problem: divergence



# Key problem: divergence

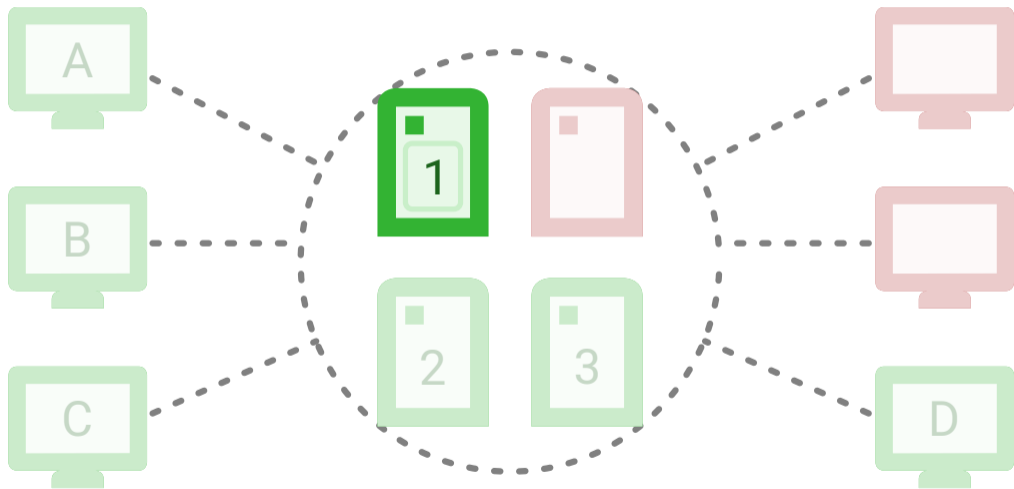


# Key problem: divergence

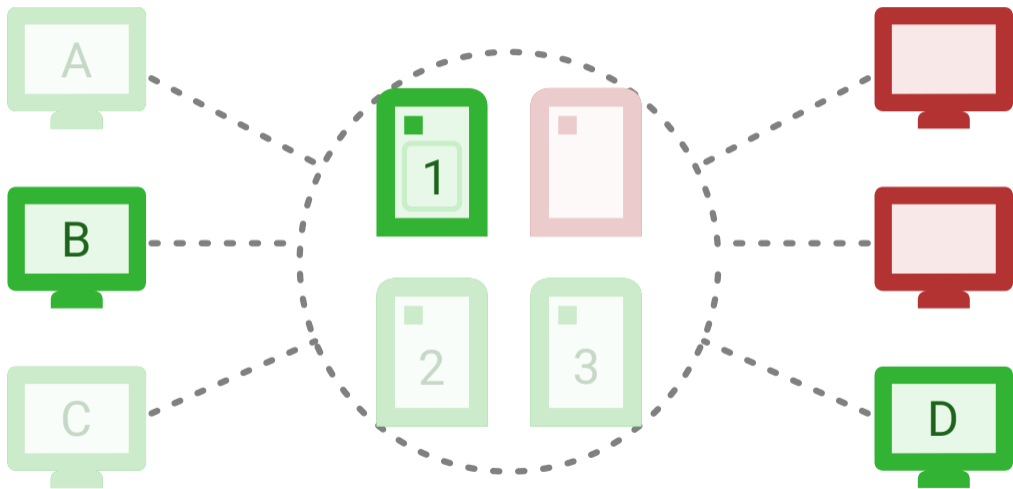




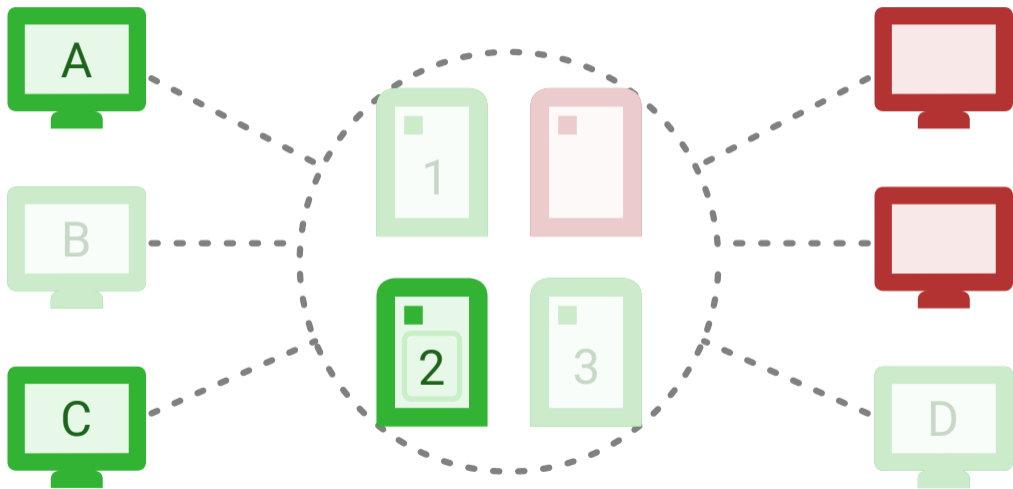
# Key problem: divergence



# Key problem: divergence



# Key problem: divergence

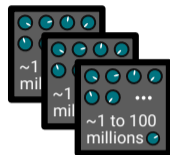


# Key problem: divergence



# The goal

Can we keep the



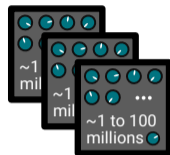
**"close"** to each other...

...despite network **asynchrony**...

...and **Byzantine** behaviors?

## Key approach

Can we **bring** the

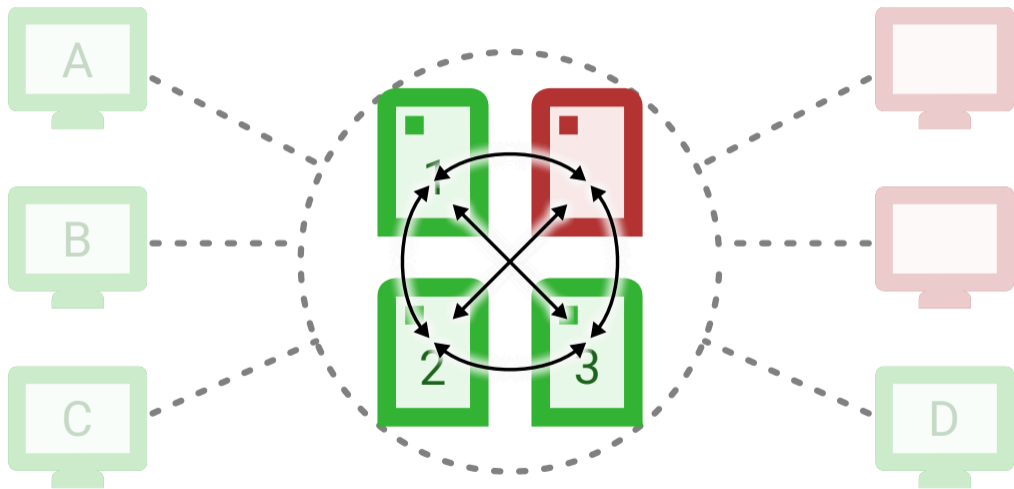


**back** closer to each other...

...despite network **asynchrony**...

...and **Byzantine** behaviors?

# Key approach: +1 round




# Key approach: toy example



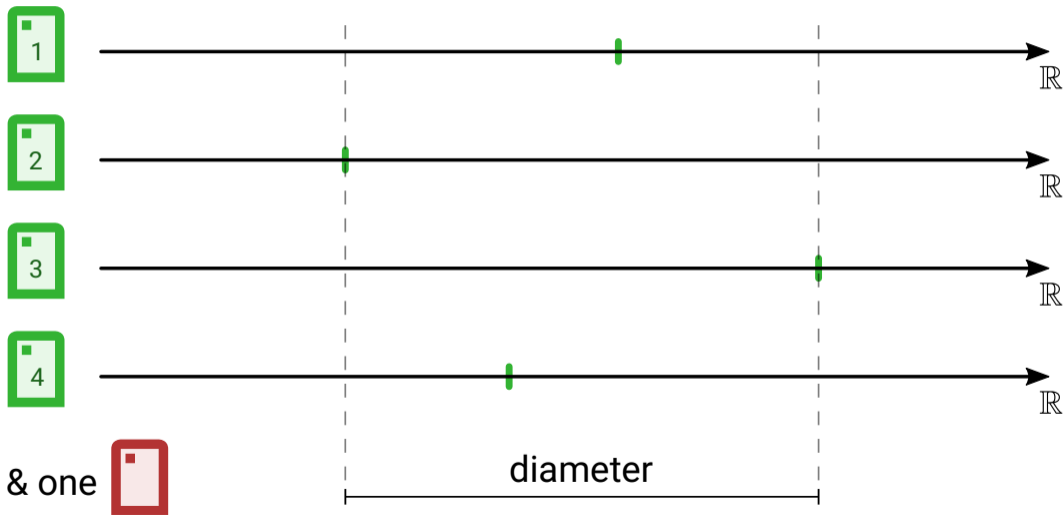
& one 

1-parameter model:

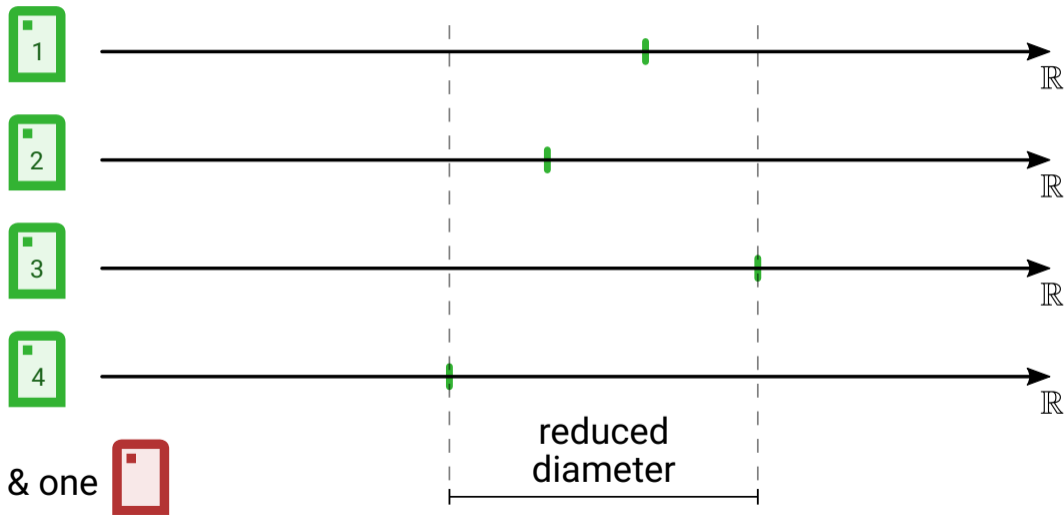
 =  $\mathbb{R}$  ↷



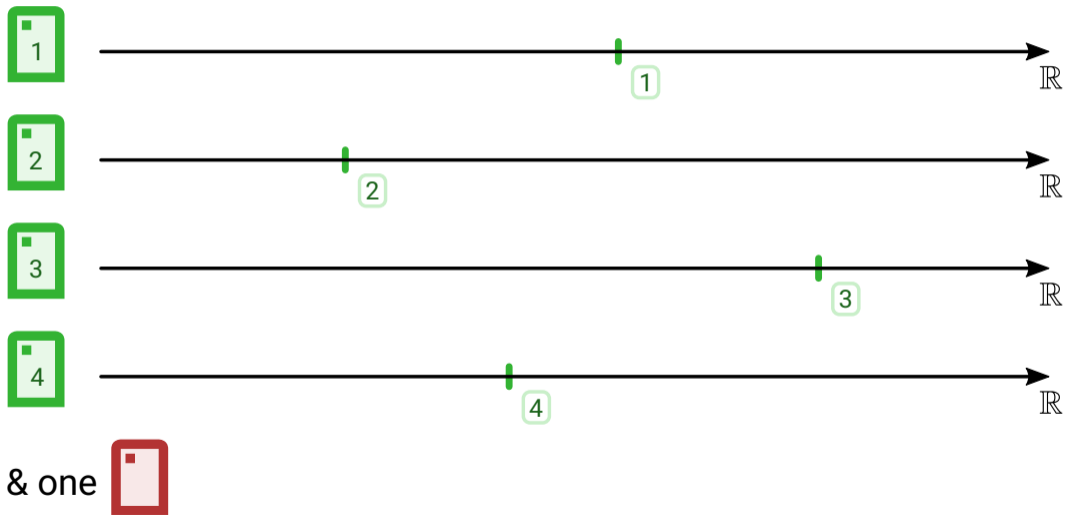
# Key approach: toy example



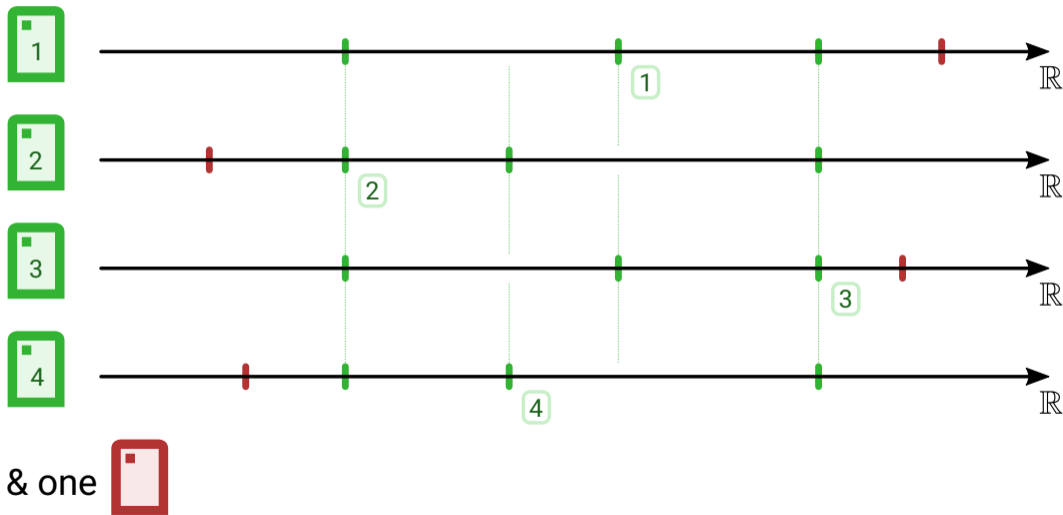
# Key approach: toy example



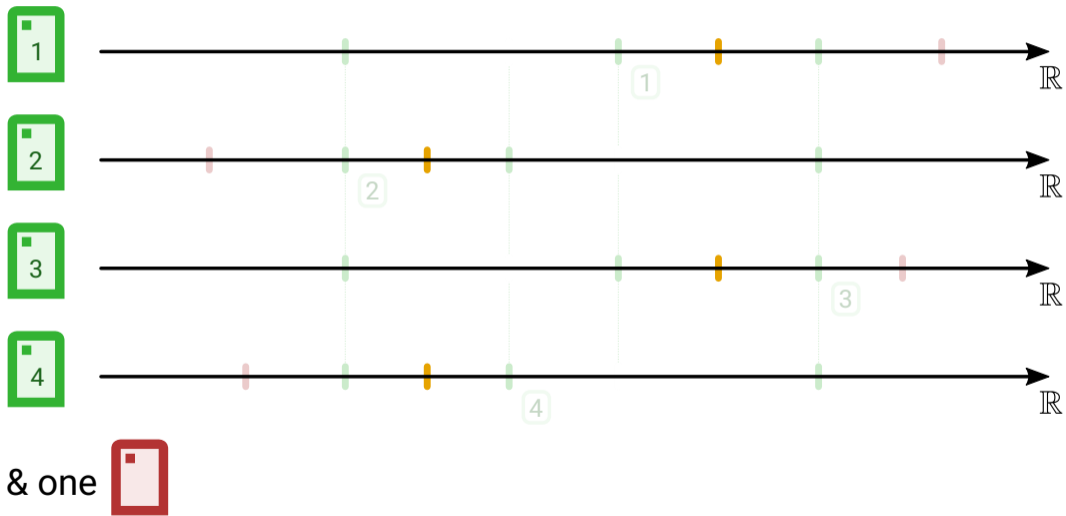
# Key approach: toy example



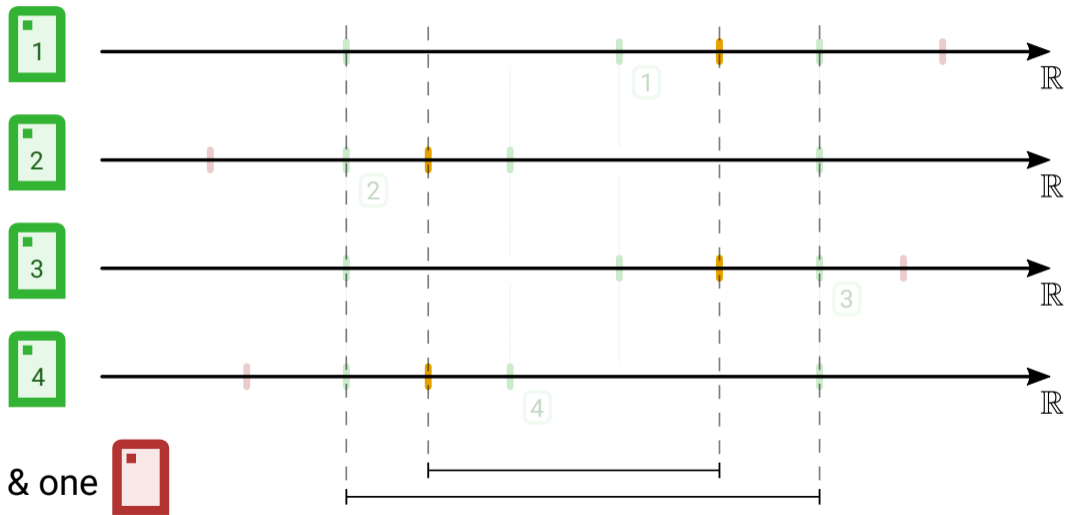
# Key approach: toy example



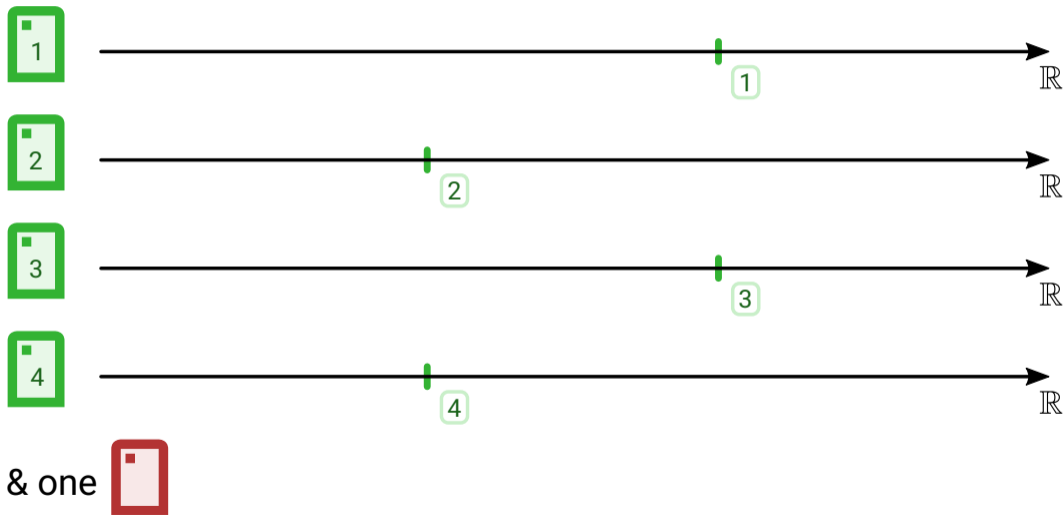
# Key approach: toy example



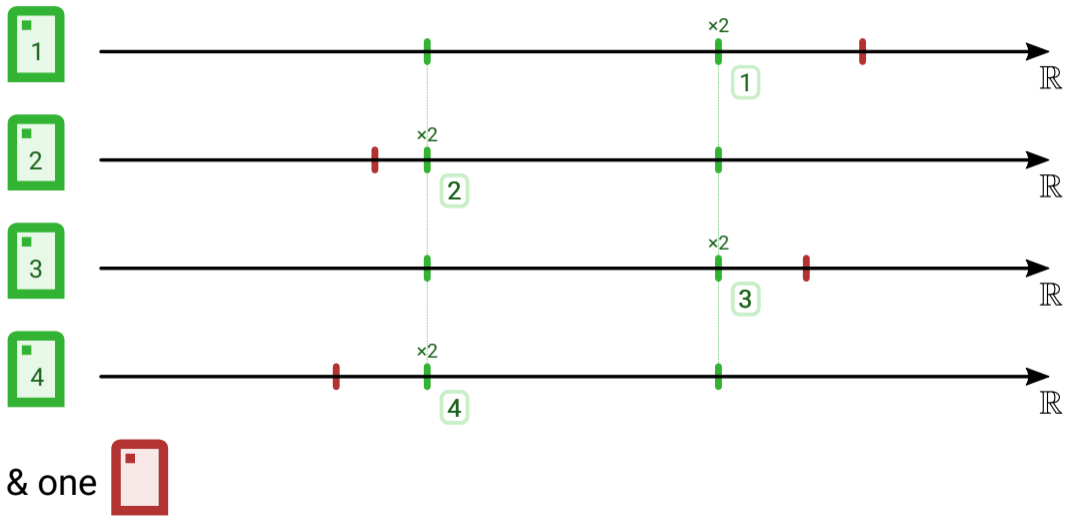
# Key approach: toy example



# Key approach: last remark



# Key approach: last remark





# Key approach: last remark

