Routing Optimization for Asynchronous Federated Learning on Heterogeneous Resources

Abdelkrim Alahyane[1,3], Céline Comte[2,3], Matthieu Jonckheere[2,3], Éric Moulines[4]

[1]EMINES-UM6P, Ben Guerir, Morocco
[2]CNRS, France
[3]LAAS, Toulouse, France
[4]CMAP, Ecole Polytechnique, France

Atelier en Évaluation des Performances – December 2, 2024

Outline

Outline

Federated Learning (FL)

Goal: Optimize the average performance of a model using stochastic gradient descent, across multiple clients, under the supervision of a central server.

Federated Learning (FL)

Goal: Optimize the average performance of a model using stochastic gradient descent, across multiple clients, under the supervision of a central server.

Solve the following **optimization problem**:

$$\underset{w \in \mathbb{R}^d}{\text{Minimize}} \left\{ f(w) \triangleq \frac{1}{n} \sum_{i=1}^{n} f_i(w) \right\}.$$

Federated Learning (FL)

Goal: Optimize the average performance of a model using stochastic gradient descent, across multiple clients, under the supervision of a central server.

Solve the following **optimization problem**:

$$\operatorname*{Minimize}_{w \in \mathbb{R}^d} \left\{ f(w) \triangleq \frac{1}{n} \sum_{i=1}^{n} f_i(w) \right\}.$$

Each client $i \in \{1, 2, \ldots, n\}$:

- Has local objective function $f_i(w) = \mathbb{E}_{(x,y) \in \mathcal{D}_i}[\ell_i(NN(x, w), y)]$.
- Approximates the gradient $\nabla_w f_i(w)$ with a **stochastic gradient estimate** $g_i(w)$.

Synchronous FL

Decentralized SGD: The central server sends (gradient estimation) tasks to $m \leq n$ clients, waits until all m clients answer, and updates the model parameter.

Decentralized SGD: The central server sends (gradient estimation) tasks to $m \leq n$ clients, waits until all m clients answer, and updates the model parameter.

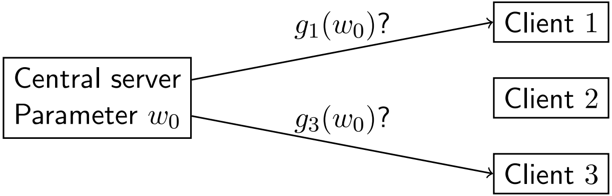
Central server
Parameter w_0

Client 1

Client 2

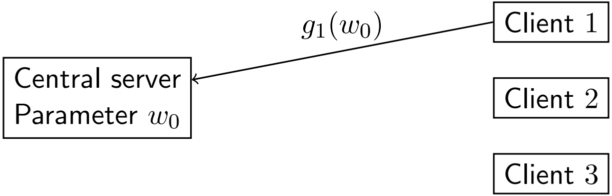
Client 3

Decentralized SGD: The central server sends (gradient estimation) tasks to $m \leq n$ clients, waits until all m clients answer, and updates the model parameter.



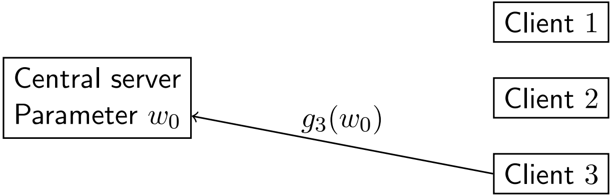
Synchronous FL

Decentralized SGD: The central server sends (gradient estimation) tasks to $m \leq n$ clients, waits until all m clients answer, and updates the model parameter.



Synchronous FL

Decentralized SGD: The central server sends (gradient estimation) tasks to $m \leq n$ clients, waits until all m clients answer, and updates the model parameter.

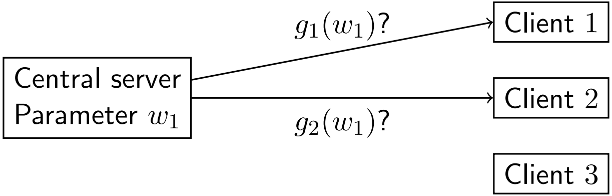


Synchronous FL

Decentralized SGD: The central server sends (gradient estimation) tasks to $m \leq n$ clients, waits until all m clients answer, and updates the model parameter.

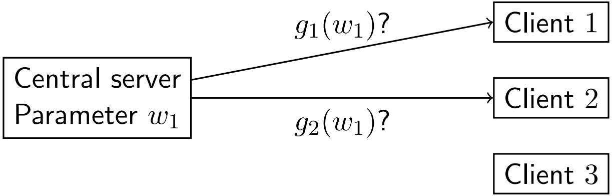
Synchronous FL

Decentralized SGD: The central server sends (gradient estimation) tasks to $m \leq n$ clients, waits until all m clients answer, and updates the model parameter.



Synchronous FL

Decentralized SGD: The central server sends (gradient estimation) tasks to $m \leq n$ clients, waits until all m clients answer, and updates the model parameter.



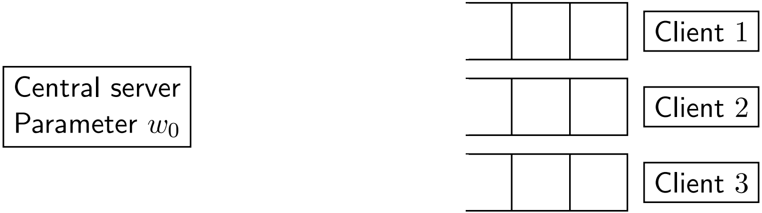
Problems:
1. The straggler effect
2. Coordination

Asynchronous FL

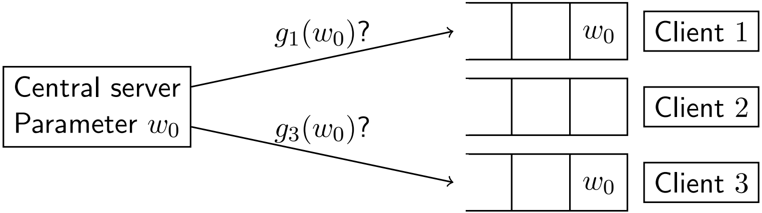
Asynchronicity: The central server updates the model parameter *au fil de l'eau* (that is, as gradient estimates are received)

Asynchronous FL

Asynchronicity: The central server updates the model parameter *au fil de l'eau* (that is, as gradient estimates are received)

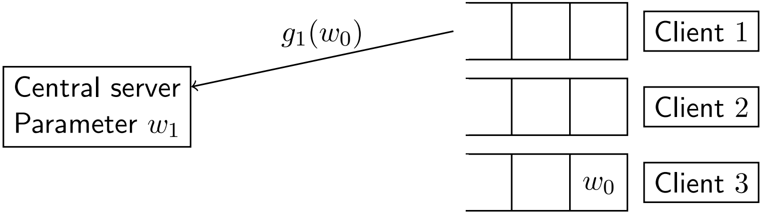


Asynchronicity: The central server updates the model parameter *au fil de l'eau* (that is, as gradient estimates are received)



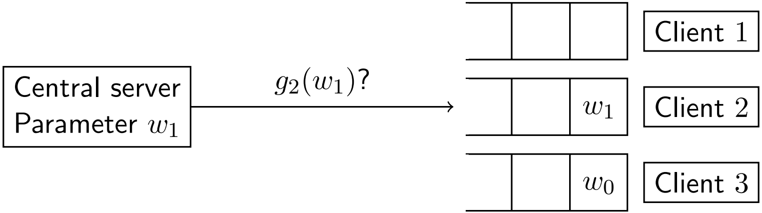
Asynchronous FL

Asynchronicity: The central server updates the model parameter *au fil de l'eau* (that is, as gradient estimates are received)



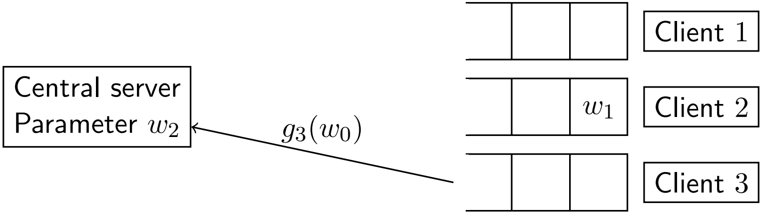
Asynchronous FL

Asynchronicity: The central server updates the model parameter *au fil de l'eau* (that is, as gradient estimates are received)



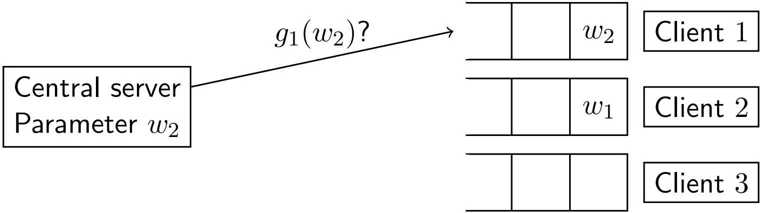
Asynchronous FL

Asynchronicity: The central server updates the model parameter *au fil de l'eau* (that is, as gradient estimates are received)



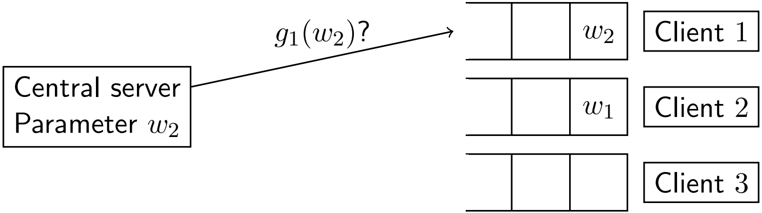
Asynchronous FL

Asynchronicity: The central server updates the model parameter *au fil de l'eau* (that is, as gradient estimates are received)



Asynchronous FL

Asynchronicity: The central server updates the model parameter *au fil de l'eau* (that is, as gradient estimates are received)



Challenge: Find a tradeoff between two antagonistic performance objectives

- Staleness, measured by the **mean relative delay**
- Speed, measured by the **throughput**, i.e., mean time between successive updates

Related works on FL

- **Introduction of FL** (Konečný, McMahan, and Ramage 2015; McMahan et al. 2017; Wang et al. 2020; Qu, Song, and Tsui 2022; Makarenko et al. 2022; Mao et al. 2022; Tyurin and Richtárik 2022)
- **Limitations of synchronous FL** (Xie, Koyejo, and Gupta 2020; Chen et al. 2021)
- **Algorithms for asynchronous FL**
  - `FedAsync` and its variants (Xie, Koyejo, and Gupta 2020; Chen et al. 2020; Xu et al. 2023)
  - `FedBuff` (Nguyen et al. 2022)
  - `AsyncSGD` (Koloskova, Stich, and Jaggi 2022)
  - `Generalized AsyncSGD` (Leconte et al. 2024)
  - `AsGrad` (Islamov, Safaryan, and Alistarh 2024)
- **Performance analysis of asynchronous FL**
  - Impact of dataset heterogeneity (Chen et al. 2020; Chen et al. 2021; Xu et al. 2023; Koloskova, Stich, and Jaggi 2022; Agarwal, Joshi, and Pileggi 2024)
  - Impact of queuing dynamics (Leconte et al. 2024), (Agarwal, Joshi, and Pileggi 2024)

Related works on FL

- **Introduction of FL** (Konečný, McMahan, and Ramage 2015; McMahan et al. 2017; Wang et al. 2020; Qu, Song, and Tsui 2022; Makarenko et al. 2022; Mao et al. 2022; Tyurin and Richtárik 2022)
- **Limitations of synchronous FL** (Xie, Koyejo, and Gupta 2020; Chen et al. 2021)
- **Algorithms for asynchronous FL**
  - `FedAsync` and its variants (Xie, Koyejo, and Gupta 2020; Chen et al. 2020; Xu et al. 2023)
  - `FedBuff` (Nguyen et al. 2022)
  - `AsyncSGD` (Koloskova, Stich, and Jaggi 2022)
  - `Generalized AsyncSGD` (Leconte et al. 2024)
  - `AsGrad` (Islamov, Safaryan, and Alistarh 2024)
- **Performance analysis of asynchronous FL**
  - Impact of dataset heterogeneity (Chen et al. 2020; Chen et al. 2021; Xu et al. 2023; Koloskova, Stich, and Jaggi 2022; Agarwal, Joshi, and Pileggi 2024)
  - Impact of queuing dynamics (Leconte et al. 2024), (Agarwal, Joshi, and Pileggi 2024)

Outline

Generalized AsyncSGD

Algorithm 1: `Central server`

Input: $T = \#$ model updates, $n = \#$ clients, $m = \#$ tasks,
$p =$ routing vector, $\eta =$ learning rate

`// Initialization`
Initialize model parameter w_0 randomly;
Send m (gradient estimation) tasks to the clients based on w_0;
`// End Initialization`
for $t = 0, \dots, T$ **do**
  Receive stochastic gradient $g_{C_t}(w_{I_t})$ from a client C_t;
  Update $w_{t+1} \leftarrow w_t - \frac{\eta}{np_{C_t}} g_{C_t}(w_{I_t})$;
  Sample a new client A_{t+1} with $\mathbb{P}(A_{t+1} = i) = p_i$;
  Send new model w_{t+1} to A_{t+1};
end

Algorithm 2: `Client` i

Input: Queue of tasks, $\mathcal{D}_i =$ local dataset

if *queue is not empty* **then**
  Take received parameter w from queue in FIFO order;
  Compute gradient estimate $g_i(w)$;
  Send the gradient estimate to CS;
  Repeat;
end

Assumptions

Queuing assumptions

- Tasks are processed in first-in-first-out order at each client.
- Processing times at client i are i.i.d. exponentially distributed with rate $\mu_i > 0$.
- Neglected: processing times at the central server, communication times.

Assumptions

Queuing assumptions

- Tasks are processed in first-in-first-out order at each client.
- Processing times at client i are i.i.d. exponentially distributed with rate $\mu_i > 0$.
- Neglected: processing times at the central server, communication times.

Dataset assumptions

- Lower boundedness: The objective function f is bounded from below by some f^*.
- Gradient smoothness: f_i has an L-Lipschitz continuous gradient, for each client i.
- Stochastic gradient properties: $g_i(w)$ is unbiased with variance bounded by $\sigma^2 > 0$, for each i.
- Bounded client heterogeneity: $\mathbb{E}[\|\nabla f(w) - \nabla f_i(w)\|^2] \leq \zeta^2$ for each client i, parameter w.

Definitions and notation

For each $t \in \{1, 2, \dots, T\}$, **step** t proceeds as follows:

- A task is assigned to client A_t, with $\mathbb{P}(A_t = i) = p_i$.
- A task is completed at client C_t.
- The system state at the end of this step is $X_t^{m-1} = (X_{1,t}^{m-1}, X_{2,t}^{m-1}, \dots, X_{n,t}^{m-1})$, with

$$X_{i,t}^{m-1} = X_{i,t-1}^{m-1} + \mathbb{1}[A_t = i] - \mathbb{1}[C_t - i].$$

Definitions and notation

For each $t \in \{1, 2, \dots, T\}$, **step** t proceeds as follows:

- A task is assigned to client A_t, with $\mathbb{P}(A_t = i) = p_i$.
- A task is completed at client C_t.
- The system state at the end of this step is $X_t^{m-1} = (X_{1,t}^{m-1}, X_{2,t}^{m-1}, \dots, X_{n,t}^{m-1})$, with

$$X_{i,t}^{m-1} = X_{i,t-1}^{m-1} + \mathbb{1}[A_t = i] - \mathbb{1}[C_t - i].$$

The **relative delay** at time t and client i is defined as follows:

$$D_{i,t} = \mathbb{1}[A_t = i] R_{i,t}, \text{ where } R_{i,t} = \min \left\{ r \in \mathbb{N} : \sum_{s=t}^{t+r} \mathbb{1}[C_s = i] = X_{i,t-1}^{m-1} + \mathbb{1}[A_t = i] \right\}.$$

Definitions and notation

For each $t \in \{1, 2, \ldots, T\}$, **step** t proceeds as follows:

- A task is assigned to client A_t, with $\mathbb{P}(A_t = i) = p_i$.
- A task is completed at client C_t.
- The system state at the end of this step is $X_t^{m-1} = (X_{1,t}^{m-1}, X_{2,t}^{m-1}, \ldots, X_{n,t}^{m-1})$, with

$$X_{i,t}^{m-1} = X_{i,t-1}^{m-1} + \mathbb{1}[A_t = i] - \mathbb{1}[C_t - i].$$

The **relative delay** at time t and client i is defined as follows:

$$D_{i,t} = \mathbb{1}[A_t = i] R_{i,t}, \text{ where } R_{i,t} = \min \left\{ r \in \mathbb{N} : \sum_{s=t}^{t+r} \mathbb{1}[C_s = i] = X_{i,t-1}^{m-1} + \mathbb{1}[A_t = i] \right\}.$$

We assume the Markov chain $(X_t^{m-1}, t \geq 0)$ is **stationary**, hence we drop the t index.

Outline

Results from (Leconte et al. 2024)

- The system dynamics $(X_t^{m-1}, t \geq 0)$ are given by a **Jackson network** (Jackson 1957).

Results from (Leconte et al. 2024)

- The system dynamics $(X_t^{m-1}, t \geq 0)$ are given by a **Jackson network** (Jackson 1957).
- **Upper bound on the empirical mean of the norm-square of the gradient of f:**
  There is $\eta_{\mathsf{max}}(p) > 0$ so that, for any $\eta \in (0, \eta_{\mathsf{max}}(p))$,

$$\frac{1}{T+1} \sum_{t=0}^{T} \mathbb{E}[\|\nabla f(w_t)\|^2] \leq G,$$

where $G = \dfrac{A}{\eta(T+1)} + \dfrac{\eta L B}{n^2} \displaystyle\sum_{i=1}^{n} \dfrac{1}{p_i} + \dfrac{\eta^2 L^2 B m}{n^2} \displaystyle\sum_{i=1}^{n} \dfrac{\mathbb{E}[D_i]}{p_i^2}$.

The variables A, B, and L depend on the learning problem and are estimated heuristically.

Mean relative delay

Theorem

The mean relative delay $\mathbb{E}[D_i]$ *and its gradient are given by*

$$\mathbb{E}[D_i] = \mathbb{E}[X_i^{m-1}], \qquad \frac{\partial \mathbb{E}[D_i]}{\partial p_j} = \frac{1}{p_j} \operatorname{Cov}[X_i^{m-1}, X_j^{m-1}], \qquad i, j \in \{1, 2, \ldots, n\}$$

Theorem

The mean relative delay $\mathbb{E}[D_i]$ and its gradient are given by

$$\mathbb{E}[D_i] = \mathbb{E}[X_i^{m-1}], \qquad \frac{\partial \mathbb{E}[D_i]}{\partial p_j} = \frac{1}{p_j} \operatorname{Cov}[X_i^{m-1}, X_j^{m-1}], \qquad i, j \in \{1, 2, \ldots, n\},$$

where for each $i, j \in \{1, 2, \ldots, n\}$,

$$\mathbb{E}[X_i^{m-1}] = \sum_{k=1}^{m-1} \left(\frac{p_i}{\mu_i}\right)^k \frac{Z_{n,m-1-k}}{Z_{n,m-1}}, \quad \mathbb{E}[X_i^{m-1} X_j^{m-1}] = \sum_{\substack{k,\ell=1 \\ k+\ell \le m-1}}^{m-1} \left(\frac{p_i}{\mu_i}\right)^k \left(\frac{p_j}{\mu_j}\right)^\ell \frac{Z_{n,m-1-k-\ell}}{Z_{n,m-1}},$$

and the $Z_{n,\mathfrak{m}}$'s are computed using Buzen's algorithm.

Numerical results

$$G = \frac{A}{\eta(T+1)} + \frac{\eta L B}{n^2} \sum_{i=1}^{n} \frac{1}{p_i} + \frac{\eta^2 L^2 B m}{n^2} \sum_{i=1}^{n} \frac{\mathbb{E}[D_i]}{p_i^2}$$

Procedure

- Optimize G using the Adam gradient-descent algorithm, initialized with p^{uniform}.
- Simulate the dynamics of the Jackson network.
- Evaluate `Generalized AsyncSGD` on image classification tasks using the Fashion-MNIST and CIFAR-10 datasets, each containing 10 equally distributed image classes.

$$G = \frac{A}{\eta(T+1)} + \frac{\eta L B}{n^2} \sum_{i=1}^{n} \frac{1}{p_i} + \frac{\eta^2 L^2 B m}{n^2} \sum_{i=1}^{n} \frac{\mathbb{E}[D_i]}{p_i^2}$$

Procedure

- Optimize G using the Adam gradient-descent algorithm, initialized with p^{uniform}.
- Simulate the dynamics of the Jackson network.
- Evaluate `Generalized AsyncSGD` on image classification tasks using the Fashion-MNIST and CIFAR-10 datasets, each containing 10 equally distributed image classes.

Datasets

- **Homogeneous:** Data is distributed i.i.d. across clients, and all clients have the same number of data points.
- **Heterogeneous:** For each k, we sample a vector $p_k \sim \text{Dir}_n(0.5)$, where $p_{k,j}$ is the proportion of class-k instances allocated to client j and $\text{Dir}_n(\beta)$ the Dirichlet distribution with dimension n and concentration parameter $\beta > 0$.

Numerical results

$$G = \frac{A}{\eta(T+1)} + \frac{\eta L B}{n^2}\sum_{i=1}^{n}\frac{1}{p_i} + \frac{\eta^2 L^2 B m}{n^2}\sum_{i=1}^{n}\frac{\mathbb{E}[D_i]}{p_i^2}$$

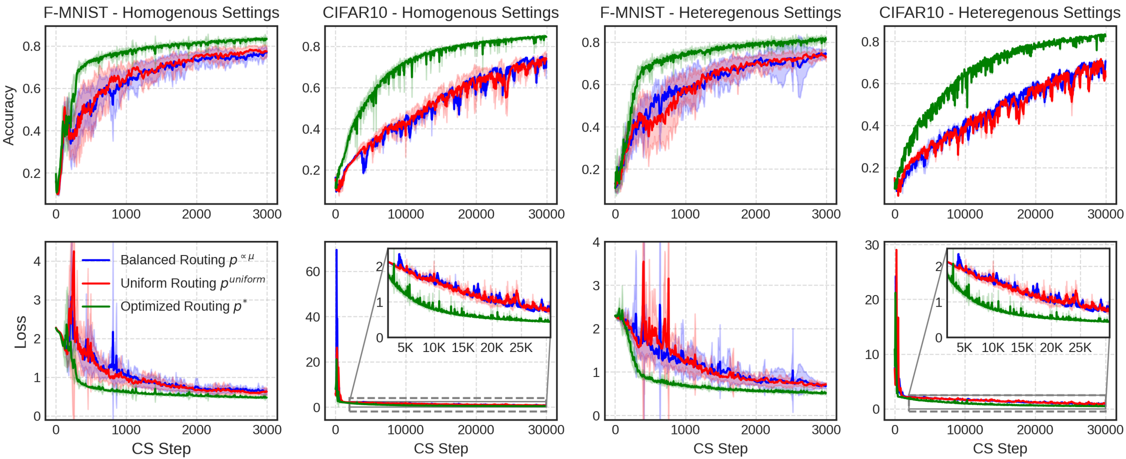


Figure: Performance on the validation set. Parameters: $n = 20$, $m = 100$, $\mu_i = e^{i/100}$, $\eta = 0.01$, $L = 1$.

Outline

Performance bound and mean delay

Conjecture

There exists $\eta_{max}(p) > 0$ *such that, for any* $\eta \in (0, \eta_{max}(p))$,

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\left[\tau_t\|\nabla f(w_t)\|^2\right] \le H,$$

where $H = \dfrac{\tilde{A}}{\eta\lambda(p)} + \dfrac{\eta L B}{\lambda(p)n^2}\displaystyle\sum_{i=1}^{n}\frac{1}{p_i} + \dfrac{\eta^2 L^2 B m}{\lambda(p)n^2}\displaystyle\sum_{i=1}^{n}\frac{\mathbb{E}[X_i^m]}{p_i^2}.$

The variables $\tilde{A}$, B, *and* L *depend on the learning problem, and*

$$\lambda(p) = \sum_{i=1}^{n}\mu_i\mathbb{P}(X_i^m > 0) = \frac{Z_{n,m-1}}{Z_{n,m}}.$$

$$H = \frac{\tilde{A}}{\eta\lambda(p)} + \frac{\eta L B}{\lambda(p) n^2} \sum_{i=1}^{n} \frac{1}{p_i} + \frac{\eta^2 L^2 B m}{\lambda(p) n^2} \sum_{i=1}^{n} \frac{\mathbb{E}[X_i^m]}{p_i^2}$$

Procedure

- Optimize H using the Adam gradient-descent algorithm, initialized with p^{uniform}.
- Simulate the dynamics of the Jackson network.
- Evaluate `Generalized AsyncSGD` on image classification tasks using the KMNIST and CIFAR-10 datasets, each containing 10 equally distributed image classes.

Datasets

- **Homogeneous:** Data is distributed i.i.d. across clients, and all clients have the same number of data points.
- **Heterogeneous:** For each k, we sample a vector $p_k \sim \text{Dir}_n(0.5)$, where $p_{k,j}$ is the proportion of class-k instances allocated to client j and $\text{Dir}_n(\beta)$ the Dirichlet distribution with dimension n and concentration parameter $\beta > 0$.

Numerical results

$$H = \frac{\tilde{A}}{\eta\lambda(p)} + \frac{\eta L B}{\lambda(p) n^2}\sum_{i=1}^{n}\frac{1}{p_i} + \frac{\eta^2 L^2 B m}{\lambda(p) n^2}\sum_{i=1}^{n}\frac{\mathbb{E}[X_i^m]}{p_i^2}$$

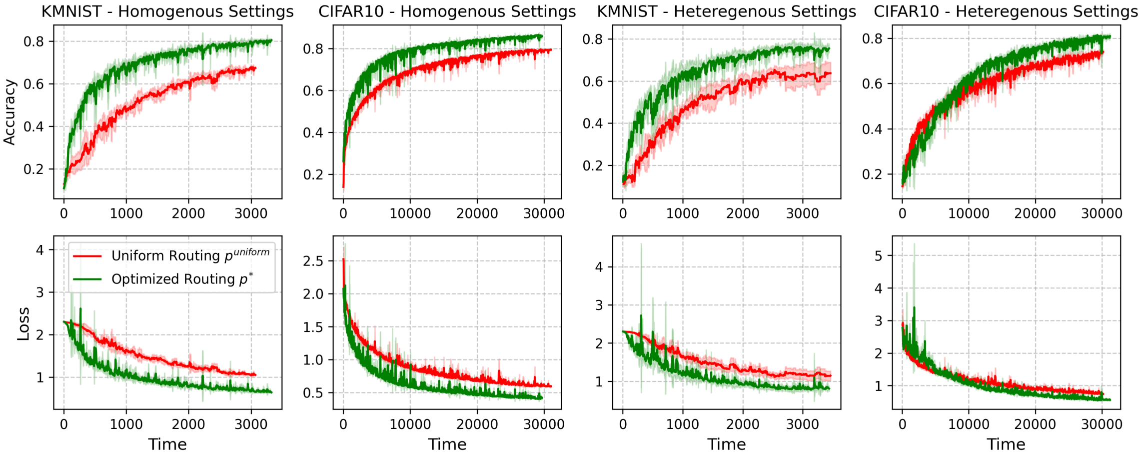


Figure: Performance on the validation set. Parameters: $n = m = 30$, $\mu_{1\ldots10} = \frac{1}{100}$, $\mu_{11\ldots20} = \frac{1}{10}$, $\mu_{21\ldots30} = 1$, $\tilde{A} = 15$, $L = 1$, $\sigma = 3$, $G = 10$, $\eta = 0.01$.

Conclusion

Contributions

- Computed mean relative delay by applying (a variant of) Little's law in Jackson networks.
- Computed its gradient by rewriting it as an expectation.
- Introduced a new clock-time-aware performance metric and derive an upper-bound.
- Designed a gradient-descent algorithm to optimize the upper-bounds.
- Numerically evaluated performance on image-classification tasks (Fashion-MNIST, KMNIST, and CIFAR-10 datasets).

Conclusion

Contributions

- Computed mean relative delay by applying (a variant of) Little's law in Jackson networks.
- Computed its gradient by rewriting it as an expectation.
- Introduced a new clock-time-aware performance metric and derive an upper-bound.
- Designed a gradient-descent algorithm to optimize the upper-bounds.
- Numerically evaluated performance on image-classification tasks (Fashion-MNIST, KMNIST, and CIFAR-10 datasets).

Take-away: Find a tradeoff between two antagonistic performance objectives.

Conclusion

Contributions

- Computed mean relative delay by applying (a variant of) Little's law in Jackson networks.
- Computed its gradient by rewriting it as an expectation.
- Introduced a new clock-time-aware performance metric and derive an upper-bound.
- Designed a gradient-descent algorithm to optimize the upper-bounds.
- Numerically evaluated performance on image-classification tasks (Fashion-MNIST, KMNIST, and CIFAR-10 datasets).

Take-away: Find a tradeoff between two antagonistic performance objectives.

Future works:
1. More tailored scheduling that accounts for staleness?
2. State-dependeng routing mechanism?

References I

- Jackson, James R (1957). "Networks of waiting lines". In: *Operations research.*
- Konečný, Jakub, Brendan McMahan, and Daniel Ramage (2015). *Federated optimization: Distributed optimization beyond the datacenter*. arXiv: 1511.03575.
- McMahan, Brendan et al. (2017). "Communication-efficient learning of deep networks from decentralized data". In: *Artificial intelligence and statistics.* PMLR.
- Chen, Yujing et al. (2020). "Asynchronous online federated learning for edge devices with non-iid data". In: *2020 IEEE International Conference on Big Data (Big Data).* IEEE.
- Wang, Jianyu et al. (2020). "Tackling the objective inconsistency problem in heterogeneous federated optimization". In: *Advances in neural information processing systems* 33.
- Xie, Cong, Sanmi Koyejo, and Indranil Gupta (2020). *Asynchronous federated optimization*. arXiv: 1903.03934.
- Chen, Zheyi et al. (2021). "Towards asynchronous federated learning for heterogeneous edge-powered internet of things". In: *Digital Communications and Networks* 7.3.

References II

- Koloskova, Anastasiia, Sebastian U Stich, and Martin Jaggi (2022). "Sharper convergence guarantees for asynchronous SGD for distributed and federated learning". In: *Advances in Neural Information Processing Systems* 35.
- Makarenko, Maksim et al. (2022). "Adaptive compression for communication-efficient distributed training". In: *arXiv preprint arXiv:2211.00188*.
- Mao, Yuzhu et al. (2022). "Communication-efficient federated learning with adaptive quantization". In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 13.4.
- Nguyen, John et al. (2022). "Federated learning with buffered asynchronous aggregation". In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*. Ed. by Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera. PMLR.
- Qu, Linping, Shenghui Song, and Chi-Ying Tsui (2022). "FedDQ: Communication-efficient federated learning with descending quantization". In: GLOBECOM 2022 - 2022 IEEE Global Communications Conference.

References III

- Tyurin, Alexander and Peter Richtárik (2022). *DASHA: Distributed nNonconvex optimization with communication compression, oOptimal oracle complexity, and no client synchronization*. arXiv: 2202.01268.
- Xu, Chenhao et al. (2023). "Asynchronous federated learning on heterogeneous devices: A survey". In: *Computer Science Review* 50.
- Agarwal, Aayushya, Gauri Joshi, and Larry Pileggi (2024). *FedECADO: A dynamical system model of federated learning*. arXiv: 2410.09933.
- Islamov, Rustem, Mher Safaryan, and Dan Alistarh (2024). "AsGrad: A sharp unified analysis of asynchronous-SGD algorithms". In: International Conference on Artificial Intelligence and Statistics (AISTATS). PMLR.
- Leconte, Louis et al. (2024). "Queuing dynamics of asynchronous federated learning". In: International Conference on Artificial Intelligence and Statistics (AISTATS). PMLR.

Routing Optimization for Asynchronous Federated Learning on Heterogeneous Resources

Abdelkrim Alahyane, Céline Comte, Matthieu Jonckheere, and Éric Moulines

To appear soon on HAL and arXiv

Céline Comte
CNRS & LAAS, Toulouse, France

celine.comte@cnrs.fr
https://homepages.laas.fr/ccomte/

SOLACE Team, Toulouse, France
https://solace.cnrs.fr/

$$G = \frac{A}{\eta(T+1)} + \frac{\eta L B}{n^2} \sum_{i=1}^{n} \frac{1}{p_i} + \frac{\eta^2 L^2 B m}{n^2} \sum_{i=1}^{n} \frac{\mathbb{E}[D_i]}{p_i^2}$$

Dependency on the number m of tasks?

- $\mathbb{E}[D_i]$ and G are increasing with m, and

$$\sum_{i=1}^{n} \mathbb{E}[D_i] = \sum_{i=1}^{n} \mathbb{E}[X_i^{m-1}] = m - 1.$$

- Performance is best with $m = 1$ task!

Dependency on the routing policy p?

- Second term minimized by p^{uniform}
- Third term is non-monotonic

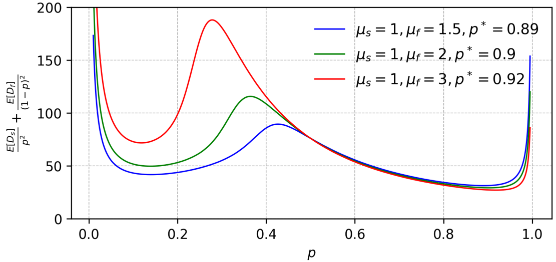


Figure: Third term of the bound G vs. the routing probability to the slowest client, in a toy example with $n = 2$ clients and $m = 20$ tasks, for various speed vectors $\mu = (\mu_s, \mu_f)$.

Uniform routing p^{uniform}

- The default in many applications: $p_i^{\text{uniform}} = \frac{1}{n}$.
- $G(p^{\text{uniform}}) = \dfrac{A}{\eta(T+1)} + \eta L B + \eta^2 L^2 B m(m-1)$.

Proportional routing $p^{\propto \mu}$

- A load-balancing heuristic: $p_i^{\propto \mu} = \frac{\mu_i}{\sum_j \mu_j}$.
- $G(p^{\propto \mu}) = \dfrac{A}{\eta(T+1)} + \dfrac{\eta L B |\mu|}{n^2} \sum_i \dfrac{1}{\mu_i} + \dfrac{\eta^2 L^2 B m(m-1)|\mu|^2}{n^3} \sum_i \dfrac{1}{\mu_i^2}$.

$\mathbb{E}[S_i] = \mathbb{E}[X_i^m]$: Consequences

$$H = \frac{\tilde{A}}{\eta\lambda(p)} + \frac{\eta L B}{\lambda(p) n^2} \sum_{i=1}^{n} \frac{1}{p_i} + \frac{\eta^2 L^2 B m}{\lambda(p) n^2} \sum_{i=1}^{n} \frac{\mathbb{E}[X_i^m]}{p_i^2}$$

Non-monotonicity with respect to the number m of tasks

- Contrary to G, H is not increasing in m, as it accounts for the duration of a step.
- H is minimized by some $m^* > 1$.
- m^* decreases with the learning rate η.

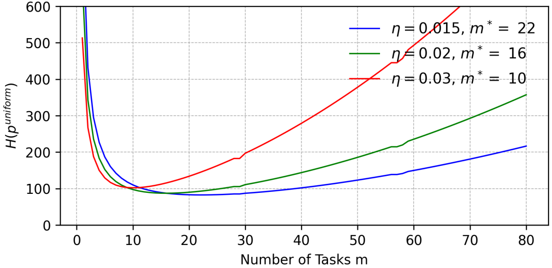


Figure: Bound $H(p^{\text{uniform}})$ as a function of the number m of tasks for different values of the step size η. The system consists of $n = 50$ clients, and the service speeds are $\mu_i = e^{i/100}$.