Throughput-Optimal Topology Design for Cross-Silo Federated Learning

Othmane Marfoq, Chuan Xu, Giovanni Neglia, Richard Vidal

Inria · Université Côte d'Azur · Accenture Labs

Paper

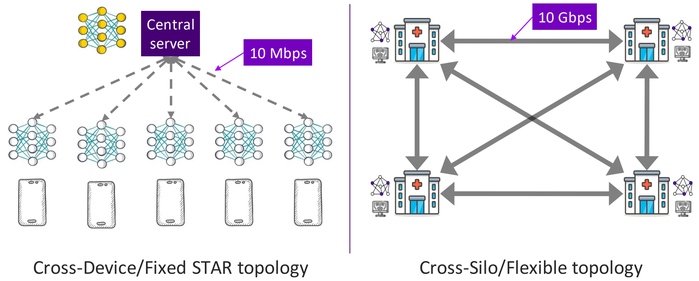
Code

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" because of privacy concerns or limited communication resources. The definition implicitly distinguishes two different settings:

- **Cross-Device**: Includes a large number of unreliable mobile devices, with limited computing resources and slow internet connections.
- **Cross-Silo**: Considers a few hundreds of reliable data silos with powerful computing resources and high-speed access links.

While cross-device scenario requires a **client-server** architecture where mobiles communicate only with the server, flexible **peer-to-peer** communications are possible in the cross-silo setting.



Cross-Device/Fixed STAR topology | Cross-Silo/Flexible topology

In this work we consider the following problem:

How to design the communication topology for cross-silo FL to achieve the fastest convergence?

Training time

Training Time = #Iterations × Iteration Time

Two contrasting effect of communication topology on the training duration:

1) A more connected topology leads to fewer iterations, because information can flow faster among nodes.
2) A more connected topology leads to slower iterations, not only because of congestion, but also because node needs to wait for more neighbors.

Recent experimental and theoretical works suggest this second effect may dominate the first one. Thus, in this work we focus on:

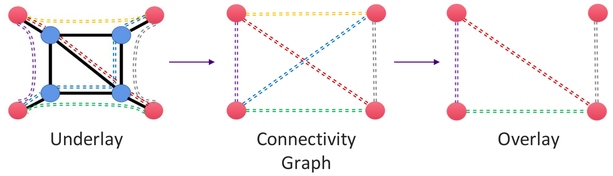
Finding the topology that minimizes the iteration time.

Problem formulation

Consider a network of N siloed data centers who collaboratively train a global machine learning model, solving:

$$\min_{w\in\mathbb{R}^d} \sum_{i=1}^{N} \mathbb{E}_{\xi_i\sim\mathcal{D}_i}\left[f_i(w,\xi_i)\right]$$

Silos are connected by a communication infrastructure which we call **underlay**. The underlay determines the set of possible connections among silos, namely the **connectivity graph**. The subset of connections used by silos to communicate form the **overlay**.



The delay associated with each connection can be expressed as:

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

Computation time · Latency · Model size M · Capacities

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

1) It updates the local model through a minibatch gradient step.
2) It sends the new model to its out-neighbors in the overlay.
3) It updates its model and those received from its in-neighbors.

The following recurrence holds:

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

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 and coincides with 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 $\mathcal{G}_o$.

Minimizing the iteration time is then equivalent to the following problem:

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: A strong spanning subdigraph of G_c with minimal cycle time.

Analysis

We prove that Minimal Cycle Time is NP-hard in general and provide exact polynomial algorithms.

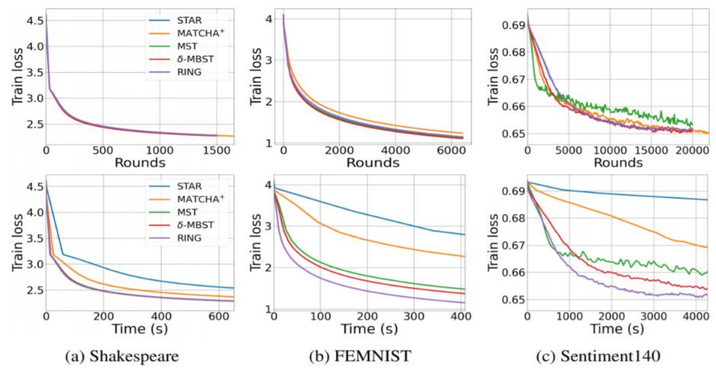
Network	Conditions	Algorithm	Complexity	Guarantees
Edge-capacitated	Undirected $\mathcal{G}_o$	Prim's Algorithm	$\mathcal{O}(\lvert\mathcal{E}_c\rvert + \lvert\mathcal{V}\rvert \log \lvert\mathcal{V}\rvert)$	Optimal solution
Edge/Node-capacitated	Euclidean $\mathcal{G}_o$	Christofides' Algorithm	$\mathcal{O}(\lvert\mathcal{V}\rvert^2 \log \lvert\mathcal{V}\rvert)$	$3N$-approximation
Node-capacitated	Euclidean & undirected $\mathcal{G}_o$	Algorithm 1	$\mathcal{O}(\lvert\mathcal{E}_c\rvert \lvert\mathcal{V}\rvert \log \lvert\mathcal{V}\rvert)$	6-approximation

Proposed algorithms output a **ring** or a **tree with constrained degrees.**

Experimental results

We conducted experiments on a multitude of federated learning tasks simulated on real networks. The proposed approach leads to significant training speed-up in comparison to the client-server topology and state-of-the-art **MATCHA**, both in terms of iteration time and overall training time.

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



(a) Shakespeare | (b) FEMNIST | (c) Sentiment140

Conclusion

Our algorithms speed up training by a factor 9 and 1,5 in comparison to the server-client architecture and to state-of-the-art MATCHA.

Counter-intuitively, sparser topologies may lead to faster convergence even in the absence of congestion.

Main References

Wang, Jianyu, et al. (2019). "MATCHA: Speeding up decentralized SGD via matching decomposition sampling." In: 2019 Sixth Indian Control Conference (ICC): 299-300.

Neglia, Giovanni, et al. (2020). "Decentralized gradient methods: does topology matter?." In: AISTATS 2020-23rd International Conference on Artificial Intelligence and Statistics.