AsyncFLEO: Asynchronous Federated Learning for LEO Satellite Constellations with High-Altitude Platforms

Mohamed Elmahallawy and Tie Luo*
Computer Science department, Missouri University of Science and Technology, USA
E-mail: {meqxk, tluo}@mst.edu

***Abstract*—Low Earth Orbit (LEO) constellations, each comprising a large number of satellites, have become a new source of big data "from the sky". Downloading such data to a ground station (GS) for big data analytics demands very high bandwidth and involves large propagation delays. Federated Learning (FL) offers a promising solution because it allows data to stay in-situ (never leaving satellites) and it only needs to transmit machine learning model parameters (trained on the satellites' data). However, the conventional, synchronous FL process can take several days to train a single FL model in the context of satellite communication (Satcom), due to a bottleneck caused by *straggler satellites*. In this paper, we propose an asynchronous FL framework for LEO constellations called *AsyncFLEO* to improve FL efficiency in Satcom. Not only does AsynFLEO address the bottleneck (idle waiting) in synchronous FL, but it also solves the issue of *model staleness* caused by straggler satellites. AsyncFLEO utilizes high altitude platforms (HAPs) positioned "in the sky" as parameter servers, and consists of three technical components: (1) a *ring-of-stars* communication topology, (2) a *model propagation* algorithm, and (3) a *model aggregation* algorithm with *satellite grouping* and *staleness discounting*. Our extensive evaluation with both IID and non-IID data shows that AsyncFLEO outperforms the state of the art by a large margin, cutting down convergence delay by 22 times and increasing accuracy by 40%.**

***Index Terms*—Low-Earth orbit (LEO), satellite communications, federated learning, high-altitude platform (HAP)**

I. Introduction

Recent years have seen a surge in the deployment of Low Earth Orbit (LEO) satellites, which float at an altitude of 500–2000 km and form multiple (mega) constellations. Many of these satellites continuously collect Earth observational data in high volume, speed, and heterogeneity, constituting a new source of big data "from the sky". To extract tremendous value from such data and thereby support a wide range of applications such as urban planning, weather forecasting, and disaster management [1, 2], machine learning (ML) plays a crucial role in big data analytics. However, downloading a massive amount of data (e.g., millions of satellite images) to a ground station (GS) to train ML models is not practical: 1) satellite communication (Satcom) can barely afford such a high demand on network bandwidth; 2) the propagation delay between LEO satellites and GS is large; 3) transmission of raw data involves significant privacy and security risks (e.g., for military applications).

Federated learning (FL) [3] offers a promising solution by allowing each satellite to train an ML model *locally* on its own data and only need to upload the resulting model parameters to a parameter server (PS). The PS then aggregates all the satellites' *local models* into a *global model*. This eliminates the need for transmitting raw data, thus coping with the bandwidth and privacy issues mentioned above quite well.

However, applying FL to Satcom or more specifically LEO constellations is not straightforward and faces significant challenges. First, FL is an iterative process and typically requires hundreds of communication rounds between clients (i.e., satellites) and the PS. While this does not present as a big issue in large-capacity networks, it causes a severe slow-convergence problem in the context of Satcom due to the long propagation and transmission delay, and more saliently because of the *highly sporadic and irregular visit pattern* of LEO satellites to the PS. This visit pattern results from the distinction between satellites' and the PS' travel trajectories and the distinction between satellite orbiting speeds and the Earth rotation speed. Ultimately, the iterative FL process will incur vast delay in Satcom, taking several days or even longer to converge [4, 5].

To address this severely slow convergence problem of FL when applying to Satcom, we propose *AsyncFLEO*, a novel asynchronous FL framework tailored for LEO satellites, in this paper. AsyncFLEO exploits the availability of satellite local models opportunistically without waiting for all the models to be available (as in synchronous FL), and tackle *straggler satellites* and *model staleness* as follows. In the first epoch, it groups satellites from various orbits based on their data distributions (which we infer using their model weights since data is not accessible to PS in FL). Then in subsequent epochs, the PS selects a subset of satellite models from each group based on their freshness, to be included in model aggregation. Apart from these, AsyncFLEO has two other technical components for efficiency improvement: a ring-of-stars communication topology and an intra-orbit model propagation algorithm. Overall, AsyncFLEO can accelerate the FL convergence speed by several orders of magnitude, while increasing model quality (in terms of classification accuracy) at the same time.

AsyncFLEO leverages high altitude platforms (HAPs) positioned "in the sky" as PSs in lieu of a GS [6]. A HAP is a semi-static aircraft or airship situated in the stratosphere (17–22 km above the Earth's surface) and thus offers slightly better visibility of satellites than locating the PS on the ground [6–8]. However, AsyncFLEO does not *rely on* HAPs and it works

* Corresponding Author.
This work is supported by the National Science Foundation (NSF) under Grant No. 2008878.

with GS exactly the same way as in its single-HAP case.

In summary, this paper makes the following contributions:

- We propose for LEO Satcom an asynchronous FL approach that overcomes the large convergence delay caused by the highly sporadic connectivity and irregular visit pattern between satellites and the FL server.
- AsyncFLEO introduces three new technical components: (1) a model aggregation algorithm based on satellite grouping and model selection, which tackles *straggler satellites* and *stale models*; (2) a *ring-of-stars* topology in substitution of the conventional FL's star topology; (3) a *model propagation algorithm* that alleviates sporadic connectivity using intra-orbit model relay.
- We demonstrate via simulations that AsyncFLEO significantly accelerates FL convergence and outperforms state-of-the-art FL-Satcom algorithms by large margins (up to 22 times faster in convergence and 40% increase in accuracy).

Paper Organization. Section II discusses related work. Section III describes a general system model for FL in LEO constellations with HAPs. Section IV describes the design of AsyncFLEO in great detail. The performance evaluation of AsyncFLEO with other state-of-the-art methods is provided in Section V. Finally, we conclude in Section VI.

II. Related Work

While the research on big data problems in FL-Satcom is still in its infancy, some initial progress has been made recently by a number of interesting studies [4–6, 9, 10].

Synchronous FL. Chen et al. [9] applied the traditional FL approach (i.e., FedAvg [11]) to LEO constellations and compared it with centralized training obtained from downloading the data to a GS. Razmi et al. [5] proposed an FL approach called FedISL which uses inter-satellite link (ISL) for communication among satellites within the same orbit to reduce FL delay. It assumes that the PS is either a GS located at the North Pole (NP) or a satellite in medium Earth orbit (MEO) at an altitude of 20,000 km, in order to simplify the problem and increase the visibility between satellites and PS. However, NP is the ideal location which was only available during the early years of LEO deployment, and PS at MEO of that altitude travels at very high speed with non-negligible *Doppler shift* which is not addressed in their study. FedHAP [6] is an alternative synchronous FL approach proposed to address the above limitations. It introduces HAPs in place of the traditional GS, to act as PSs to oversee the model aggregation process collaboratively. However, FedHAP [6] still requires more than a day to converge due to its synchronicity.

Asynchronous FL. The authors of [10] proposed an asynchronous FL approach called FedSat to speed up FL convergence. Similar to [5], it also assumes the ideal setup where the GS is located at the NP, so that every satellite visits the GS at regular intervals, which again over-simplifies the problem. So et al. [4] proposed an approach called FedSpace to trade-off between the idle connectivity in synchronous FL and the staleness issue in asynchronous FL. The limitation of FedSpace is that it requires each satellite to upload a fraction of its captured images (or all the images in a lower resolution) to the GS to schedule model aggregation, which contradicts the important FL principle on privacy protection and communication efficiency.

III. System Model

Consider an LEO satellite constellation with O orbits where each orbit o contains N_o satellites equally spaced. Any satellite in orbit o has an orbital period $T_o = \frac{2\pi(R_E+h_o)}{v_o}$, where $R_E = 6371$km is the radius of the Earth, h_o is the orbital altitude, and v_o is the satellite velocity given by $v_o = \sqrt{\frac{GM}{(R_E+h_o)}}$, in which G is the gravitational constant and M is the mass of the Earth. There are a few HAPs $\mathcal{H} = \{h_1, h_2, \ldots, H\}$ that act as PSs and communicate with a varying number of satellites at different times depending on the *irregular* visit pattern. As mentioned in Section I, AsyncFLEO does not rely on HAPs and can operate with GS the same way as the single-HAP case; we include HAPs just to present a better design scheme (slightly better visibility than GS due to its elevated altitude, as well as more flexibility since it is deployed at a fixed location "in the sky"). In fact, our performance evaluation in Section V covers both HAP and GS scenarios with AsyncFLEO. Fig. 1 gives an illustration.

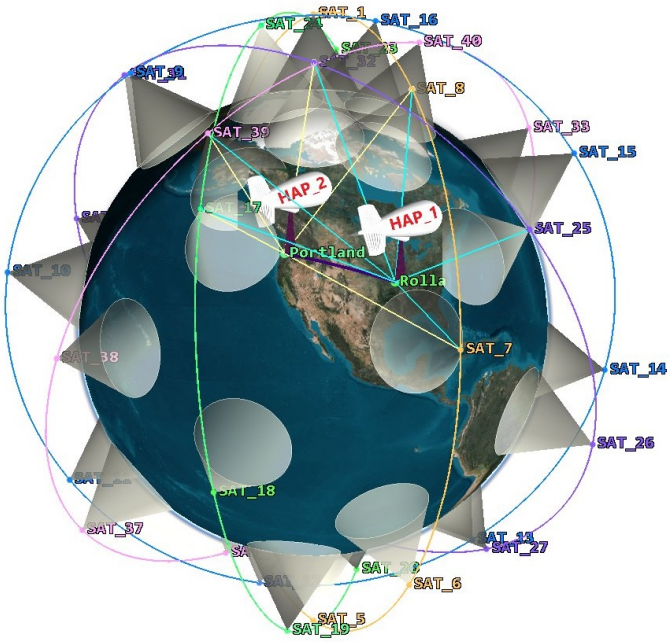


Fig. 1: An example of Walker-delta constellation [12] consists of $O = 5$ orbits, each having $N_o = 8$ satellites, orchestrated by $H = 2$ HAPs. Gray cones depict LEO satellites' coverage.

A. FL in Satellite Communication Networks

In this section, we assume a single PS (either HAP or GS) for ease of description and more consistency with standard FL. With this background knowledge, it would lead to a

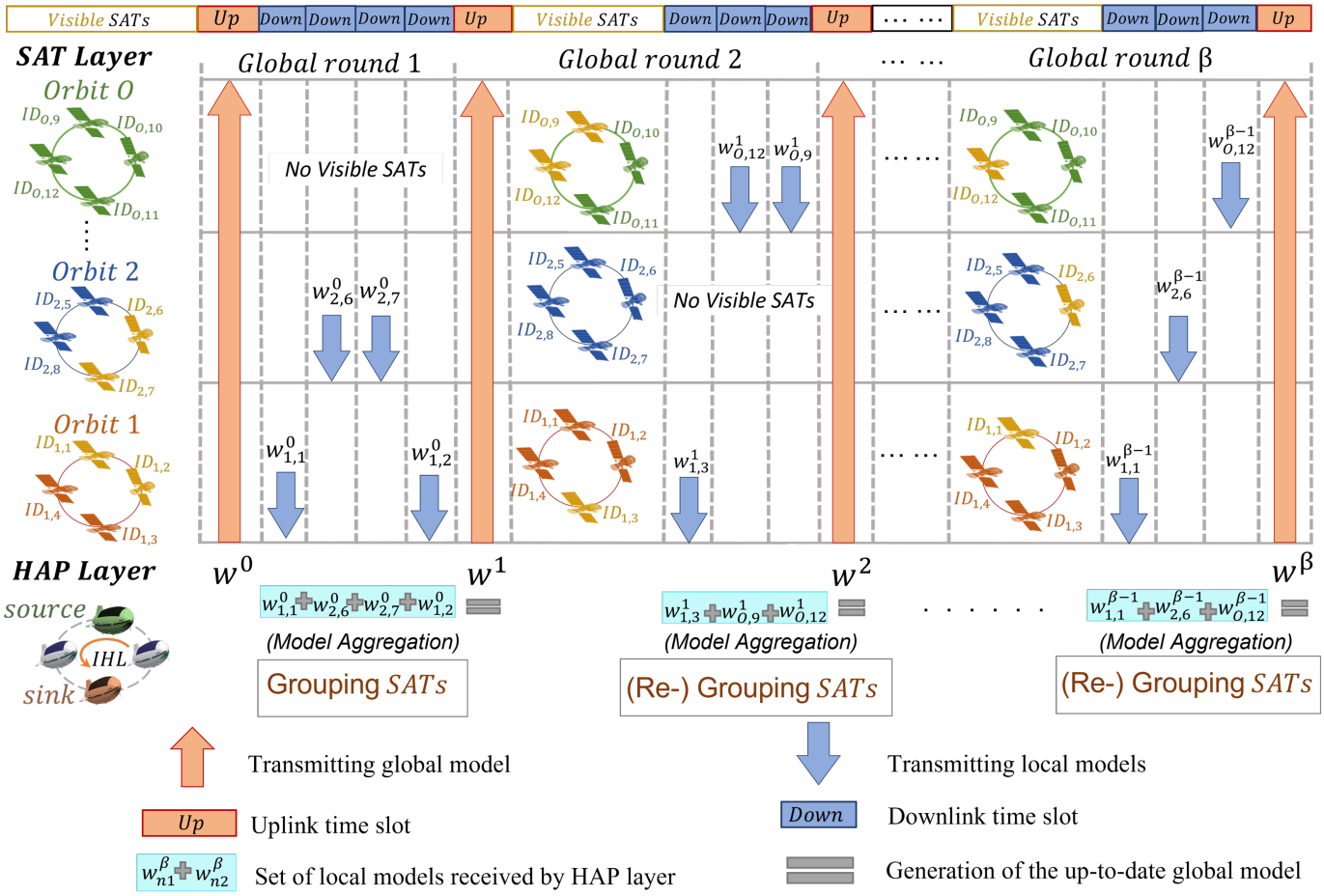


Fig. 2: The AsyncFLEO framework illustrated using a sequential diagram. **Yellow** satellites represent **visible** satellites.

more natural flow to Section IV, where we present a more sophisticated scenario with multiple HAPs.

The overall goal of FL for a constellation $\mathcal{N}$ of LEO satellites is to collaboratively train a global ML model under the orchestration of a PS, using each satellite's data *locally*, by minimizing the following global objective function:

$$\min_{w \in \mathbb{R}^d} F(w) = \sum_{n \in \mathcal{N}} \frac{m_n}{m} F_n(w) \tag{1}$$

where w is the parameters of the target global model, m_n is the size of satellite n's dataset D_n, $m = \sum_{n \in \mathcal{N}} m_n$ is the total size of all the satellites' data, and F_n is the local loss function at satellite n resulting from training w over D_n, which can be expressed as

$$F_n(w) = \frac{1}{m_n} \sum_{x \in D_n} f_n(w; x) \tag{2}$$

where $f_n(w; x)$ is the training loss for a data sample x and model parameters w at satellite n. Each satellite n solves (2) by applying a local optimizer such as stochastic gradient descent (SGD), for J local iterations, in the following way:

$$w_n^{\beta,j+1} = w_n^{\beta,j} - \frac{\eta}{b} \sum_{i=1}^{b} \nabla f_n(w_n^{\beta,j}; x_n^i) \tag{3}$$

where $j = 0, 1, 2, ..., J-1$ is a local training iteration in a global training round $\beta = 0, 1, 2, \ldots$, $w_n^{\beta,j}$ is the local model of satellite n at iteration j, η is the learning rate, x_n^i is the i-th training sample in the current mini-batch of size b.

In general, there are two approaches for the PS to aggregate all the local ML models $w_n^{\beta,J}$ collected from the satellites $\mathcal{N}$, into a global model. *Synchronous FL* follows the same principle as McMahan et al. [11], in which the PS waits to receive all the satellites' local models and then aggregates them as:

$$w^{\beta+1} = \sum_{n \in \mathcal{N}} \frac{m_n}{m} w_n^{\beta} \tag{4}$$

where we write w_n^{β} in place of $w_n^{\beta,J}$ for notation simplicity. Since this approach requires all the satellites in a constellation to become successively visible to the PS, it incurs a large delay in each global communication round and hence a much-prolonged convergence time after all the rounds.

Asynchronous FL is an approach that aims to address this limitation in synchronous FL. With this approach [13], the PS aggregates just a subset of local models as soon as they have been received, thus mitigating the long idle waiting as in synchronous FL and speeding up the global model convergence. However, it introduces a *model staleness* problem,

where some received models could come from earlier rounds from *straggler satellites* with limited visibility and hence are outdated. According to [4], this problem makes asynchronous FL unable to achieve comparable accuracy to synchronous FL although the convergence time would be reduced; in other words, one has to make a trade-off. In contrast, our proposed asynchronous FL approach AsyncFLEO is able to accelerate convergence *and* improve accuracy simultaneously. AsyncFLEO determines the subset of satellites of each round based on their data distributions (inferred from their model weights) using a satellite grouping scheme, and uses a staleness discounting factor to progressively aggregate models as soon as a model becomes available (details are given in Section IV-C).

B. Communication Links

Without loss of generality, consider a satellite *n* and a PS *g*, where the communication link between them can only be established if $\vartheta_{n,g}(t) = \angle(r_g(t), (r_n(t) - r_g(t))) \leq \frac{\pi}{2} - \vartheta_{min}$, where $r_n(t)$ and $r_g(t)$ are the trajectory of satellite *n* and GS *g*, respectively, and ϑ_{min} is the minimum elevation angle (a constant depending on the device).

Below, we model all the communication links (among satellites or HAPs or between them) as radio frequency (RF) links rather than free-space optical (FSO) links for the purpose of a fair comparison with prior work, but we note that, in practice, AsyncFLEO can actually benefit from FSO links which enjoy a much higher data rate (as high as Terabytes per second) than RF links and are also much more resistant to radio interference.

Assuming that the wireless channels are symmetric with additive white Gaussian noise, then the signal-to-noise ratio (SNR) between two objects x and y (e.g., satellite and GS) in free space is given as

$$SNR_{RF}(x, y) = \frac{P_t G_x G_y}{K_B T B \mathcal{L}_{x,y}} \tag{5}$$

where P_t is the transmission power, G_x and G_y denote the total antenna gain of the transmitter and the receiver, respectively, K_B is the Boltzmann constant, *T* is the noise temperature at the receiver, *B* is the channel bandwidth, and $\mathcal{L}_{x,y}$ is the free-space pass loss which can be expressed as

$$\mathcal{L}_{x,y} = \begin{cases} \left(\frac{4\pi \|x,y\|_2 f}{c}\right)^2, & \text{if LoS}(x,\, y), \\ \infty \quad , & \text{otherwise.} \end{cases} \tag{6}$$

where $\|x, y\|_2$ is the Euclidean distance between x and y, f is the carrier frequency, *c* is the speed of the light, and LoS is the line-of-sight link between x and y. The total delay t_c of sending data from x to y or vice versa, can be computed as follows:

$$t_c = t_t + t_p + t_x + t_y \tag{7}$$

$$t_t = \frac{b|\mathcal{D}|}{R}, \quad t_p = \frac{\|x, y\|_2}{c} \tag{8}$$

where t_t is the transmission delay, t_p is the propagation delay, t_x and t_y are the processing delay at x and y, respectively, $|\mathcal{D}|$ is the number of data samples and b is the size in bits of each sample, and R is the data rate that can be given by the Shannon formula as

$$R \approx B \log_2(1 + SNR) \tag{9}$$

In our simulation (Section V), we set the parameters for the formulae presented above.

IV. AsyncFLEO

AsyncFLEO is an asynchronous FL approach tailored for LEO satellites to speed up FL model convergence without sacrificing the model performance. It achieves this by (1) alleviating the negative impact of stale models received from straggler satellites due to irregular and sporadic connectivity, (2) overcoming the long waiting time for each satellite to visit the PS, and (3) reducing the large number of communication rounds via a smarter selection of satellites in each round. AsyncFLEO consists of three technical components described in subsections A–C: (A) a new, *ring-of-stars* topology for communication between satellites and HAPs, in lieu of the *star topology* used in traditional FL; (B) a *model propagation* algorithm that relays local and global ML models among satellites (intra-orbit) and HAPs; (C) a *model aggregation* algorithm based on satellite grouping and a smarter model selection of the fresh satellite models from each group to be included in each asynchronous round as well as a *discounting* scheme designed for and applied to stale models. Fig. 2 gives an overview of the AsyncFLEO framework.

A. SAT-HAP Communication Topology

The conventional FL adopts a star topology, where a PS communicates with all the clients (satellites in our case) directly. In AsyncFLEO, we introduce parallelism by using a ring-of-stars topology as follows. First, we cast the network into a layered structure where the first layer is an SAT layer that consists of all the LEO satellites $\mathcal{N}$, and the second layer is a HAP layer that consists of all the HAPs $\mathcal{H}$ (or a GS, which is just a special case). In the HAP layer, HAPs form a *ring topology* in which they can communicate with their adjacent (two) neighbors; at the same time, each HAP can also communicate with all of its currently visible satellites from *different orbits*, thereby forming a (small) star topology. Thus overall, all the HAPs and their currently visible satellites form a *ring-of-stars* topology, as shown in Fig. 3. In the SAT layer, we allow satellites in the same orbit to communicate with their adjacent neighbors (thus forming a ring too), but not those from different orbits. The reason is that satellites from different orbits have very high *relative velocity* and hence the impact of *Doppler shift* will become prominent and make communication unstable.

B. Propagation of Local and Global Models

We propose a model propagation algorithm that relays local and global models within the SAT layer and the HAP layer to speed up the FL training process under the severe constraint of highly intermittent satellite connectivity.

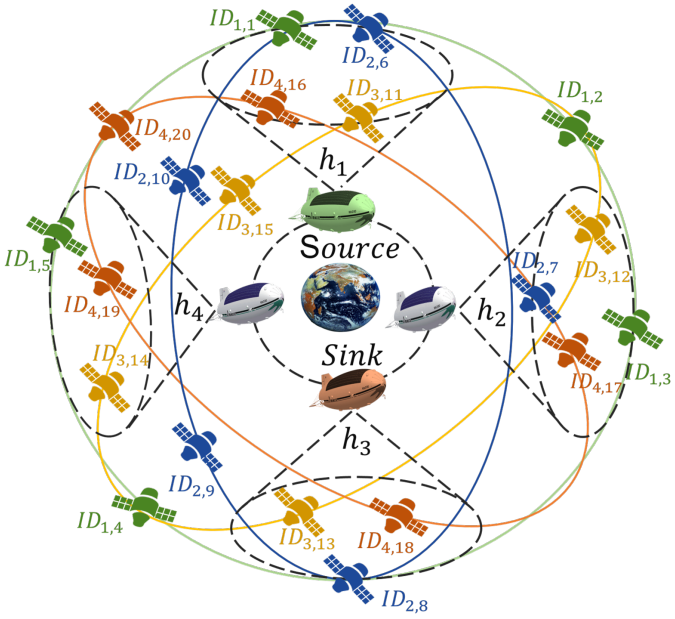


Fig. 3: Illustration of the *ring-of-stars* topology (indicated by the black dotted lines). The 4 HAPs form the "backbone" ring and each HAP orchestrates a star topology consisting of its currently visible satellites (as in the cone). Satellite IDs are in the format of ($ID_{Orbit\#,\ Satellite\#}$). The large colored ovals represent orbits.

1) **Relaying global model in the HAP layer:** When there are multiple HAPs, we pre-designate one HAP to be a *source* and another one to be a *sink* (typically the farthest from the source); they will also swap roles at appropriate times (see Section IV-B3). The source HAP generates a global model w^β (where $\beta = 0$ means the initial global model) and transmits this model to its two adjacent HAPs via inter-HAP link (IHL). Each of these two HAPs will then pass w^β to its next-hop neighbor (singular, since there is no need to send it back to the source). This relay continues on the "ring" until the model w^β reaches the sink HAP, as illustrated in Fig. 4a. Along the way, each HAP will also broadcast the model w^β to all of its visible satellites via the "star" topology[1].

2) **Relaying global and local models in the SAT Layer:** Once a visible satellite n receives w^β from a HAP, it performs two tasks simultaneously. One is to train w^β using n's local dataset D_n to obtain an updated local model w_n^β, and the other is to send the global model w^β to its neighboring satellites using ISL (without waiting for the model training task to complete). The second task is important because not all the satellites are visible to a HAP, and hence this model relay will kick start all the model training processes (each on a satellite) with *minimal delay*, using the latest version of the global model w^β. If a satellite receives the *same global model* from its two adjacent neighbors, the model relaying will cease at that satellite, as shown in Fig.4b.

[1]When there is only a single HAP, no model relay happens (just like the conventional GS case) and only the model broadcast will take place.

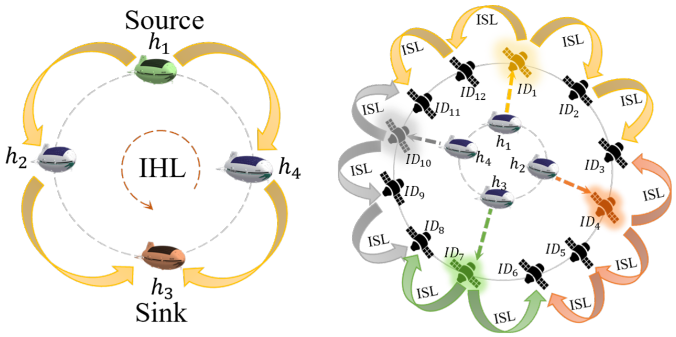
Algorithm 1: Propagation Algorithm of local and global models

Initialize: epoch β = 0, global model w^β, visible $\mathcal{N}_s = \phi$
while *Stopping criterion not met* **do**
  foreach $h \in \mathcal{H}$ **do** ▷ `propagate models via HAPs`
    if h *is source HAP* **then**
      Transmit w^β to its adjacent HAPs
    else if h *is sink HAP* **then**
      Stop relaying w^β
    else
      Transmit w^β to its next-hop HAP
    Transmit w^β to all its visible satellites $\mathcal{N}_h$
    Update $\mathcal{N}_s \leftarrow \mathcal{N}_s \cup \mathcal{N}_h$
  foreach $n \in \mathcal{N}$ **do** ▷ `Propagate models via SATs`
    if $n \in \mathcal{N}_s$ **then**
      ▷ `Model propagation via visible SATs`
      Transmit w^β to its two neighboring satellites
      Train w^β to obtain w_n^β
      if n *still visible to* h **then**
        Transmit w_n^β and its metadata to h
      else
        Transmit w_n^β and metadata to next-hop satellite
    else
      ▷ `invisible SAT`
      Wait until n receives w^β from a neighbor
      Train w^β to obtain w_n^β
      Transmit w^β, w_n^β, and metadata to next-hop satellite
  $\beta \leftarrow \beta + 1$

Upon completion of the local training process, each satellite n will send its trained local model w_n^β to its currently visible HAP (random selection if multiple), together with some metadata (described in Section IV-C1). If it does not have a visible HAP at the moment, it will send w_n^β and metadata to its two adjacent satellites, who will either transmit the model to a HAP if they are visible to one; otherwise, it continues to relay w_n^β to their respective neighbors too (but each to a single neighbor only because they would know the direction). This will substantially reduce the waiting time for each satellite to directly enter the visible zone of a HAP.

3) **Relaying local models in the HAP layer:** Following the above process, each HAP will receive a set of local models $\{w_n^\beta\}$ with associated metadata from its visible satellites. Once this set reaches a certain point (determined in Section IV-C), the HAP will propagate these models to the sink HAP for model aggregation. This follows the same route shown in Fig. 4a, from source to sink, but instead of relaying a global model, now each HAP is relaying a set of local models $\{w_n^\beta\}$ and metadata. Note that the set $\{w_n^\beta\}$ includes not only visible satellites' models but also some invisible ones', because of the local model relay in the SAT layer. Finally, when the sink HAP receives all the local models from other HAPs, it aggregates the local models into an updated global model $w^{\beta+1}$.

Next, the sink HAP will switch its role to a source, and



(a) Model relay in the HAP layer. (b) Model relay in the SAT layer.

Fig. 4: Illustration of the proposed model propagation. The curved arrows represent model propagation directions. In (b), the four visible satellites $ID_{1-4-7-10}$ initiate the model relay, and the four satellites $ID_{3-6-8-11}$ cease the model relay.

propagates the updated global model $w^{\beta+1}$ to its neighbors until reaching the source HAP which has switched its role to a sink, following the reverse path of propagating w^{β} as described in Section IV-B1. Along the way, each HAP also transmits $w^{\beta+1}$ to its visible satellites.

Algorithm 1 summarizes the entire propagation process. Note that model training and relaying take place concurrently.

C. Convergence Operations

Without loss of generality, we assume satellite data are non-IID since they are collected from different orbits. As described in Section IV-B, the sink HAP will eventually collect all the local models and satellites' metadata from other HAPs, obtaining $\mathcal{U} = \bigcup_{h\in\mathcal{H}} u_h$, where u_h contains all the local models and satellites' metadata collected by HAP h. However, simply aggregating these local models into a global model will result in poor performance because it fails to address three issues: (1) *model staleness* resulting from satellite models that were trained using an outdated global model, (2) the *non-IID nature* of data collected from different orbits, and (3) the *data size variation* among visible satellites from different orbits. These will render the global model *biased* towards orbits that have frequently visible satellites and have larger data sizes.

Furthermore, unlike traditional asynchronous FL approaches, AsyncFLEO has another responsibility for each HAP to decide when to stop collecting the set of models u_h and how to "clean" it, i.e., decide which orbits to include in the current global epoch (and what discount factor to apply), and which orbits to discard.

Therefore, AsyncFLEO performs two new functions: (1) grouping the satellites based on the diversity of their local models, and (2) aggregating the received local models in such a way that stale models do not adversely affect the FL model convergence.

1) **Satellite Grouping:** Once the sink HAP has collected all the local models, it organizes them as follows:

$$\mathcal{U} = \{\mathcal{S}_{o_1}, \mathcal{S}_{o_2}, \ldots, \mathcal{S}_{O}\}, \tag{10}$$

where $\mathcal{S}_o$ is the set of all local models collected from orbit o via HAPs $h_1, ..., h_p$, which can be expressed as:

$$S_o = \{\underbrace{\{w^{\beta}_{n_1}, w^{\beta}_{n_2}, .., w^{\beta}_{N}\}_{h_1}}_{u_{h_1}}, .., \underbrace{\{w^{\beta}_{n_1}, w^{\beta}_{n_2}, .., w^{\beta}_{N}\}_{H}}_{u_H}\}_o \tag{11}$$

In addition, $\mathcal{U}$ also contains the *metadata* of each satellite n in the collected set: a tuple $\langle ID, size, loc, ts, epoch\rangle_n$, where ID is satellite n's ID, $size$ is satellite n's training data size, loc is the satellite's current location (in an angular coordinate system) which is used to calculate its next visit time to PS, ts is the time stamp when satellite n transmits its local model to the PS, and $epoch$ is the last global epoch when satellite n was included in updating the global model w^{β} (i.e., if the latest $\beta = epoch$, then this local model is considered fresh).

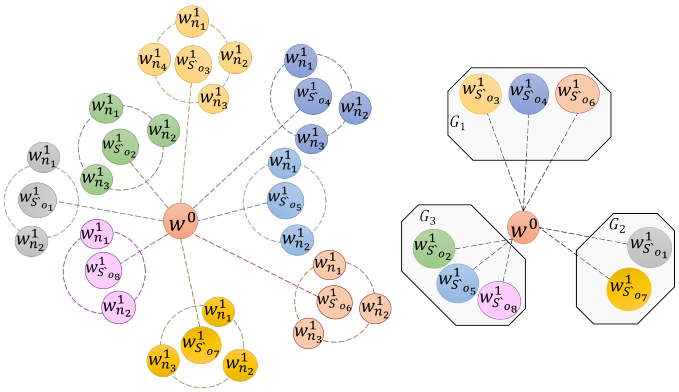
Note that, each $\mathcal{S}_o$ and ultimately $\mathcal{U}$ could contain duplicate satellites' models and metadata, due to the possibility of some satellites being visible to more than one HAP at the same time. Thus, AsyncFLEO will filter out these duplicate models and obtain a cleaned set $\mathcal{U}$ which is composed of unique local models and their metadata (i.e., $\{u_{hi}\}_o \cap \{u_{hj}\}_o = \phi$). Given the metadata, the sink HAP is able to determine the total data size of all the satellites in orbit o as

$$D_{\mathcal{S}_o} = \sum_{h=1}^{H} \sum_{n\in u_h} D_n \tag{12}$$

where D_n is the metadata $size$ of satellite n.

To deal with the straggler satellite problem and model staleness, AsyncFLEO groups satellites based on the similarity among their data distributions, which helps it to determine which satellites' models must be included in generating the global model and which could be discarded.

Definition. For a grouping $\mathcal{G} = \{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_n\}$ over the set of all the satellites $\mathcal{N}$, a satellite $n \in \mathcal{N}$ is said to be *grouped* if $n \in \mathcal{G}_i$ for some group $\mathcal{G}_i \in \mathcal{G}$, and *ungrouped* if $n \notin \mathcal{G}_i$ for all groups $\mathcal{G}_i \in \mathcal{G}$.



(a) Models before grouping. (b) Models after grouping.

Fig. 5: Satellite (model) grouping of $O = 8$ orbits into 3 groups $\mathcal{G}$. Dotted lines indicate the Euclidean distance between an orbit-wise aggregated model and the initial global model w^0.

In general, data collected by satellites in the same orbit tend to be similar to each other, while data collected from different orbits are more likely non-IID. This is due to the fact that satellites in the same orbit travel with the same orbital velocity v_o (about 25,000 km/h) which is much faster than the rotation speed of the Earth (about 1,600 km/h); whereas, different orbits have different altitudes and inclination angles, thus resulting in different travel speeds and geographic areas covered by different orbits.

Since PS has no access to satellites' data as dictated by FL, AsyncFLEO groups satellites based on satellites' local models w_n^β and the initial global model w^0. Specifically, during the first global epoch, local models generated by satellites from different orbits ($w_n^1,\ n = 1, 2, \ldots, N$) are based solely on their local training dataset and are not affected (biased) by any other satellites' local models. Therefore, the weight divergence between each satellite's local model and the initial global model w^0 tends to be the greatest *in the first epoch*, thereby allowing for a most effective differentiation among satellites' models. This is also the reason why we choose w^0 instead of the latest global model w^β.

Based on this concept, the sink HAP first generates a *partial global model* $\mathcal{S}'_o$ for each orbit o by performing a weighted average (according to data size) of the received models collected from orbit o (see Fig. 5a: each partial global model is at the center of a dashed-line circle, while those on the circle are the "member" models participated in the weighted average). Next, it calculates the Euclidean distance between each $\mathcal{S}'_o$ and the initial global model w^0, $\|d_{w^1_{\mathcal{S}'_o}} - d_{w^0}\|_2$. Those orbits with similar Euclidean distances to w^0 will be grouped together into a group $\mathcal{G}_i$ (see Fig. 5b). The sink HAP then stores this grouping scheme $\mathcal{G} = \{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_n\}$ for use in subsequent epochs.

In the next global epoch, when the sink HAP receives an updated version of satellites' models for any orbit, it checks whether that orbit is already in one of the stored groups. If so, this orbit of models will be directly assigned to the associated group. Otherwise, a partial global model will be computed on these models like the above, and its Euclidean distance to w^0 will again be computed, based on which this orbit will be assigned to the group that has the *minimum difference* from this orbit in terms of the average distance of its existing group members. The grouping procedure continues this way during each global epoch until all orbits have been grouped (hence typically taking only a few epochs). An illustration is given in Fig. 5.

2) **Model Aggregation:** During each global epoch, AsyncFLEO selects which satellites' models should be included in generating the global model (i.e., model aggregation) and which should be excluded. The selection approach takes into account three main factors: (1) the staleness of each model (determined by its epoch as indicated by metadata $epoch$, relative to its group, (2) the number of satellites of each group, and (3) the total $size$ of all the satellites' data in each group. Based on these factors, and in principle, AsyncFLEO selects

Algorithm 2: Model aggregation operation of AsyncFLEO

Initialize: epoch $\beta = 0$, global model w^β, $\mathcal{G} = \phi$
while *Termination criterion is not met* **do**
  Wait for receiving $\mathcal{S}_o\ \forall\ o \in O$
  Build $\mathcal{S}_o \leftarrow \bigcup_{u_{h_1}}^{u_H} \{w_{n_1}^\beta, \ldots, w_N^\beta\}$
  Update $\mathcal{U}$ using (10)
  Filter $\mathcal{U}$ by removing redundant models
  foreach $n \in \mathcal{N}$ **do** ▷ `grouping satellites`
    if n *is not grouped* **then**
      Compute $d_n = \|d_{w_n^0} - d_{w^0}\|_2$
      foreach $n' \in \mathcal{G}_i$ *and all* i **do** ▷ `compare similarity with grouped satellites`
        Assign n to group $\mathcal{G}_i$ according to the similarity between d_n and $d_{n'}$
    Update the grouping scheme $\mathcal{G}$
  foreach $\mathcal{G}_i \in \mathcal{G}$ **do** ▷ `model aggregation`
    if *none of the models in* $\mathcal{G}_i$ *is fresh* **then**
      Compute γ for all $w_n^\beta \in \mathcal{G}_i$ using (13)
    else
      ▷ `some or all are fresh`
      Select fresh models $\{w_n^\beta\}$
    Generate $w^{\beta+1}$ using selected models via (14)
  $\beta \leftarrow \beta + 1$

satellite models that are fresh and were trained with considerable data sizes to be included in model aggregation (i.e., generating the global model), while discarding stale models for this epoch only. The rationale is that the group possesses enough fresh models to compensate for the discarded stale models to participate in the global model aggregation. If, however, a group contains only stale models (no fresh models), AsyncFLEO will utilize these stale models but with a *staleness discounting factor* γ for the group $\mathcal{G}_i$, defined as follows:

$$\gamma = \sum_{\mathcal{G}_i \in \mathcal{G}} \sum_{n \in \mathcal{G}_i} \left(\frac{D_n}{D}\right)\left(\frac{k_n}{\beta}\right) \tag{13}$$

where n indexes the model using the pertaining satellite ID, D_n/D is the ratio between the data size of this satellite n and the total data size of all the satellites, and k_n/β is the ratio between the last global epoch where satellite n was included in generating the global model and the current global epoch β.

Once the model selection has been completed, AsyncFLEO updates the global model as follows:

$$w^{\beta+1} = (1-\gamma)w^\beta + \sum_{g=\mathcal{G}_1}^{\mathcal{G}_n} \sum_{n=1}^{N'_g} \gamma w_n^\beta \tag{14}$$

where N'_g is the total number of selected satellites from a group $g = G_i$. The rationale is that, when the current local models have grown stale, give the previous-round global model w^β higher weight while local models lower weight, and vice versa. This way, the FL training and aggregation process repeats in each global epoch until reaching a termination criterion (e.g., a target accuracy or a maximum number of

epochs) after multiple communication rounds (global epochs). As an optional step, the *final* FL model could be sent to a GS (e.g., by a HAP) if needed.

Algorithm 2 summarizes the entire process of AsyncFLEO for grouping the satellites and generating the global model.

V. Performance Evaluation

A. Simulation setup

LEO Constellation. We consider an LEO constellation consisting of 40 satellites equally distributed over five orbits. Each orbit is located at a height h_o of 2000 km above the Earth's surface, with an inclination angle of 80°. Two scenarios are considered for the PSs. One is a single GS or HAP located in Rolla, Missouri, USA (HAP floats above the city). The second scenario involves two HAPs, one floating above Rolla and the other above Portland, Oregon, USA. Each HAP hovers at an altitude of 20 km above the Earth's surface, with a minimum elevation angle $\vartheta_{min} = 10°$ which is the same as the GS. A two-line element (TLE) set [14] of each satellite is used by each PS to predict the satellite location on its trajectory. For each of the above two scenarios, the trajectories are determined over the course of three days. We select all communication link parameters discussed in Section III-B to be consistent with the baselines that we compare with, as summarized in Table I.

Baselines. AsyncFLEO is compared to the most recent state-of-the-art methods, as reviewed in Section II: FedISL [5] and FedHAP [6] which are synchronous approaches, and FedSat [10] and FedSpace [4] which are asynchronous approaches.

TABLE I: Simulation Parameters

Parameters	**Values**
Transmission power P_t	40 dBm
Antenna gain of G_t, G_r	6.98 dBi
Carrier frequency f	2.4 GHz
Noise temperature T	354.81 K
Transmission data rate R	16 Mb/sec
Number of local training epochs I	100
Learning rate η	0.01
Mini-batch size b_k	32

Dataset and ML models. In line with most peer studies on FL-Satcom, we use the same two datasets in our evaluation. The first one is the MNIST dataset [15] which contains 70,000 grayscale images of handwritten digits of size 28×28, and the other one is the CIFAR-10 dataset [16] which consists of 60,000 color images of 10 classes with a resolution of 32×32 pixels. We consider both IID and non-IID data distributions among satellites. In the IID setting, training data samples are randomly shuffled and evenly distributed among all the satellites (each having all 10 classes of images). In the non-IID setting, satellites from two orbits have four classes of data, while satellites from the other three orbits have the remaining six classes. We consider two neural networks to train satellites: convolutional neural network (CNN) and fully connected multi-layer perceptron (MLP). The hyperparameters used for training are listed at bottom of Table I.

TABLE II: Comparison with SOTA approaches

FL scheme	Accuracy (%)	Convergence time (h:mm)	Remark
FedISL [5]	63.51	72	GS at arbitrary location
FedISL [5] (ideal setup)	81.74	3:30	PS is a GS at the NP or an MEO satellite above the Equator
FedSat [10] (ideal setup)	88.83	12	GS at NP so that all satellites visit it at regular intervals
FedSpace [4]	46.10	72	GS needs satellites' local data
FedHAP [6]	87.29	30	HAP at arbitrary location
AsyncFLEO-GS	**80.62**	**6**	GS at arbitrary location
AsyncFLEO-HAP	**81.36**	**5**	HAP at arbitrary location
AsyncFLEO-twoHAP	**82.94**	**3:20**	HAP at arbitrary location

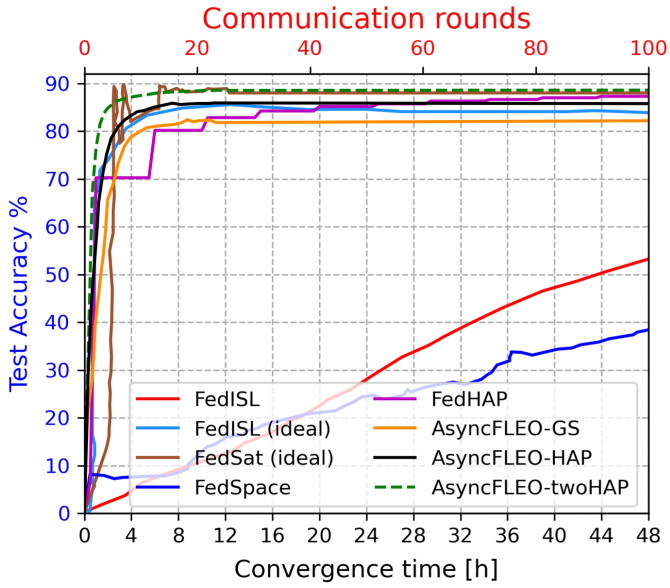
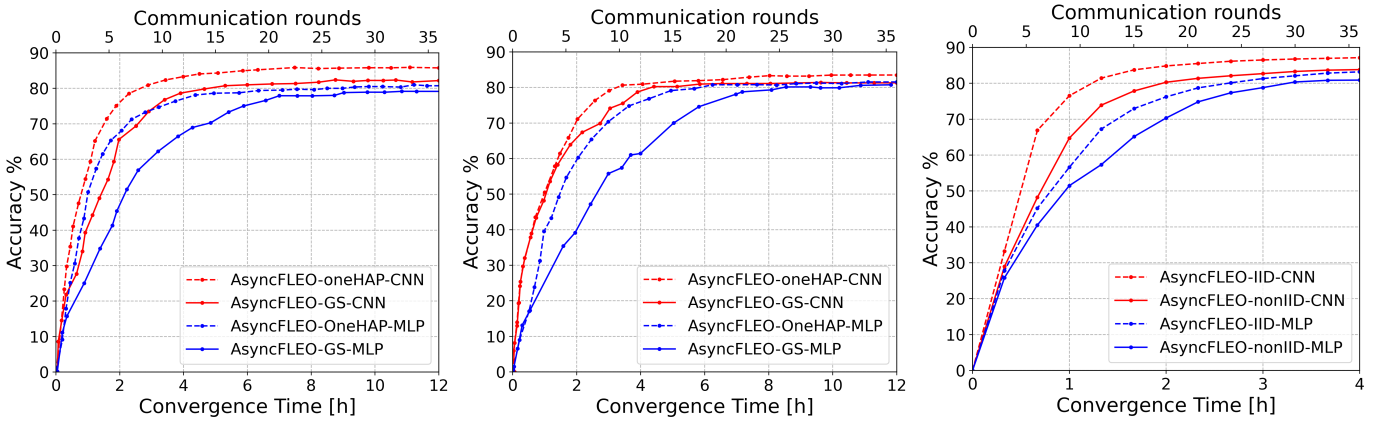


Fig. 6: Accuracy vs. Convergence time: Comparison with state-of-the-art baselines using the MNIST dataset.

B. Results

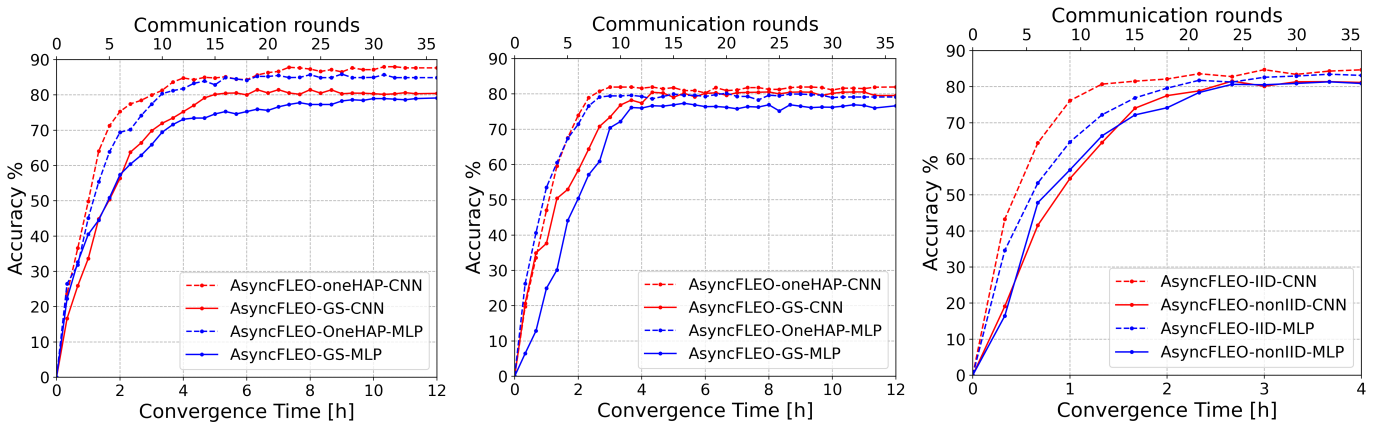
Comparison with State of the Art. Table II and Fig. 6 summarize the comparison results on the MNIST dataset under the non-IID setting with CNN as a training network. Note that AsyncFLEO has two versions, AsyncFLEO-GS, and AsyncFLEO-HAP, since AsyncFLEO has the flexibility of using GS or HAP as its PS. For a fair comparison, AsyncFLEO-HAP uses a single HAP only (the multi-HAP case has better performance and is evaluated in the next subsection as shown in Fig. 7c and Fig. 8c).

From Table II and Fig. 6, we observe that AsyncFLEO-twoHAP takes the shortest time 3:20 hours to converge and



(a) IID data. (b) Non-IID data. (c) Two HAPs (Rolla and Portland).

Fig. 7: AsyncFLEO evaluation on MNIST in various settings: IID/non-IID data, CNN vs. MLP, HAP vs. GS, one/two HAPs.



(a) IID data. (b) Non-IID data. (c) Two HAPs (Rolla and Portland).

Fig. 8: AsyncFLEO evaluation on CIFAR-10 in various settings: IID/non-IID data, CNN vs. MLP, HAP vs. GS, one/two HAPs.

achieves a high accuracy of 82.94%, and when it uses only a single HAP (for comparison with baselines), it achieves an accuracy of 81.36% within only 5 hours. Although FedISL [5] is faster than AsyncFLEO-HAP for convergent time 3:30 hours, it assumes an ideal setup (as discussed in SectionII), and even though given that condition, it only attains a lower accuracy of 81.7%. When FedISL places the GS elsewhere, it takes as long as 72 hours and only achieves an accuracy of 63.5%. While FedSat [10] and FedHAP [6] obtain slightly higher accuracy than AsyncFLEO-HAP, they take about 2.4 and 6 times longer convergence time, respectively, than ours to complete the learning process. In addition, FedSat [10] assumes the similar ideal condition as FedISL [5] which is very restrictive.

Among the two versions of AsyncFLEO, AsyncFLEO-HAP performs better than AsyncFLEO-GS as it is able to take advantage of the HAP's better satellite visibility due to its slightly elevated altitude. Nevertheless, AsyncFLEO-GS still performs fairly well, converging in just a few hours (6 hours) which is faster than most baselines and achieving a satisfactory accuracy (80.6%).

Evaluating AsyncFLEO in more extensive settings. Here we evaluate AsyncFLEO more extensively under various settings: IID vs. non-IID data, CNN vs. MLP, and single- vs. multi-HAP, as shown in Fig. 7 and Fig. 8, using MNIST and CIFAR-10 datasets, respectively. Fig. 7a gives the results obtained with the IID setting of the MNIST dataset. It shows that by just using a single HAP as the PS, AsyncFLEO can achieve an accuracy of 85.5% within five hours, while AsyncFLEO using GS can also achieve an accuracy of 82.1% after six hours. These are values when satellites employ CNN as their ML models. When using MLP, the accuracy slightly decreases by 4% in both the GS and HAP cases, but they still converge within only a few hours. When the dataset changes from MNIST to CIFAR-10, AsyncFLEO again achieves good results. As shown in Fig. 8a, AsyncFLEO attains an accuracy of 84.47% in 4 hours and then 86.82% after 8 hours using only a single HAP. This demonstrates AsyncFLEO's robustness to the change of datasets.

Fig. 7b and Fig. 8b demonstrate the robustness of AsyncFLEO to non-IID data settings. As can be observed from the two figures, AsyncFLEO converges in 12-15 asynchronous

global epochs (four-six hours) with an accuracy of 79.67%-81.36% attained with a single HAP, and converges in 15-18 global epochs (five-six hours) with an accuracy of 78.43%-80.15% achieved with a GS. This difference comes again from the fact the HAP has better visibility of LEO satellites (about 1-5 satellites more at the same location) than the GS. When the training model changes from CNN to MLP, the accuracy only decreases negligibly while the convergence time increases by only 1-2 hours, which is still much more acceptable than those baseline methods.

Finally, in Fig. 7c, we present the results for two HAPs, under both IID and non-IID data distributions. In the IID case, AsyncFLEO achieves an accuracy of 87.79% in only 2:40 hours, while in the non-IID case, AsyncFLEO converges in 3:20 hours with an accuracy of 82.9%. When CNN is substituted by MLP, the accuracy drops about 2-6% for the IID case and the non-IID, respectively but is still above 80% and the convergence time increased slightly to be around 6 hours, which is still rather desirable. When CIFAR-10 is used in place of the MNIST dataset as shown in Fig. 8c, the accuracy is slightly decreased and the convergence time is marginally increased to four hours, which is still significantly faster than the baselines. When we compare the results of one HAP with those of two HAPs (between (a) and (c) as well as between (b) and (c) in both Figs. 7 and 8), it can be observed that the latter further speeds up the convergence process and improves the performance of FL.

VI. Conclusion

To usher FL into Satcom with maximal effectiveness and efficiency, we propose a novel asynchronous FL framework, AsyncFLEO, for LEO constellations. We address the challenges of highly sporadic connectivity and irregular visit patterns between satellites and PS, as well as model staleness caused by straggler satellites, which altogether lead to large convergence delays and poor model performance. AsyncFLEO groups satellites from different orbits based on the similarity of their data distribution inferred from model weights. Furthermore, AsyncFLEO introduces a ring-of-star communication topology and a model propagation algorithm. Our extensive simulations show that AsyncFLEO accelerates FL model convergence by up to 22 times and at the same time improves accuracy by up to 40%, in comparison with several state-of-the-art approaches. The results also reveal that AsyncFLEO is robust to non-IID data, attaining an accuracy of 81.36% in 5 hours which is quite on a par with its 85.57% accuracy under the IID setting.

References

[1] H. Wu, J. Chen, C. Zhou, W. Shi, N. Cheng, W. Xu, W. Zhuang, and X. S. Shen, "Resource management in space-air-ground integrated vehicular networks: Sdn control and ai algorithm design," *IEEE Wireless Communications*, vol. 27, no. 6, pp. 52–60, 2020.

[2] A. Perez-Portero, J. F. Munoz-Martin, H. Park, and A. Camps, "Airborne gnss-r: A key enabling technology for environmental monitoring," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 14, pp. 6652–6661, 2021.

[3] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[4] 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.

[5] N. Razmi, B. Matthiesen, A. Dekorsy, and P. Popovski, "On-board federated learning for dense leo constellations," in *IEEE International Conference on Communications (ICC)*, Seoul, Southkorea, May 2022. [Online]. Available: https://arxiv.org/abs/2111.12769

[6] M. Elmahallawy and T. Luo, "FedHAP: Fast federated learning for LEO constellations using collaborative HAPs," in *2022 14th International Conference on Wireless Communications and Signal Processing*. IEEE, Nov 2022. [Online]. Available: https://arxiv.org/abs/2205.07216

[7] F. Hsieh, F. Jardel, E. Visotsky, F. Vook, A. Ghosh, and B. Picha, "Uav-based multi-cell haps communication: System design and performance evaluation," in *GLOBECOM 2020-2020 IEEE Global Communications Conference*. IEEE, 2020, pp. 1–6.

[8] S. C. Arum, D. Grace, and P. D. Mitchell, "A review of wireless communication using high-altitude platforms for extended coverage and capacity," *Computer Communications*, vol. 157, pp. 232–256, 2020.

[9] H. Chen, M. Xiao, and Z. Pang, "Satellite-based computing networks with federated learning," *IEEE Wireless Communications*, vol. 29, no. 1, pp. 78–84, 2022.

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

[11] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[12] J. G. Walker, "Satellite constellations," *Journal of the British Interplanetary Society*, vol. 37, p. 559, 1984.

[13] C. Xie, O. Koyejo, and I. Gupta, "Asynchronous federated optimization," in *12th OPT Workshop on Optimization for Machine Learning*, 2020.

[14] "Nasa, definition of two-line element set coordinate system [online]," Available: https://https://tinyurl.com/yc36cwju, May 2022.

[15] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[16] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10 (canadian institute for advanced research)," *http://www. cs. toronto. edu/kriz/cifar. html*, vol. 5, no. 4, p. 1.