Scheduling for Ground-Assisted Federated Learning in LEO Satellite Constellations

Nasrin Razmi*†, Bho Matthiesen*†, Armin Dekorsy*†, and Petar Popovski‡*
*University of Bremen, U Bremen Excellence Chair, Dept. of Communications Engineering, Germany
†Gauss-Olbers Center c/o University of Bremen, Dept. of Communications Engineering, Germany
‡Department of Electronic Systems, Aalborg University, Aalborg, Denmark
Email:{razmi, matthiesen, dekorsy}@ant.uni-bremen.de, petarp@es.aau.dk

***Abstract*—Distributed training of machine learning models directly on satellites in low Earth orbit (LEO) is considered. Based on a federated learning (FL) algorithm specifically targeted at the unique challenges of the satellite scenario, we design a scheduler that exploits the predictability of visiting times between ground stations (GS) and satellites to reduce model staleness. Numerical experiments show that this can improve the convergence speed by a factor three.**

***Index Terms*—LEO constellation, Federated learning, Scheduling.**

I. Introduction

The small low Earth orbit (LEO) satellites, efficient in terms of cost and deployment, are marking a new era in satellite communications, as well as their integration with terrestrial networks. These LEO satellites are commonly deployed in large constellations, thereby creating a moving infrastructure for a seamless global coverage for, various applications such as communication services and Earth observation [1]. Many of these applications are data-intensive. For instance, in Earth observation, the high spatial, spectral, and temporal resolution of the imaging equipment, lead to large amounts of collected data [2]. These data are used in a variety of applications, such as disaster prevention, environmental monitoring, and urban planning. Transmitting such a large amount of data to the Earth may not be practical due to scarcity of the radio frequency resources or stringent delay requirements [3].

To address these constraints, a plausible solution is to process the data directly on-board of the satellites and only transmit the abstracted information to the ground station (GS). In this regard, it is viable to use federated learning (FL) [4], as a cooperative machine learning (ML) scheme in which the satellites only need to transmit the model parameters to the server instead of the raw data. Fig. 1 depicts the usage of FL for Earth observation. In the original FL setup, the user's participation in the training process is intermittent and randomized, based on user activity and communication availability. In satellite scenarios, the link (un)availability is predictable and related to the visiting pattern of the satellites to the location at which the GS is positioned. This GS orchestrated FL scenario was first identified in [5], where a new asynchronous FL procedure is proposed that addresses the unique challenges of FL on satellites. In [6], it is shown that the presence of inter-satellite links (ISL) improves the convergence speed of FL on satellites considerably. Without this capability, several extensions of the approach in [5] are possible that might lead to faster convergence speed. Indeed, the authors of [7] consider a heuristic GS update procedure and gradient buffering to speed up convergence.

This work was funded in part by by the German Research Foundation (DFG) under Germany's Excellence Strategy (EXC 2077 at University of Bremen, University Allowance).

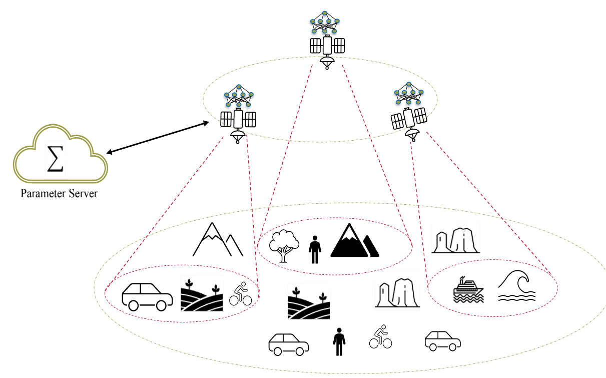

Fig. 1. FL for Earth observation applications.

In this paper, we take a different approach and design a scheduler based on the predictability of GS-satellite visiting lengths to reduce the model staleness. This leads to significantly faster convergence speed than the baseline approach in [5] and can be combined with different aggregation rules, such as the one in [7].

II. System Model

We consider a LEO satellite constellation with P circular orbits where the p-th orbit contains K_p satellites. The set $\mathcal{K} = \{S_1, S_2, \dots, S_K\}$ denotes all $K = \sum_{p=1}^{P} K_p$ satellites. The altitude and inclination of orbit p are denoted by h_p and i_p, respectively. Let $T_p = \frac{2\pi(r_E + h_p)}{v_p}$ and $v_p = \sqrt{\frac{\mu}{h_p + r_E}}\,\mathrm{m/s}$, respectively, denote the orbital period and the speed of the satellites in orbit p where $r_E = 6371\,\mathrm{km}$ is the Earth radius and $\mu = 3.98 \times 10^{14}\,\mathrm{m}^3/\mathrm{s}^2$ is the geocentric gravitational constant. A satellite can communicate with the GS if it is in a *visiting state*, i.e., the line-of-sight link between it and the GS is not blocked by the Earth. Otherwise, it is in a *non-visiting state*. The line-of-sight link between the satellite k and the GS is available when $\frac{\pi}{2} - \angle(\vec{r}_{GS}, \vec{r}_k - \vec{r}_{GS}) \geq \alpha_e$, where

$\vec{r}_k$ and $\vec{r}_{GS}$ denote the positions of satellite k and the GS, respectively, and α_e is the minimum elevation angle.

We define two time instants of significance. The *rise time* of the k-th satellite $t^n_{r,k}$ is the time instant at which the satellite enters in its n-th visit, while the *set-time* $t^n_{s,k}$ is the time instant at which the satellite finishes its n-th visit. The rise-time sequences of all K satellites τ_{rise} is expressed as

$$\tau_{\text{rise}} = \left(\left\{t^n_{r,1}\right\}^{N_1}_{n=1}, \left\{t^n_{r,2}\right\}^{N_2}_{n=2}, ..., \left\{t^n_{r,K}\right\}^{N_K}_{n=K}\right), \quad (1)$$

where $t_{r,k} = \left\{t^n_{r,k}\right\}^{N_k}_{n=1}$ is the rise-time sequence of the k-th satellite with N_k being the number of visiting states of the k-th satellite in the considered time interval $[T_b, T_f]$. Without loss of generality, we assume both T_b and T_f are located in the off-time of all satellites. Similarly, the set-time sequences of all K satellites τ_{set} is given by

$$\tau_{\text{set}} = \left(\left\{t^n_{s,1}\right\}^{N_1}_{n=1}, \left\{t^n_{s,2}\right\}^{N_2}_{n=2}, \ldots, \left\{t^n_{s,K}\right\}^{N_K}_{n=K}\right), \quad (2)$$

where $t_{s,k} = \left\{t^n_{s,k}\right\}^{N_k}_{n=1}$ is the set-time sequence of the k-th satellite.

Next, we define two types of time intervals associated with these sequences. The *on-time* is the time interval between the rise- and set-time, which corresponds to the duration of a visiting state. The *off-time* is a time interval between two visiting states. On-time sequence for all K satellites τ_{on} is defined as

$$\tau_{\text{on}} = \left(\left\{[t^n_{r,1}, t^n_{s,1}]\right\}^{N_1}_{n=1}, ..., \left\{[t^n_{r,K}, t^n_{s,K}]\right\}^{N_K}_{n=1}\right), \quad (3)$$

where $t_{\text{on},k} = \left\{[t^n_{r,k}, t^n_{s,k}]\right\}^{N_1}_{n=1}$ denotes the on-time sequence of the k-th satellite. Off-time sequences of the satellites τ_{off} is also expressed as

$$\tau_{\text{off}} = \left(\left\{t^n_{\text{off},1}\right\}^{N_1+1}_{n=1}, \ldots, \left\{t^n_{\text{off},k}\right\}^{N_K+1}_{n=1}\right) \quad (4)$$

where

$$t_{\text{off},k} = \left\{t^n_{\text{off},k}\right\}^{N_k+1}_{n=1} = \left\{[T_b, t^1_{r,k}], [t^1_{s,k}, t^2_{r,k}], \ldots, [t^{N_k}_{s,k}, T_f]\right\}. \quad (5)$$

The state $E(t, k)$ denotes the visiting state of the k-th satellite at time instant t, defined as

$$E(t, k) = \begin{cases} E_{\text{on}}, & t \in t_{\text{on,k}} \\ E_{\text{off}}, & t \in t_{\text{off,k}} \end{cases} \quad (6)$$

where E_{on} and E_{off} refer to on-time and off-time states, respectively. For simplicity, in the following we remove the time component t and denote the state of satellite k by $E(k)$.

A. Computation Model

Each satellite k gathers a local dataset $\mathcal{D}_k = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_{D_k}\}$ from the Earth where $\boldsymbol{x}_i$ and D_k denote the i-th sample and the number of samples of this satellite, respectively. This data is used to train a ML model in which each satellite k builds a loss function $F_k(\boldsymbol{w})$ expressed as

$$F_k(\boldsymbol{w}) = \frac{1}{D_k} \sum\nolimits_{\boldsymbol{x} \in \mathcal{D}_k} f_k(\boldsymbol{x}, \boldsymbol{w}), \quad (7)$$

where $f_k(\boldsymbol{x}, \boldsymbol{w})$ is the per-sample loss function at satellite k and builds upon the learning target which can be any convex or non-convex function. The vector $\boldsymbol{w}$ denotes the parameter describing the model.

Local raw dataset of each satellite is kept private, i.e., it is shared neither with other satellites nor with the GS. Satellites aim to collaboratively minimize a global loss function

$$F(\boldsymbol{w}) = \sum\nolimits_{k \in \mathcal{K}} \frac{D_k}{D} F_k(\boldsymbol{w}), \quad (8)$$

where $D = \sum_{k \in \mathcal{K}} D_k$ is the total number of samples. Unlike the well-known FedAvg algorithm [4], which has only one counter for the global epoch, we define counters for the GS and each satellite. The global epoch of the model is denoted as n which is kept track by the GS. In addition, for any satellite k, a local counter n_k is defined to track the satellite participation. For example, in a scenario with synchronous FL and full client participation, $n_k = n$ for all k. We further define $\boldsymbol{w}^n$ to be the global model parameters at epoch n. Assuming each satellites trains the model locally for I iterations using stochastic gradient descent (SGD), the local model parameters of satellite k at iteration $i \geq 1$ are

$$\boldsymbol{w}^{n_k,i}_k = \boldsymbol{w}^{n_k,i-1}_k - \eta \bigtriangledown F_k(\boldsymbol{w}^{n_k,i-1}_k), \quad (9)$$

where $\boldsymbol{w}^{n_k,0}_k$ is the global model received by satellite k in its n_k-th update and η is the learning rate. Following the linear computation time model from [8], the time $t_l(k)$ required by satellite k to compute an update to the global model is

$$t_l(k) = \frac{c_k I S(D_k)}{\nu_k}, \quad (10)$$

where c_k is the number of CPU cycles required to process a single data bit, $S(D_k)$ is the size of data in bits, and ν_k is the CPU frequency.

B. Communication Model

Communication between a satellite and the GS is possible if the line of sight between them is not blocked by the Earth, i.e., satellite k is in the on-time period with $E(k) = E_{\text{on}}$. The signal to noise ratio (SNR) between the k-th satellite and the GS is written as [9]

$$\text{SNR}(k, GS) = \begin{cases} \frac{P_t G_k G_{GS}}{N_0 L(k, GS)}, & \text{if } E(k) = E_{\text{on}} \\ 0, & \text{if } E(k) = E_{\text{off}}, \end{cases} \quad (11)$$

where P_t is the transmission power, G_k and G_{GS} are the average antenna gains of satellite k towards GS and vice versa, $N_0 = k_B T B$ is the total noise power with $k_B = 1.380649 \times 10^{-23}$ J/K being the Boltzmann constant, T is the receiver noise temperature, and B is the channel bandwidth. Free space path loss $L(k, GS)$ between the k-th satellite and the GS is expressed as

$$L(k, GS) = \left(\frac{4\pi f_c d(k, GS)}{c}\right)^2, \quad (12)$$

where f_c is the carrier frequency, c is the speed of light, and $d(k, GS)$ is the distance between satellite k and the

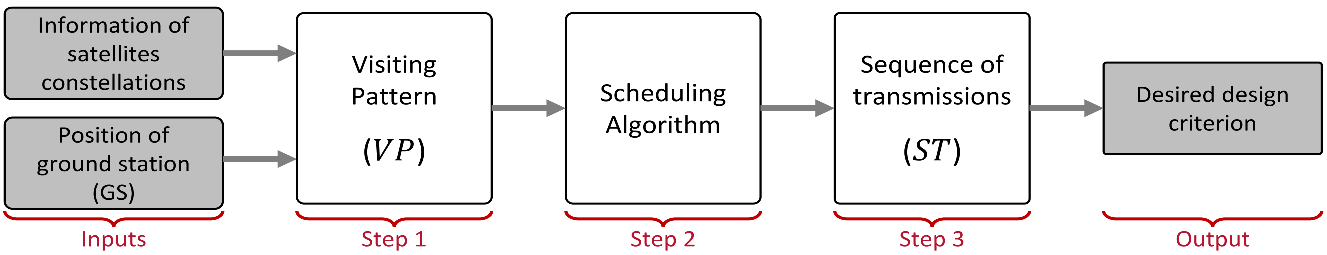

Fig. 2. Satellite scheduling Algorithm.

GS. Maximum achievable data rate for satellite k under the Gaussian channel assumption is

$$R(k,GS) = B\log_2\left(1+\mathrm{SNR}(k,GS)\right). \tag{13}$$

We use the longest distance between each satellite and the GS in each on-time duration to derive the SNR and rate. The time for exchanging the model parameters $\boldsymbol{w}$ between satellite k and the GS then is

$$t_c(k,GS) = \frac{S(\boldsymbol{w})}{R(k,GS)} + \frac{d(k,GS)}{c}, \tag{14}$$

where $\frac{S(\boldsymbol{w})}{R(k,GS)}$ and $\frac{d(k,GS)}{c}$ are the required time for transmission and propagation, respectively, and $S(\boldsymbol{w})$ is the data size of $\boldsymbol{w}$ in bits.

III. The Proposed Scheduling Algorithm

As mentioned above, a satellite can communicate with the GS when there is a line of sight link between them which means satellite k is in the E_{on} state. As a noteworthy fact, the rotation of Earth causes duration between visits of a satellite to the same GS to be different from its orbital period T_p.

Federated Averaging (FedAvg) algorithm is a well-known and widely employed FL procedure [4], [10]. Using it to train a FL model on satellites with full client participation [6] roughly works as follows: 1) The GS transmits the global model parameters to all satellites when they visit; 2) Satellites train the model using local SGD; 3) Satellites send the updated local parameters to the GS upon their next visit; and 4) The GS aggregates received model parameters from all satellites.

Implementing FedAvg in ground-assisted FL on satellites scenarios leads to very slow model convergence because satellites visit the GS at different times and the GS has to wait for all updates to receive before starting a new global epoch. An asynchronous version of FedAvg algorithm, named FedSat, is proposed in [5] for the satellite scenarios and shown to significantly reduce the convergence time. In FedSat, the GS updates the global model parameters whenever it receives updated local parameters from one of the satellites.

In this paper, we propose a general approach that helps in implementing the FL for any form of satellite constellation. This approach, as shown in Fig. 2, consists of three consecutive steps. The inputs are the satellites and the GS information such as the number of satellites and their altitudes, inclinations, and initial positions, plus the position of the GS.

With this input data, in the first step, the visiting pattern between each satellite and the GS can be obtained in the considered time i.e. $[T_b, T_f]$. An example of this visiting pattern is illustrated in Fig. 3. This figure presents the visiting pattern for a period of 24-hour between the GS, located in Bremen, and ten satellites. Five of the satellites, S_1 to S_5, are at altitude 500 km and the other five, S_6 to S_{10}, are at altitude 2000 km. The rise-time, τ_{rise}, set-time, τ_{set}, on-time, τ_{on}, and off-time, τ_{off}, of all satellites are derived in this step. Let us define the visiting pattern, $\mathcal{VP}$, as

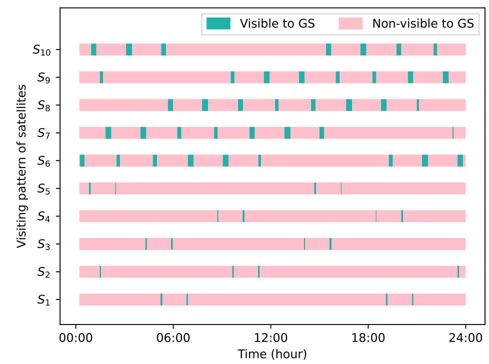

Fig. 3. Visiting pattern of 10 satellites and the GS in Bremen in one day. Satellites S_1 to S_5 are at altitude 500 km and S_6 to S_{10} are at altitude 2000 km.

$$\mathcal{VP} = \left(\tau_{\mathrm{rise}}, \tau_{\mathrm{set}}\right). \tag{15}$$

In the second step, a scheduling algorithm is designed based on the derived $\mathcal{VP}$. For example, the algorithm that will be proposed in Section III-B uses $\mathcal{VP}$ to determine whether the satellite trains the next model iteration while being offline or during its next visit to the GS. This is illustrated in Fig. 4. Scheduling algorithm in the second step leads to determining the transmission times between the satellites and the GS in the third step, i.e., the time intervals in which the UL and DL transmissions to exchange the model parameters happen are extracted. Let us define the sequence of transmission referring to these time intervals as

$$\mathcal{ST} = \left(\tau_{\mathrm{UL}}, \tau_{\mathrm{DL}}\right), \tag{16}$$

where τ_{UL} and τ_{DL} are tuples specified by

$$\tau_{\mathrm{UL}} = \left(t_{u,1}, t_{u,2}, ..., t_{u,K}\right), \tag{17}$$

$$\tau_{\mathrm{DL}} = \left(t_{d,1}, t_{d,2}, ..., t_{d,K}\right), \tag{18}$$

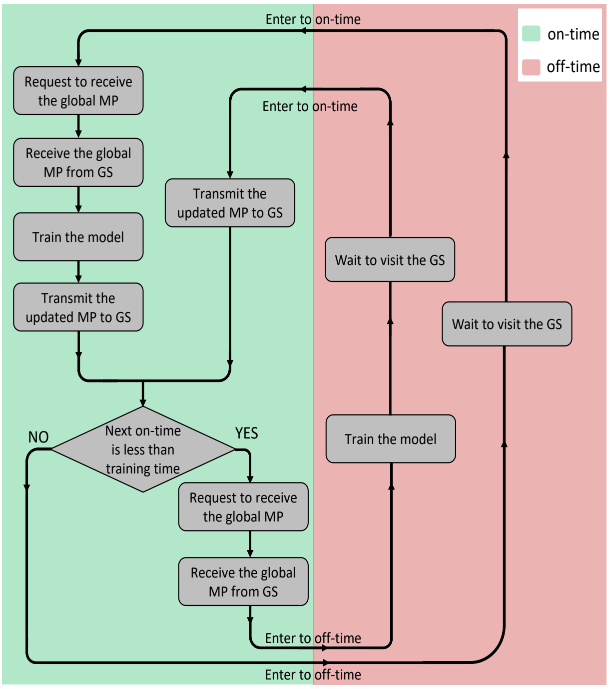

Fig. 4. Flow chart of FedSatSchedule algorithm, the red and green colors represent the off-time and on-time intervals, respectively. MP stands for model parameters.

where $t_{u,k}$ and $t_{d,k}$ are sequences of the UL and DL transmission times associated with the k-th satellites, given by

$$t_{u,k} = \left\{t_{u,k}^{n}\right\}_{n=1}^{U_k}, \tag{19}$$

$$t_{d,k} = \left\{t_{d,k}^{n}\right\}_{n=1}^{D_k}. \tag{20}$$

In (19) and (20), U_k and D_k stand for the total number of UL and DL transmissions of the k-th satellite, respectively. To obtain the optimal $\mathcal{ST}$, we formulate an optimization problem expressed as

$$\mathcal{ST}^* = \arg\max_{\mathcal{ST}} C(\mathcal{VP}, \mathcal{ST}), \tag{21}$$

where C, as a function of $\mathcal{VP}$ and $\mathcal{ST}$, is a desired design criterion which should be defined based on the requirements of any specific problem. An example of this criterion function is given in section III-B.

By using the proposed three-step thorough model, in the following, we present a new scheduling algorithm named as FedSatSchedule. To define this scheme, at first, we briefly explain FedSat, the scheme that we proposed in our previous work [5].

A. Federated Learning for Satellite Constellations (FedSat)

One way to implement FL for the satellite constellations, is using an asynchronous algorithm as presented in FedSat [5]. By this approach, we can benefit from the predictability of satellites visiting pattern which helps to overcome the intermittent connectivity between the GS and satellites.

In FedSat, each satellite exchanges the model parameters with the GS when they visit each other. This means in the rise-time, satellite k transmits the updated local model parameters to the GS. Then, the GS updates global model parameters by

$$\boldsymbol{w}^{n+1} = \boldsymbol{w}^{n} - \alpha_k(\boldsymbol{w}_k^{n_k-1,I} - \boldsymbol{w}_k^{n_k,I}). \tag{22}$$

where α_k is $\frac{D_k}{D}$. Then, the GS transmits the updated model parameters to that satellite. Again, the satellite trains the model in the off-time period and transmits the model parameters to the GS in the next rise-time. This algorithm does not take on-time and off-time durations into account. However, if the satellite's next visit to the GS will be long enough to complete the training during that visit, obtaining the global model already at the current visit will lead to considerable model staleness, which has a negative impact on convergence. Exploiting this simple observation is the key idea behind the FedSatSchedule algorithm proposed next.

B. Federated Learning Scheduling for Satellite Constellations (FedSatSchedule)

In FedSat scheme, as mentioned above, the duration of each visit i.e. $t_{\text{on},k}$ is not taken into account when deriving $t_{u,k}$ and $t_{d,k}$ for $\mathcal{ST}$. However, due to the fact that the length of $t_{\text{on},k}$ and $t_{\text{off},k}$ are completely predictable, $\mathcal{ST}$ can be determined such that a higher training accuracy can be achieved in a shorter time frame. The FedSatSchedule scheme uses these times to schedule the FL aimed at convergence time reduction. Formalizing this in our general framework, (21) can be converted to

$$\mathcal{ST}^* = \arg\min_{\mathcal{ST}} CT(\mathcal{VP}, \mathcal{ST}) \tag{23}$$

where CT is the convergence time of the model which, in its turn, is a function of $\mathcal{VP}$ and $\mathcal{ST}$. Solving this problem exactly is challenging, as even the functional relation CT is difficult to define. Instead, we take a heuristic approach that aims to reduce the model staleness at the satellites while still ensuring that every satellite provides a model update during each visit to the GS. In particular, the scheduler predicts whether the next visit to the GS is long enough to complete a local model update. If this is the case, the satellite will receive the current global model parameters upon its next contact to the GS. Otherwise, it will receive them immediately and compute its update during its off-time. We design this procedure named "FedSatSchedule" explicitly next.

In FedSatSchedule algorithm, in the current on-time i.e. $[t_{r,k}^n, t_{s,k}^n]$, the k-th satellite decides about the required operations based on comparing the duration of the next on-time and the necessary time for training; whether $t_{s,k}^{n+1} - t_{r,k}^{n+1} < t_l(k)$ or not. The flow chart in Fig. 4, in detail shows the tasks to be done during the n-th on-time period.

If the next on-time period, $t_{s,k}^{n+1} - t_{r,k}^{n+1}$, is shorter than the required training time, $t_l(k)$, the satellite requests that the GS sends the global model parameters in the same visit i.e. (n)-th on-time period. Then, the satellite by using the received global parameters, trains the model in the coming off-time period i.e. $[t_{s,k}^n, t_{r,k}^{n+1}]$. Afterwards, in the $(n+1)$-th on-time interval, it transmits the updated parameters to the GS.

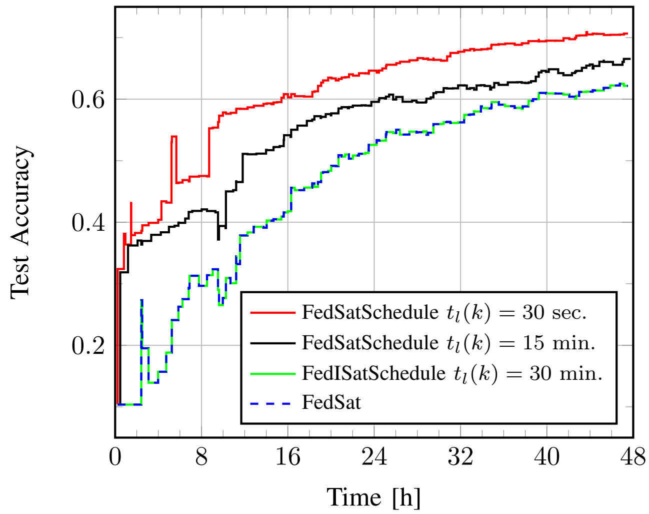

Fig. 5. Test Accuracy of a LEO constellation with 10 satellites and a GS located in Bremen.

Instead, if the next on-time period is longer than the required time for training, the satellite will have enough time for training using more up-to-date parameters in the coming on-time interval. Note that, in the off-time interval, the GS keeps updating the model parameters based on the received parameters from other satellites. Then, the k-th satellite had better wait and receive up-to-date model parameters exactly before starting to train in the next on-time interval. Hence, the satellite instead of requesting for receiving the new model parameters in the n-th on-time interval, will do it in the $(n+1)$-th on-time. With the received parameters, the satellite trains the model and transmits the updated model parameters to the GS in the $(n+1)$-th on-time. This approach results in higher accuracy without adding more delay or using extra resources.

IV. Numerical Results

In this section, we present simulation results to show the effectiveness of the proposed scheme. We consider ten satellites in 10 orbits; five of them are at altitude 500 km and the other five are at altitude 2000 km with a GS located in Bremen. The minimum difference in right ascension of the ascending node (RAAN) between two near orbits of different altitudes is 36°. The inclination and minimum elevation angles of all satellites are set to 80° and 10°, respectively. All satellites and the GS transmit the model parameters on channels with bandwidth of 20 MHz with the transmission power set to 40 dBm. The transmit and receive antenna gains are both set to 6.98 dBi. The carrier frequency and the receiver noise temperature are $f_c = 2.4$GHz and $T = 290$ K, respectively.

For training process based on [11], the well-known CIFAR dataset with the ResNet-18 model is considered. The learning rate, η, and the batch sizes are set to 0.1 and 10, respectively. The whole CIFAR dataset is divided between all satellites with Non-IID settings such that five labels are given to the satellites at altitude 500 and the other five labels to the other five at altitude 2000 km.

We examine the impact of the training time of each satellite, $t_l(k)$, on the test-accuracy. Fig. 5 shows the test accuracy for three different training time, 30 seconds, 15 minutes and 30 minutes, for a period of two days. It depicts that our proposed scheduling algorithm can noticeably improve the test accuracy for the cases $t_l(k) = 30$ seconds and $t_l(k) = 15$ minutes compared to the FedSat.

We observe if $t_l(k) = 30$ seconds, it takes 48 hours for the FedSat to have a test accuracy around 62%, while for the FedSatSchedule, it takes only 16 hours, improving the convergcne speed by a factor of three. FedSatSchedule outperforms FedSat due to a proper scheduling to receive more up-to-date model parameters.

By increasing the training time interval, as we see in the case with the $t_l(k) = 30$ minutes, the performances of the FedSat and FedSatSchedule converge together. In such cases, all satellites have, in practice, to train the model in their off-time period. So, the FedSatSchedule cannot benefit from having more up-to-date model parameters.

V. Conclusion

In this paper, we have presented a general approach for optimally scheduling the transmission and reception time of the model parameters between the satellites and the GS for implementing FL in any constellation. Then, we have specifically designed a scheduling algorithm, FedSatSchedule, by considering the duration of each on-time. The numerical results have shown that this scheme can accelerate the convergence of FL.

References

[1] I. Leyva-Mayorga, B. Soret, M. Röper, D. Wübben, B. Matthiesen, A. Dekorsy, and P. Popovski, "LEO small-satellite constellations for 5G and beyond-5G communications," *IEEE Access*, vol. 8, pp. 184 955–184 964, 2020.

[2] J. M. Haut, M. E. Paoletti, S. Moreno-Álvarez, J. Plaza, J.-A. Rico-Gallego, and A. Plaza, "Distributed deep learning for remote sensing data interpretation," *Proceedings of the IEEE*, vol. 109, no. 8, pp. 1320–1349, 2021.

[3] G. Curzi, D. Modenini, and P. Tortora, "Large constellations of small satellites: A survey of near future challenges and missions," *Aerospace*, vol. 7, no. 9, p. 133, 2020.

[4] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Aguera y Arcas, "Communication-efficient learning of deep networks from decentralized data," ser. Proc. Mach. Learn. Res. (PMLR), vol. 54, 2017.

[5] N. Razmi, B. Matthiesen, A. Dekorsy, and P. Popovski, "Ground-assisted federated learning in LEO satellite constellations," *IEEE Wireless Communications Letters*, pp. 1–1, 2022.

[6] N. Razmi, B. Matthiesen, A. Dekorsy, P. Popovski, "On-board federated learning for dense LEO constellations," in *ICC 2022 - IEEE International Conference on Communications (ICC)*, 2022.

[7] J. So, K. Hsieh, B. Arzani, S. Noghabi, S. Avestimehr, and R. Chandra, "Fedspace: An efficient federated learning framework at satellites and ground stations," *arXiv preprint arXiv:2202.01267*, 2022.

[8] N. H. Tran, W. Bao, A. Zomaya, M. N. H. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 1387–1395.

[9] L. J. Ippolito Jr, *Satellite Communications Systems Engineering*. John Wiley & Sons, 2017.

[10] Z. Li and P. Richtárik, "A unified analysis of stochastic gradient methods for nonconvex federated optimization," *arXiv preprint arXiv:2006.07013*, 2020.

[11] C. He, S. Li, J. So, X. Zeng, M. Zhang, H. Wang, X. Wang, P. Vepakomma, A. Singh, H. Qiu *et al.*, "FedML: A research library and benchmark for federated machine learning," *arXiv preprint arXiv:2007.13518*, 2020.