Identification-based first-order algorithms for distributed learning

Dmitry Grishchenko

Problem

$$\min_{x \in \mathbb{R}^n} \sum_{i=1}^{M} f_i(x) + r(x)$$

Problem

convex
L-smooth

$$\min_{x \in \mathbb{R}^n} \sum_{i=1}^{M} f_i(x) + r(x)$$

convex
non-smooth

Problem

$$\min_{x \in \mathbb{R}^n} \sum_{i=1}^{M} f_i(x) + r(x)$$

convex
L-smooth

convex
non-smooth

In ML context

$$f_i(x) = \sum_{j \in S_i} \pi_i l_j(x)$$

Problem

$$\min_{x \in \mathbb{R}^n} \sum_{i=1}^{M} f_i(x) + r(x)$$

convex
L-smooth

convex
non-smooth

In ML context

$$f_i(x) = \sum_{j \in S_i} \pi_i l_j(x)$$

l_j – loss associated with j^{th} example
S_i – i^{th} set of examples
π_i – proportion of examples in i^{th} set

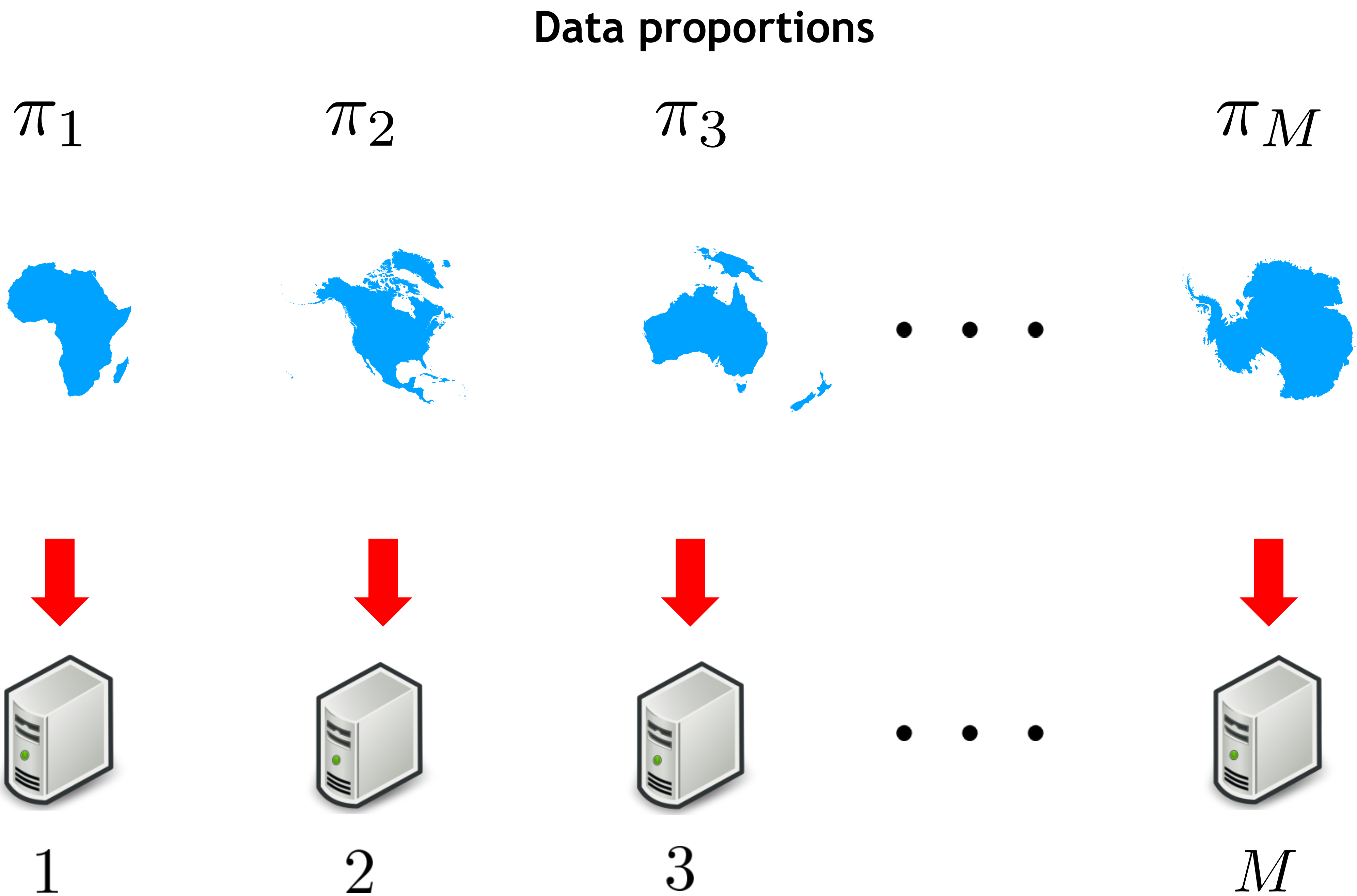
Distribute It!

Distribute It!

Data proportions

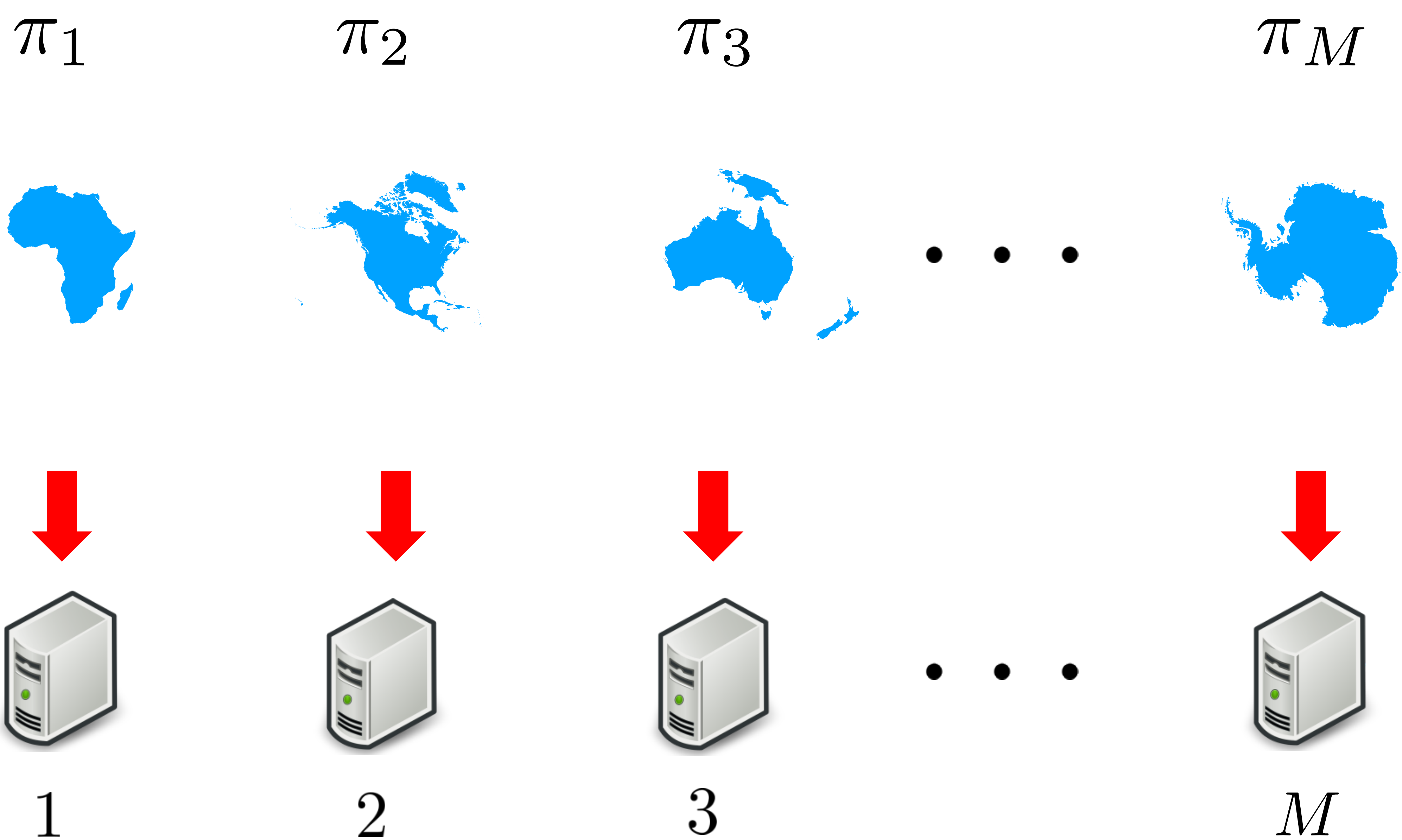
π_1 π_2 π_3 π_M

Distribute It!



Distribute It! Centralized

Data proportions



Parallel or Distributed?

Parallel or Distributed?

What is a difference?

Parallel or Distributed?

What is a difference?

	Parallel	Distributed

Parallel or Distributed?

What is a difference?

	Parallel	Distributed
Machines		

Parallel or Distributed?

What is a difference?

	Parallel	Distributed
Machines	Single	Multiple

Parallel or Distributed?

What is a difference?

	Parallel	Distributed
Machines	Single	Multiple
Storage		

Parallel or Distributed?

What is a difference?

	Parallel	Distributed
Machines	Single	Multiple
Storage	Limited	Unlimited

Parallel or Distributed?

What is a difference?

	Parallel	Distributed
Machines	Single	Multiple
Storage	Limited	Unlimited
Communication		

Parallel or Distributed?

What is a difference?

	Parallel	Distributed
Machines	Single	Multiple
Storage	Limited	Unlimited
Communication	Free	Bottleneck

Parallel or Distributed?

What is a difference?

	Parallel	Distributed
Machines	Single	
Storage	Limited	Unlimited
Communication	Free	Bottleneck
	Multiple	

Parallel or Distributed?

What is a difference?

	Parallel	Distributed
Machines	Single	
Storage	Limited	Unlimited
Communication	Free	Bottleneck

Multiple

Private data

Parallel or Distributed?

What is a difference?

	Parallel	Distributed
Machines	Single	
Storage	Limited	Unlimited
Communication	Free	Bottleneck

Multiple

Private data

Phones = Machines

Parallel or Distributed?

What is a difference?

	Parallel	Distributed
Machines	Single	
Storage	Limited	Unlimited
Communication	Free	**Bottleneck**

Multiple

Private data

Phones = Machines

Parallel or Distributed?

What is a difference?

	Parallel	Distributed
Machines	Single	
Storage	Limited	Unlimited
Communication	Free	Bottleneck

Multiple

Private data

Phones = Machines

Result

Parallel or Distributed?

What is a difference?

	Parallel	Distributed
Machines	Single	
Storage	Limited	Unlimited
Communication	Free	**Bottleneck**

Multiple

Private data

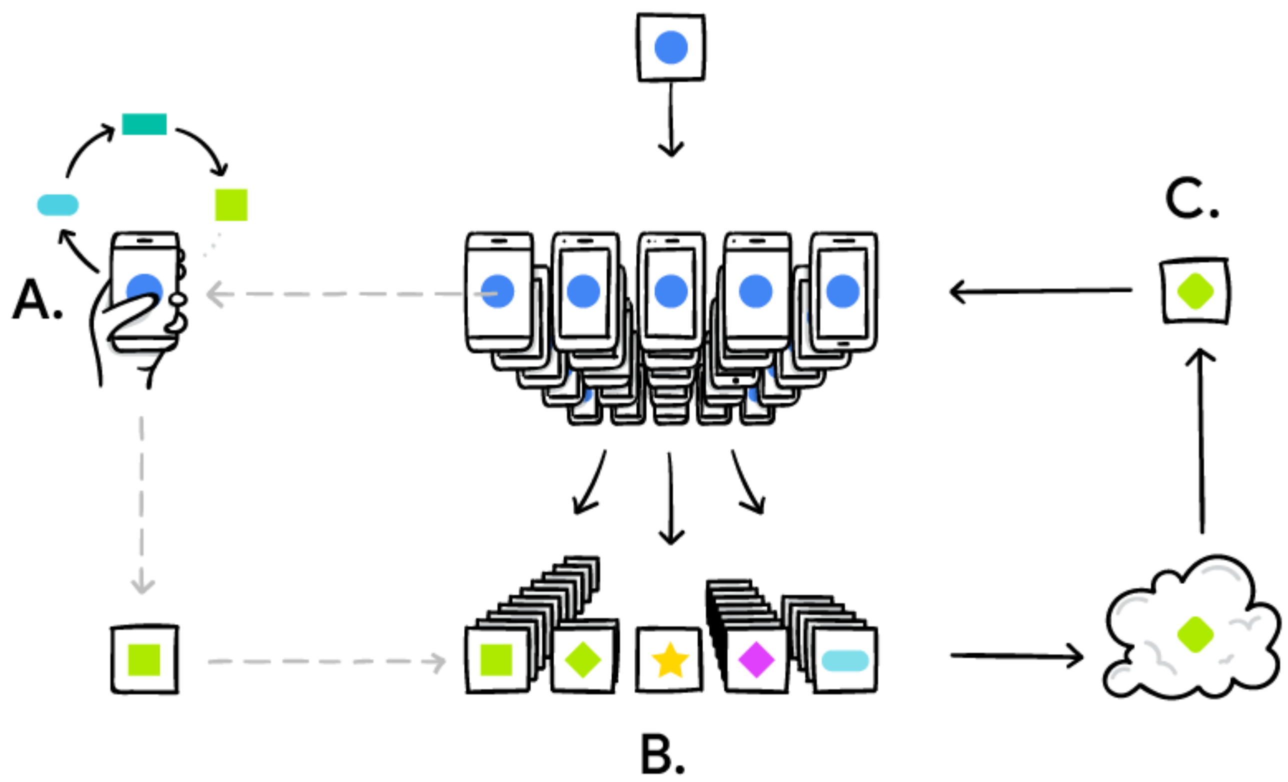
Phones = Machines

Result

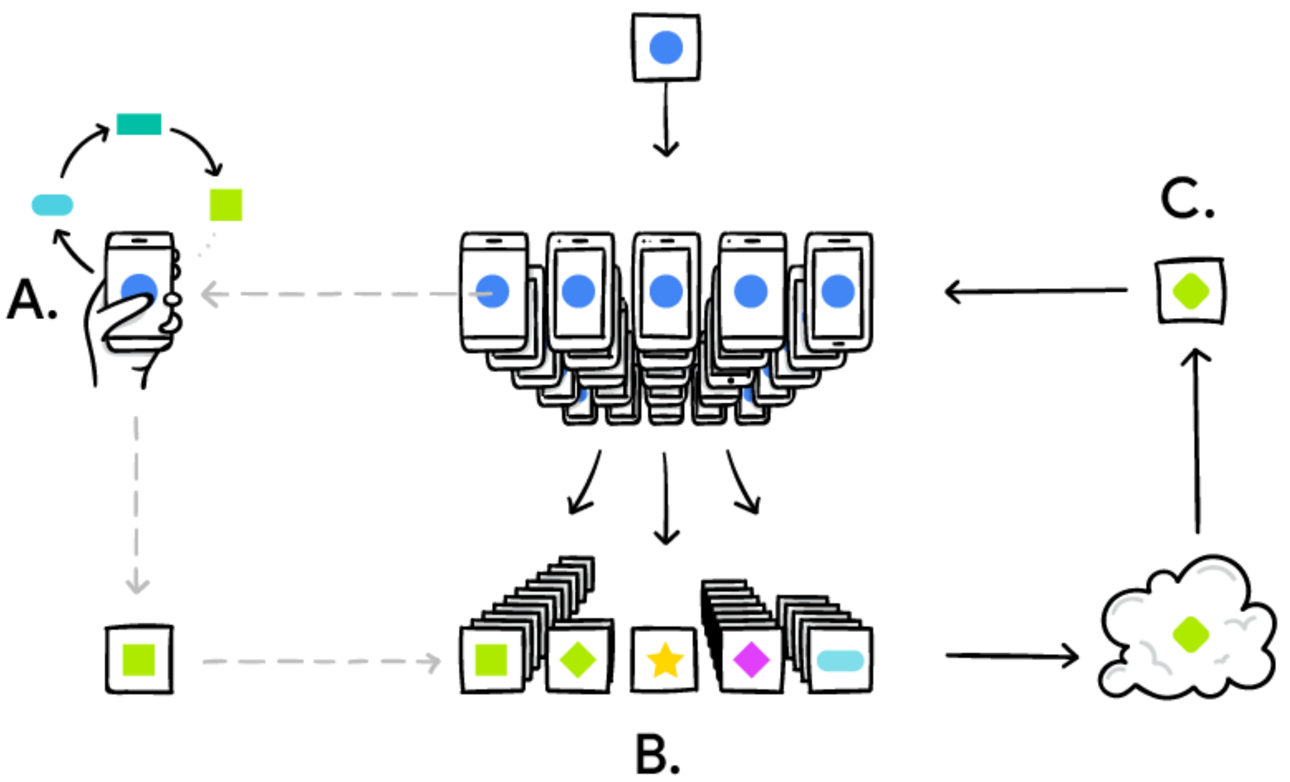
<u>Computations are cheaper than communications</u>

Federated Learning

Federated Learning

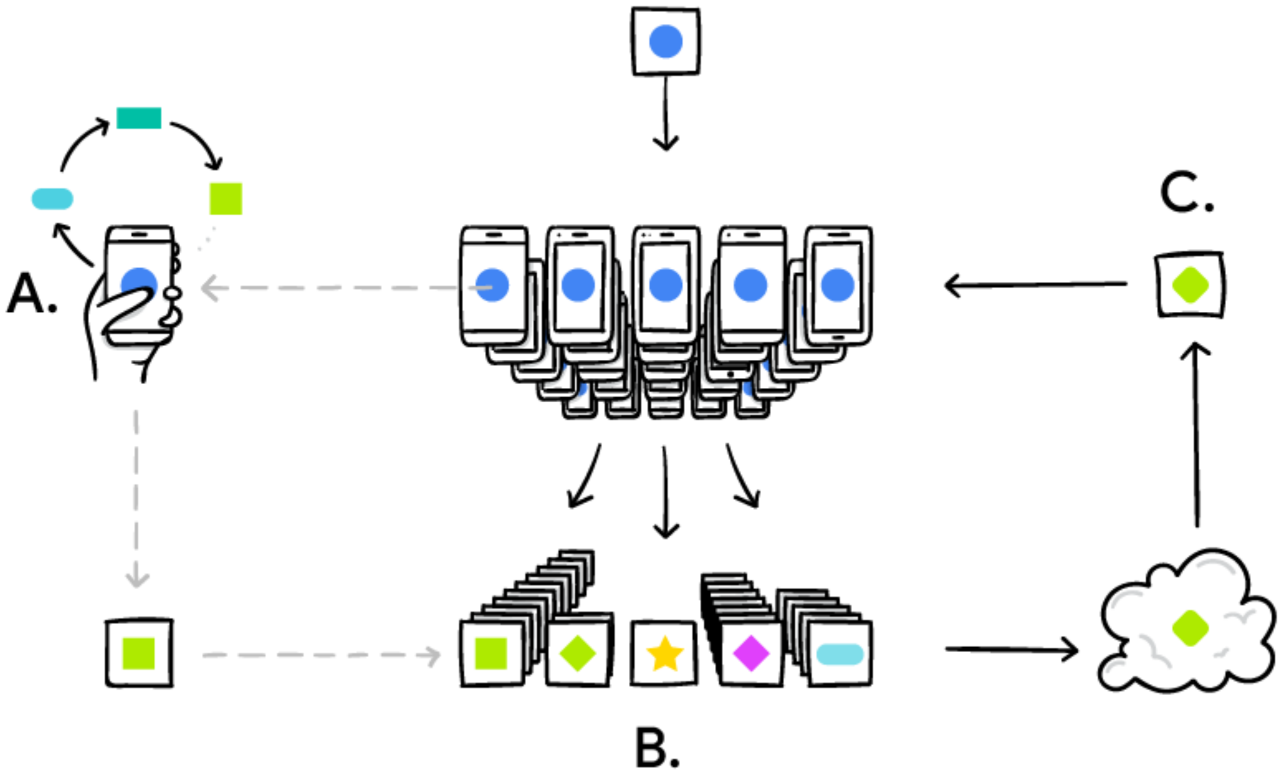


Federated Learning



A. - phone personalizes the model locally, based on user's usage

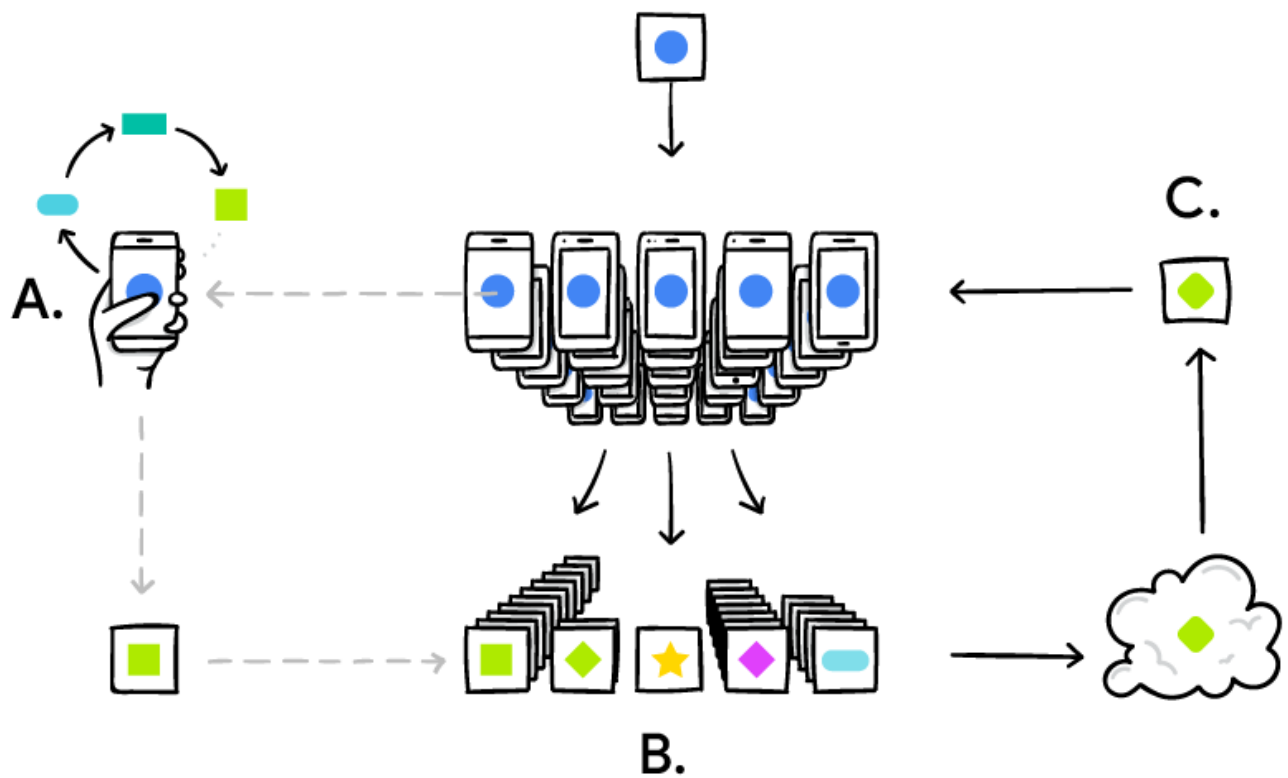
Federated Learning



A. - phone personalizes the model locally, based on user’s usage

B. - many users' updates are aggregated

Federated Learning

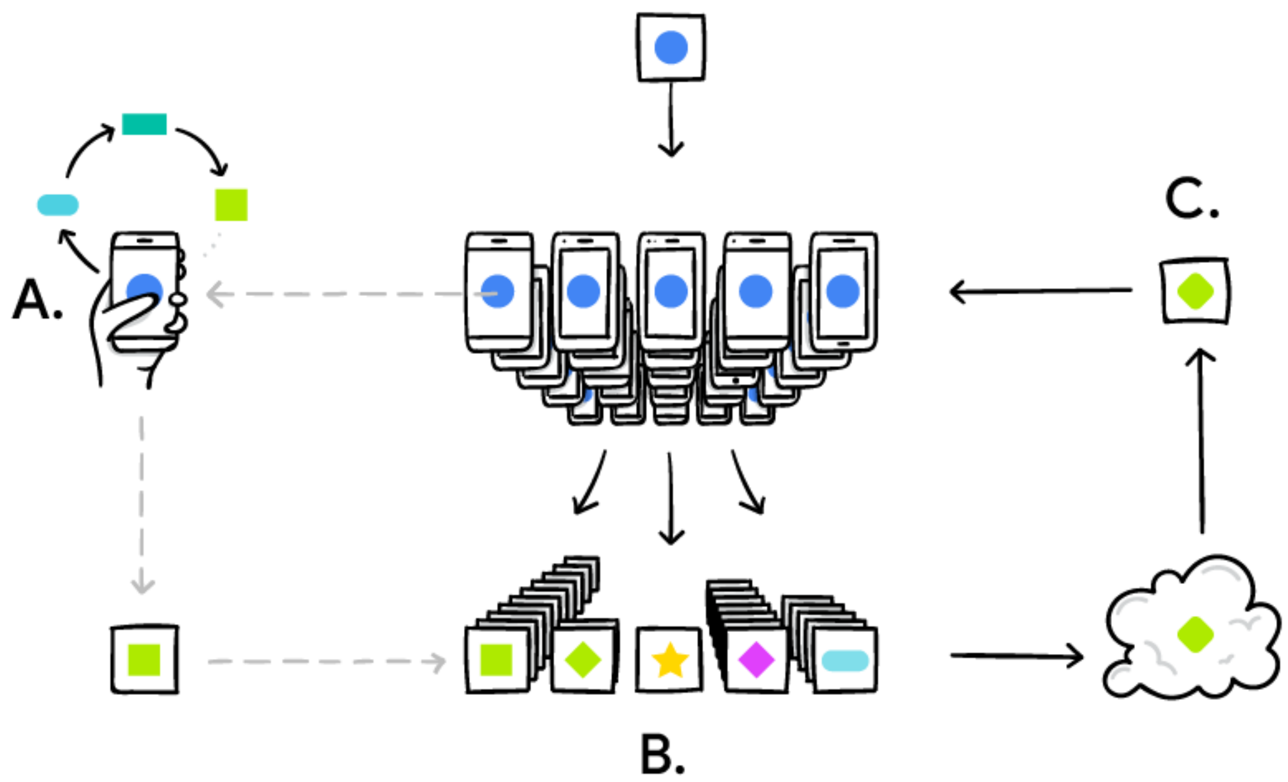


A. - phone personalizes the model locally, based on user’s usage

B. - many users' updates are aggregated

C. - consensus change to the shared model

Federated Learning



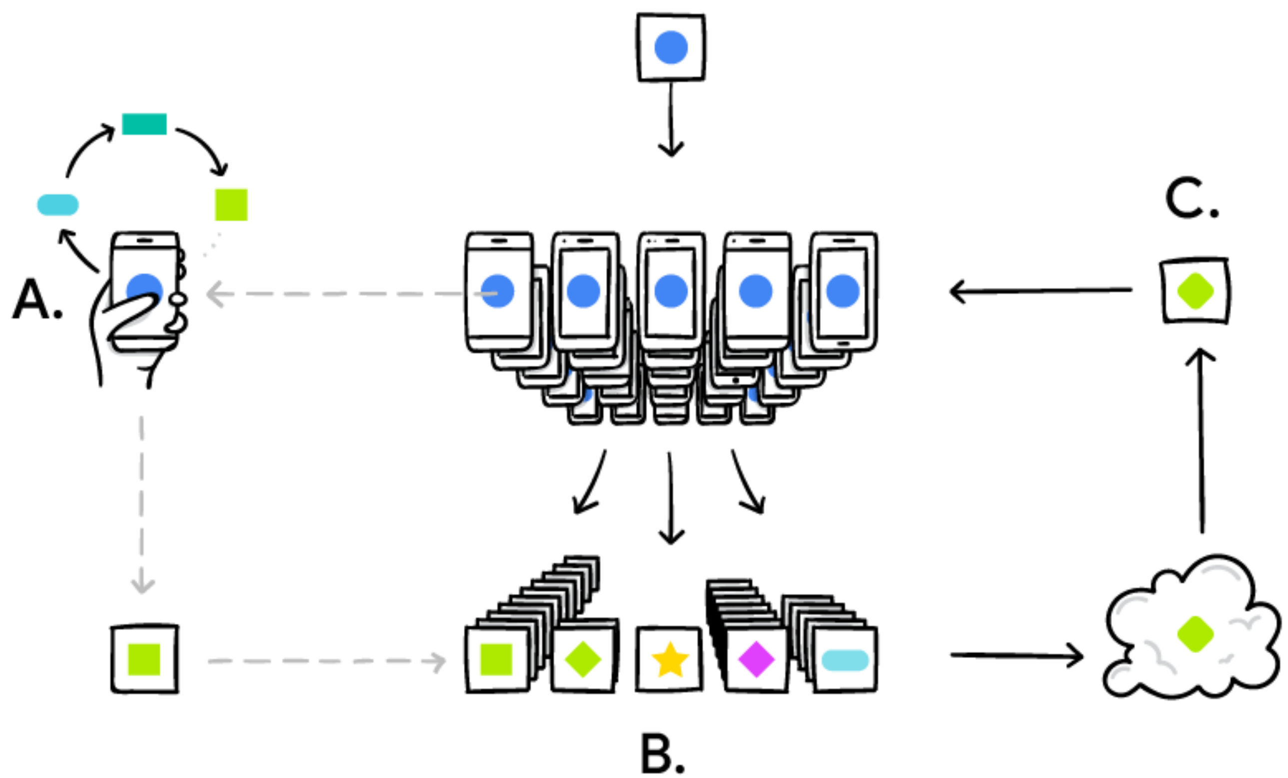
A. - phone personalizes the model locally, based on user’s usage

B. - many users' updates are aggregated

C. - consensus change to the shared model

As a result: a lot of updates, with cheap computations

Federated Learning



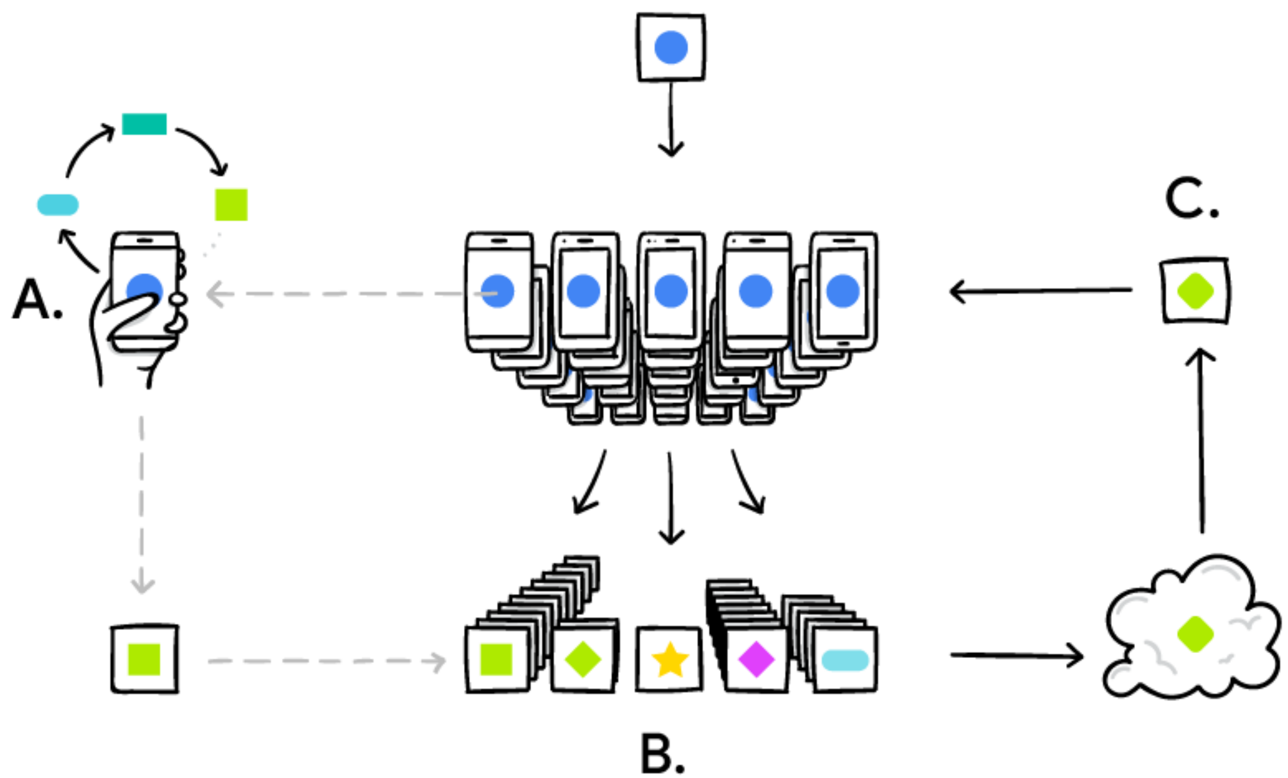
A. - phone personalizes the model locally, based on user's usage

B. - many users' updates are aggregated

C. - consensus change to the shared model

Let's make communications cheaper!

Federated Learning



A. - phone personalizes the model locally, based on user's usage

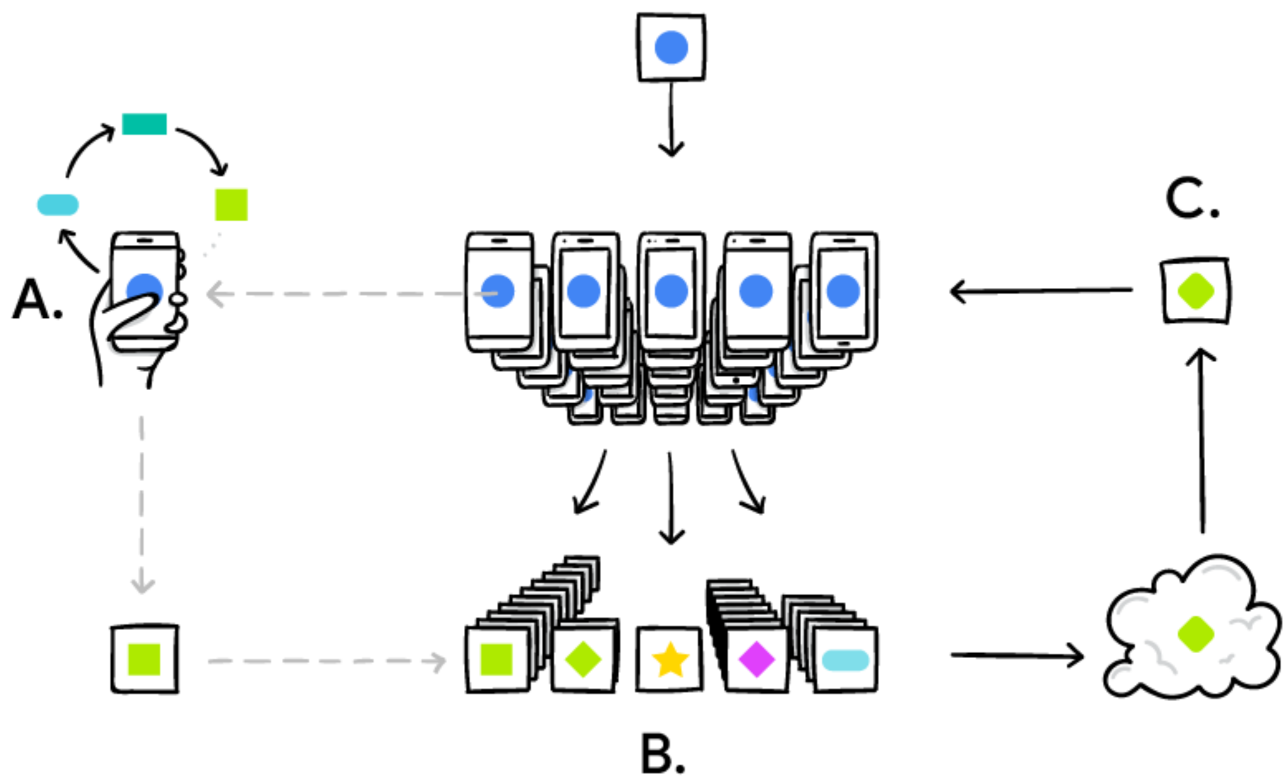
B. - many users' updates are aggregated

C. - consensus change to the shared model

Let's make communications cheaper!

Less often

Federated Learning



A. - phone personalizes the model locally, based on user's usage

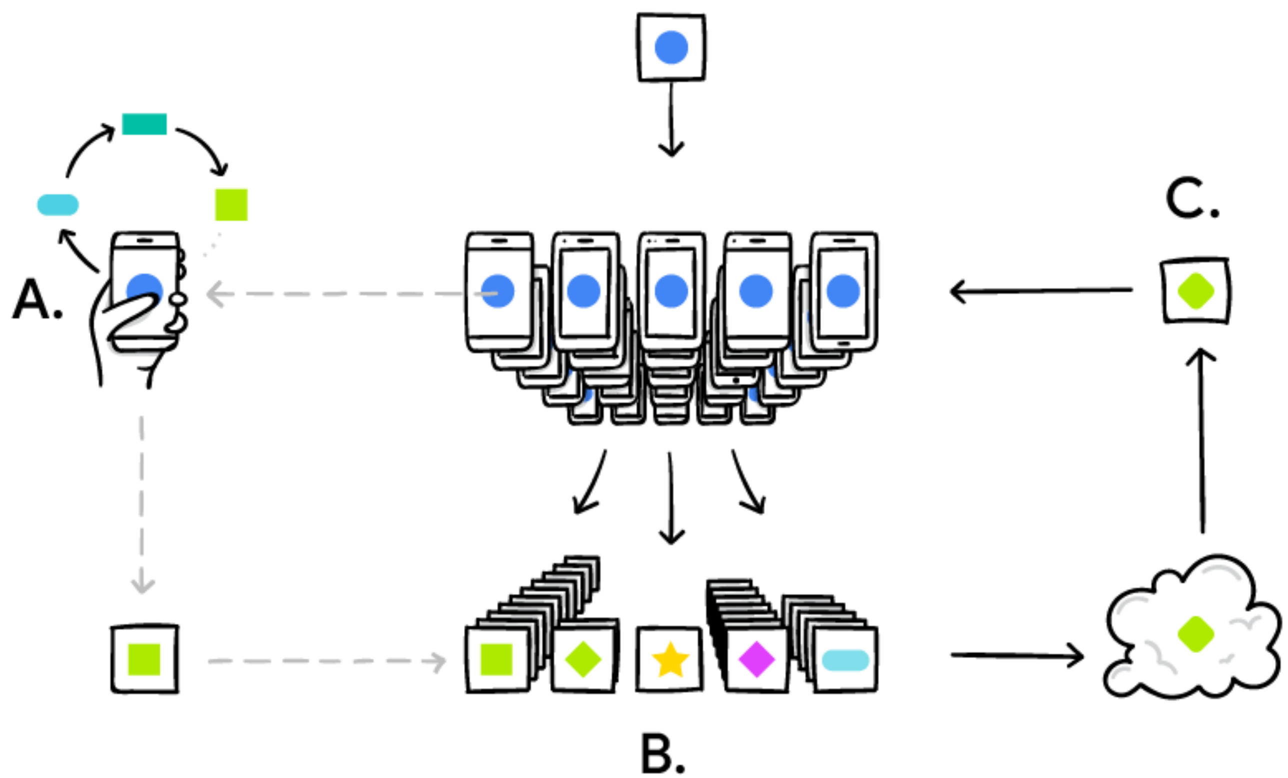
B. - many users' updates are aggregated

C. - consensus change to the shared model

Let's make communications cheaper!

Less often **Smaller size**

Federated Learning



A. - phone personalizes the model locally, based on user's usage

B. - many users' updates are aggregated

C. - consensus change to the shared model

Let's make communications cheaper!

Less often

Smaller size

The Size Is Important!

The Size Is Important!

The Size Is Important!

Not so many fans

The Size Is Important!

Not so many fans → Everyone can go to the stadium and watch the game

The Size Is Important!

Not so many fans → Everyone can go to the stadium and watch the game

The Size Is Important!

~~Not so many fans~~ → Everyone can go to the stadium and watch the game

A lot of fans

The Size Is Important!

~~Not so many fans~~

~~Everyone can go to the stadium and watch the game~~

A lot of fans → Need to make a ballot for the tickets

The Size Is Important!

~~Not so many fans~~ ~~Everyone can go to the stadium and watch the game~~

A lot of fans → Need to make a <u>ballot</u> for the tickets

||

<u>**sparsification**</u>

Synchronous?

Synchronous?

Drawbacks of synchronous algorithms

Synchronous?

Drawbacks of synchronous algorithms

Wasting of time

All machines have to wait the
moment, when everyone finishes

Synchronous?

Drawbacks of synchronous algorithms

Wasting of time

All machines have to wait the
moment, when everyone finishes

A lot of communications in one time

Master machine communicates with
all machines in the same time

ASynchronous?

Drawbacks of synchronous algorithms

Wasting of time

All machines have to wait the
moment, when everyone finishes

A lot of communications in one time

Master machine communicates with
all machines in the same time

ASynchronous?

Drawbacks of synchronous algorithms

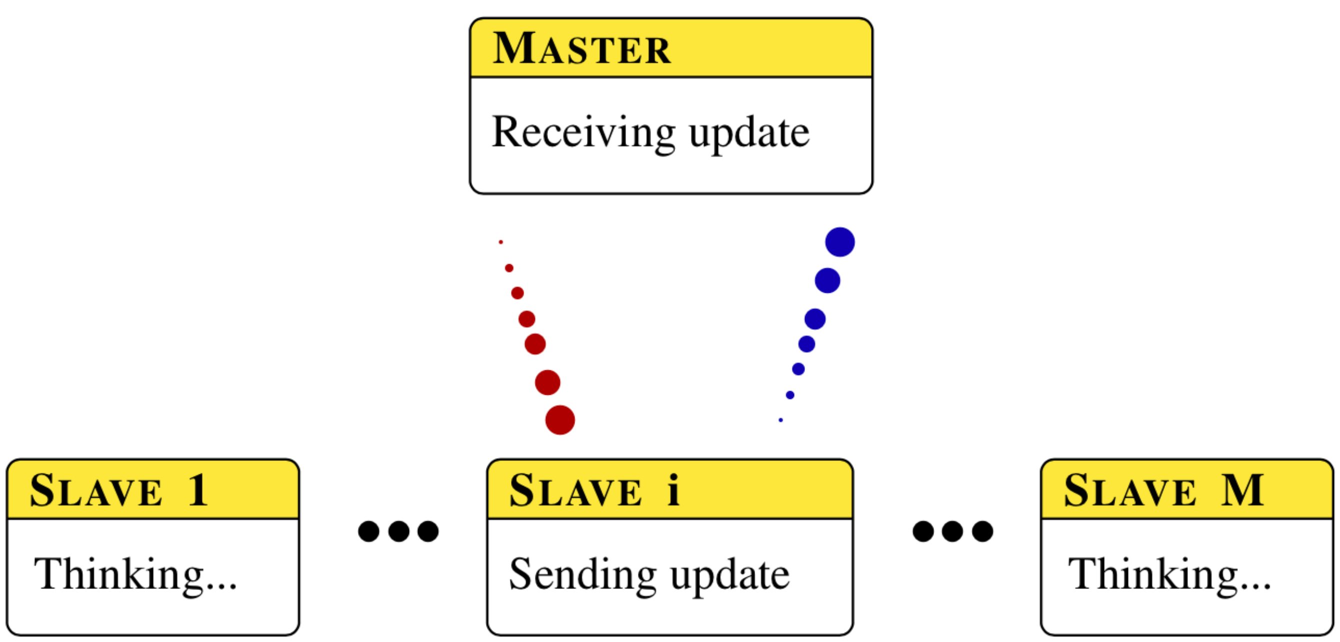
Wasting of time

All machines have to wait the moment, when everyone finishes

A lot of communications in one time

Master machine communicates with all machines in the same time

Let's kill 2 birds with one stone



Back to Basics

Problems of the form

$$\min_{x \in \mathbb{R}^n} f(x) + r(x)$$

convex
L-smooth

convex
non-smooth

Back to Basics

Problems of the form

$$\min_{x \in \mathbb{R}^n} f(x) + r(x)$$

convex
L-smooth

convex
non-smooth

Call for

First-order proximal methods (for example Proximal Gradient Descent)

Back to Basics

Problems of the form

$$\min_{x \in \mathbb{R}^n} f(x) + r(x)$$

convex
L-smooth

convex
non-smooth

Call for

First-order proximal methods (for example Proximal Gradient Descent)

$$x^{k+1} = \underset{\gamma g}{\mathbf{prox}}(x^k - \gamma \nabla f(x^k))$$

Back to Basics

Problems of the form

$$\min_{x \in \mathbb{R}^n} f(x) + r(x)$$

convex
L-smooth

convex
non-smooth

Call for

First-order proximal methods (for example Proximal Gradient Descent)

$$x^{k+1} = \mathbf{prox}_{\gamma g}(x^k - \gamma \nabla f(x^k))$$

where $\mathbf{prox}_{\gamma g}(x) := \operatorname{argmin}_u \left\{ g(u) + \frac{1}{2\gamma}\|u - x\|^2 \right\}$

Back to Basics

Problems of the form

$$\min_{x \in \mathbb{R}^n} f(x) + r(x)$$

convex
L-smooth

convex
non-smooth

Call for

First-order proximal methods (for example Proximal Gradient Descent)

$$x^{k+1} = \mathbf{prox}_{\gamma g}(x^k - \gamma \nabla f(x^k))$$

where $\mathbf{prox}_{\gamma g}(x) := \operatorname*{argmin}_u \left\{ g(u) + \frac{1}{2\gamma} \|u - x\|^2 \right\}$

In synchronous case it will be the same

Each worker send its gradient and master calculate the weighted average

Back to Basics

Problems of the form

$$\min_{x \in \mathbb{R}^n} f(x) + r(x)$$

convex
L-smooth

convex
non-smooth

Call for

First-order proximal methods (for example Proximal Gradient Descent)

$$x^{k+1} = \mathbf{prox}_{\gamma g}(x^k - \gamma \nabla f(x^k))$$

where $\mathbf{prox}_{\gamma g}(x) := \operatorname*{argmin}_u \left\{ g(u) + \frac{1}{2\gamma}\|u - x\|^2 \right\}$

In synchronous case it will be the same

Each worker send its gradient and master calculate the weighted average

$$F(x) = \sum_{i=1}^{M} \pi_i f_i(x)$$

Back to Basics

convex
L-smooth

Problems of the form

$$\min_{x \in \mathbb{R}^n} f(x) + r(x)$$

convex
non-smooth

Call for

First-order proximal methods (for example Proximal Gradient Descent)

$$x^{k+1} = \mathbf{prox}_{\gamma g}(x^k - \gamma \nabla f(x^k))$$

where $\quad \mathbf{prox}_{\gamma g}(x) := \operatorname*{argmin}_u \left\{ g(u) + \frac{1}{2\gamma} \|u - x\|^2 \right\}$

In synchronous case it will be the same

Each worker send its gradient and master calculate the weighted average

$$F(x) = \sum_{i=1}^{M} \pi_i f_i(x) \quad \longrightarrow \quad \nabla F(x) = \sum_{i=1}^{M} \pi_i \nabla f_i(x)$$

Delays

Delays

Asynchronous updates bring some delays

Gradient that master receive from worker is computed in one of previous iterate points

Delays

Asynchronous updates bring some delays

Gradient that master receive from worker is computed in one of previous iterate points

Let’s define a master’s timeline

Delays

Asynchronous updates bring some delays

Gradient that master receive from worker is computed in one of previous iterate points

Let's define a master's timeline

k **- the number of updates master receive**

i^k**- an agent, that communicated with master at time**

Delays

Asynchronous updates bring some delays

Gradient that master receive from worker is computed in one of previous iterate points

Let's define a master's timeline

k **- the number of updates master receive**

i^k**- an agent, that communicated with master at time**

Let's define a worker i timeline

Delays

Asynchronous updates bring some delays

Gradient that master receive from worker is computed in one of previous iterate points

Let's define a master's timeline

k **- the number of updates master receive**

i^k**- an agent, that communicated with master at time**

Let's define a worker i timeline

d_i^k**- time elapsed from the last update**

D_i^k**- the time of penultimate update**

Delays

Asynchronous updates bring some delays

Gradient that master receive from worker is computed in one of previous iterate points

Let's define a master's timeline

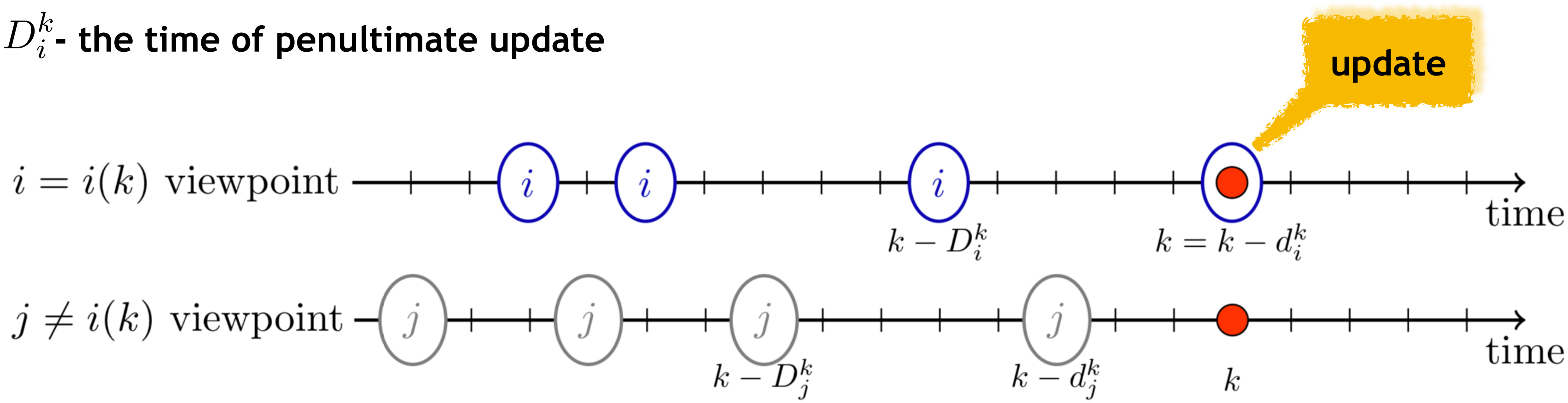
k **- the number of updates master receive**

i^k**- an agent, that communicated with master at time**

Let's define a worker i timeline

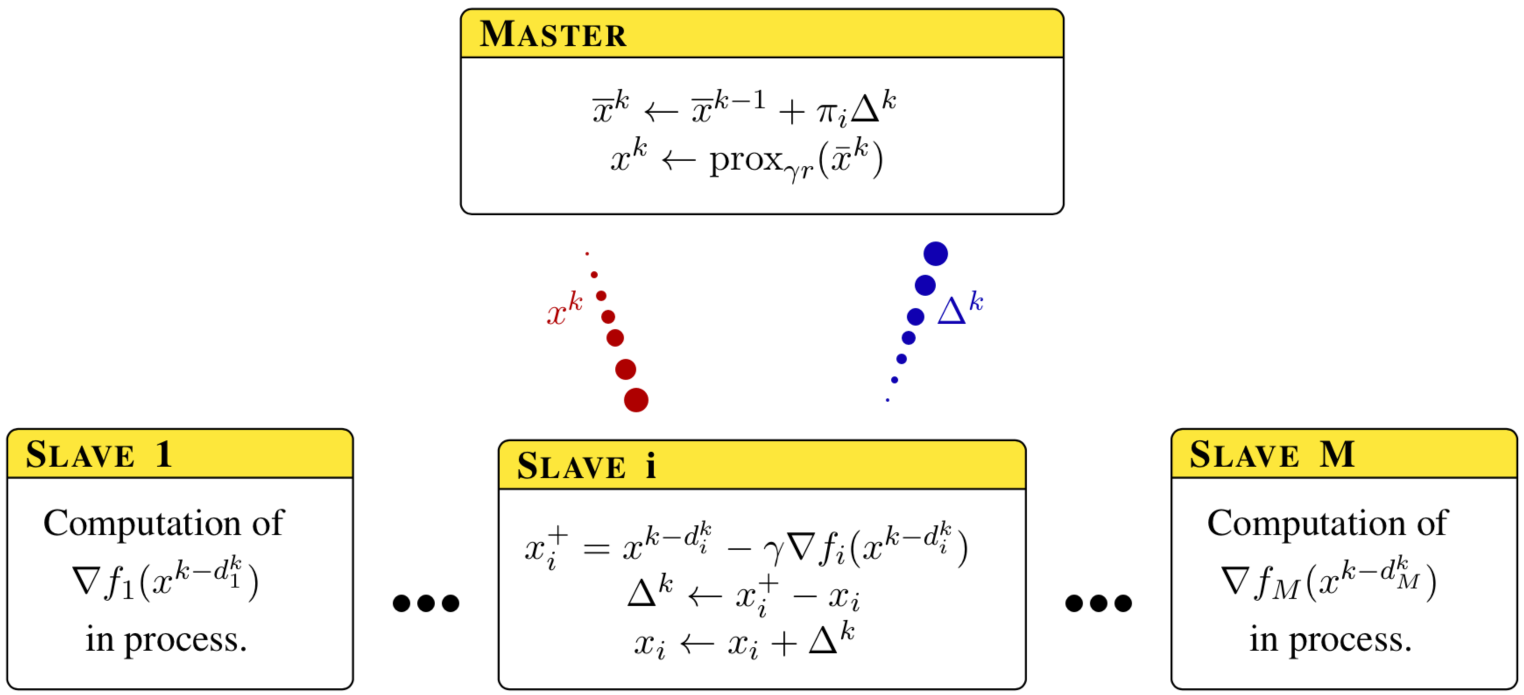
d_i^k**- time elapsed from the last update**

D_i^k**- the time of penultimate update**



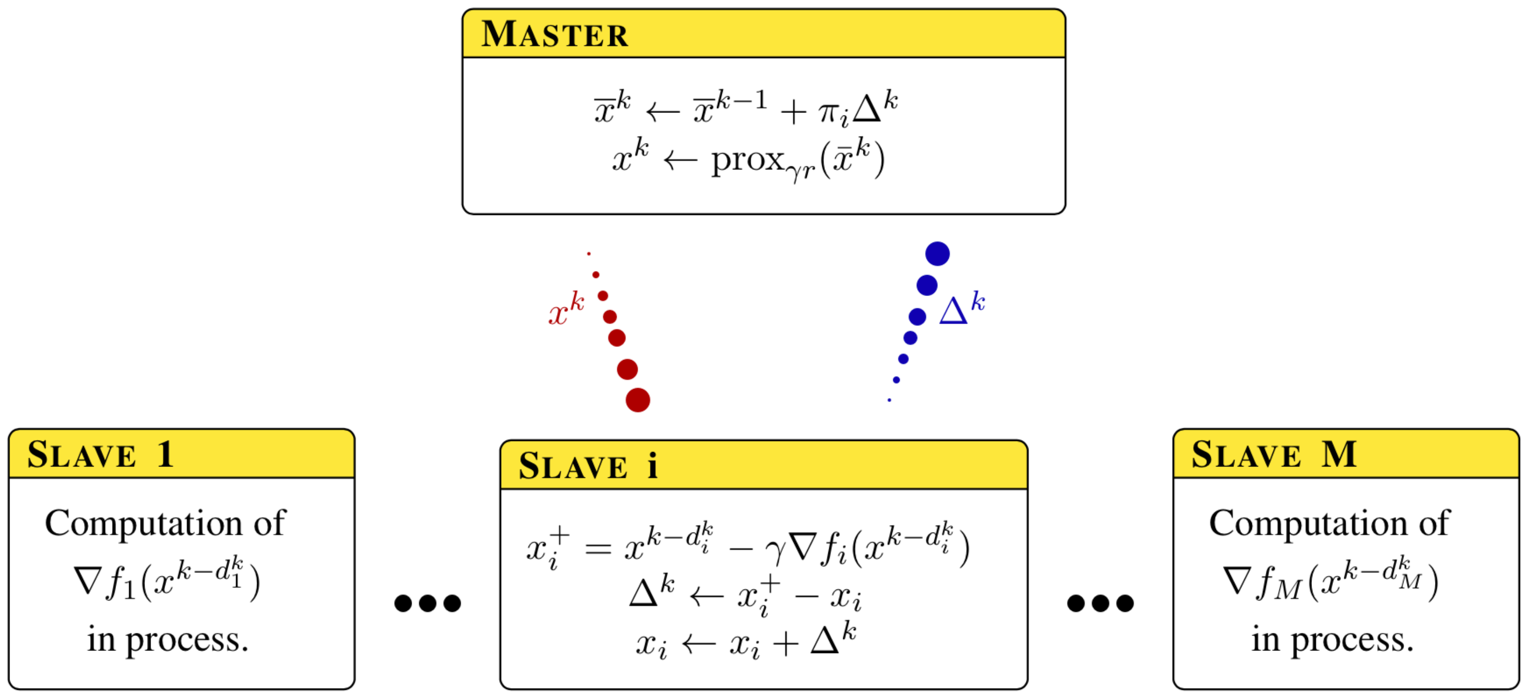
Non-sparsified Algorithm

Non-sparsified Algorithm



Mishchenko, K., Iutzeler, F., Malick, J., & Amini, M. R. (2018, July).
A delay-tolerant proximal-gradient algorithm for distributed learning.
In *International Conference on Machine Learning* (pp. 3584-3592).

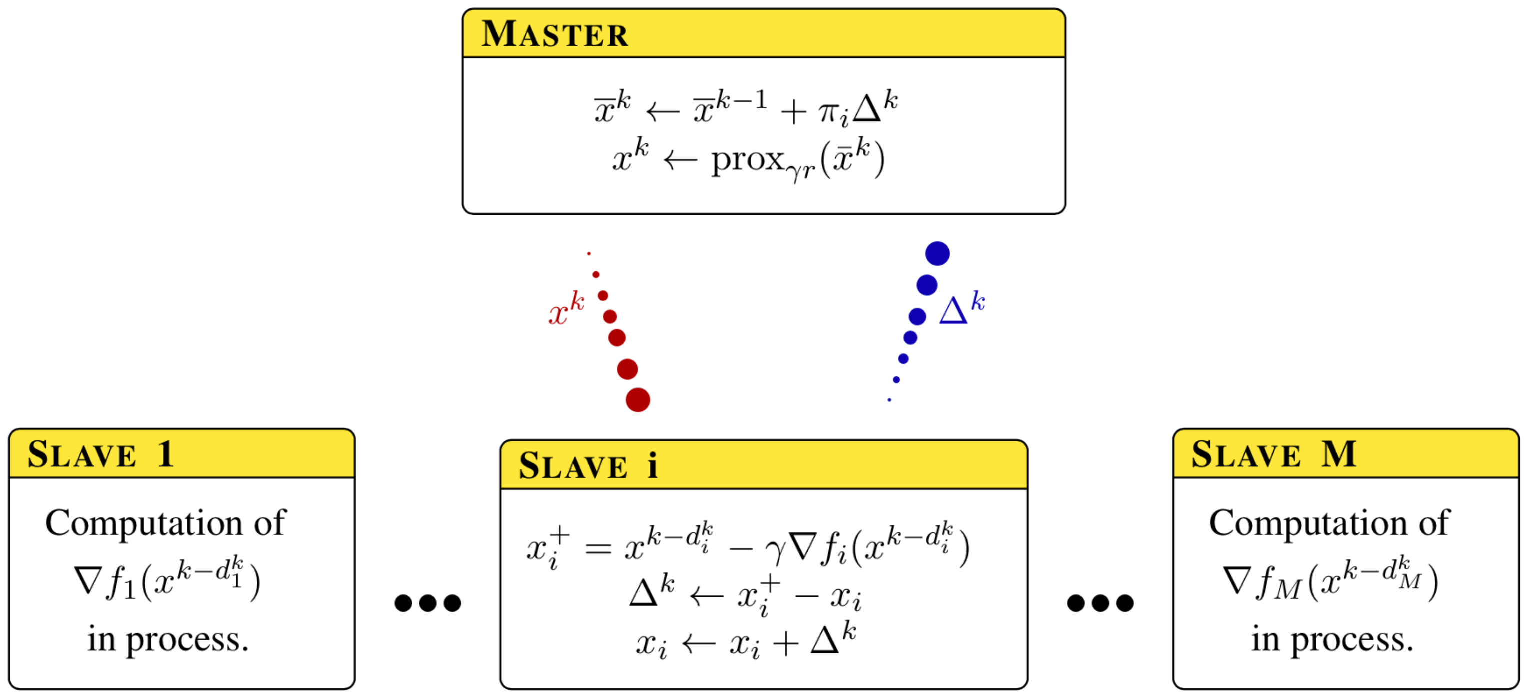
Non-sparsified Algorithm



If dimension is very big, it is very expansive to send full gradient

Mishchenko, K., Iutzeler, F., Malick, J., & Amini, M. R. (2018, July).
A delay-tolerant proximal-gradient algorithm for distributed learning.
In *International Conference on Machine Learning* (pp. 3584-3592).

Non-sparsified Algorithm



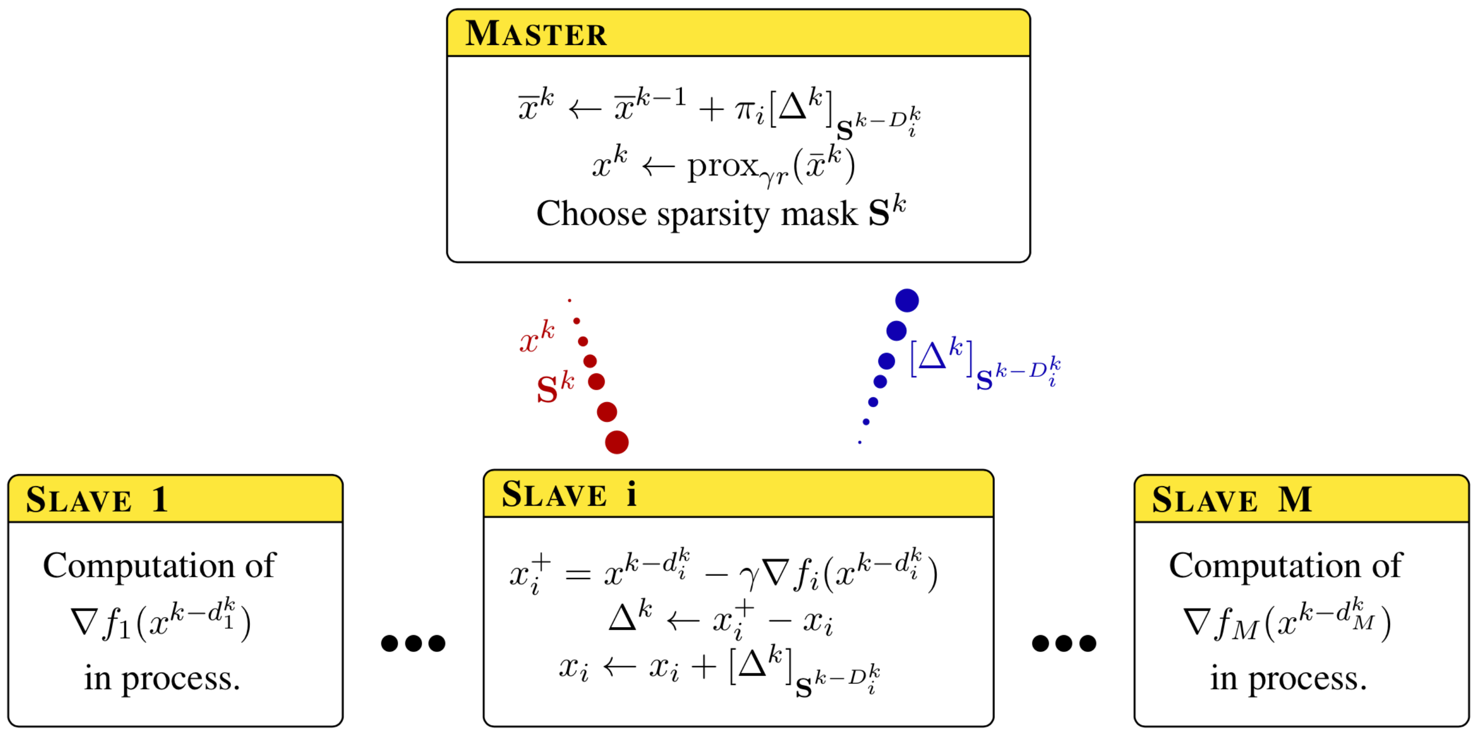
If dimension is very big, it is very expansive to send full gradient

If regulariser is sparsity enforcing, no need to sparsify master $\rightarrow$ worker

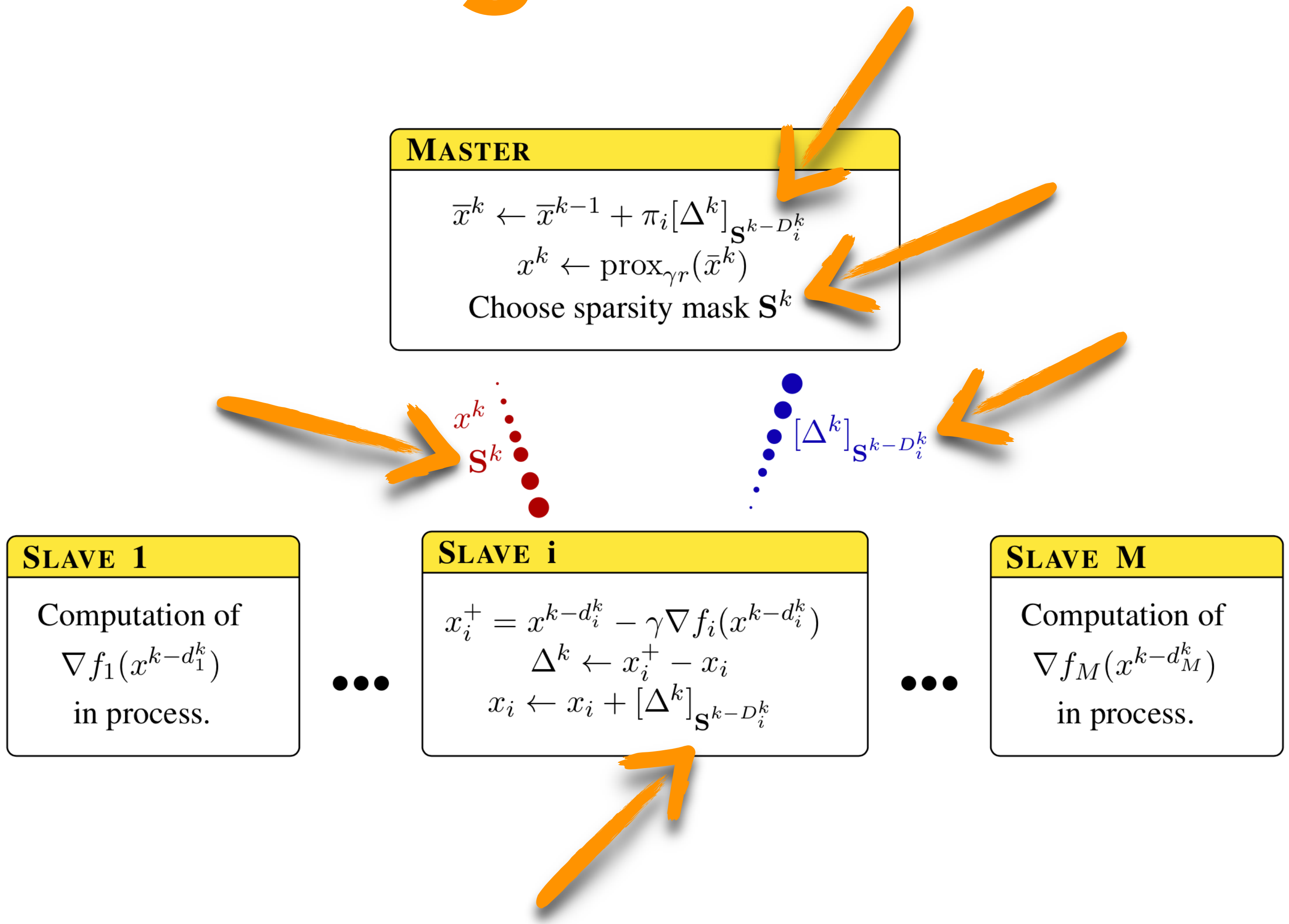
Mishchenko, K., Iutzeler, F., Malick, J., & Amini, M. R. (2018, July).
A delay-tolerant proximal-gradient algorithm for distributed learning.
In *International Conference on Machine Learning* (pp. 3584-3592).

Fadili J., Malick J., Peyré G. Sensitivity analysis for mirror-stratifiable convex functions
SIAM Journal on Optimization. – 2018. – T. 28. – №. 4. – C. 2975-3000.

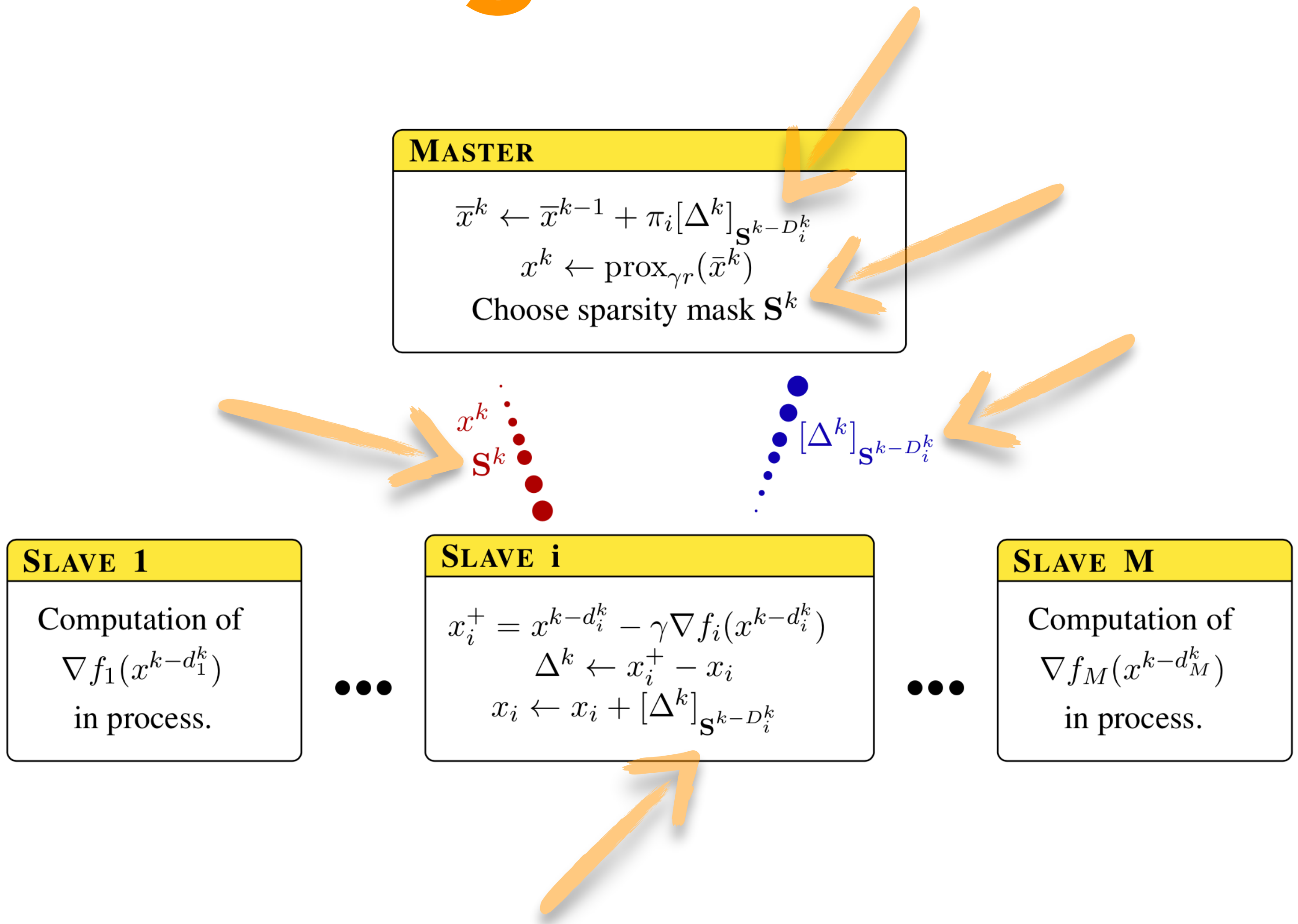
Sparsified Algorithm



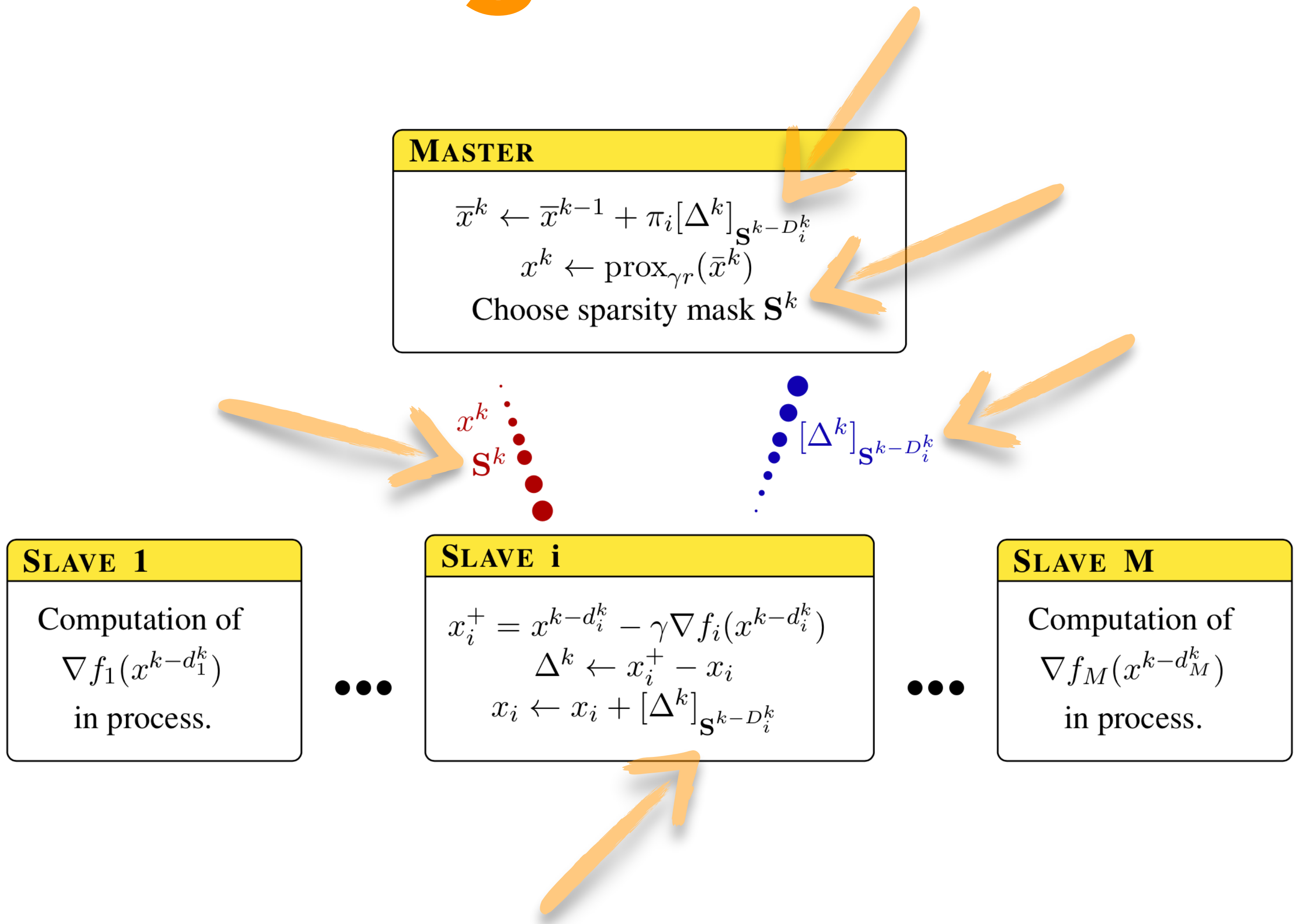
Sparsified Algorithm



Sparsified Algorithm

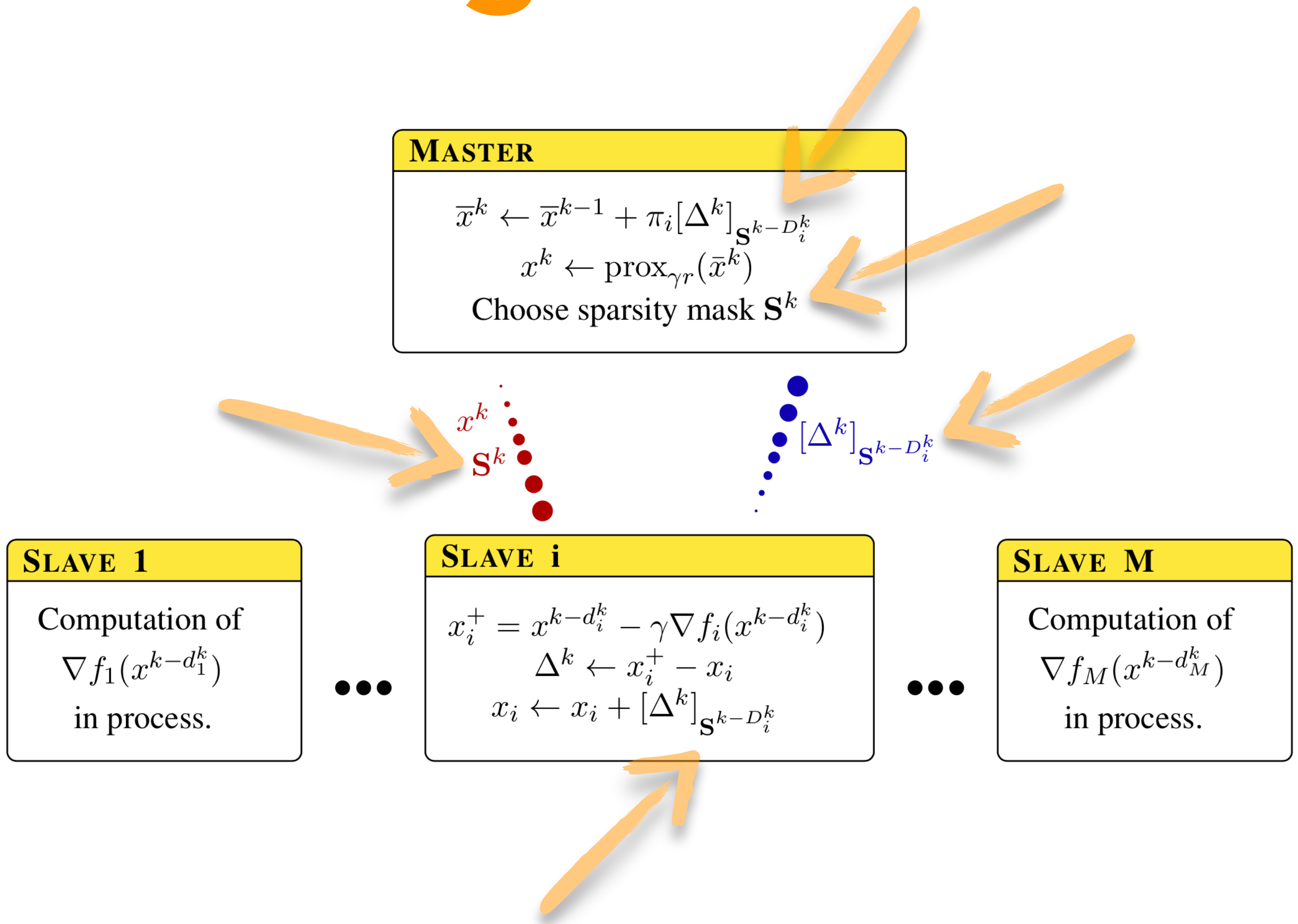


Sparsified Algorithm



How to chose the sparsity mask?

Sparsified Algorithm

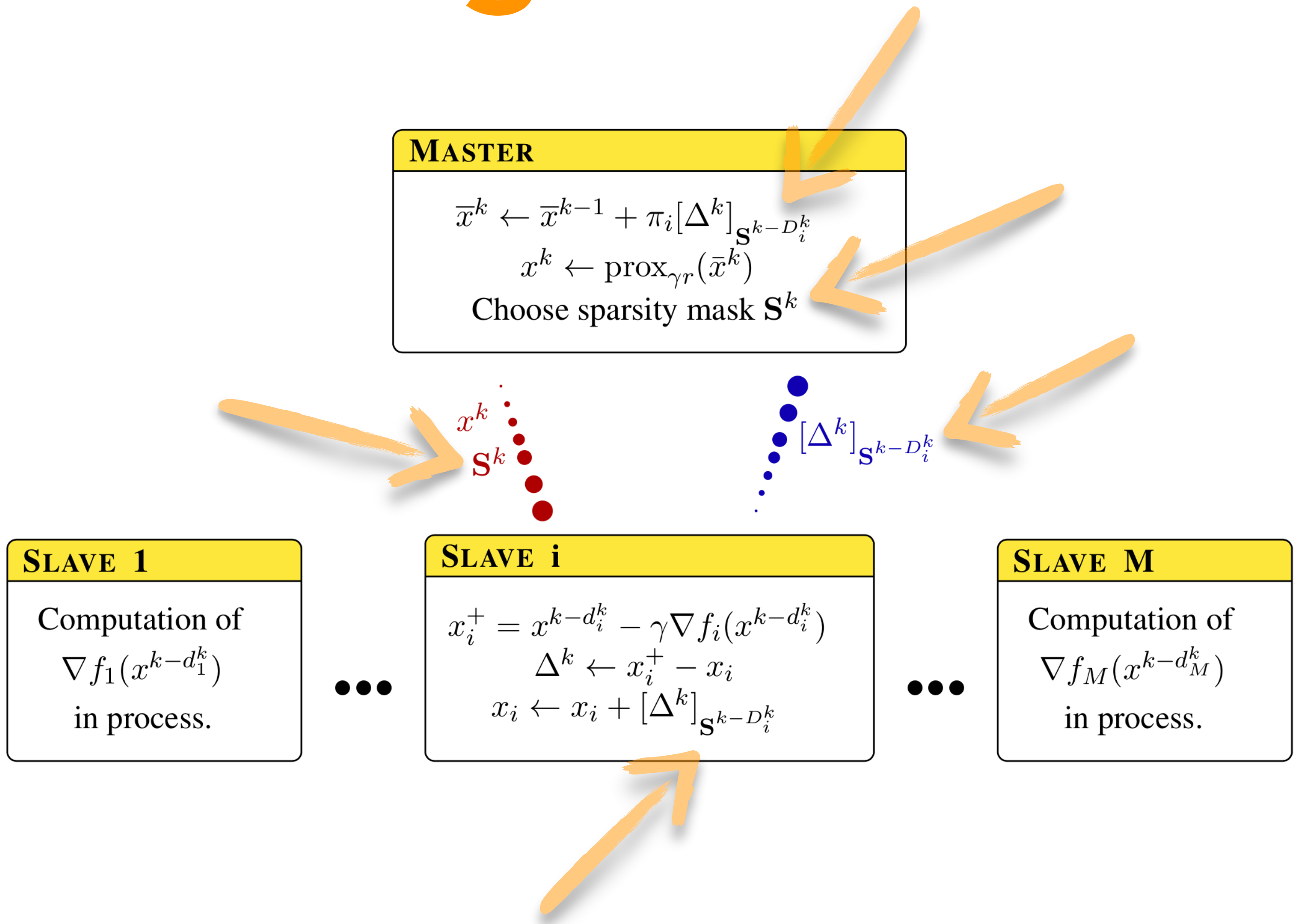


How to chose the sparsity mask?

Random uniform coordinate selection (Coordinate Descent with uniform probabilities)

Nesterov Y. Efficiency of coordinate descent methods on huge-scale optimization problems
SIAM Journal on Optimization. – 2012. – Т. 22. – №. 2. – С. 341-362.

Sparsified Algorithm



How to chose the sparsity mask?

Random uniform coordinate selection (Coordinate Descent with uniform probabilities)

OR…

Nesterov Y. Efficiency of coordinate descent methods on huge-scale optimization problems
SIAM Journal on Optimization. – 2012. – T. 22. – №. 2. – C. 341-362.

Back to Ballot

It’s an example of identification from real world

Back to Ballot

1 wave - most loyal fans (the biggest amount of games visited)

It’s an example of identification from real world

Back to Ballot

1 wave - most loyal fans (the biggest amount of games visited)

2 wave (after cancellations) - uniformly random from all

It’s an example of identification from real world

Back to Ballot

1 wave - most loyal fans (the biggest amount of games visited)

2 wave (after cancellations) - uniformly random from all

This system gives a chance for everyone to watch a game

It’s an example of identification from real world

Back to Ballot

1 wave - most loyal fans (the biggest amount of games visited)

2 wave (after cancellations) - uniformly random from all

This system gives a chance for everyone to watch a game

Most loyal have probability 1, all the others much smaller

It’s an example of identification from real world

l_1-regularizer*

* - l_1-regularizer enforces coordinate sparsity

l_1-regularizer*

Let's introduce our adaptive way of mask selection

*** - l_1-regularizer enforces coordinate sparsity**

l_1-regularizer*

Let's introduce our adaptive way of mask selection

1 - select all the coordinates that are in master's current iterate

*** - l_1-regularizer enforces coordinate sparsity**

l_1-regularizer*

Let's introduce our adaptive way of mask selection

1 - select all the coordinates that are in master's current iterate

2 - add some other coordinates with uniform probability

*** - l_1-regularizer enforces coordinate sparsity**

l_1-regularizer*

Let's introduce our adaptive way of mask selection

1 - select all the coordinates that are in master's current iterate

2 - add some other coordinates with uniform probability

As a result, all the coordinates from the support of the final solution would be selected

*** - l_1-regularizer enforces coordinate sparsity**

l_1-regularizer*

Let's introduce our adaptive way of mask selection

1 - select all the coordinates that are in master's current iterate

2 - add some other coordinates with uniform probability

As a result, all the coordinates from the support of the final solution would be selected

Dimension reductions without loss of speed (but only from some <u>finite</u> moment of time)

* - l_1-regularizer enforces coordinate sparsity

l_1-regularizer*

Let's introduce our adaptive way of mask selection

1 - select all the coordinates that are in master's current iterate

2 - add some other coordinates with uniform probability

As a result, all the coordinates from the support of the final solution would be selected

Dimension reductions without loss of speed (but only from some <u>finite</u> moment of time)

Theoretical result (for s.c. objective)

Grishchenko, D., Iutzeler, F., Malick, J., & Amini, M. R. (2018).
Asynchronous Distributed Learning with Sparse Communications and Identification.
arXiv preprint arXiv:1812.03871.

*** - l_1-regularizer enforces coordinate sparsity**

l_1-regularizer*

Let's introduce our adaptive way of mask selection

1 - select all the coordinates that are in master's current iterate

2 - add some other coordinates with uniform probability

As a result, all the coordinates from the support of the final solution would be selected

Dimension reductions without loss of speed (but only from some finite moment of time)

Theoretical result (for s.c. objective)

$$\mathbb{E}\|x^k - x^\star\|^2 \leq \left(1 - 2\frac{\gamma p \mu L}{\mu + L}\right)^m \max_{i=1,..,M} \|x_i^0 - x_i^\star\|^2$$

where $k \in [k_m, k_{m+1})$ **and** $k_{m+1} = \min\left\{k : k - D_i^k \geq k_m \text{ for all } i\right\}$

Grishchenko, D., Iutzeler, F., Malick, J., & Amini, M. R. (2018).
Asynchronous Distributed Learning with Sparse Communications and Identification.
arXiv preprint arXiv:1812.03871.

*** - l_1-regularizer enforces coordinate sparsity**

Numerical Experiments

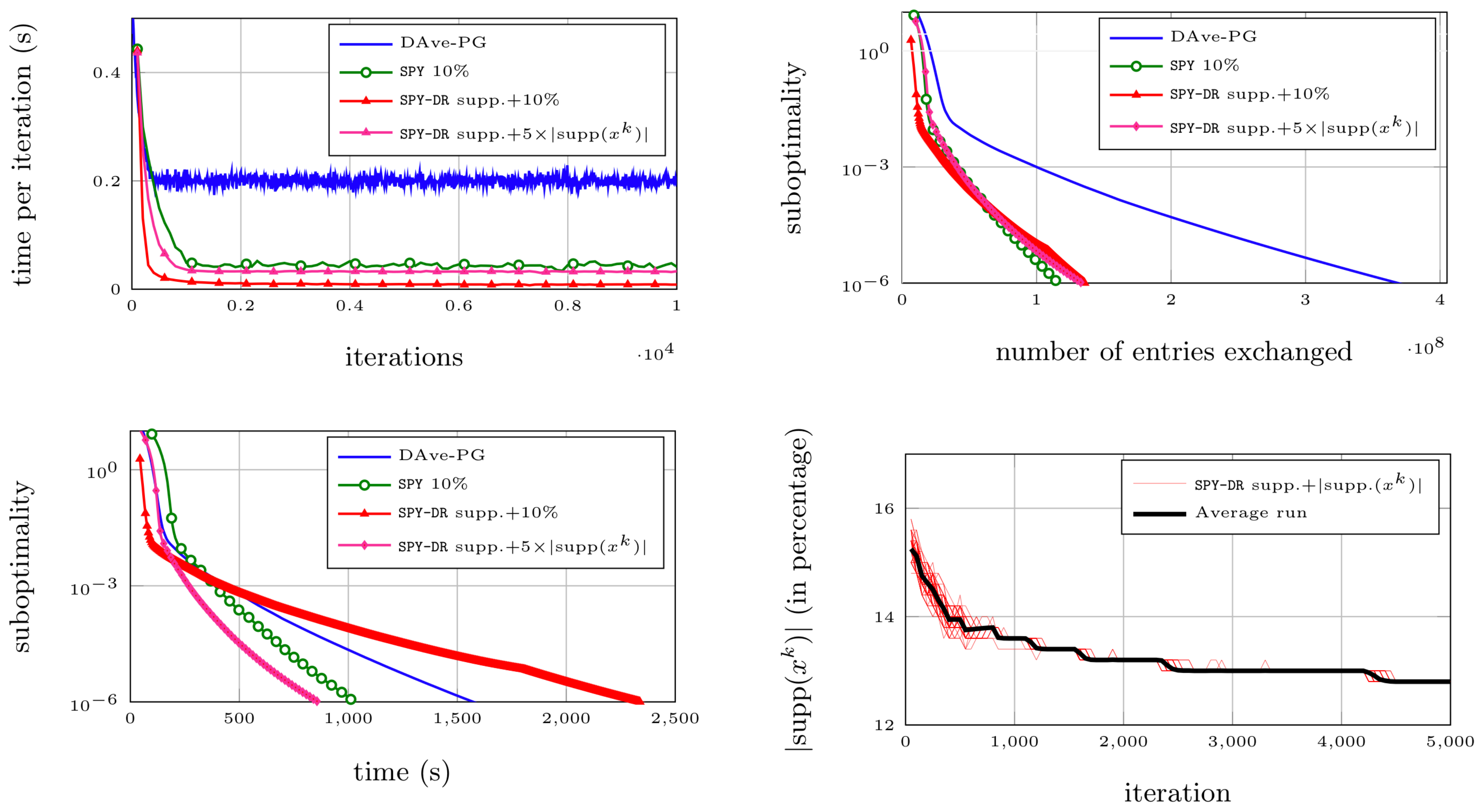


Figure 1: Logistic regression for the rcv1 dataset: evolution of the time per iteration, wallclock time performance, suboptimality vs communication, and robustness of identification.

Thank You For Your Attention