Throughput-Optimal Topology Design for Cross-Silo Federated Learning

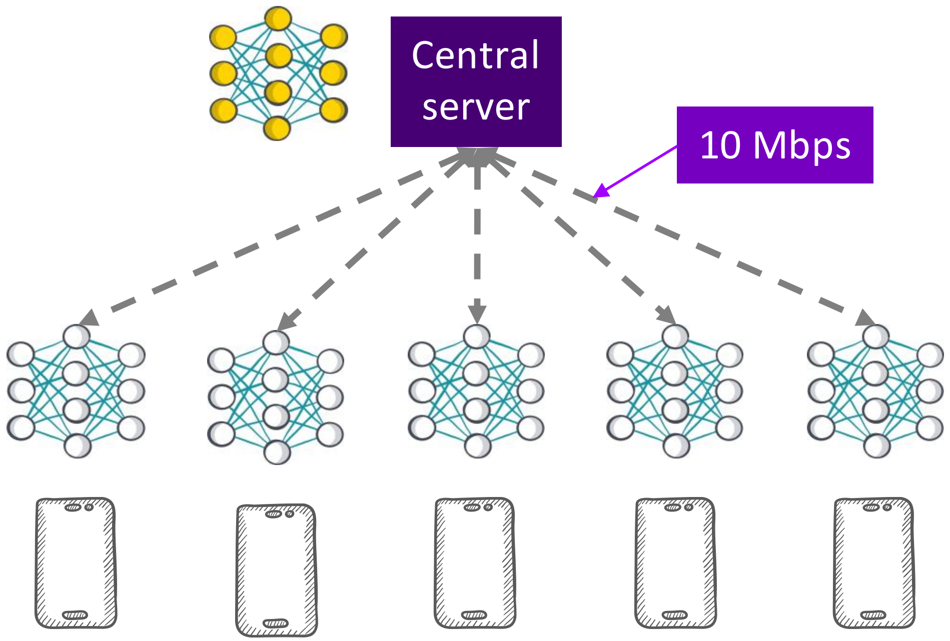
Othmane Marfoq (Inria&Accenture), Chuan Xu (Inria),
Giovanni Neglia (Inria), Richard Vidal (Accenture)

Federated Learning

Federated learning involves "*Training statistical models over remote devices or siloed data centers, such as mobile phones or hospitals, while keeping data localized*" (Li et al. 2020).

Federated Learning

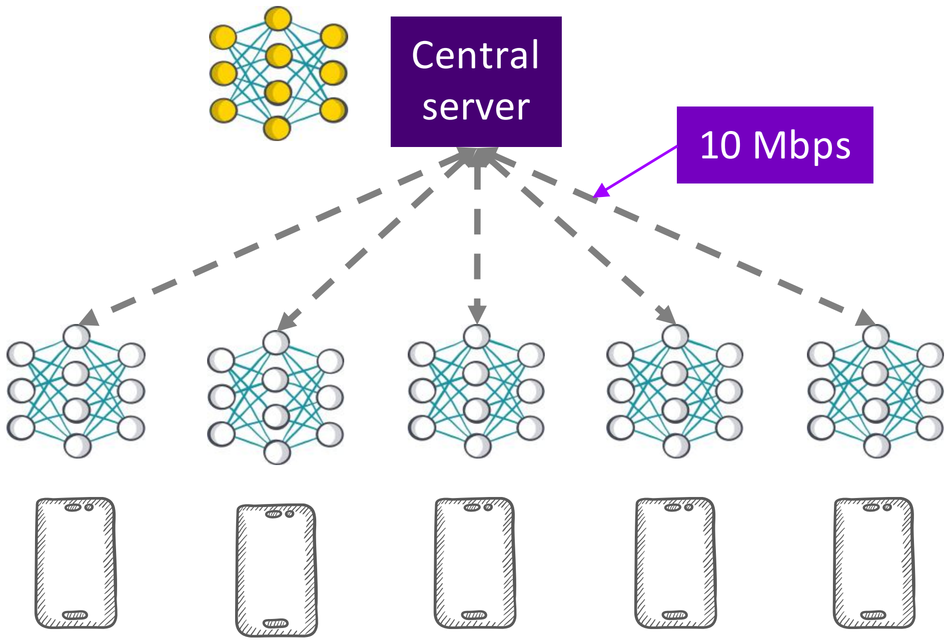
Federated learning involves "*Training statistical models over remote devices or siloed data centers, such as mobile phones or hospitals, while keeping data localized*" (Li et al. 2020).



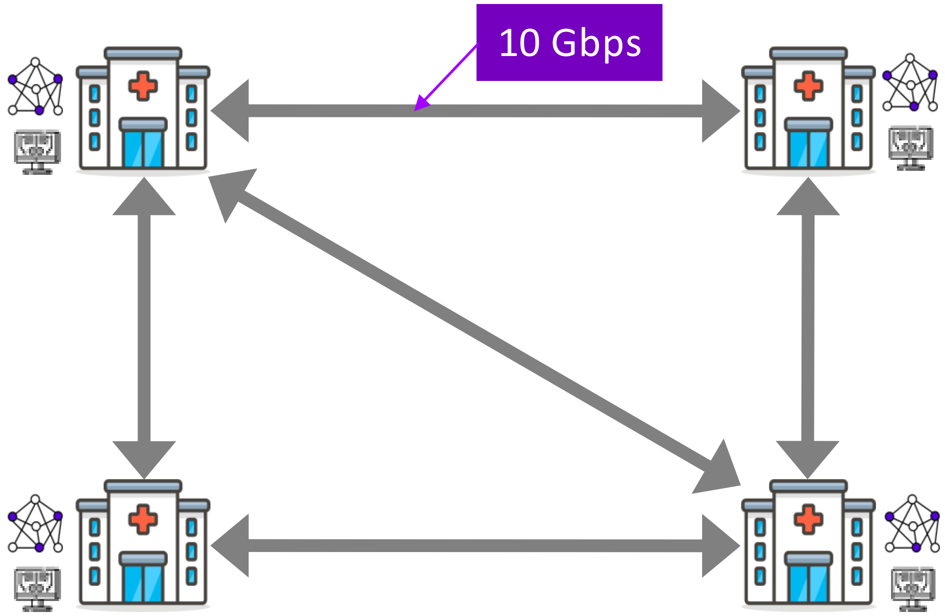
Cross-Device/Fixed STAR topology

Federated Learning

Federated learning involves "*Training statistical models over remote devices or siloed data centers, such as mobile phones or hospitals, while keeping data localized*" (Li et al. 2020).



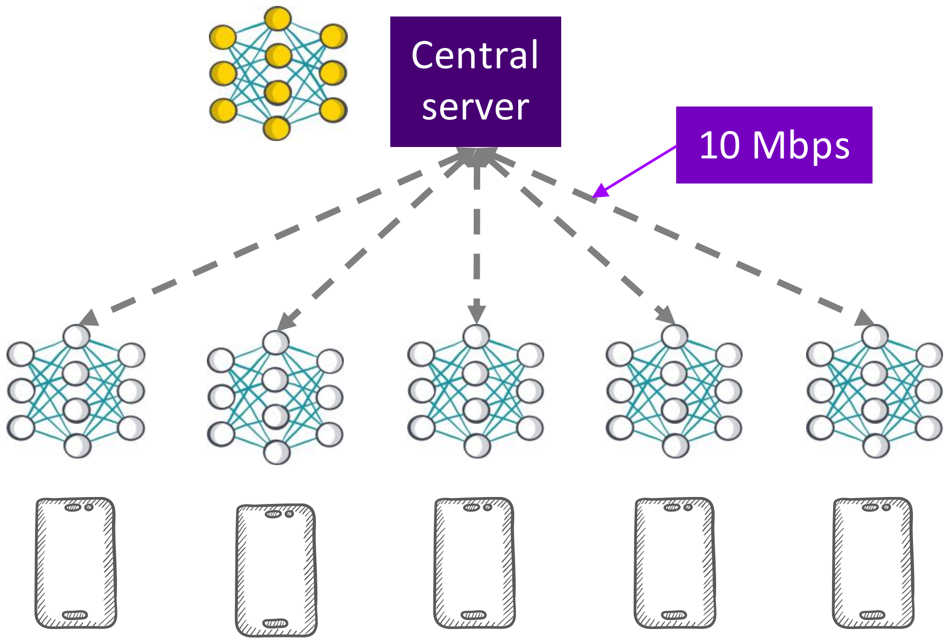
Cross-Device/Fixed STAR topology



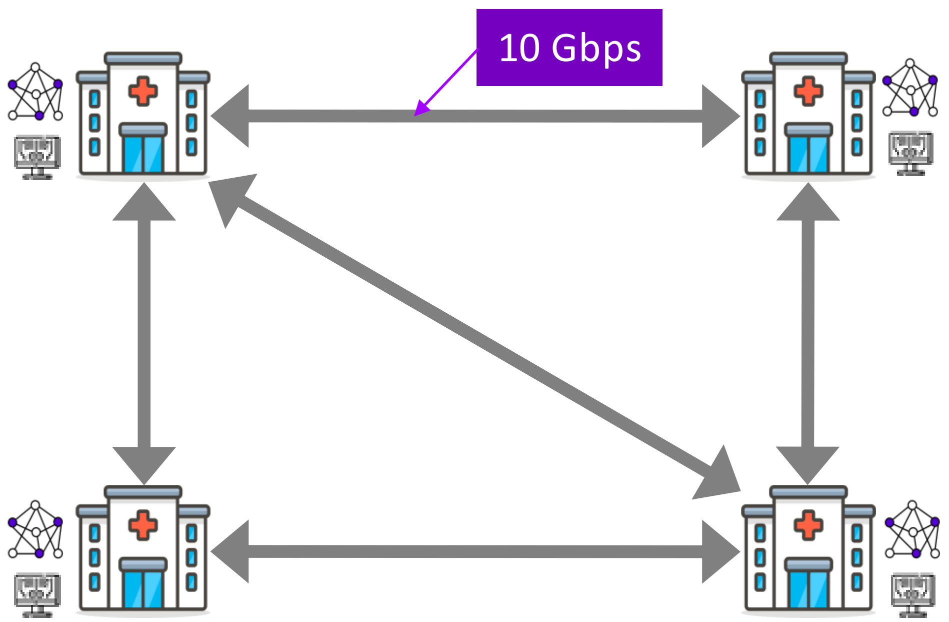
Cross-Silo/Flexible topology

Federated Learning

Federated learning involves "*Training statistical models over remote devices or siloed data centers, such as mobile phones or hospitals, while keeping data localized*" (Li et al. 2020).



Cross-Device/Fixed STAR topology



Cross-Silo/Flexible topology

Communication Topology & Training Time

$$Training\,Time = \#Iterations \; \times \; Iteration\,Time$$

Communication Topology & Training Time

$$Training\,Time = \#Iterations \times Iteration\,Time$$

The communication topology has two contrasting effects on the training time:

Communication Topology & Training Time

$$Training\ Time = \#Iterations \times Iteration\ Time$$

The communication topology has two contrasting effects on the training time:

More connected topology → Fewer iterations

Communication Topology & Training Time

$$Training\ Time = \#Iterations\ \times\ Iteration\ Time$$

The communication topology has two contrasting effects on the training time:

More connected topology

Fewer iterations

Slower iterations

Communication Topology & Training Time

$$Training\ Time = \#Iterations\ \times\ Iteration\ Time$$

The communication topology has two contrasting effects on the training time:

More connected topology → Fewer iterations

More connected topology → Slower iterations

Recent experimental and theoretical works suggest that the second effect may dominate the first one.

Communication Topology & Training Time

$$Training\ Time = \#Iterations \times \boxed{Iteration\ Time}$$

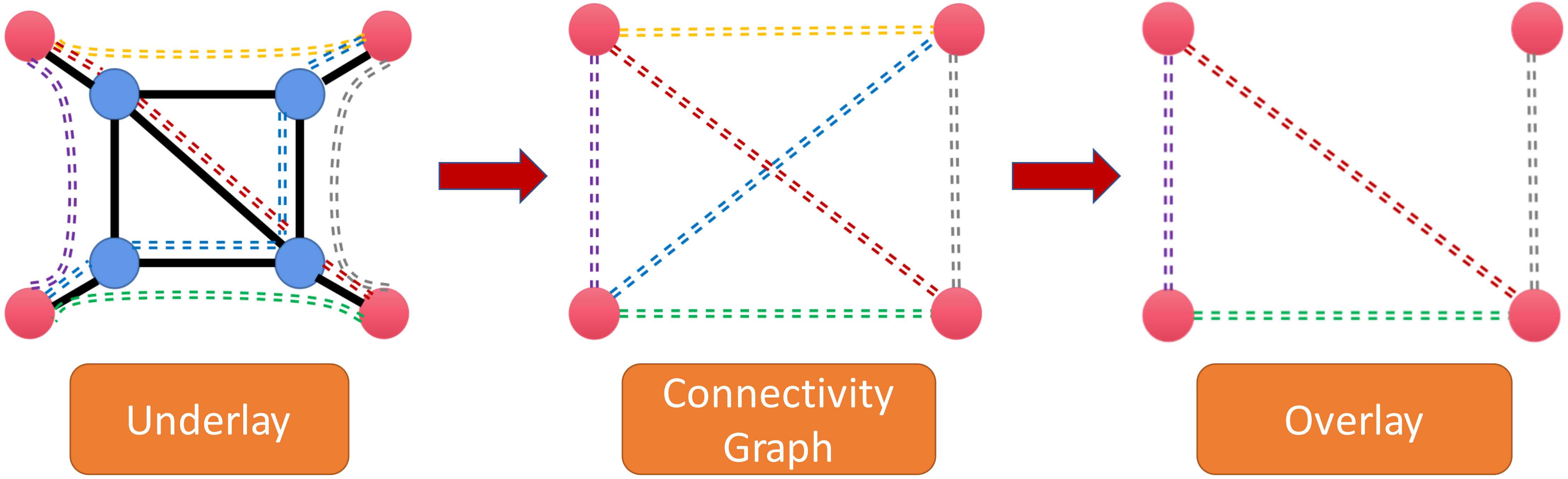
The communication topology has two contrasting effects on the training time:

More connected topology → Fewer iterations

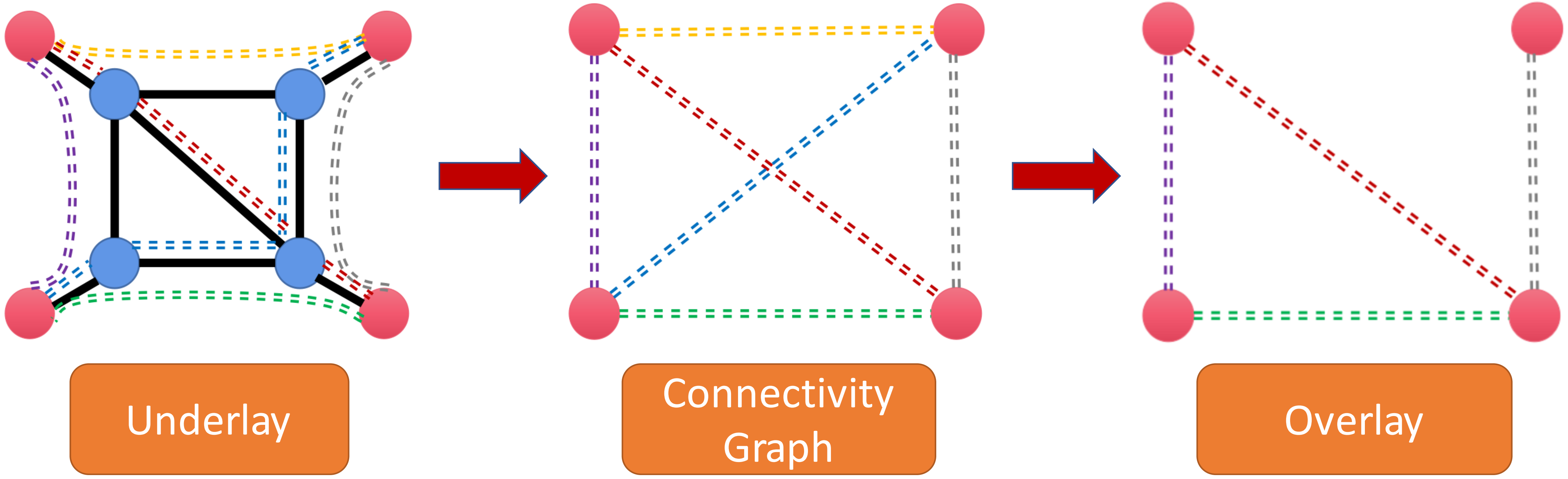
More connected topology → Slower iterations

Recent experimental and theoretical works suggest that the second effect may dominate the first one.

Problem Formulation

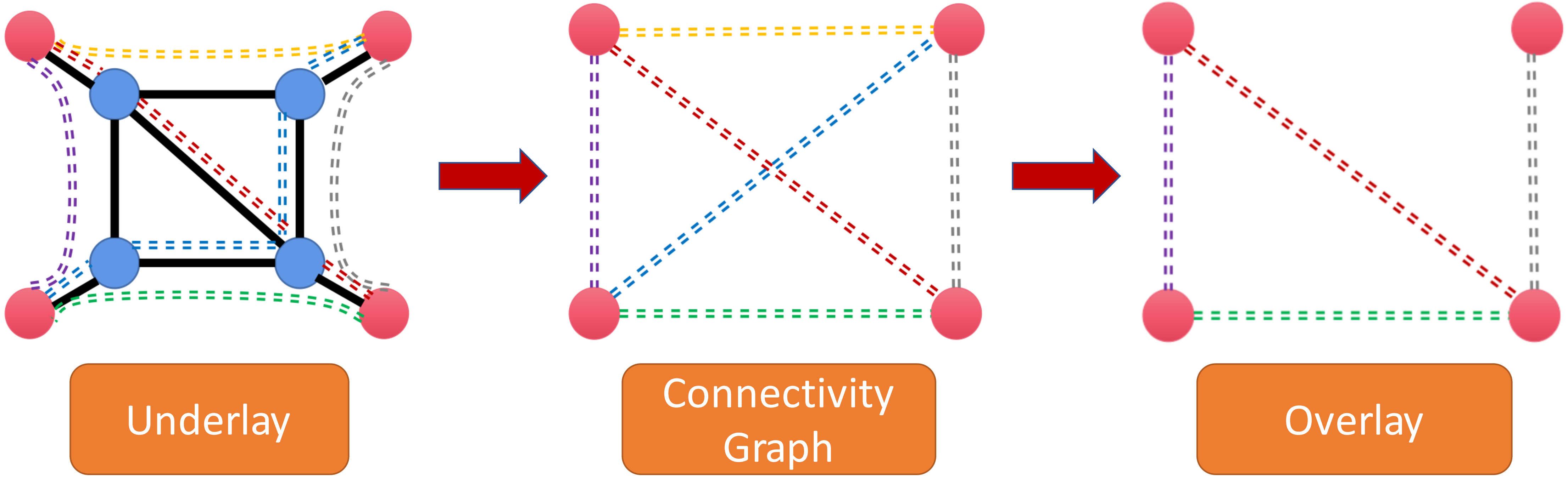


Problem Formulation



$$d_o(i,j) = s \times T_c(i) + l(i,j) + \frac{M}{\min\left(\frac{C_{\mathrm{UP}}(i)}{|\mathcal{N}_i^-|}, \frac{C_{\mathrm{DN}}(j)}{|\mathcal{N}_j^+|}, A(i',j')\right)}$$

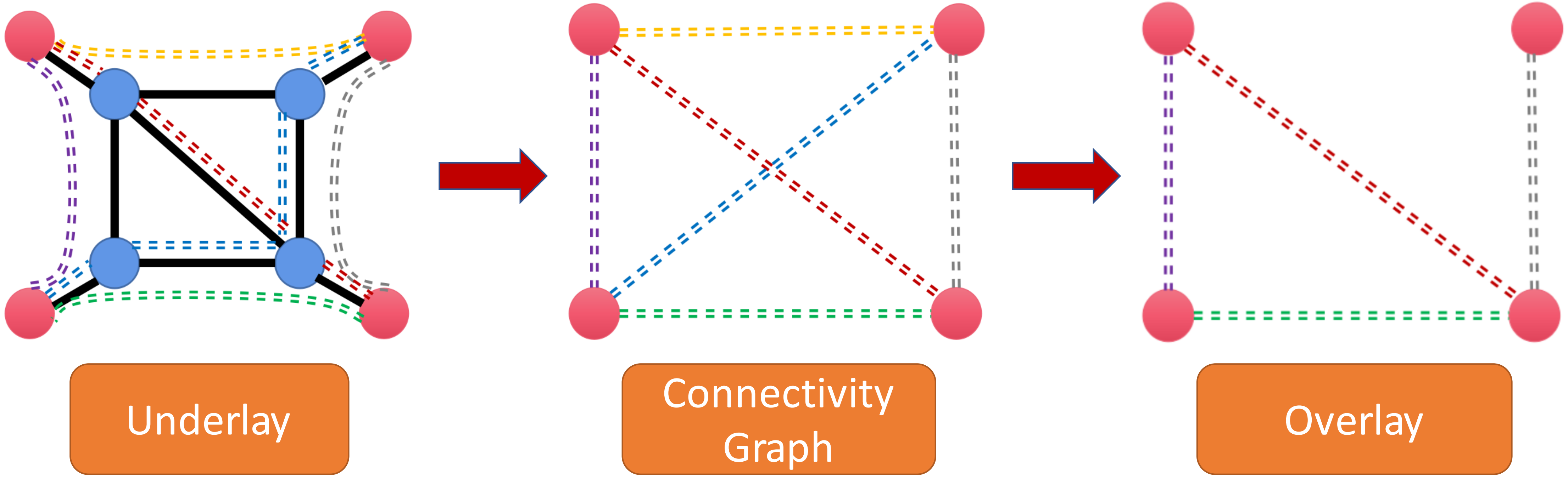
Problem Formulation



$$d_o(i,j) = s \times T_c(i) + \boxed{l(i,j)} + \frac{M}{\min\left(\frac{C_{\mathrm{UP}}(i)}{|\mathcal{N}_i^-|}, \frac{C_{\mathrm{DN}}(j)}{|\mathcal{N}_j^+|}, A(i',j')\right)}$$

Latency

Problem Formulation



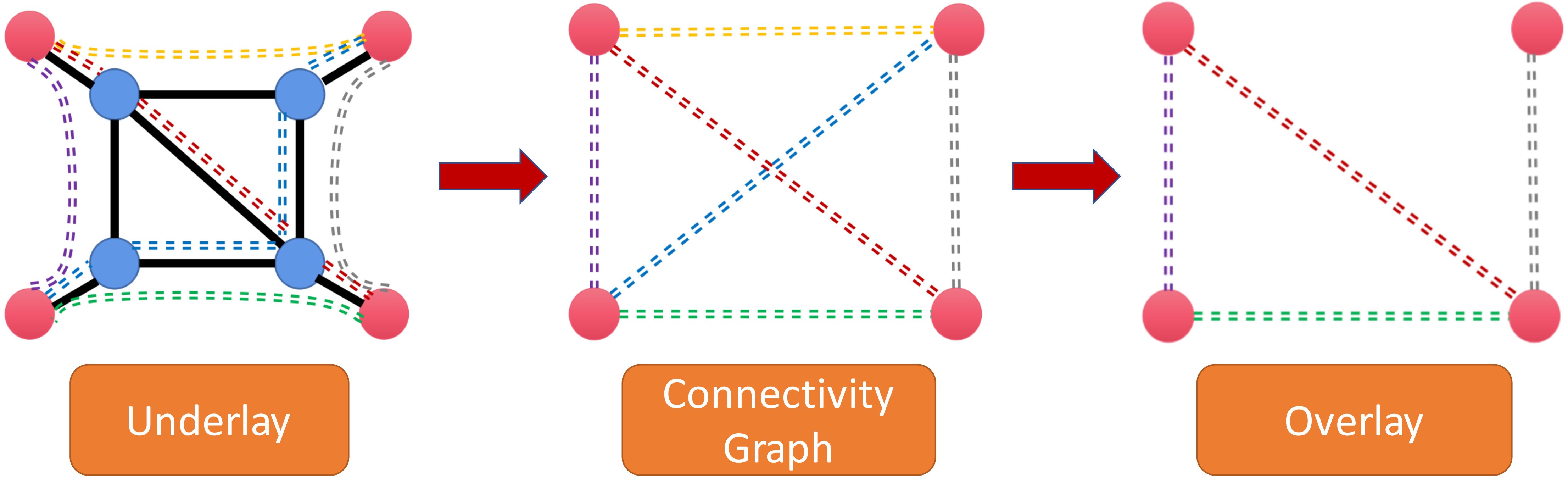
Model size

Capacities

$$d_o(i,j) = s \times T_c(i) + \boxed{l(i,j)} + \frac{\boxed{M}}{\boxed{\min\left(\frac{C_{\mathrm{UP}}(i)}{|\mathcal{N}_i^-|}, \frac{C_{\mathrm{DN}}(j)}{|\mathcal{N}_j^+|}, A(i',j')\right)}}$$

Latency

Problem Formulation



$$d_o(i,j) = \underbrace{\boxed{s \times T_c(i)}}_{\text{Computation Time}} + \underbrace{\boxed{l(i,j)}}_{\text{Latency}} + \frac{\overbrace{\boxed{M}}^{\text{Model size}}}{\underbrace{\boxed{\min\left(\frac{C_{\mathrm{UP}}(i)}{|\mathcal{N}_i^-|}, \frac{C_{\mathrm{DN}}(j)}{|\mathcal{N}_j^+|}, A(i',j')\right)}}_{\text{Capacities}}}$$

Problem Formulation

Each silo maintains a local copy of the model. At time $t_i(k)$ silo i starts its k-th iteration, it

1) updates the local model through minibatch gradient descent.
2) sends the new model to its out-neighbors in the overlay.
3) aggregates the models received from its in-neighbors into a new local model.

Problem Formulation

Each silo maintains a local copy of the model. At time $t_i(k)$ silo i starts its k-th iteration, it

1) updates the local model through minibatch gradient descent.
2) sends the new model to its out-neighbors in the overlay.
3) aggregates the models received from its in-neighbors into a new local model.

This is a **synchronous system.** The following recurrence holds:

$$t_i(k+1) \;=\; \max_{j \in \mathcal{N}_i^+ \cup \{i\}} \left(t_j(k) + d_o(i,j)\right)$$

Problem Formulation

The duration of an iteration at silo i is defined as $\tau_i = \lim_{k \to +\infty} t_i(k)/k$.

Problem Formulation

The duration of an iteration at silo i is defined as $\tau_i = \lim_{k \to +\infty} t_i(k)/k$.

Max-plus algebra & **synchronization theory** show that:

- τ_i does not depend on the specific silo.
- τ_i is the cycle time of the graph $\mathcal{G}_o$, defined as $\tau(\mathcal{G}_o) = \max_{\gamma} \frac{d_o(\gamma)}{|\gamma|}$,
  where γ is a circuit of $\mathcal{G}_o$.

Analysis

From Max-Plus algebra & Synchronization theory

Minimal Cycle Time (MCT)

Input: A strong directed graph $G_c = (V, E_c)$,

$\{C_{UP},(i) C_{DN}(j), l(i,j), A(i',j'), T_c(i), \forall\ (i,j) \in\ E_c\}$

Output: Strong spanning subdigraph of G_c with minimal cycle time

Analysis

<u>Minimal Cycle Time (MCT)</u>

Input: A strong directed graph $G_c = (V, E_c)$,

$\{C_{UP}, (i) C_{DN}(j), l(i,j), A(i',j'), T_c(i), \forall\ (i,j) \in\ E_c\}$

Output: Strong spanning subdigraph of G_c with minimal cycle time

From Max-Plus algebra & Synchronization theory

Table 1: Algorithms to design the overlay $\mathcal{G}_o$ from the connectivity graph $\mathcal{G}_c$.

Network	Conditions	Algorithm	Complexity	Guarantees
Edge-capacitated	Undirected $\mathcal{G}_o$	Prim's Algorithm [80]	$\mathcal{O}(\lvert\mathcal{E}_c\rvert + \lvert\mathcal{V}\rvert \log \lvert\mathcal{V}\rvert)$	Optimal solution (Prop. 3.1)
Edge/Node-capacitated	Euclidean $\mathcal{G}_c$	Christofides' Algorithm [69]	$\mathcal{O}(\lvert\mathcal{V}\rvert^2 \log \lvert\mathcal{V}\rvert)$	$3N$-approximation (Prop. 3.3,3.6)
Node-capacitated	Euclidean $\mathcal{G}_c$ and undirected $\mathcal{G}_o$	Algorithm 1 (App. D)	$\mathcal{O}(\lvert\mathcal{E}_c\rvert\lvert\mathcal{V}\rvert \log \lvert\mathcal{V}\rvert)$	6-approximation (Prop. 3.5)

Analysis

From Max-Plus algebra & Synchronization theory

Minimal Cycle Time (MCT)

Input: A strong directed graph $G_c = (V, E_c)$,

$\{C_{UP},(i)C_{DN}(j), l(i,j), A(i',j'), T_c(i), \forall\,(i,j) \in E_c\}$

Output: Strong spanning subdigraph of G_c with minimal cycle time

The proposed algorithms output either a **ring** or a **tree** with constrained degree.

Table 1: Algorithms to design the overlay $\mathcal{G}_o$ from the connectivity graph $\mathcal{G}_c$.

Network	Conditions	Algorithm	Complexity	Guarantees
Edge-capacitated	Undirected $\mathcal{G}_o$	Prim's Algorithm [80]	$\mathcal{O}(\lvert\mathcal{E}_c\rvert + \lvert\mathcal{V}\rvert \log \lvert\mathcal{V}\rvert)$	Optimal solution (Prop. 3.1)
Edge/Node-capacitated	Euclidean $\mathcal{G}_c$	Christofides' Algorithm [69]	$\mathcal{O}(\lvert\mathcal{V}\rvert^2 \log \lvert\mathcal{V}\rvert)$	$3N$-approximation (Prop. 3.3,3.6)
Node-capacitated	Euclidean $\mathcal{G}_c$ and undirected $\mathcal{G}_o$	Algorithm 1 (App. D)	$\mathcal{O}(\lvert\mathcal{E}_c\rvert\lvert\mathcal{V}\rvert \log \lvert\mathcal{V}\rvert)$	6-approximation (Prop. 3.5)

Numerical Experiments

We considered three **real topologies** from *Rocketfuel engine* (**Exodus** and **Ebone**) and from *The Internet Topology Zoo* [48] (**Géant**), and two synthetic topologies (**AWS North-America** and **Gaia**) built from the geographical locations of *AWS* data centers.

Numerical Experiments

We considered three **real topologies** from *Rocketfuel engine* (**Exodus** and **Ebone**) and from *The Internet Topology Zoo* [48] (**Géant**), and two synthetic topologies (**AWS North-America** and **Gaia**) built from the geographical locations of *AWS* data centers.

Table 2: Datasets and Models. Mini-batch gradient computation time with NVIDIA Tesla P100.

Dataset	**Task**	**Samples** (x 10^3)	**Batch Size**	**Model**	**Parameters** (x 10^3)	**Model Size** (Mbits)	**Computation Time** (ms)
Shakespeare [14, 72]	Next-Character Prediction	$4,226$	512	Stacked-GRU [17]	840	3.23	389.6
FEMNIST [14]	Image classification	805	128	2-layers CNN	$1,207$	4.62	4.6
Sentiment140 [30]	Sentiment analysis	$1,600$	512	GloVe [82]+ LSTM [37]	$4,810$	18.38	9.8
iNaturalist [99]	Image classification	450	16	ResNet-18 [35]	$11,217$	42.88	25.4

Numerical Experiments

We considered three **real topologies** from *Rocketfuel engine* (**Exodus** and **Ebone**) and from *The Internet Topology Zoo* [48] (**Géant**), and two synthetic topologies (**AWS North-America** and **Gaia**) built from the geographical locations of *AWS* data centers.

Table 2: Datasets and Models. Mini-batch gradient computation time with NVIDIA Tesla P100.

Dataset	**Task**	**Samples** (x 10^3)	**Batch Size**	**Model**	**Parameters** (x 10^3)	**Model Size** (Mbits)	**Computation Time** (ms)
Shakespeare [14, 72]	Next-Character Prediction	$4,226$	512	Stacked-GRU [17]	840	3.23	389.6
FEMNIST [14]	Image classification	805	128	2-layers CNN	$1,207$	4.62	4.6
Sentiment140 [30]	Sentiment analysis	$1,600$	512	GloVe [82]+ LSTM [37]	$4,810$	18.38	9.8
iNaturalist [99]	Image classification	450	16	ResNet-18 [35]	$11,217$	42.88	25.4

Code: https://github.com/omarfoq/communication-in-cross-silo-fl

Numerical Experiments

Table 3: iNaturalist training over different networks. 1 Gbps core links capacities, 10 Gbps access links capacities. One local computation step ($s = 1$).

Network name	**Silos**	**Links**	**Cycle time (ms)**					**Ring's training speed-up**	
			STAR	MATCHA$^{(+)}$	MST	δ-MBST	RING	vs STAR	vs MATCHA$^{(+)}$
Gaia [36]	11	55	391	228 (228)	138	138	**118**	2.65	1.54 (1.54)
AWS North America [91]	22	231	288	124 (124)	90	90	**81**	3.41	1.47 (1.47)
Géant [27]	40	61	634	452 (106)	**101**	**101**	109	4.85	3.46 (0.81)
Exodus [64]	79	147	912	593 (142)	145	145	**103**	8.78	5.71 (1.37)
Ebone [64]	87	161	902	580 (123)	122	122	**95**	8.83	6.09 (1.29)

Numerical Experiments

Table 3: iNaturalist training over different networks. 1 Gbps core links capacities, 10 Gbps access links capacities. One local computation step ($s = 1$).

Network name	**Silos**	**Links**	**Cycle time (ms)**					**Ring's training speed-up**	
			STAR	MATCHA$^{(+)}$	MST	δ-MBST	RING	vs STAR	vs MATCHA$^{(+)}$
Gaia [36]	11	55	391	228 (228)	138	138	**118**	2.65	1.54 (1.54)
AWS North America [91]	22	231	288	124 (124)	90	90	**81**	3.41	1.47 (1.47)
Géant [27]	40	61	634	452 (106)	**101**	**101**	109	4.85	3.46 (0.81)
Exodus [64]	79	147	912	593 (142)	145	145	**103**	8.78	5.71 (1.37)
Ebone [64]	87	161	902	580 (123)	122	122	**95**	8.83	6.09 (1.29)

Numerical Experiments

Table 3: iNaturalist training over different networks. 1 Gbps core links capacities, 10 Gbps access links capacities. One local computation step ($s = 1$).

Network name	**Silos**	**Links**	**Cycle time (ms)**					**Ring's training speed-up**	
			STAR	MATCHA$^{(+)}$	MST	δ-MBST	RING	vs STAR	vs MATCHA$^{(+)}$
Gaia [36]	11	55	391	228 (228)	138	138	**118**	2.65	1.54 (1.54)
AWS North America [91]	22	231	288	124 (124)	90	90	**81**	3.41	1.47 (1.47)
Géant [27]	40	61	634	452 (106)	**101**	**101**	109	4.85	3.46 (0.81)
Exodus [64]	79	147	912	593 (142)	145	145	**103**	8.78	5.71 (1.37)
Ebone [64]	87	161	902	580 (123)	122	122	**95**	8.83	6.09 (1.29)

Numerical Experiments

Table 3: iNaturalist training over different networks. 1 Gbps core links capacities, 10 Gbps access links capacities. One local computation step ($s = 1$).

Network name	Silos	Links	Cycle time (ms)					Ring's training speed-up	
			STAR	MATCHA$^{(+)}$	MST	δ-MBST	RING	vs STAR	vs MATCHA$^{(+)}$
Gaia [36]	11	55	391	228 (228)	138	138	**118**	2.65	1.54 (1.54)
AWS North America [91]	22	231	288	124 (124)	90	90	**81**	3.41	1.47 (1.47)
Géant [27]	40	61	634	452 (106)	**101**	**101**	109	4.85	3.46 (0.81)
Exodus [64]	79	147	912	593 (142)	145	145	**103**	8.78	5.71 (1.37)
Ebone [64]	87	161	902	580 (123)	122	122	**95**	8.83	6.09 (1.29)

Sparser topologies can lead to a faster convergence **even in the absence of congestion**.

Conclusion

- Synchronization theory & max-plus algebra to model and optimize iteration time.
- In cross-silo setting, replacing server by peer-to-peer communication, results in significant speed ups ($\times 9$).
- Counter-intuitively, sparser topologies may lead to faster convergence even in the absence of congestion.

Thank you for your attention

Code: https://github.com/omarfoq/communication-in-cross-silo-fl

Email: othmane.marfoq@inria.fr